actuators

MDPI

Article

Tracking Control of Unforced and Forced Equilibrium Positions of the Pendubot System: A Nonlinear MHE and MPC Approach

Martin Gulan [1,*], Michal Salaj [2] and Boris Rohaľ-Ilkiv [1]

[1] Institute of Automation, Informatization, and Measurement, Faculty of Mechanical Engineering, Slovak University of Technology in Bratislava, 812 31 Bratislava, Slovakia; boris.rohal-ilkiv@stuba.sk
[2] Danfoss Power Solutions, Havnbjerg, Lundenvej v Grønvej, 6430 Nordborg, Denmark; michal.salaj@danfoss.com
* Correspondence: martin.gulan@stuba.sk

Abstract: This paper presents a unified control scheme of the Pendubot based on nonlinear model predictive control (NMPC) and nonlinear moving horizon estimation (NMHE) with the objective of point-to-point tracking its unstable unforced and ultimately forced equilibrium positions. In order to implement it on this fast, underactuated mechatronic system, we employ the Gauss–Newton real-time iteration scheme tailored to obtain the efficient solution of the underlying nonlinear optimization problems via sequential quadratic programming. The control performance is experimentally assessed on a real-world laboratory setup featuring an execution timing analysis and hints how to further improve the computational efficiency of the proposed nonlinear estimation control scheme. Even nowadays, the number of practical NMPC applications in the millisecond range is still rather limited, and the presented NMHE-based NMPC of the Pendubot thus also represents a unique case study for control practitioners.

Keywords: Pendubot; predictive control; state estimation; underactuated system; nonlinear optimization

Citation: Gulan, M.; Salaj, M.; Rohaľ-Ilkiv, B. Tracking Control of Unforced and Forced Equilibrium Positions of the Pendubot System: A Nonlinear MHE and MPC Approach. *Actuators* **2023**, *12*, 343. https://doi.org/10.3390/act12090343

Academic Editors: He Chen, Yinan Wu and Yougang Sun

Received: 27 June 2023
Revised: 14 August 2023
Accepted: 21 August 2023
Published: 26 August 2023

1. Introduction

During recent decades, a variety of scientific, industrial, and military applications have motivated the analysis and the rigorous derivation of control algorithms for mechanical systems. Naturally, this research field has also attracted the attention of mathematicians since the majority of the systems possess a global nonlinear characteristic, and their linear approximation seems to be deficient. Putting their efforts together, both researchers and industry practitioners have developed several control design methodologies that include linear control; optimal control; adaptive control; nonlinear control; and, more recently, robust control in order to account for uncertainties in a practical context. In fact, interest in mechanical systems especially grew when researchers realized that they can be underactuated [1].

The design of their functionality, in the case of both fully actuated and underactuated systems, is similar in concept. The usual problem is planning the motion for the task to be performed and designing the control system to execute it. However, unlike in the case of fully actuated robots, where there exist well-established results to solve both tasks, the trajectory planning and control of underactuated robots is theoretically more complex and less general. Despite this difficulty, the design of such mechanisms that can perform complex tasks with less actuators allows the reduction of the cost and weight by using fewer actuators and allows the optimization of energy consumption and increase in maneuverability as well as tolerance to actuator failure. However, their control is challenging due to the nonlinear dynamics, nonholonomic behavior, and lack of linearizability exhibited by these systems, and therefore, they are also of great importance in both control theory and applications. The convenient use of such systems in both academia and industry, supported

by intensive research activities aiming at the exploitation of the underactuation properties, formed the main motivation behind this study.

In this article, our focus is to employ advanced model-based control designs for nonlinear systems that arise from the control of an important and broad class of mechanical systems known as underactuated systems. A mechanical system is said to be underactuated when the number of control inputs is less than the number of degrees of freedom to be controlled. The underactuation property of underactuated mechanical systems (UMSs) can be due to one of the following reasons [2]:

- It can be naturally due to the dynamics of the system, such as those of aircrafts, spacecrafts, helicopters, or underwater vehicles;
- It can be imposed by design, aiming at cost or weight reduction, particularly in the case when actuators are expensive and/or heavy and therefore sometimes avoided in a system design, e.g., satellites with two thrusters or flexible-link robots;
- It can be due to actuator failure; for example, if a fully actuated system becomes faulty in a critical situation, the underactuation property can be properly exploited to increase its reliability or even to avoid the complete failure of the system instead of the uneconomical addition of redundant actuators;
- It can be imposed artificially to create complex low-order nonlinear systems in order to gain insight into the control of high-order UMSs.

This class of systems has varied and rich applications, at both practical and theoretical levels, in various fields such as robotics, aeronautical and spatial systems, marine and underwater systems, and flexible and mobile systems. In contrast to systems that have direct practical applications, pendulum-type systems have applications more in terms of academic benchmarks for nonlinear control where classical procedures cannot be applied. Systems like the inverted pendulum (cart–pole system), the rotational inverted (Furuta) pendulum, the Pendubot, or the ball and beam tend to be part of a standard control laboratory. Other examples of UMSs include the TORA (translational oscillator with a rotational actuator) system, the VTOL (vertical take off and landing) aircraft, a convey-crane system, and hovercraft or helicopter models. Despite their relatively simple mechanical structure, they represent a challenge to the nonlinear control community. This has given rise to a number of studies devoted to control design and stability analyses for particular classes of nonlinear UMSs, such as the essential overview in [3]; the relevant books [1,4]; and several recent distinct publications, e.g., [5–7].

The Pendubot, introduced in [8], is a well-known academic benchmark from the class of underactuated robotic systems. During the past few decades, these systems have been studied in many application areas and utilized to demonstrate various concepts in linear and nonlinear control [4]. They are typically characterized by strongly nonlinear and fast dynamics and dynamic couplings between the unactuated and actuated state variables; therefore, they may clearly benefit from a complex control strategy capable of addressing these phenomena, such as the optimization-based model predictive control (MPC).

