algorithms

MDPI

Article

Experimental Validation of a Guaranteed Nonlinear Model Predictive Control

Mohamed Fnadi [1,*] and Julien Alexandre dit Sandretto [2]

[1] LISIC Laboratory—EA 4491, University of Littoral Côte d'Opale, 62100 Calais, France
[2] lENSTA Paris, Institut Polytechnique de Paris, U2IS Laboratory, 828 bd des Maréchaux, CEDEX, 91762 Palaiseau, France; julien.alexandre-dit-sandretto@ensta-paris.fr
* Corresponding: mohamed.fnadi@univ-littoral.fr

Abstract: This paper combines the interval analysis tools with the nonlinear model predictive control (NMPC). The NMPC strategy is formulated based on an uncertain dynamic model expressed as nonlinear ordinary differential equations (ODEs). All the dynamic parameters are identified in a guaranteed way considering the various uncertainties on the embedded sensors and the system's design. The NMPC problem is solved at each time step using validated simulation and interval analysis methods to compute the optimal and safe control inputs over a finite prediction horizon. This approach considers several constraints which are crucial for the system's safety and stability, namely the state and the control limits. The proposed controller consists of two steps: filtering and branching procedures enabling to find the input intervals that fulfill the state constraints and ensure the convergence to the reference set. Then, the optimization procedure allows for computing the optimal and punctual control input that must be sent to the system's actuators for the pendulum stabilization. The validated NMPC capabilities are illustrated through several simulations under the DynIbex library and experiments using an inverted pendulum.

Keywords: validated NMPC; ordinary differential equations; interval analysis; uncertain dynamic model

Citation: Fnadi, M.; Alexandre dit Sandretto, J. Experimental Validation of a Guaranteed Nonlinear Model Predictive Control. *Algorithms* **2021**, *14*, 248. https://doi.org/10.3390/a14080248

Academic Editor: Mircea-Bogdan Radac

Received: 27 July 2021
Accepted: 19 August 2021
Published: 20 August 2021

1. Introduction

Despite their maturing technology, robotic solutions still face many challenges related mainly to their design and control techniques. In practical applications, control of automated devices is often formulated depending on the nature of the operational task for which they are designed. Several research works have been realized in the past few years to develop optimal and reliable controllers, ensuring various reliability goals. For instance, authors in [1] investigated enhancing the reliability of control systems by using multiple controllers for a given plant, which is a delicate way to improve the reliability in control structures subject to both the controllers and plant failures. In [2], a design approach of reliable centralized and decentralized controllers is proposed, such that the resulting control systems remain stable and accurate despite sensor or actuator disturbances. Moreover, reliable H-infinity controllers for linear systems were designed in [3], in such a way that they ensure the asymptotic stability of the control systems, mainly when perturbations occur in some control components (sensors and actuators). These classical control strategies are generally designed based on dynamic and/or kinematic models and simplified control schemes. However, the drawback of these deterministic approaches is the lack of techniques to prove the robustness analysis of such controllers to different uncertain parameters in the plant.

Significant advances in stabilizing uncertain systems with parametric uncertainties have been recently treated in a very different way. Among the techniques used to achieve this end, interval analysis methods (IA) can provide guaranteed solutions to deal with complex nonlinear problems and to manage various uncertainties involved in the hardware

and software parts of the automated device [4]. Indeed, IA tools turn out to be more promising and reliable to handle all the noises and uncertainties in the experimental and theoretical data, contrary to these classical approaches that are generally based on simplified models with deterministic values. IA methods are extremely popular in several research fields due to their efficiency and reliability in order to achieve a robust control in the presence of model uncertainties [5,6] or for state estimation [7]. For example, a robust controller based on interval arithmetic for constrained discrete-time nonlinear systems with additive uncertainties is presented in [5]. In addition, authors in [6] developed a robust feedback controller synthesis such that all the unknown parameters are assumed to vary between known bounds. This controller guarantees at each time step that the closed-loop system behavior is greater than the lower bound of the set-point interval and is lower than its upper bound.

Model-based Predictive Control (MPC) methodology (also known as the receding horizon control or the moving horizon control) has become a powerful control technique widely used in a large number of areas (e.g., the automotive industry and chemical processes [8]). There are several reasons for this: (i) its capability to anticipate the future on the reference trajectory (i.e., future changes in set-point), and (ii) its ability to handle nonlinearities of multi-variable systems as well as all the system's constraints, i.e., input and state limits. The primary purpose of the linear and nonlinear MPC is to apply a plant model to predict its behavior along a receding horizon. At each sampling time, the MPC technique computes the optimal control input that minimizes some cost functions and fulfills all the safety constraints (e.g., actuators position, speed, and acceleration bounds). Overall, it has been proven that the MPC technique is much more practical and sophisticated to solve linear and nonlinear control problems [9–13]. Amongst the existing works, a real-time constrained MPC with safety and stability constraints is proposed in [9,10], where all the constraints are expressed as inequalities with respect to the optimization variable. Furthermore, a linear time-varying MPC method is designed in [12,13]. It is based on the online exact linearization of the nonlinear model so as to formulate the optimization problem completely as a Quadratic Program (QP). It can be easily handled by linear solvers to decrease the computational complexity. Nevertheless, they assume that uncertainties, related to the system's modeling and the degree of correctness of measurements, are omitted. Consequently, a control strategy in a guaranteed way is required to ensure robustness toward all the uncertainties occurring in dynamic parameters' estimation.

Currently, many researchers apply IA methods to synthesize a guaranteed linear or nonlinear MPC controllers [14,15]. These guaranteed methods allow the calculation of smooth and safe input intervals over the prediction horizon, i.e., the control vector is expressed here as an interval-vector (or a box that is the Cartesian product of intervals). Appendix A gives a brief presentation outlining different IA tools. For instance, authors in [14] use IA methods to build a new nonlinear MPC law considering the discrete system. It is based on forward-backward contraction using a dynamic model to compute feasible and admissible inputs. This method allows authors to calculate a validated control for an inverse pendulum system. Other techniques include the adjoint method for sensitivity analysis enabling the handling of the rate of change of the system dynamics considering the control variable [6,16–18]. Otherwise, this last approach is used to calculate the derivative of dynamics with respect to the input variable that must be limited between some bounds. In addition, a robust and guaranteed MPC of constrained discrete-time nonlinear systems with additive uncertainties is designed in [6,17] to quantify different uncertainties. Moreover, a branch-and-bound algorithm was applied in [15] to synthesize a NMPC controller. This combinatorial optimization algorithm is promising but needs high computation time to obtain optimal solutions. In addition, a reliable nonlinear MPC via interval arithmetic is developed in [19]. This algorithm uses validated simulation and branching methods to create input interval-vectors (or input boxes) and respect the system's constraints by consecutive bisection of the initial feasible boxes of actuator's inputs.

Contributions : This paper focuses on developing a new validated and reliable nonlinear model predictive control (NMPC), which is more specifically designed using an uncertain mathematical model. It encompasses two main research dimensions. (i) This paper is an extension of our previous work introduced in [20] that aims to identify in a guaranteed way the inertial and frictional parameters of our new inverted pendulum (manufactured in our lab and shown in Figure 1). The IA and set-inversion tools have been deployed to express all the needed parameters as intervals instead of constants' values so that the coverage ratio between the physical reality and the mathematical model is significant. (ii) The second contribution lies in the synthesis of a validated NMPC strategy relying on the previous guaranteed identification to compute smooth and safe input intervals. These approaches have been investigated through various simulations with uncertain high-order ordinary differential equations (ODEs) describing the system's behavior. The validated simulations are performed using our DynIbex (DynIbex Library: https://perso.ensta-paris.fr/chapoutot/dynibex/ (accessed on 19 August 2021)) library (a sophisticated solver of ODEs using Runge–Kutta schemes [21]) and several experiments on the actual nonlinear inverted pendulum.