The typical control objectives of the Pendubot and pendulum-like underactuated systems are to swing up and stabilize the unstable unforced equilibrium positions, as well as possible reference tracking. We refer the interested reader to [9] for an overview of the various control techniques used to approach them. The usual approach is to use a strategy of switching between the swing-up and balancing controller; see, e.g., our earlier work [10] using energy-based control and a linear MPC or the more sophisticated robust switching strategies of [11,12]. As first shown in [13], it is, however, much more challenging to propose a unified control strategy that would continuously stabilize the Pendubot in one of its unforced equilibrium configurations. This problem was addressed in our early study [14] by using nonlinear MPC (NMPC) to exploit the full nonlinear dynamic model of the system, and more recently in [15] by using NMPC combined with reinforcement learning and in [16] by using a motion control approach based on the integrated trajectory.

Nevertheless, the ultimate challenge is to perform rest-to-rest maneuvers of the Pendubot considering not only its three unstable unforced equilibria, but also its forced equi-

librium configurations, which is not common in traditional control approaches of the linear and nonlinear class. This problem was addressed in [17], which is so far one of very few studies providing experimental results. The authors, however, focused on a specific swing-up strategy to a desired forced equilibrium setpoint—assuming the control to be taken over by a stabilizing linear-quadratic (LQ) controller once the system enters its region of attraction. Only very recently, the authors of [18] proposed an observer-based active disturbance rejection control to perform rest-to-rest maneuvers taking the system far from the unforced equilibrium position, and also verified it experimentally in a laboratory setting.

In this paper, we propose to accomplish this task by nonlinear model predictive control. NMPC is currently one of the most powerful tools that can be used to address nonlinear control problems, which in fact arise in nearly all engineering applications, often due to nonlinear dynamics. Many predictive control applications are, however, still based on the use of linear or linearized models, which facilitates the identification and optimization tasks and leads to a reasonably good behavior in the neighborhood of the respective operating point. After decades of research, linear MPC is nowadays considered to be a mature technique for linear but rather slow systems, such as those usually encountered in the process industry. However, more complex systems such as nonlinear or very fast processes still call for research in the field of nonlinear MPC. Solving a nonlinear optimal control problem usually comes with a rapidly increased computational complexity, which, due to the enormous progress achieved recently in nonlinear optimization, is now possible to be addressed even in the case of processes with very short sampling times. Nevertheless, the number of NMPC applications is still limited, in particular for systems with fast dynamics, which makes this topic very actual and relevant.

To implement NMPC under hard real-time constraints, we use the real-time iteration (RTI) scheme of [19] implemented in the ACADO toolkit [20], namely its Code Generation tool [21]. The efficacy of the RTI scheme stems primarily from performing only one Newton-type sequential quadratic programming (SQP) iteration per sampling instant. The NMHE-based NMPC scheme exploits the nonlinear dynamics and hence allows the perforance of the above tasks within a unified control strategy. We remark that a similar NMHE–NMPC approach has been used lately in [22] for trajectory tracking of a quadrocopter. In order to demonstrate the performance of the proposed estimator-control scheme, we revisit our earlier study [23] to account for the above control objective and present experimental results with a detailed timing analysis and ways to make the implementation more efficient by employing different solvers and parallelization of particular algorithmic steps. The control scheme is moreover augmented by a nonlinear estimation scheme, in particular nonlinear moving horizon estimation (NMHE), to obtain the unmeasured states, a nonlinear friction model, and a parallelization of particular algorithmic routines to speed up the online execution.

2. System Model

The Pendubot, schematically illustrated in Figure 1, is a two-link planar robot belonging to the class of underactuated systems, as it has less control inputs than degrees of freedom. The actuator is located in the joint of the first (active) link while the unactuated joint between the two links allows the free movement of the second (passive) link. Its mathematical model can be derived by means of the Lagrange formalism:

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial}{\partial \dot{q}_k}\mathcal{L}(q,\dot{q}) - \frac{\partial}{\partial q_k}\mathcal{L}(q,\dot{q}) = \tau \quad k = 1,2, \tag{1}$$

with the Lagrange function $\mathcal{L}(q,\dot{q}) = \mathcal{T}(q,\dot{q}) - \mathcal{V}(q)$ defined as the difference between kinetic energy and potential energy. The generalized coordinates in $q = [q_1, q_2]^\mathsf{T}$ stand for angular positions of the two links, and $\tau = [\tau_1, 0]^\mathsf{T}$ denotes the external control force vector. By applying (1), the resulting equation of motion can be obtained in the following form:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau, \tag{2}$$

where $D(q)$ is the symmetric positive definite inertia matrix, $C(q,\dot{q})$ contains the Coriolis and centrifugal terms, and $g(q)$ denotes the vector of gravitational terms. For the Pendubot in Figure 1, the following quantities are obtained:

$$D(q) = \begin{bmatrix} \theta_1 + \theta_2 + 2\theta_3 \cos q_2 & \theta_2 + \theta_3 \cos\ q_2 \\ \theta_2 + \theta_3 \cos q_2 & \theta_2 \end{bmatrix},$$
$$C(q,\dot{q}) = \begin{bmatrix} -\theta_3 \sin(q_2)\dot{q}_2 & -\theta_3 \sin(q_2)\dot{q}_2 - \theta_3 \sin(q_2)\dot{q}_1 \\ \theta_3 \sin(q_2)\dot{q}_1 & 0 \end{bmatrix},$$
$$g(q) = \begin{bmatrix} \theta_4 g \cos q_1 + \theta_5 g \cos(q_1 + q_2) \\ \theta_5 g \cos(q_1 + q_2) \end{bmatrix},$$

with parameters $\theta_1 = m_1 l_{c1}^2 + m_2 l_1^2 + I_1$, $\theta_2 = m_2 l_{c2}^2 + I_2$, $\theta_3 = m_2 l_1 l_{c2}$, $\theta_4 = m_1 l_{c1} + m_2 l_1$, and $\theta_5 = m_2 l_{c2}$.