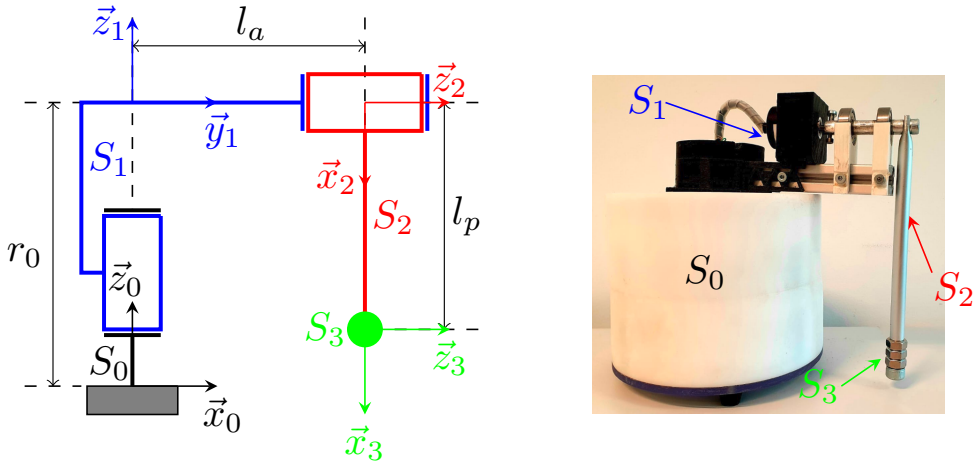

Figure 1. (**Left**) Definition of links frame (configuration $y_1 = y_3 = 0°$); (**Right**) representation of the real rotary inverted pendulum manufactured in our laboratory (S_0: pendulum housing, S_1: horizontal arm, S_2: pendulum arm, S_3: balancing mass).

The remainder of this paper is organized as follows: firstly, an overview of the direct dynamic model of our inverted pendulum is presented in Section 2. Secondly, Section 3 introduces the guaranteed NMPC approach based on IA. Thirdly, simulation and experimental results and discussions are reported in Section 4, on which the guaranteed identification results using the SIVIA algorithm (Set-Inversion Via Interval Analysis [7]) are recapped, and the capabilities of the proposed controller are illustrated via the nonlinear inverted pendulum. Finally, Section 5 closes the paper by summarizing the contribution and providing an overview of the future work.

2. Background of Dynamic Modeling of the Inverted Pendulum

2.1. Robotic Device

The inverted pendulum is a simple system with two serial rotational joints (active and passive). We have designed and manufactured one in our laboratory (see Figure 1). Our experimental device is equipped with a brushless DC motor to actuate the first joint and two incremental encoders to measure the rotational angles of both the motor shaft and the passive joint. Its kinematics is sketched in Figure 1. This device must be identified and stabilized in a validated and guaranteed way, considering all the errors and uncertainties related to the system modeling and data measurements.

2.2. Dynamic Modeling of the Inverted Pendulum

The Euler–Lagrange equation is a systematic method used here to express the system's equations of motion, describing various nonlinear derivatives to get the ODEs (for more details, see [22]). After going through this process, the nonlinear equations of a direct dynamic model for rotary nonlinear inverted pendulum are given by

$$\begin{cases} \dot{y}_1 = y_2, \\ \dot{y}_2 = \dfrac{1}{\Delta}\Big\{ -\mu_3 \cos(y_3)\left[\mu_1 \sin(y_3)\cos(y_3)y_2^2 + \Gamma_2 - \mu_g \sin(y_3)\right] + \\ \qquad\qquad \mu_4\left[\mu_3 \sin(y_3)y_4^2 - \mu_1 \sin(2y_3)y_2 y_4 + \Gamma_1\right]\Big\}, \\ \dot{y}_3 = y_4, \\ \dot{y}_4 = \dfrac{1}{\Delta}\Big\{ \left[\mu_1 \sin(y_3)^2 + \mu_2\right]\left[\mu_1 \cos(y_3)\sin(y_3)y_2^2 + \Gamma_2 - \mu_g \sin(y_3)\right] - \\ \qquad\qquad \mu_3 \cos(y_3)\left[\mu_3 \sin(y_3)y_4^2 - 2\mu_1 \cos(y_3)\sin(y_3)y_2 y_4 + \Gamma_1\right]\Big\}, \end{cases} \tag{1}$$

where y_1 and y_2 indicate, respectively, the angular position and velocity of the rotative arm. Likewise, y_3 and y_4 are the angular and speed variables of the pendulum arm. We draw all these agreements as $y = [y_1, y_2, y_3, y_4] \in \mathbb{R}^4$ referring to the state vector. Γ_i is the torque applied at the joint i, which depends on the viscous friction variables, and

$$\Delta = -\mu_3^2 \cos(y_3)^2 + \mu_1\mu_4 \sin(y_3)^2 + \mu_2\mu_4,$$

$$\mu_1 = l_p^2\left[\frac{m_p}{4} + M\right], \quad \mu_2 = l_a^2[m_p + M] + \frac{J_m}{N_g^2} + J_a,$$

$$\mu_3 = l_p l_a\left[\frac{m_p}{2} + M,\right], \quad \mu_4 = l_p^2\left[\frac{m_p}{4} + M\right] + J_p, \quad \mu_g = \left[\frac{m_p}{2} + M\right] l_p g,$$

where m_a and J_a denote the horizontal arm's mass and its inertia, respectively—similarly for the pendulum arm with the mass m_p and inertia J_p. M is the mass of the load attached to the pendulum arm, J_m is the DC motor inertia, and N_g its gear ratio. l_a, l_p, and r_0 are the device's dimensional parameters (as defined in Figure 1), and g is the gravitational acceleration.

Since the second joint is passive, Γ_2 can only be written as a function of friction conditions. The relationship between friction and velocity is well modeled in the literature [20]. In this paper, we use a simple friction model, in which it is related to the viscous terms, which are proportional to the joint speed:

$$\begin{bmatrix} \Gamma_1 \\ \\ \Gamma_2 \end{bmatrix} = \begin{bmatrix} \tau - f_{v1} y_2 \\ \\ -f_{v2} y_4 \end{bmatrix}, \tag{2}$$

where τ is the motor's torque and f_{v_i} is the viscous Coulomb friction coefficient of the joint i. These coefficients are often considered as constant values in the majority of literature works. Here, the main goal is to identify them in a guaranteed way by reducing the error between the theoretical model output and measured data output, i.e., they must belong to specific subsets consistent with the model and sensors' outputs. The guaranteed estimated parameters are reported in [20] and outlined in Table 1.

Table 1. Identified inertial and frictional parameters of the inverted pendulum.

Symbol	Measurement Unit	Interval Enclosure
μ_1	kg·m^2	$[9.63318e^{-4};\ 1.22604e^{-3}]$
μ_2	kg·m^2	$[2.19585e^{-3};\ 2.79471e^{-3}]$
μ_3	kg·m^2	$[7.227e^{-4};\ 8.833e^{-4}]$
μ_4	kg·m^2	$[1.08271e^{-3};\ 1.37799e^{-3}]$
μ_g	kg·m^2·s^{-2}	$[6.24299e^{-2};\ 8.44641e^{-2}]$
f_{v_1}	N·m·s·rad^{-1}	[0.043012; 0.13002]
f_{v_2}	N·m·s·rad^{-1}	[0.000454; 0.001174]

2.3. Torque, Speed, and Acceleration Computation

Because our experimental datasets have noises, the recorded signals are consequently processed by applying an eighth-order low-pass Butterworth filter. The input of the system (torque signal) and the joints' velocity and acceleration are required to stabilize the nonlinear inverted pendulum. The speed and acceleration are computed using a non-causal derivative filter. The function *firpm* of Matlab allows us to synthesize this kind of filter to determine the suitable filter's parameters (impulse response coefficients). The second stage is the actual application of the designed filter to the joints' positions (y_1 and y_3), using *filtfilt* command, enabling to process a data in both the forward and reverse directions (zero-phase distortion).

On the other hand, the torque generated at the base of the rotary arm (i.e., at the load gear) can be expressed as a function of the angular velocity (input variable) denoted u as below:

$$\tau = \frac{N_m N_g \eta_m \eta_g (u_0 - u)}{R_v} \tag{3}$$

where N_m and N_g are respectively the motor and gear ratios, η_m and η_g are the motor and gearbox efficiencies, u and u_0 are the instantaneous and no-load angular velocities, respectively, and R_v is the motor speed-torque constant.

3. Guaranteed Controller Design via Validated Simulation

In this section, a validated Nonlinear Model Predictive Control (NMPC) is developed to stabilize the inverted pendulum in the upright position considering the system's constraints and uncertainties. Firstly, we give a general overview of the validated numerical integration methods in Section 3.1. Secondly, we present the validated NMPC in Section 3.2.