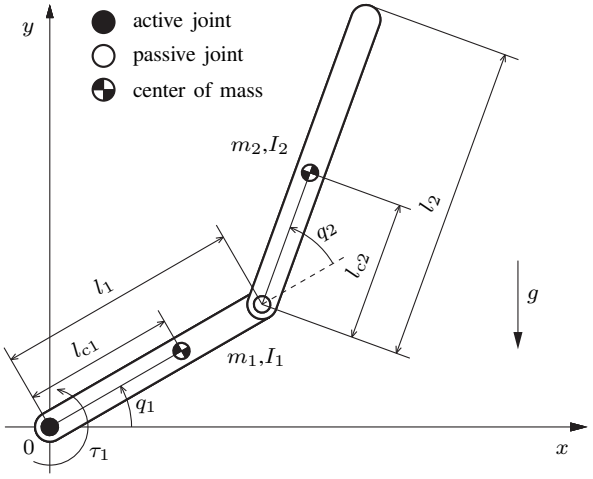

Figure 1. Schematic diagram of the Pendubot.

Although the friction in the passive joint of a laboratory Pendubot system is typically negligible, as observed in the experiments presented later in Section 4.2, there is a more significant friction present in the actuating element. Considering friction only in the active joint, the friction vector is given by $f_r(\dot{q}) = [f_r(\dot{q}_1),\ 0]^T$. Based on extensive experimental testing, we used the following nonlinear dynamic friction model proposed in [24]:

$$f_r(\dot{q}_1) = \gamma_1(\tanh \gamma_2 \dot{q}_1 - \tanh \gamma_3 \dot{q}_1) + \gamma_4 \tanh \gamma_5 \dot{q}_1 + \gamma_6 \dot{q}_1, \tag{3}$$

where $\gamma_i \in \mathbb{R}$, $i = 1, \ldots, 6$ denote positive constant coefficients. This model is continuously differentiable and symmetric about the origin. The static friction coefficient is approximated by the term $\gamma_1 + \gamma_2$, whereas the viscous friction is modelled by the term $\gamma_6 \dot{q}_1$. The friction model also includes the Coulomb friction via the term $\gamma_4 \tanh \gamma_5 \dot{q}_1$ and the Stribeck effect via the term $\tanh \gamma_2 \dot{q}_1 - \tanh \gamma_3 \dot{q}_1$.

When plugging the friction model (3) into (2), we obtain

$$\ddot{q}_1 = \frac{1}{\theta_1\theta_2 - \theta_3^2 \cos^2 q_2} \Big[\theta_2\theta_3 \sin q_2 (\dot{q}_1 + \dot{q}_2)^2 + \theta_3^2 \cos q_2 \sin(q_2) \dot{q}_1^2 - \theta_2\theta_4 g \cos q_1 + \theta_3\theta_5 g \cos q_2 \cos(q_1 + q_2) + \theta_2 \tau_1 - \theta_2 f_r(\dot{q}_1) \Big], \tag{4a}$$

$$\ddot{q}_2 = \frac{1}{\theta_1\theta_2 - \theta_3^2 \cos^2 q_2} \Big[-\theta_3(\theta_2 + \theta_3 \cos q_2) \sin q_2 (\dot{q}_1 + \dot{q}_2)^2 - (\theta_1 + \theta_3 \cos q_2)\theta_3 \sin(q_2)\dot{q}_1^2 + (\theta_2 + \theta_3 \cos q_2)(\theta_4 g \cos q_1 - \tau_1) - (\theta_1 + \theta_3 \cos q_2)\theta_5 g \cos(q_1 + q_2) + (\theta_2 + \theta_3 \cos q_2) f_r(\dot{q}_1) \Big], \tag{4b}$$

representing Pendubot's two nonlinear equations of motion which are used as the starting point for the design of a nonlinear predictive controller in the following sections. One may also notice the dynamic coupling between the actuated and unactuated variables.

The Pendubot system is said to have an equilibrium configuration when a control input and state variables satisfy particular conditions for which the Pendubot is at rest, i.e., $\dot{q} = 0$. Solving the nominal system dynamics in (2) under this condition, the equilibrium points are given as

$$\begin{aligned} \theta_4 \cos q_1 + \theta_5 \cos(q_1 + q_2) &= \tau_1 \\ \theta_5 \cos(q_1 + q_2) &= 0. \end{aligned} \tag{5}$$

It can be shown that as long as input τ_1 is constrained to satisfy $\tau_1 \leqslant \theta_4$, the solution of (5) for the equilibrium states can be obtained as

$$q_1 = \arccos\left(\frac{\tau_1}{\theta_4}\right) \quad \text{and} \quad q_2 = k\frac{\pi}{2} - q_1, \quad k = 1, 3, 5, \ldots \tag{6}$$

For an arbitrary value of q_1, there are thus two equilibrium configurations; see Figure 2.

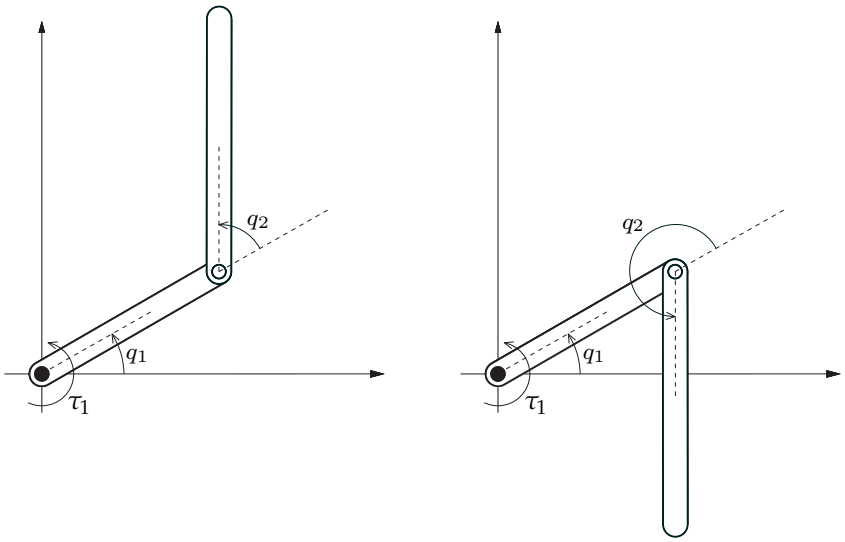

Figure 2. Illustration of Pendubot's equilibrium configurations for an arbitrary value of q_1 and corresponding input τ_1.