3.1. Validated Numerical Integration

The dynamic model given by Equation (1) can simply be written by this nonlinear ODEs:

$$\dot{y}_t = \mathbf{f}(t, y_t, u_t) \tag{4}$$

At each validated numerical integration, we assume that u_t is constant to compute corresponding tight enclosures of the state variable. We are now considering an Initial Value Problem for ODEs (IVP-ODEs) over the time interval $t \in [0, T]$,

$$\begin{cases} \dot{y}_t = \mathbf{f}(t, y_t, u_t) \\ y_0 \in [\mathbf{y}_0] \subseteq \mathbb{R}^n \\ u_t \in [\mathbf{u}] \subseteq \mathbb{R}^m \end{cases} \tag{5}$$

where the sets $[\mathbf{y}_0]$ and $[\mathbf{u}]$, both expressed as boxes, are respectively the initial condition related to the initial state vector and the considering input interval. This IVP-ODE has a unique solution $y_t(t; y_0, u_t)$ at $t > 0$ since $\mathbf{f} : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is continuous in t and Lipschitz in y_t, but, for our purpose, we assume $\mathbf{f}$ to be smooth enough, i.e., of class C^k.

Compared to the classical numerical integration for the IVP-ODE problem, validated approaches consist of solving this problem in a complete and validated way (i.e., each actual guaranteed solution is rigorously returned and enclosed in a tight interval). These methods are generally based on Taylor series [23] or Runge–Kutta methods [21]. Basically, the main principle of the validated methods is to obtain the tight enclosure of the IVP-ODEs problem. As presented in [23], the purpose of a validated numerical algorithm is to solve Equation (5) so as to get a sequence of boxes $[\mathbf{y}_0], \ldots, [\mathbf{y}_n]$ at time iterations $t_0 = 0 < \ldots < t_n = T$ and at given input boxes $[\mathbf{u}_0], \ldots, [\mathbf{u}_n]$. It is achieved in such a way that the inclusion function verifies $[\mathbf{y}_{j+1}] \supseteq [\mathbf{f}]\left(t_j, [\mathbf{y}_j], [\mathbf{u}_j]\right)$, $\forall j \in [0, n]$. Such approaches work in two stages at each integration step to compute the guaranteed solutions. They are summarized as:

i Computation of a *prior* enclosure of the solution $[\tilde{\mathbf{y}}_{j+1}]$, such that $\mathbf{f}\left(t_j, [\mathbf{y}_j], [\mathbf{u}_j]\right) \in [\tilde{\mathbf{y}}_{j+1}]$ for all t in the time interval $t \in [t_j, t_{j+1}]$. This stage enables proving the existence and the uniqueness of the solution.

ii Computation of a *tight* enclosure of state variable $[\mathbf{y}_{j+1}]$ at time instance t_{j+1}, such that $\mathbf{f}\left(t_{j+1}, [\mathbf{y}_j], [\mathbf{u}_j]\right) \in [\mathbf{y}_{j+1}]$. It uses the solution $[\tilde{\mathbf{y}}_{j+1}]$ to bound the *truncation error*, i.e., the distance between the exact and the numerical solutions.

The *tight* and *prior* enclosures calculated along one integration step between t_j and t_{j+1} (of size $h_j = t_{j+1} - t_j > 0$) can be visualized in Figure 2.

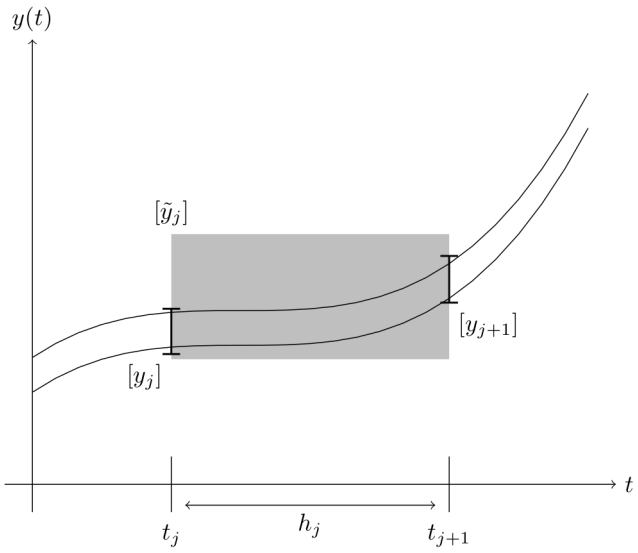

Figure 2. Prior and tight enclosures computed along one iteration using validated methods.

3.2. Validated Nonlinear Model Predictive Control

The purpose of this section is to apply the validated simulation algorithm presented in the previous section to compute a solution satisfying the NMPC constraints and minimizing an interval cost function.

The validated NMPC aims to stabilize the nonlinear inverted pendulum in the upright position. It is based on interval analysis tools to compute guaranteed control variables over the receding horizon using the validated numerical integration method of the IVP-ODEs problem (5). The control intervals are calculated such that the convergence to the set-point interval is ensured (i.e., $y_i \rightarrow [\mathbf{y}_r]$, $\forall i$), and all the state and input constraints are satisfied (i.e., $y_i \in [\mathbf{y}_i]$ and $u_j \in [\mathbf{u}_j]$, $\forall i, j$). The validated NMPC is decomposed into two steps:

- The IVP-ODEs problem (5) is simulated in a complete and validated way starting from the initial domain of inputs referring to the actuators' saturation. Then, for each tested input set, we check if the constraints on state variables are respected using validated simulation and branching methods (Branch-and-Prune algorithm).

- Since safe inputs are computed over a finite time horizon, the optimal controller box is calculated by minimizing the interval objective function, optimizing the error between the predicted outputs and the reference set, and the norm of the inputs.

3.2.1. Filtering and Branching Procedures

The validated NMPC strategy should provide an optimal input box at each time-step k. This latter can be denoted as $[\mathbf{U}]_k^\star = [\mathbf{u}]_k^\star \times [\mathbf{u}]_{k+1}^\star \times \ldots \times [\mathbf{u}]_{k+N_p-1}^\star$, where N_p is the number of input samples generated over the prediction horizon. Branching procedure allows us to generate safe input intervals $[\mathbf{U}]_k^\star$ over the prediction horizon that satisfies all the state constraints and convergence to the reference interval $[\mathbf{y}_r]$. Therefore, the system can be described using a simple interval notation by the following constraint differential equation:

$$\begin{aligned} &\dot{y}_t = \mathbf{f}(t, y_t, u_t), \\ &\text{s.t.}: \quad u_t \in \mathbb{U}, \quad \forall t \geq 0 \\ &\qquad\quad y_t \in \mathbb{Y}, \quad \forall t \geq 0 \end{aligned} \tag{6}$$

where $\mathbb{U}$ and $\mathbb{Y}$ are respectively the input and state variables saturation. They can be written as

$$\mathbb{U} = \{u_t \in \mathbb{R}^m \mid u_{\min} \leq u_t \leq u_{\max}\} \quad \text{and} \quad \mathbb{Y} = \{y_t \in \mathbb{R}^n \mid y_{\min} \leq y_t \leq y_{\max}\} \tag{7}$$

where $u_{\min}$ and $u_{\max}$ are the admissible domain bounds for the control variable. Similarly, $y_{\min}$ and $y_{\max}$ are those of the state components.