While (6) holds, there are infinitely many equilibrium configurations $(q_1, 0, q_2, 0)$, in which the Pendubot balances if a constant torque $\tau_1 \neq 0$ is applied. We refer to them as *forced* unstable equilibrium configurations. In addition, there are four well-known *unforced* equilibrium configurations which are obtained by solving (6) for $\tau_1 = 0$:

1. $(q_1, \dot{q}_1, q_2, \dot{q}_2) = (-\frac{\pi}{2}, 0, 0, 0)$: both actuated and unactuated links are in their low positions, further denoted as $\downarrow\downarrow$ or 'bottom' configuration.
2. $(q_1, \dot{q}_1, q_2, \dot{q}_2) = (-\frac{\pi}{2}, 0, \pi, 0)$: actuated link is in its low position, and unactuated link is in its high position, further denoted as $\downarrow\uparrow$ or 'mid' configuration.
3. $(q_1, \dot{q}_1, q_2, \dot{q}_2) = (\frac{\pi}{2}, 0, \pi, 0)$: actuated link is in its high position, and unactuated link is in its low position, further denoted as $\uparrow\downarrow$.
4. $(q_1, \dot{q}_1, q_2, \dot{q}_2) = (\frac{\pi}{2}, 0, 0, 0)$: both actuated and unactuated links are in their high positions, further denoted as $\uparrow\uparrow$ or 'top' configuration.

Note that the first one represents a stable and the remaining three unstable unforced equilibrium configurations of the Pendubot. The interest of researchers is usually aimed at steering the system to and/or balancing it in the mid or top unforced equilibrium position.

In this paper, we are, however, interested in controlling the Pendubot also to its forced equilibrium configurations and transitioning between them, representing an even more challenging task.

3. Control Design and Implementation

This section introduces the formulation and efficient implementation of the nonlinear MPC scheme, proposed in view of a unified Pendubot control strategy. Due to several practical reasons stated further, the NMPC controller is moreover advantageously augmented by a nonlinear moving horizon estimation (NMHE) scheme. This gives rise to a complementary NMHE-based NMPC framework, jointly exploiting nonlinear optimization techniques that were outlined in the introduction. For clarity of presentation, we start the formulation with a brief description of the estimation scheme.

In order to achieve this, let us introduce the state vector $x(t) = [x_1, x_2, x_3, x_4]^\mathsf{T} = [q_1, \dot{q}_1, q_2, \dot{q}_2]^\mathsf{T}$, lumping together the angular positions and velocities of both links, and the control input $u = \tau_1$. The equations of motion of System (4) may be then reformulated in the following state-space form:

$$\dot{x} = f(x, u), \tag{7a}$$

$$y = h(x), \tag{7b}$$

where (7a) and (7b) represent the system's nonlinear state and output equation, respectively.

3.1. NMHE Problem Formulation

In most practical control applications, the number of available measurements is typically smaller than the number of states. This is also the case for the Pendubot system, where the dedicated sensors provide information about the angular positions of both links, leaving the respective angular velocities to be estimated. Unlike most other nonlinear estimation approaches, the NMHE allows for the incorporation of constraints into its underlying least-squares (LSQ) dynamic optimization problem:

$$\min_{x(\cdot),u(\cdot)} \ \frac{1}{2} \int_{t_0-T_\mathrm{E}}^{t_0} (\|h(x) - \bar{y}\|^2_{Q_\mathrm{E}} + \|u - \bar{u}\|^2_{R_\mathrm{E}})\, \mathrm{d}t, \tag{8a}$$

$$\text{s.t.} \ \ \dot{x} = f(x, u), \tag{8b}$$

$$g(x, u) \leqslant 0. \tag{8c}$$

This nonlinear optimization problem is solved repeatedly at every time instant $t_k = kT_\mathrm{s}$ $(k = 0, 1, \dots)$, where T_s is the sampling time and T_E denotes the NMHE estimation horizon. The moving horizon objective (8a) weighs the deviation between the measurement model $h(\cdot)$ and the set of measurement data $\bar{y}$. The inclusion of control variables in the objective accounts for any noise collected during signal transfer, actuator inaccuracies and unmodeled dynamics. The appropriately chosen matrices $Q_\mathrm{E} \succeq 0$ and $R_\mathrm{E} \succ 0$ weigh the LSQ terms. Propagation of the system states is described by equality constraint (8b), whereas the upper and lower bounds for both states and input can be imposed via (8c). The output of the estimator is the initial state estimate $\hat{x}$, which is subsequently fed to the NMPC controller.

3.2. NMPC Problem Formulation

Now, our NMPC controller is based on repeatedly solving the following optimization problem in an effort to find an optimal control input function:

$$u^\star(\cdot) = \underset{u(\cdot),x(\cdot)}{\arg\min} \ \frac{1}{2}(\|x - x^\mathrm{ref}\|^2_{P_\mathrm{C}})_{t=t_0+T_\mathrm{C}} + \frac{1}{2} \int_{t_0}^{t_0+T_\mathrm{C}} (\|x - x^\mathrm{ref}\|^2_{Q_\mathrm{C}} + \|u - u^\mathrm{ref}\|^2_{R_\mathrm{C}})\, \mathrm{d}t, \tag{9a}$$

$$\text{s.t.} \ \ x(t_0) = \hat{x}(t_0), \tag{9b}$$

$$\dot{x} = f(x, u), \tag{9c}$$

$$g(x, u) \leqslant 0. \tag{9d}$$

Within the LSQ objective (9a), the deviation of the control inputs u and states x from their reference trajectories, $u_{\mathrm{R}}^{\mathrm{ref}}$ and x^{ref}, respectively, is penalized. The first term evaluates the final costs raised by the controlled variables at the given end time $t_{k+T_{\mathrm{C}}}$, with T_{C} denoting the NMPC control horizon. This so-called cost-to-go or terminal penalty function is included to guarantee stability. As usual in tracking MPC applications, the norms in the objective function are weighed with penalty matrices $Q_{\mathrm{C}}, P_{\mathrm{C}} \succeq 0$ and $R_{\mathrm{C}} \succ 0$. System dynamics is embedded into the problem by equality constraint (9c). To obtain the current state x_k, we employ the NMHE scheme implied by initial state constraint (9b). In this light, the NMPC problem (9) can be viewed as a parametric nonlinear optimal control problem. Similar to NMHE, state and input constraints are imposed by means of inequality (9d).