Starting from the current state boxes $[\mathbf{y}_t]$ computed at time t and input feasible domain $[\mathbf{u}_k] = [u_{\min}, u_{\max}]$, the computation of a new state domain $[\mathbf{y}_{t+T_c}]$ is performed by the validated simulation algorithm, where T_c denotes the sampling time of the control signal, and then the horizon time $T_p = T_c \times N_p$. This strategy bisects the initial domain $[\mathbf{u}_k]$ on the left and right intervals that reduces the width of the initial interval $[\mathbf{u}_k]$. For each bisected input interval, the validated simulation algorithm provides a state domain of which the safety and convergence criteria are verified, as can be seen in this equation:

$$[\mathbf{f}](t, [\mathbf{u}_k], [\mathbf{y}_t]) = [\mathbf{y}_{t+T_c}] \quad \text{and} \quad [\mathbf{y}_{t+T_c}] \subseteq [y_{\min}, y_{\max}] \tag{8}$$

The filtering and branching algorithm is given in Algorithm 1. The function bisect-check is a pruning function that performs a bisection procedure of the initial input interval if its width exceeds some tolerance *tol* value. One side of bisected intervals is kept by considering these criteria: (i) A branch leading to unsafe state is avoided (i.e., $[\mathbf{y}_{t+T_c}] \nsubseteq [\mathbf{y}_{\min}, \mathbf{y}_{\max}]$); and (ii) a branch leading to a state far from the reference interval $[\mathbf{y}_r]$ or partially a cost greater than the other branch is eliminated (see interval optimization in Section 3.2.2). Moreover, the function Test sensitivity() enables us to simulate the adjoint function $x(u) = \dfrac{\partial y}{\partial u}$ expressing the rate of variation of the state concerning the control variable. Its derivative can be derived as below:

$$\dot{x} = \frac{\partial \mathbf{f}}{\partial y} \times x + \frac{\partial \mathbf{f}}{\partial u} \tag{9}$$

Using the Test sensitivity() function, it returns the Boolean formula:

$$\{\exists \epsilon > 0,\ 0 \in x([\mathbf{u_k}])\ \text{AND}\ x([\mathbf{u}_k]) \nsubseteq [-\epsilon, \epsilon]\}\ \text{OR}\ \{\exists \tau \in [t, t + T_c],\ y(\tau) \nsubseteq \mathbb{Y}\} \tag{10}$$

Sensitivity analysis allows for checking if the computed input $[\mathbf{u_k}]$ is leading the system to move in forward and backward directions at the same time, as can be seen by the first part of the Boolean expression. The second part of this formula enables us to verify if the state constraints are respected. Since the safe inputs' intervals are computed using the validated simulation, the interval cost function is optimized to calculate its sub-optimal solution. The optimization part is detailed in the next section.

Algorithm 1 Filtering and branching

Require: $[\mathbf{y}_0], [\mathbf{y}_r], T_c, N_p, tol$
  while not success **do**
    while $k \leq N_p$ **do**
      if $w([\mathbf{u}_k]) > tol$ **then**
        $[\mathbf{u}_k] =$ bisect-check$([\mathbf{u}_k]_{\text{left}}, [\mathbf{u}_k]_{\text{right}})$
      end if
      $[\mathbf{y}_{t+T_c}] =$Simulation$([\mathbf{y}_t], [\mathbf{u}_k], T_c)$
      if $[\mathbf{y}_{t+T_c}] \subseteq [\mathbf{y}_r]$ **then**
        $k = k+1, \ [\mathbf{y}_t] = [\mathbf{y}_{t+T_c}], \ t = t+T_c$
        $success = true$
      else
        if Test sensitivity() **then**
          if $w([\mathbf{u}_k]) > tol$ **then**
            $[\mathbf{u}_k] =$ bisect-check$([\mathbf{u}_k]_{\text{left}}, [\mathbf{u}_k]_{\text{right}})$
          else
            $success = false$
          end if
        else
          $[\mathbf{y}_t] = [\mathbf{y}_{t+T_c}]$
          $k = k+1, \quad t = t+T_c$
          $success = true$
        end if
      end if
    end while
  end while

3.2.2. Interval Cost Function Formulation

Since the feasible intervals of inputs are computed fulfilling different constraints, the optimization procedure is required to find the optimal control box and then the punctual input that we inject into the actuator's control unit u_1. The continuous cost function can be expressed over the prediction horizon $T_p = T_c \times N_p$ as

$$J(y_t, u_t) = \int_t^{t+T_p} F(y(\tau), u(\tau)) d\tau \tag{11}$$

where F is in general a quadratic function that minimizes the norm of the inputs and the error between the predicted outputs y and the reference y_r. F can be expressed in the following quadratic form:

$$F(y(\tau), u(\tau)) = (y(\tau) - y_r)^T \mathbf{Q}(y - y(\tau)_r) + u(\tau)^T \mathbf{R} u(\tau) \tag{12}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are both positive semi-definite weighting matrices.

The continuous objective function (11) then takes the form

$$\begin{aligned} J(y_t, u_t) &= \int_t^{t+T_p} \Big[(y(\tau) - y_r)^T \mathbf{Q}(y(\tau) - y_r) + u(\tau)^T \mathbf{R} u(\tau) \Big] d\tau \\ &= \sum_{k=1}^{N_p} \int_{t+(k-1)T_c}^{t+kT_c} \Big[(y(\tau) - y_r)^T \mathbf{Q}(y(\tau) - y_r) \Big] d\tau + \sum_{k=1}^{N_p} \int_{t+(k-1)T_c}^{t+kT_c} u(\tau)^T \mathbf{R} u(\tau) d\tau \end{aligned} \tag{13}$$

The purpose is to express the cost function (13) in a discrete form using interval analysis methods. Since the control signal is constant along the input sampling period $[t + (k-1)T_c, t + kT_c]$, the second integral can be simplified as

$$J(y_t, u_t) \quad = \sum_{k=1}^{N_p} \int_{t+(k-1)T_c}^{t+kT_c} \Big[(y(\tau) - y_r)^T \mathbf{Q}(y(\tau) - y_r) \Big] d\tau + \ T_c \sum_{k=1}^{N_p} \Big[u_k^T \mathbf{R} u_k \Big] \tag{14}$$

A validated simulation method generates a list of tight enclosures' $[\mathbf{y}_0], [\mathbf{y}_1], \ldots, [\mathbf{y}_n]$ solutions of the IVP-ODEs (5) from the initial conditions $[\mathbf{y}_0]$ and $[\mathbf{u}_0]$. Therefore, each solution y can be enclosed into a box thanks to the validated methods (i.e., $\forall t \in [t + (k-1)T_c, t + kT_c]$, $y(t) \in [\mathbf{y}_t]$). Hence, one useful consequence of the Rectangle Property is that it allows us to bound the first integral of the cost (14) between two bounds as can be defined below:

$$\sum_{k=1}^{N_p} \int_{t+(k-1)T_c}^{t+kT_c} \Big[(y(\tau) - y_r)^T \mathbf{Q}(y(\tau) - y_r)\Big] d\tau \quad \in \quad T_c \sum_{k=1}^{N_p} \Big[([\mathbf{y}_t] - y_r)^T \mathbf{Q}([\mathbf{y}_t] - y_r)\Big] \tag{15}$$

By exploiting (15), the interval cost function can be approximated as

$$\begin{aligned} J(y_t, u_t) \quad &\in \quad T_c \sum_{k=1}^{N_p} \Big[([\mathbf{y}_t] - y_r)^T \mathbf{Q}([\mathbf{y}_t] - y_r) \; + \; [\mathbf{u}_k]^T \mathbf{R}[\mathbf{u}_k]\Big] \\ &\leq \quad ub\left\{ T_c \sum_{k=1}^{N_p} \Big[([\mathbf{y}_t] - y_r)^T \mathbf{Q}([\mathbf{y}_t] - y_r) \; + \; [\mathbf{u}_k]^T \mathbf{R}[\mathbf{u}_k]\Big] \right\} \end{aligned} \tag{16}$$

where ub signifies the upper bound of the interval (lb denotes its lower bound).

Algorithm 2 recaps the optimization procedure to determine the inputs boxes allowing for minimizing the cost function (16). It returns the sub-optimal solutions that minimize the formulated cost function. It is based on consecutive bisections of the first input interval $[\mathbf{u_1}]$ until achieving a small width value tol. The punctual and guaranteed input value u_1 that we send to the DC motor represents the smaller value of the upper and lower bound of this interval.

Algorithm 2 Optimization

Require: $[\mathbf{y}], [\mathbf{u}_1], [\mathbf{u}_2], \ldots, [\mathbf{u}_{N_p}], tol$
  while $w([\mathbf{u}_1]) \geq tol$ **do**
    $[\mathbf{U}]_{\text{left}} = [\mathbf{u}_1]_{\text{left}} \times [\mathbf{u}_2] \times \ldots \times [\mathbf{u}_{N_p}]$
    $[\mathbf{U}]_{\text{right}} = [\mathbf{u}_1]_{\text{right}} \times [\mathbf{u}_2] \times \ldots \times [\mathbf{u}_{N_p}]$
    if $J([\mathbf{y}], [\mathbf{U}]_{\text{left}}) \geq J([\mathbf{y}], [\mathbf{U}]_{\text{right}})$ **then**
      $[\mathbf{U}]^\star = [\mathbf{U}]_{\text{right}}$
    else
      $[\mathbf{U}]^\star = [\mathbf{U}]_{\text{left}}$
    end if
  end while
  $u_1 = \min\{lb([\mathbf{u}_1]), ub([\mathbf{u}_1])\}$
  $send(u_1)$

To summarize, the proposed approach is focused on two main stages :

Algorithm 1:

returns the safe input tubes (N_p intervals) that allow for respecting the state constraints. This algorithm uses the validated simulation methods to compute the guaranteed inputs. As already mentioned, we assume that **the input interval** $[\mathbf{u}_t]$ **is constant** for each validated simulation to calculate the state tubes equivalent to each input interval. If the state limits are not fulfilled, the bisection procedure of the control interval is performed, and validated simulations are re-started.

Algorithm 2:

Because of the guaranteed input tubes (N_p intervals), the optimization algorithm is started to compute **the sub-optimal** input interval that minimizes as much as we can the formulated interval cost-function. It is based on the consecutive bisections of the guaranteed interval $[\mathbf{u}_1]$ computed by the first algorithm.

4. Experiment Results

The control framework introduced in this paper is applied for the stabilization of the nonlinear inverted pendulum shown in Figure 1. The pendulum is actuated by a DC motor whose angular speed is the input variable. It is controlled using a Phidget Motor Control. This control card is connected to the laptop via a USB cable to control the motor's velocity, using C language.

To evaluate the validated NMPC, we will initially summarize in Section 4.1 the guaranteed identification results and its validation using the identified subsets of parameters. In the second place, we will validate the proposed controller in Section 4.2 through simulation tests and then experiments using the inverted pendulum.

4.1. Guaranteed Dynamic Model Identification

The guaranteed identification of dynamic parameters $p = [\mu_1, \mu_2, \mu_3, \mu_4, \mu_g, f_{v_1}, f_{v_2}] \in \mathbb{R}^{n_p=7}$ as can be seen by the dynamic model (1) was proposed in [20]. These parameters are estimated as intervals instead of real numbers to account for all the uncertainties and errors related to the sensors measurements and the system modeling. The identification approach is based on the IA and set-inversion techniques (SIVIA algorithm) to determine the feasible sets of coefficients [7]. A brief resume of the SIVIA algorithm principle is given in Appendix A. The feasible sets of all the dynamic parameters are recapped in Table 1.

The DC motor and gearbox are of type *MDP-Maxon*, and their characteristics are provided by the manufacturer. Here is a recap of their properties: the motor inertia J_m = 9.06e^{-7} kg·m^2, its no-load speed u_0 = 11,700 tr·min^{-1}, the slope speed-torque Rv = 34,000 tr·min^{-1}·N^{-1}·m^{-1}, the motor and gearbox efficiencies η_m = 90.6% and η_g = 90%, and gear ratios N_m = 5.2 and N_g = 5.

To demonstrate the usefulness of our identification, we can solve in a complete and validated way the IVP-ODEs (5) using the identified parameters. The set of possible tight enclosures $[\mathbf{y}_1], \ldots, [\mathbf{y}_n]$ at a sequence of time-instants $t_1, \ldots, t_n$ are calculated. To do so, our library DynIbex is employed to solve this IVP-ODEs [21]. Furthermore, the guaranteed identification is compared to the traditional least square method identification (LSMI) (where the dynamic parameters are estimated as constant values) to show its efficiency. Figure 3 shows the measured pendulum angle y_3 (black lines), as well as the ones calculated with the identified parameters with LSMI (red lines) and IA (blue lines) approaches. All the model validation tests are conducted under the same settings and conditions (e.g., same initial conditions $[\mathbf{y}_0]$ and $[\mathbf{u}_0]$, same inputs, etc.). The simulation duration is fixed to be 4 s and the precision of 10^{-7}. We can notice that the simulated tubes of the pendulum angle y_2 given by the sophisticated solver DynIbex, using IA and LSMI parameters, are quite similar to the real measured pendulum angle. Nonetheless, the simulated y_2 with IA parameters comply as closely as possible with the measured one at different initial conditions. All this is due to the fact that all the system variables are taken here as intervals characterizing different uncertainties.

In order to emphasize the potential capabilities of the predicted model, the associated coverage rates can be computed. We evaluate the number of cases where the measurements data are included in the simulated tubes, and then the percentage of inclusion is calculated for each tested scenario. In other words, it is calculated by checking at each time step whether the time variable related to the measurement is included in the simulated time interval $t_i \in [\mathbf{t_i}]$. If so, we inspect if the corresponding measure y_i is included in the simulated tube $[\mathbf{y}_i]$ given by the DynIbex library. Then, the coverage rate can be approximated to be $\frac{N_q}{N_t} \times 100$, where N_t is the number of times that each sampled time is incorporated in the simulated time interval, and N_q is the one related to the second condition regarding y_i. The computed values are recapped in Table 2. By the way, this criterion is too strict to test, which explains the small calculated percentages. This is due to the timing lag between the measurement and the estimated model. Even with this issue, the coverage ratios obtained using IA parameters indicate a good match with the measurements process.

It also confirms that the model is estimated with high precision when the uncertainties are considered.

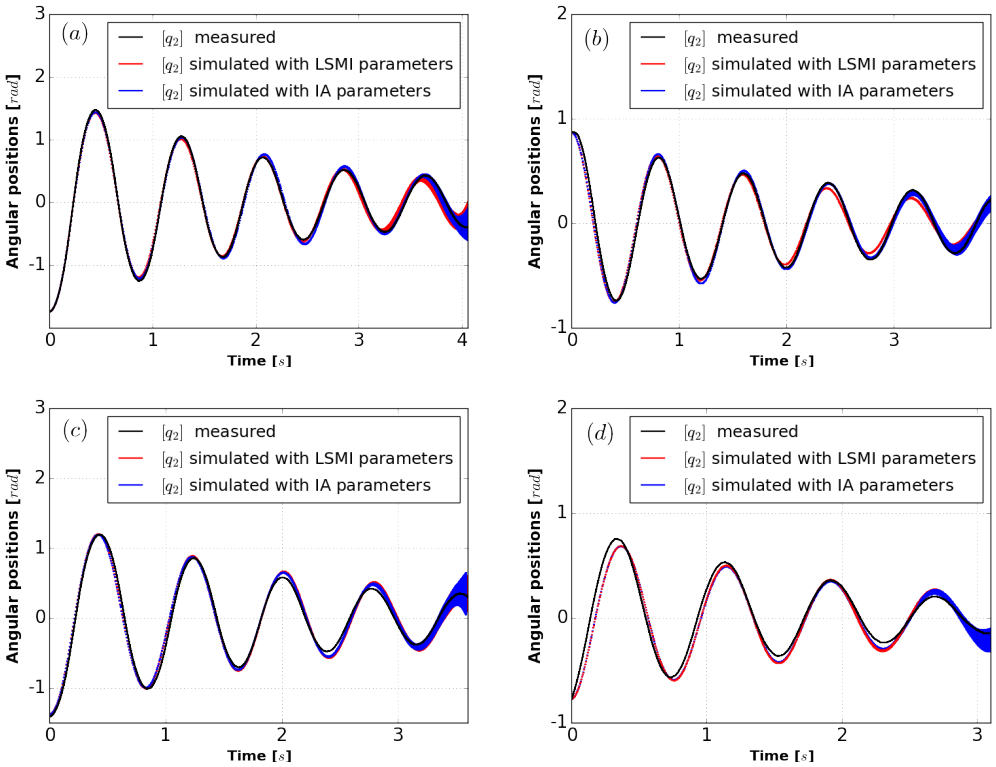

Figure 3. Open-Loop validation of the dynamic model under DynIbex library. Comparison between the real and simulated pendulum angular position (represented as tubes) at different initial conditions: (**a**) $y_0 = [0, 0, -100°, 0]^T$ and $u_0 = 0$ rad·s^{-1}; (**b**) $y_0 = [0, 0, 50°, 0]^T$ and $u_0 = 0$ rad·s^{-1}; (**c**) $y_0 = [0, 0, -90°, 0]^T$ and $u_0 = 40$ rad·s^{-1} and (**d**) $y_0 = [0, 0, -45°, 0]^T$ and $u_0 = 25$ rad·s^{-1}.