Apart from the aforementioned and clearly useful estimation properties, the nonlinear MHE and MPC problems are in this work treated together since they are almost identical in the approach and implementation, even though they solve two different yet complementary problems. Therefore, they are referred to as nearly dual problems of each other. By comparison, one may notice the apparent similarity between the two formulations. In the first place, the weighted deviation between reference and predicted system states in (9a) resembles the one between the measurements and the measurement function in the objective of (8). Moreover, the arrival cost approximation of the MHE has its theoretical counterpart in the terminal penalty term of the MPC, and both represent additional least-squares terms in the respective optimization problems. At the same time, they are two of the most powerful state-of-the-art tools that can be used to address nonlinear control and estimation problems. As evidenced by experimental results in Section 4, the state estimates provided by the nonlinear moving horizon estimator are robust and reliable, hence further improving the overall performance of the NMPC controller.

3.3. Real-Time Implementation Approach

Due to the unstable Pendubot dynamics, the NMPC problem (9) should be treated using a simultaneous transcription method, such as direct multiple shooting or direct collocation [25]. Since we are interested in a small-scale system, multiple shooting discretization was employed to transform the infinite dimensional optimal control problem (9) into a finite dimensional optimization problem. The discrete-time formulation on a uniform time grid $t_0, \ldots, t_N$ is thus obtained by numerical integration of the system dynamics over the time intervals $[t_k, t_{k+1}]$. The same time grid is used for discretizing the inequality constraints and the control input which assumes piecewise constant parametrization. The resulting discretized optimal control problem in fact renders a structured nonlinear programming (NLP) problem that can be efficiently solved with a sequential quadratic programming (SQP) method. As its objective consists of a least-squares tracking term, a Gauss–Newton approximation of the Hessian is utilized, which leads to linearization of the NLP yielding a quadratic programming (QP) subproblem. Once it is solved, the NLP variables are updated using the full Newton step.

The ultimate objective of the real-time (RTI) iteration scheme, originally developed in [19], is to reduce the feedback delay and hence to allow for short control periods. It is widely known that the computational effort needed to solve the NLP exactly may easily become intractable, in particular for mechatronic systems with fast-evolving dynamics. To overcome this issue, the RTI scheme performs only one SQP step with Gauss–Newton Hessian per sampling time. As a consequence, it produces locally suboptimal solutions. In order to solve a whole sequence of neighboring NLPs, the so-called shift initialization [25] is used to initialize the consecutive problems based on previous information. The efficacy is supported by employing the aforementioned simultaneous NLP parametrization, direct multiple shooting method with condensing, and a so-called initial value embedding which constraints the initial value in the NLP to coincide with the estimated state of the system. In this way, most of the computations can be performed before the current state estimate becomes available. The computational burden within each iteration can be thus divided into a computationally more demanding preparation phase and a shorter feedback phase

solving only a single QP problem. Even though the optimal control problem (9) is solved only approximately, these bounded approximations of the exact optimal feedback control are, in fact, iteratively refined during runtime and even contract towards the optimal feedback control, which moreover allows for a fast reaction to external disturbances [19].

It needs to be emphasized that the efficiency of the RTI scheme is largely due to its algorithmic division into the two consecutive phases. Let us therefore briefly mention their key ingredients for the case of NMPC RTI. The preparation phase starts with an optional shifting of the old solution, and proceeds with the solution of an initial value problem within the direct multiple shooting procedure, after which model sensitivities are generated. Next, the objective is evaluated, followed by optional QP condensing. Lastly, a new state feedback (estimate) is awaited. The considerably shorter feedback phase uses the initial value embedding for solving the sparse/condensed QP. Once solved, the first element of the optimal control sequence is immediately sent to the system. The optimized state trajectory may also be optionally recovered from the condensed QP.

The RTI scheme can also be adapted for solving the NMHE problems, building on the same ideas as outlined above. The essential difference lies indeed in the embedding of the current measurement into the underlying QP, which returns the current state estimate immediately sent to the controller.

In summary, in each control period, a single iteration of the RTI scheme for both the nonlinear MPC and MHE is performed. Within each of them, the respective NLP is linearized with direct multiple shooting and Gauss–Newton approximation, resulting in a block structured QP. This is solved either in its sparse form with a structure-exploiting QP solver, or in a condensed form using a dense QP solver. The main and most demanding numerical procedures involve the evaluation of nonlinear functions (objective function, model, constraints) and algorithmic differentiation, numerical integration, and the solution of the underlying QPs. Note that particularly the milli- and micro-second applications are highly dependent on the choice of the most efficient methods for the aforementioned tasks.

4. Experimental Results and Discussion

This section demonstrates the main contribution of this article—the application of nonlinear MPC and MHE algorithms to an underactuated Pendubot system. To this end, we use the efficient solution approach, implementing the real-time iteration scheme for both the controller and the estimator to control a real-world Pendubot system.

4.1. Experimental Setup

The experimental testing was performed using a laboratory Pendubot setup designed at the authors' workplace, depicted in Figure 3.

As the driving element allowing for rotation of the active Pendubot link, an AC motor with a Mitsubishi HC-KFS43 servo drive is used. It is actuated by a Mitsubishi MR-J2S-40A control unit, capable of operation in three basic modes—positioning, speed, and torque mode, where the latter is effectively used for the Pendubot control. The control unit also enables to obtain the information about the angular displacement of the motor shaft by means of an emulated encoder with adjustable resolution, while the velocities of both links are estimated. The output shaft of the motor also houses a Wittenstein alpha CP060 planetary gearbox used for reduction in revolutions and increase in torque. As stated above, the angular position of the actuated link is measured via the motor shaft position, which is obtained by means of the drive's control unit. However, for the sake of the Pendubot model, it is also necessary to read the angular position of the unactuated link. To this end, an incremental rotary encoder OMRON E6B2-C is utilized. The input–output interface is carried out by means of a Humusoft MF624 data acquisition board.

The proposed NMHE–NMPC scheme is implemented in the form of a custom C code executed on the embedded industrial computer with CPU clocked at 1.6 GHz, with 12 GB of RAM, running Ubuntu with a real-time kernel as the operating system.

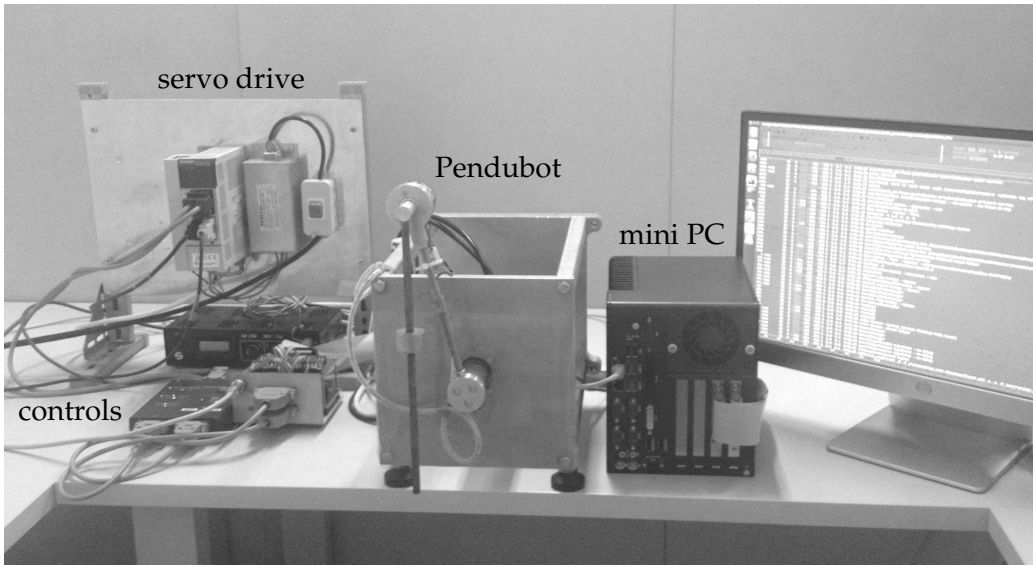

Figure 3. Photograph of the experimental Pendubot setup employing designed by the authors.

In order to numerically treat the two nonlinear optimization problems, we employ the ACADO Code Generation tool [21] exploiting direct multiple shooting, the aforementioned real-time iteration scheme and sequential quadratic programming. To solve the underlying QP subproblems of the SQP-based RTI scheme, we use efficient quadratic programming solvers, specifically the condensing-based qpOASES parametric solver implementing the online active set strategy proposed in [26], as well as the qpDUNES solver implementing a dual Newton strategy [27], which combines the warm-starting capabilities of active-set methods and the structure-exploiting features of interior-point methods. Both are used in real-time implementation to assess the timing. In addition to employing a more efficient QP strategy, the concept of parallelization is exploited as well. In particular, as it is demonstrated, in case a processor with more cores is available, certain algorithmic components of particular NMPC and NMHE RTI schemes may be executed in parallel.

By applying the multiple shooting technique, the system dynamics given by the continuous ODE model (4a) and (4b) are parameterized assuming uniform intervals of $T_\mathrm{s} = 10$ ms. For discretization over the shooting intervals, we use an implicit Gauss–Legendre Runge–Kutta integrator of the order of two. The estimation horizon is chosen as $T_\mathrm{E} = 0.5$ s, i.e., 50 steps, while the prediction horizon is chosen as $T_\mathrm{C} = 1$ s, i.e., 100 steps. At each sampling time, the respective nonlinear optimization problems are solved with the following bounds on states: $-2\,\mathrm{Nm} \leqslant u \leqslant 2\,\mathrm{Nm}$, $-3\pi\,\mathrm{rad} \leqslant x_1 \leqslant 0\pi\,\mathrm{rad}$. Within these box constraints, those imposed on the control input stem from internal limitations of the actuator, while constraints on the angular position of the actuated link are given by preventing the sensor cable from twisting onto the rotor shaft.

The sample time of 10 ms is identified as sufficient to achieve adequate Pendubot control performance. It is justified based on previous work carried out on the laboratory system. While a sampling interval as short as possible is preferable for any real-time control scheme, a lower bound on the sampling time is imposed by the computational complexity of solving the underlying optimization problems. Regarding horizon T_E, it is carefully selected such that there is no need for the arrival cost term in the NMHE cost function. Through laboratory experiments, we observe that the estimator performs just as well with a 50-sample-long window as with an arrival cost.

The proposed NMHE–NMPC scheme implicitly assumes both state and control references. For equilibria tracking, these are kept zero except of the references for angular positions of both links which are changed online, i.e., $[\cos x_1^\mathrm{ref}, \sin x_1^\mathrm{ref}, 0, \cos x_2^\mathrm{ref}, \sin x_2^\mathrm{ref}, 0]^\mathsf{T}$. This form of both state and associated reference vector is utilized within the NMPC objective to ease the reference angle tracking since each of the two links may exhibit in fact infinitely

many possible angular values (with a 2π period) corresponding to a particular equilibrium configuration, and $u^{\text{ref}} = 0$. If changing, references are adequately made available to the NMPC controller's buffer in order to fully exploit its prediction capabilities. The diagonals of respective state weighting matrices Q_{C} are therefore set appropriately so as to match the structure of the above state reference vector. The angular velocity weights are, moreover, always adequately scaled. The specific values of the weights are given separately for each control scenario in the following sections. Their units are consistent with the variables in order to yield a dimensionless cost, and are henceforth omitted for brevity.

The terminal weighting matrix P_{C} is set equal to the penalization of the state vector, while any feasibility issues are avoided by using a sufficiently long horizon T_{C}, rendering the weighting matrix obtained from the solution of the discrete-time Riccati equation unnecessary to reliably solve the problem. This choice is justified by practical experience and by other practical real-time-iteration-based NMPC works.