Table 2. Degree of coverage between the model and physical reality.

Scenario	(a)	(b)	(c)	(d)
With IA parameters	46%	61%	38%	25%
With LSMI parameters	41%	34%	20%	21%

Overall, the experimental results provided by the guaranteed identification of inertial and frictional parameters are more relevant than those obtained by the classical LSMI. Indeed, the estimation approach based on IA and set inversion techniques allows us to successfully identify these coefficients as intervals considering all the errors and uncertainties related to the sensors and the system modeling. Despite its proven effectiveness to more closely approximate the model output to the actual system's behavior, it still has some drawbacks. The main ones are related globally to the computation time, which depends mainly on the number of parameters and the initial domain. This issue can be enhanced with a contraction approach [24].

4.2. NMPC Results

4.2.1. Simulation Results

In the simulations, the parameters of this proposed validated NPMC are tuned as: prediction horizon $N_p = 10$, control sampling time $T_c = 16$ ms and the final time of the simulation $T_f = 0.3$ s. The safety saturation on state variables are: $y_1 \in [-2\pi;\ 2\pi]$, $y_2 \in [-52.4$ rad·s$^{-1};\ 52.4$ rad·s$^{-1}]$, $y_3 \in [-2\pi;\ 2\pi]$, $y_4 \in [-100$ rad·s$^{-1};\ 100$ rad·s$^{-1}]$, and the DC motor's torque constraints are $\tau \in [-8.05$ N·m; 8.05 N·m]. The reference interval of the desired pendulum arm is $y_r \in [\pi - 0.1;\ \pi + 0.1]$. The tolerance parameter applied

for the bisection procedure is adjusted as $tol = 0.25$. Since the filtering and branching algorithm begins from the admissible input domain $[-8.05$ N·m; 8.05 N·m], it can lead to approximately 64^{10} branches with $N_p = 10$. The interval cost function is computed only in the optimization process using these weighting matrices $\mathbf{Q} = diag[1000, 1000, 1000, 1000]$ and $\mathbf{R} = 1$.

Several software tools have been developed to simplify the resolution of the validated NMPC strategy via interval analysis. These tools are globally in the form of libraries to be integrated into a program (e.g., DynIbex library [21], Numerica [25] or RealPaver [26]). However, it has been shown that these solvers are not much more practical due to their high computation time. In other words, the fast convergence of this kind of solvers is not guaranteed over a finite time (i.e., extreme long response time to get the solution of the problem). That's why the real-time validation of this guaranteed approach is not easy to realize. To do so, it is divided into two steps: a numerically offline calculation of the input variables via validated simulations and then the injection of these inputs into the physical pendulum.

Two scenarios have been executed with different initial conditions on the angular pendulum position $[\mathbf{y}_{3_{ini}}]$ to discuss the capabilities of the proposed controller. Figure 4a–c display the results of the first scenario starting from $[\mathbf{y}_{3_{ini}}] = [0°;\ 0°]$ and Figure 5a–c show those of the second one starting from $[\mathbf{y}_{3_{ini}}] = [149°;\ 150°]$. As can be seen from Figure 4a, the pendulum arm starts from the downward position, and it is stabilized via the validated NMPC in its vertical position interval $[\mathbf{y}_r]$ with a small settling-time (around $t_{r_{5\%}} \approx 0.18$ s). Similarly, the simulation has also been executed when the initial position is close to the desired position π. The generated enclosures of the inverted pendulum position are plotted in Figure 5a. As the previous results, the validated NMPC succeeds in stabilizing the pendulum feasibly in its terminal position.

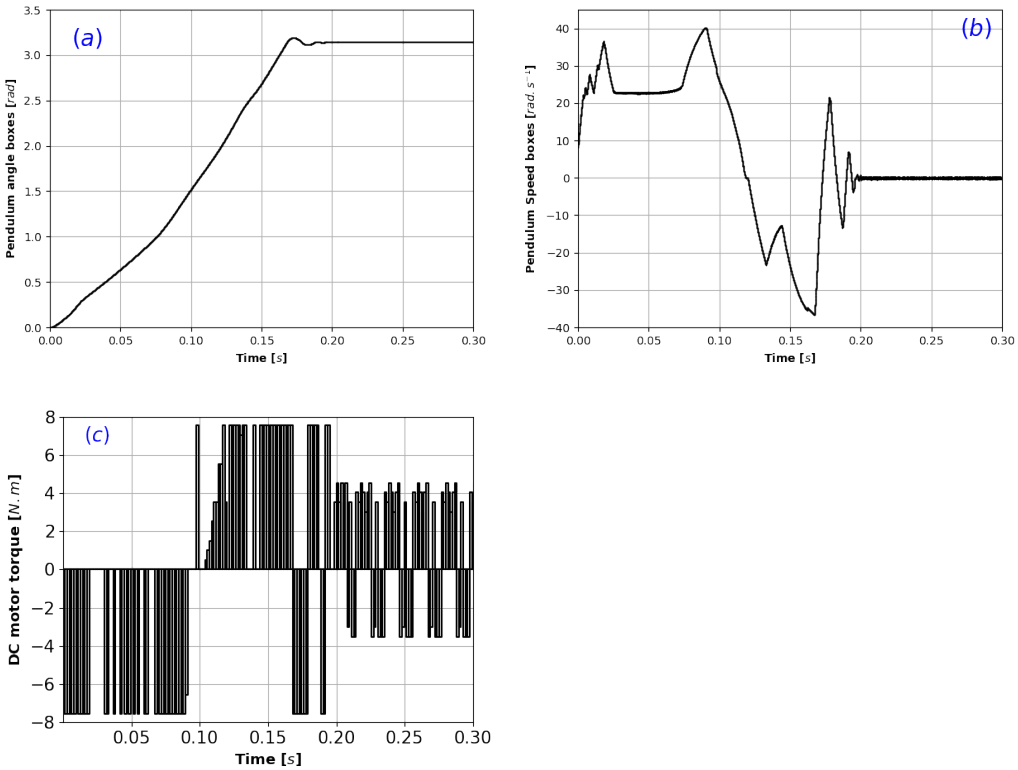

Figure 4. Validated NMPC results starting from $[\mathbf{y}_{3_{ini}}] = [0°;\ 0°]$: (**a**) pendulum angle tubes; (**b**) the corresponding pendulum speed tubes; (**c**) safe and optimal input Intervals $[\mathbf{u}_1]$.

On the other hand, the calculated tubes of the pendulum's velocity are depicted in Figures 4b and 5b. As can be noticed in both of these scenarios, the validated velocity respects the imposed constraints and stagnate around zero when the pendulum reaches its

final desired position. In addition, the torques applied by the DC motor for each tested scenario are shown in Figures 4c and 5c. The relationship between the motor's velocity and torque is given by Equation (3). In the first case, a strong input variable with the bang-bang process is applied to enable the pendulum to catch the equilibrium and stabilize in its terminal reference interval (see Figure 4c). Since the pendulum is stabilized, the torque applied by the system's actuator changes the sign to maintain the reference interval. In the second case, the actuator applies maximum inputs for a short time to try to reach the desired interval (see Figure 5c). All the calculated torques fulfill the system's intrinsic constraints.