As it is experimentally demonstrated, the proposed NMPC strategy manages to successfully merge the techniques of swing-up, balancing, and tracking within a single unified and continuous approach. However, there is one tuning parameter that seems useful to be changed at runtime—the NMPC weights—as they directly affect the controller's behavior. Herein, this one-time change of controller weights is used merely at the time of reference change—namely at the transition between different set-points. We implement this strategy within the real-time control loop by means of a set of if–then rules switching the state/input weights according to the position of the Pendubot's links. For example, we use one set of weights while the system is "swinging up" (in one swing), and replace these with another set once the linkage is brought to the vicinity of a desired position, where the controller likely needs to perform differently due to the nature of the balancing phase. This allows us direct tuning of the controller's performance by adapting the penalization of particular system states within the NMPC objective (9a) according to the current configuration of Pendubot links and required control aggressiveness. This strategy has proven itself essential in achieving a smoother control performance and transitions between the desired setpoints. It may resemble the switching strategy between the swing-up and the balancing controllers, known from traditional Pendubot control approaches; however, apart from the switching nature, there is no relation between them.

4.2. Tracking of Unstable Unforced Equilibrium Positions

The following presents the experimental results obtained for three investigated control scenarios aimed at steering the system to its selected unstable unforced equilibrium configurations.

4.2.1. Assuming a Simple Viscous Friction Model

For a better demonstration of the practical control aspects, we first present our initial experimental results where we employed the standard viscous friction model that later led us to employing a more advanced, nonlinear friction model outlined in Section 2, as well as other improvements in terms of both control and computational performance. Recall that the viscous friction model alone essentially corresponds to the last term of (3). The value of its coefficient (with respect to (3) equivalent to γ_6) was identified as 0.08.

Starting from the initial, resting $\downarrow\downarrow$ stable position, the Pendubot was first commanded to the mid $\downarrow\uparrow$ unforced equilibrium. The relevant NMPC weights were for this purpose chosen as $\text{diag}(Q_{\text{C1}}) = [2 \cdot 10^2, 2 \cdot 10^2, 1, 10^2, 10^2, 1]$ and $R_{\text{C1}} = 1.5 \cdot 10^2$. Once the linkage was brought into the vicinity of the $\downarrow\uparrow$ setpoint, the controller's performance was adjusted by switching the weights to $\text{diag}(Q_{\text{C2}}) = [10^3, 10^3, 1, 10^2, 10^2, 1, 2 \cdot 10^2]$ and $R_{\text{C2}} = 2 \cdot 10^2$. Finally, the NMHE weights were tuned as $\text{diag}(Q_{\text{E}}) = [3, 3]$ and $R_{\text{E}} = 10^{-3}$, without any online changes necessary. The resulting behavior of the system is illustrated in Figure 4. The acquired angular position measurements are denoted by gray markers (+), together with their estimates depicted with the solid red lines. The remaining states, i.e., estimated angular velocities, and the corresponding control input acting on the first link are shown

as well. Notice that, during the 50 samples prior to the main part of experiment, only the estimator's buffer was filled while the NMHE–NMPC routine itself was triggered thereafter.

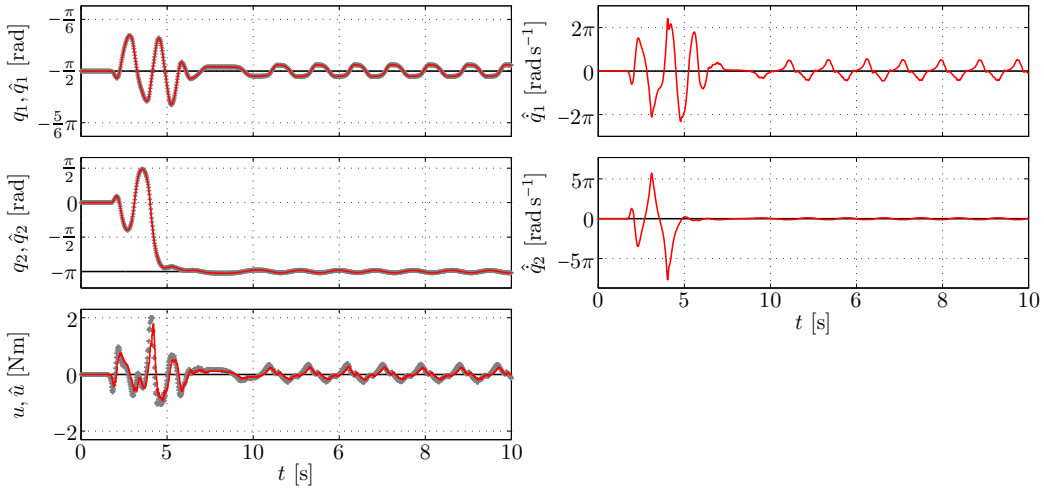

Figure 4. Experimental results for the transition from ↓↓ to ↓↑ unforced equilibrium configuration: red line—state and control input estimates, gray '+' markers—measurements, black line—references.

As can be seen in Figure 5, expectedly, the most challenging setpoint to track was the ↑↑ position when both Pendubot links simultaneously achieved their upright unstable equilibria. The controller weights were set as $\mathrm{diag}(Q_{\mathrm{C1}}) = [2 \cdot 10^2, 2 \cdot 10^2, 1, 2 \cdot 10^2, 2 \cdot 10^2, 1]$ and $R_{\mathrm{C1}} = 5$. When approaching the setpoint, the swing-up nature of the controller turned into a balancing one, with respective weights tuned as $\mathrm{diag}(Q_{\mathrm{C2}}) = [10^2, 10^2, 1, 10^2, 10^2, 1]$ and $R_{\mathrm{C2}} = 20$. We remark that the previously used NMHE weights were kept identical in this and all the control scenarios that follow, and thus are henceforth omitted for brevity.

Lastly, Figure 6 illustrates the performance of the NMPC–NMHE scheme in a point-to-point motion scenario, where the desired setpoints correspond to particular unforced equilibrium configurations of the system. Specifically, starting off from its slightly disturbed ↓↓ stable configuration, the Pendubot was shortly afterwards stepwise prompted to achieve the unstable equilibria in the order of ↓↑–↑↑. This unified swing-up–balancing–tracking task was finally concluded by performing the so-called swing-down maneuver, i.e., returning both links into the initial resting position in the shortest time possible. The sets of weights were adopted from the previous tasks, moreover augmented by similar weights valid for the swing-down maneuver.