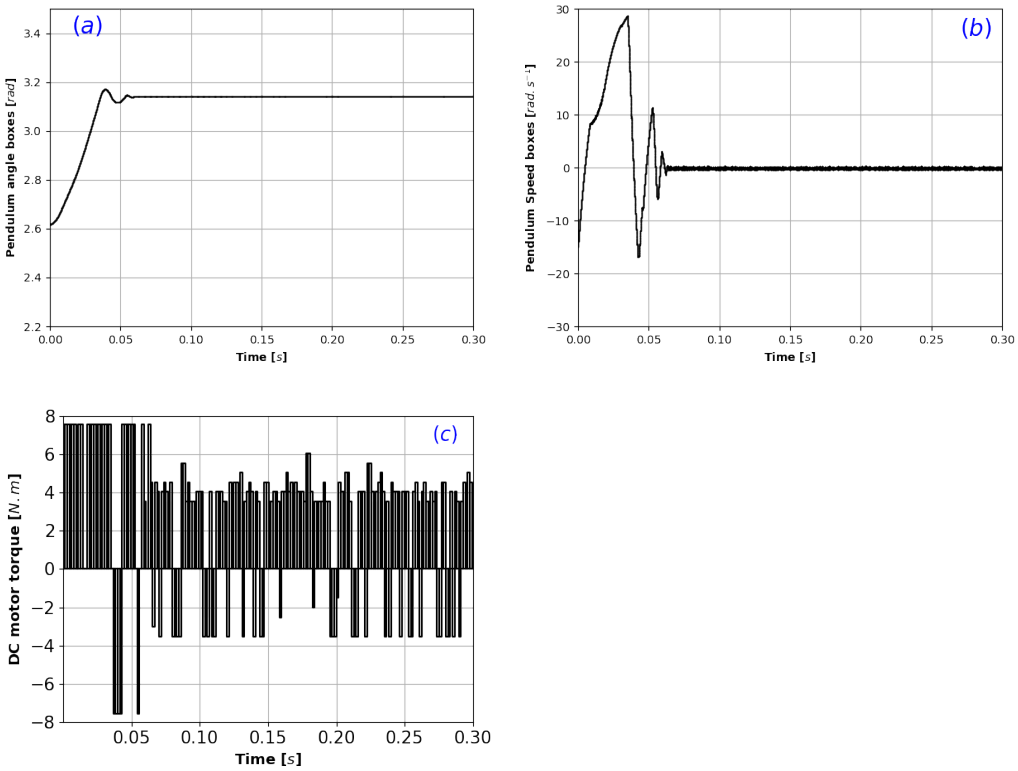

Figure 5. Validated NMPC results starting from $[\mathbf{y}_{3_{ini}}] = [149°; 150°]$: (**a**) pendulum angle tubes; (**b**) the corresponding pendulum speed tubes; (**c**) safe and optimal input Intervals $[\mathbf{u}_1]$.

4.2.2. Experimental Validation Using the Inverted Pendulum

To further show the usefulness of the proposed controller, it is tested on the experimental platform (Figure 1). As mentioned before, the real-time implementation of the validated strategy cannot be accomplished due to the high computation time of interval methods. In addition, we are currently working to improve this point by adopting parallel computation and relaxation methods. Consequently, we inject the computed input components into the control part of the pendulum that must somehow behave as the simulated model. Indeed, the dynamic model is proven to be as close as possible to the system's actual behavior (see Section 4.1).

Figure 6 shows the measured pendulum angle for the tested scenarios, starting from 0° and 150°. The stabilization is performed in an open-loop by incorporating the simulated input data into the DC motor controller (the computed inputs are drawn in Figures 4c and 5c). Compared to the simulation results, the calculated control inputs stabilize the inverted pendulum in the steady-state regime. The second scenario gives coherent results enabling the pendulum to be stabilized around the target position π thanks to the validated NMPC (see Figure 6 (Right)). However, for the first scenario (starting from 0°), we have some synchronization problems between the DC motor sampling period (around 0.028 s) and the sampling horizon of the NMPC method T_c. This is due to the lack of good deeds that the system's actuator executes at the right moment and its sensibility to weak signals

that destabilize it. Despite these issues, the proposed NMPC enables the system to reach another equilibrium position around $\pi/2$ (see Figure 6 (Left)), so we will try to handle this issue separately.

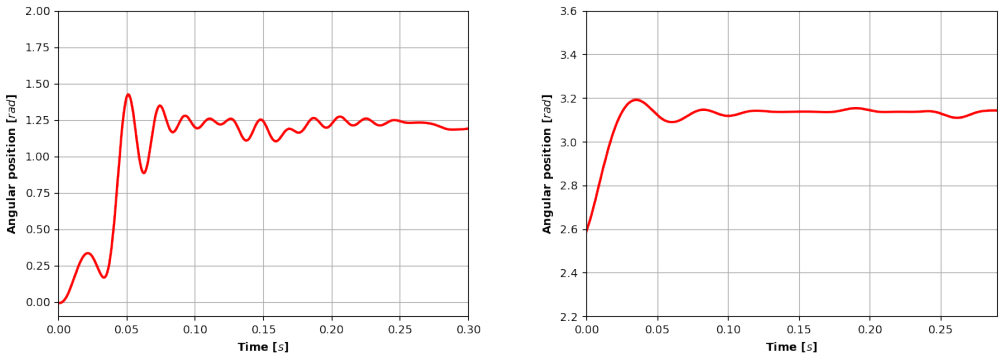

Figure 6. Experimental validation results starting from 0° (**Left**) and from 150° (**Right**).

To sum up, the validated NMPC seems more promising for enabling the system to successfully converge to the reference target with high accuracy and small settling time. The imposed state and control bounds are respected, critical, and necessary for the pendulum's stability and safety. Nevertheless, the computation time of the domain $[\mathbf{u}_k]$ is remarkable due to a large number of bisections of the initial input domain and the high computation time required by validated simulations.

5. Conclusions and Future Works

The research presented in this paper focuses on developing a reliable and validated nonlinear model predictive control (NMPC), which is more specifically designed using an uncertain mathematical model. The proposed controller is based on validated simulation mixed with interval analysis techniques to handle the system's nonlinearities and uncertainties. Two different procedures are employed: branching procedure to calculate the valid input intervals from an uncertain nonlinear model, allowing the convergence to the set-point interval and the system constraints' insurance. From the generated branches, the optimization procedure is applied to determine the sub-optimal control variable. The efficiency and robustness of the proposed method are validated through numerical and experimental tests using a new nonlinear inverted pendulum.

As regards ongoing works, we will focus on studying two issues: firstly, the computational complexity improvement of the validated simulation methods by taking into consideration contraction methods and parallel computing strategies; secondly, the projection of the validated NMPC on a complex mechatronic system (ROV—Remotely Operated underwater Vehicle). It allows us to perform some tasks (trajectory tracking) in the aquatic environment while respecting all the system's constraints. In addition, another relevant aspect enabling completing a system's autonomy is online estimation of the model's parameters via nonlinear observers and interval methods with respect to the environmental conditions, e.g., aerodynamic coefficients. This issue allows for reaching high accuracy and stability of the trajectory tracking task.

Author Contributions: Conceptualization, M.F.; software, J.A.d.S.; validation, M.F.; writing—review and editing, M.F. and J.A.d.S.; supervision, J.A.d.S. Both authors have read and agreed to the published version of the manuscript.

Funding: This research work is funded by the French LABEX Digicosme project "PREGARI" and DGA "CRa 1835 -DGA 2018-2021-Projet F".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In this appendix, we remind readers about some mathematical notions related to IA, used particularly to solve linear and/or nonlinear problems in a guaranteed way. For further details on this subject, we encourage the readers to refer to [4,7].

Appendix A.1. Interval Analysis: The Main Notations and Definitions

- A scalar interval of $\mathbb{R}$ is denoted as $[\mathbf{x}] = [\underline{x}; \overline{x}]$ where $\underline{x}$ is the lower bound and $\overline{x}$ is the upper bound. Any interval of $\mathbb{R}$ is finite, closed, and connected subsets.
- An interval vector, or a box $[\mathbf{X}]$ of $\mathbb{R}^n$ is a Cartesian product of n intervals, i.e., $[\mathbf{X}] = [\mathbf{x}_1] \times \ldots \times [\mathbf{x}_n]$. The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$.
- The width $w([\mathbf{X}])$ of a box $[\mathbf{X}]$ is the length of its largest side.
- The interval function $[f]$ from $\mathbb{IR}^n$ to $\mathbb{IR}^m$, is an inclusion function of f if $\forall [\mathbf{X}] \in \mathbb{IR}^n, \quad [f]([\mathbf{X}]) \supseteq \{f(x), x \in [\mathbf{X}]\}$.
- A subpaving of $\mathbb{R}^n$ is a list of non-overlapping boxes of $\mathbb{R}^n$, enabling for bracketing any compact set between a list of inner and outer subpavings.

Appendix A.2. Set Inversion and SIVIA Algorithm for Parameters Estimation

If $f : \mathbb{R}^n \to \mathbb{R}^m$ and $Y \subset \mathbb{R}^m$. A set inversion problem is described by the reciprocal image of a set Y by the function f that can be written as

$$\mathbb{P} = \{\mathbf{p} \in \mathbb{R}^n \mid f(\mathbf{p}) \in Y\} = f^{-1}(Y) \tag{A1}$$

The set $\mathbb{P}$ should be computed in a guaranteed way relying on the IA tools.

For this purpose, the SIVIA algorithm is an effective set-inversion problem solver via IA [7]. It allows us to get the set $\mathbb{P}$ in an effective and a guaranteed way, through successive and recursive bisections of the prior interest domain. To obtain the feasible set $\mathbb{P}$, three kinds of union boxes can be distinguished:

1. The feasible subpavings that belong to the set $\mathbb{P}$ and satisfy this implication test $[f]([\mathbf{p}]) \subset Y \quad \Rightarrow \quad [\mathbf{p}] \subset \mathbb{P}$ (denoted $[\mathbf{p_i}]$ in Figure A1);
2. The infeasible subpavings that make the empty intersection with $\mathbb{P}$, i.e., $[f]([\mathbf{p}]) \cap Y = \emptyset \Rightarrow [\mathbf{p}] \cap \mathbb{P} = \emptyset$ (denoted $[\mathbf{p_o}]$ in Figure A1);
3. The undetermined subpavings (or penumbra) for which nothing can be decided and will be bisected, except if their width is too small (denoted $[\mathbf{p_u}]$ in Figure A1).

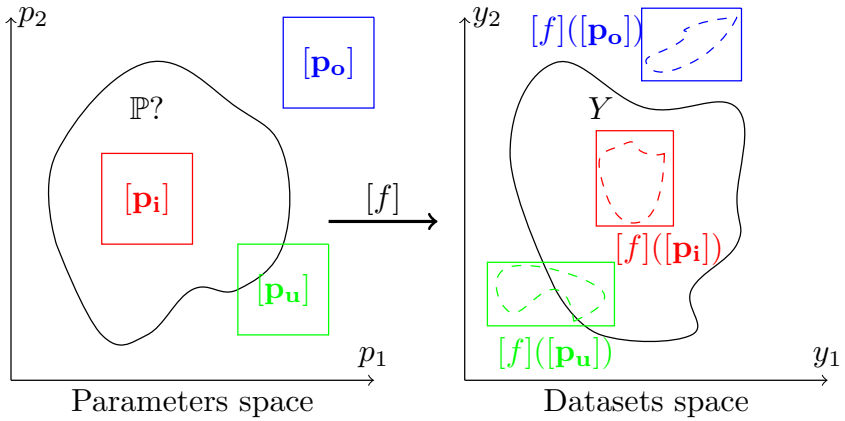

Figure A1. Illustration of the principal of the set inversion problem—Inclusion tests: infeasible (blue), undetermined (green), feasible (red).

References

1. Siljak, D. Reliable control using multiple control systems. *Int. J. Control* **1980**, *31*, 303–329. [CrossRef]
2. Veillette, R.J.; Medanic, J.V.; Perkins, W.R. Design of reliable control systems. In Proceedings of the 29th IEEE Conference on Decision and Control, Honolulu, HI, USA, 5–7 December 1990; pp. 1131–1136.
3. Yang, G.H.; Wang, J.L.; Soh, Y.C. Reliable H∞ controller design for linear systems. *Automatica* **2001**, *37*, 717–725. [CrossRef]
4. Jaulin, L.; Kieffer, M.; Didrit, O.; Walter, E. Interval analysis. In *Applied Interval Analysis*; Springer: London, UK, 2001; pp. 11–43.
5. Limon, D.; Bravo, J.M.; Alamo, T.; Camacho, E.F. Robust MPC of constrained nonlinear systems based on interval arithmetic. *IEE Proc. Control Theory Appl.* **2005**, *152*, 325–332. [CrossRef]
6. Lhommeau, M.; Hardouin, L.; Cottenceau, B.; Jaulin, L. Interval analysis and dioid: Application to robust controller design for timed event graphs. *Automatica* **2004**, *40*, 1923–1930. [CrossRef]
7. Jaulin, L.; Walter, E. Set inversion via interval analysis for nonlinear bounded-error estimation. *Automatica* **1993**, *29*, 1053–1064. [CrossRef]
8. Qin, S.J.; Badgwell, T.A. An overview of nonlinear model predictive control applications. *Nonlinear Model Predict. Control* **2000**, *26*, 369–392.
9. Fnadi, M.; Du, W.; Plumet, F.; Benamar, F. Constrained Model Predictive Control for dynamic path tracking of a bi-steerable rover on slippery grounds. *Control. Eng. Pract.* **2021**, *107*, 104693. [CrossRef]
10. Richter, S.; Jones, C.N.; Morari, M. Real-time input-constrained MPC using fast gradient methods. In Proceedings of the 48th IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference, Shanghai, China, 15–18 December 2009; pp. 7387–7393.
11. Limon, D.; Alamo, T.; Salas, F.; Camacho, E.F. On the stability of constrained MPC without terminal constraint. *IEEE Trans. Autom. Control* **2006**, *51*, 832–836. [CrossRef]
12. Allgower, F.; Findeisen, R.; Nagy, Z.K. Nonlinear model predictive control: From theory to application. *J. Chin. Inst. Chem. Eng.* **2004**, *35*, 299–316.
13. Falcone, P.; Borrelli, F.; Asgari, J.; Tseng, H.E.; Hrovat, D. Predictive active steering control for autonomous vehicle systems. *IEEE Trans. Control Syst. Technol.* **2007**, *15*, 566–580. [CrossRef]
14. Lydoire, F.; Poignet, P. Nonlinear model predictive control via interval analysis. In Proceedings of the 44th IEEE Conference on Decision and Control, Seville, Spain, 15 December 2005; pp. 3771–3776.
15. Kubica, B.J. Preliminary experiments with an interval model-predictive-control solver. In *Parallel Processing and Applied Mathematics*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 464–473.
16. Rauh, A.; Senkel, L.; Kersten, J.; Aschemann, H. Reliable control of high-temperature fuel cell systems using interval-based sliding mode techniques. *IMA J. Math. Control. Inf.* **2016**, *33*, 457–484. [CrossRef]
17. Senkel, L.; Rauh, A.; Aschemann, H. Interval-based sliding mode observer design for nonlinear systems with bounded measurement and parameter uncertainty. In Proceedings of the 2013 18th International Conference on Methods & Models in Automation & Robotics (MMAR), Miedzyzdroje, Poland, 26–29 August 2013; pp. 818–823.
18. Rauh, A.; Aschemann, H. Interval-based sliding mode control and state estimation for uncertain systems. In Proceedings of the 2012 17th International Conference on Methods & Models in Automation & Robotics (MMAR), Miedzyzdroje, Poland, 27–30 August 2012; pp. 595–600.
19. Alexandre Dit Sandretto, J. Reliable NonLinear Model-Predictive Control via Validated Simulation. In Proceedings of the American Control Conference (ACC), Milwaukee, WI, USA, 27–29 June 2018; pp. 609–614.
20. Fnadi, M.; Sandretto, J.A.d.; Ballet, G.; Fribourg, L. Guaranteed Identification of Viscous Friction for a Nonlinear Inverted Pendulum Through Interval Analysis and Set Inversion. In Proceedings of the 2021 American Control Conference (ACC), New Orleans, LA, USA, 25–28 May 2021.
21. Sandretto, J.A.D.; Chapoutot, A. Validated Explicit and Implicit Runge–Kutta Methods. 2016. Available online: https://perso.ensta-paris.fr/~chapoutot/dynibex/ (accessed on 18 August 2021).
22. Gafvert, M. *Modelling the Furuta Pendulum*; Department of Automatic Control, Lund Institute of Technology (LTH): Lausanne, Switzerland, 2006.
23. Nedialko, S.; Nedialkov, K.; Corliss, G. Validated solutions of initial value problems for ordinary differential equations. *Appl. Math. Comput. J.* **1999**, *105*, 21–68. [CrossRef]
24. Sandretto, J.A.D.; Trombettoni, G.; Daney, D.; Chabert, G. Certified calibration of a cable-driven robot using interval contractor programming. In *Computational Kinematics*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 209–217.
25. van Hentenryck, P.; Michel, L.; Deville, Y. *Numerica: A Modeling Language for Global Optimization*; MIT Press: Cambridge, MA, USA, 1997.
26. Granvilliers, L.; Benhamou, F. Algorithm 852: Realpaver: An interval solver using constraint satisfaction techniques. *ACM Trans. Math. Softw. (TOMS)* **2006**, *32*, 138–156. [CrossRef]