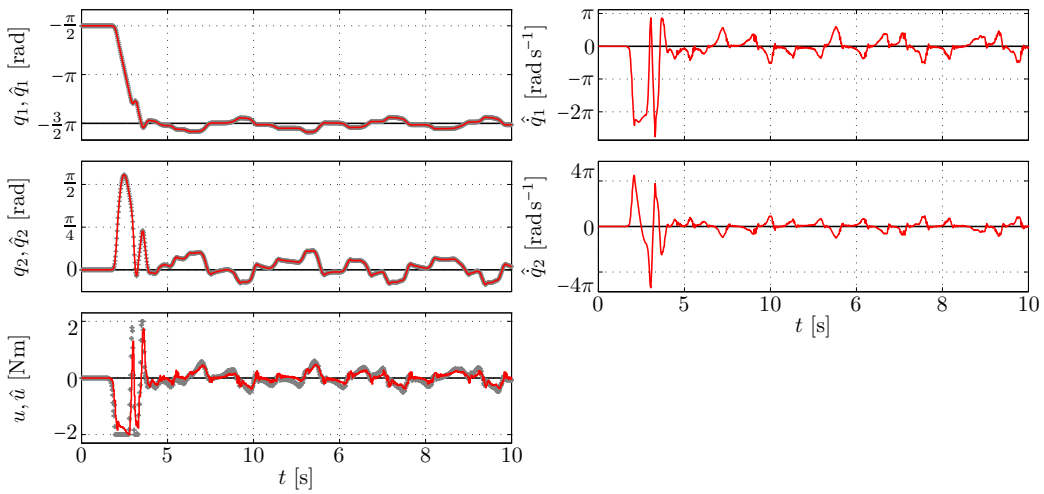

Figure 5. Experimental results for the transition from ↓↓ to ↑↑ unforced equilibrium configuration.

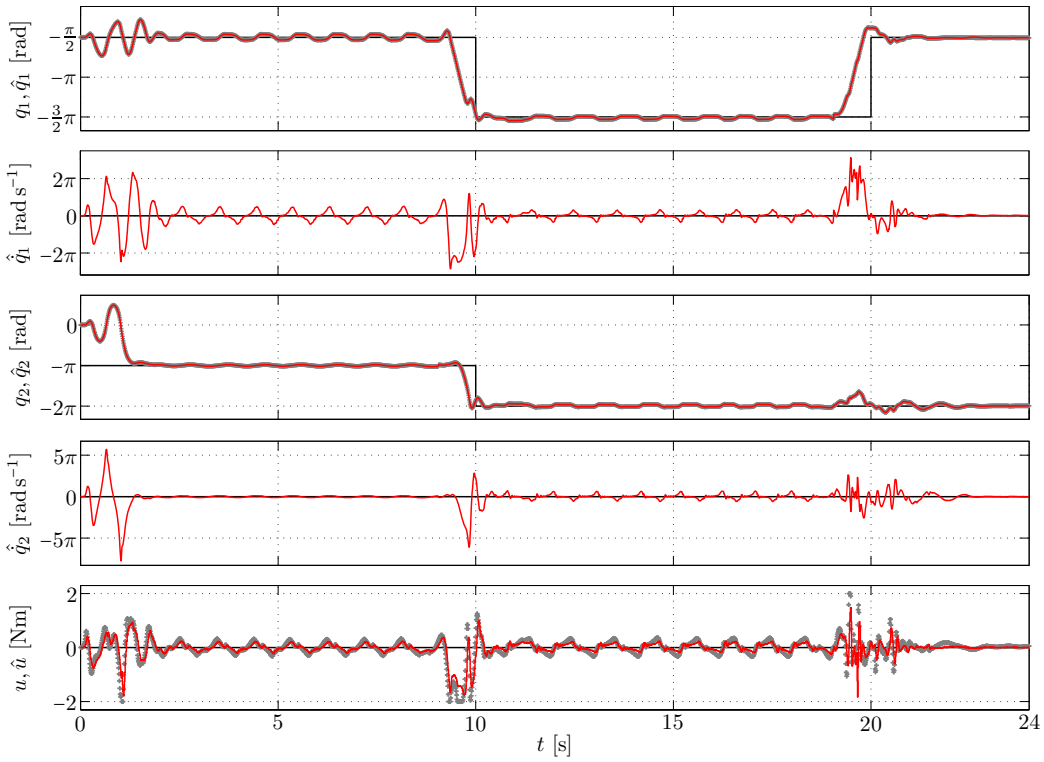

Figure 6. Experimental results for the ↓↓–↓↑–↑↑–↓↓ unforced equilibria tracking task.

Let us now review the control performance observed in Figures 4–6. First of all, and most importantly, the control objective wasfulfilled in all the aforementioned scenarios while respecting the upper and the lower bounds on both input and state variables. When appropriately tuned, the controller was able to continuously drive the system towards the desired position and maintain it unless prompted to change it, i.e., in the case of transition between the unstable equilibria depicted in Figure 6. One may also notice the prediction capabilities when a step change in the reference occurs. The reliable MHE estimates fairly contributed to the robustness of the control scheme, which was observed mainly in the otherwise critical transitions between setpoints. Similarly important are the smooth velocity estimates, fed to the controller instead of the rather noisy ones, initially calculated by means of finite differences. The accurate estimates of all the states are due to the low level of noise collected during signal transfer. The effects of unmodelled dynamics are reflected in the slightly inaccurate estimates of the control variable, which is,in fact, of no significance to the overall performance of the controller.

There was, however, one clearly observable drawback appearing during the stabilization phase, namely the oscillatory behavior of the Pendubot's actuated link. This also translated into the motion of the outer, passive link—specifically in the challenging ↑↑ unstable equilibrium, where the system was clearly most sensitive to any external disturbances, unmodelled dynamics, etc. This stick-slip behavior is due to the considerable friction present in the actuated joint. It originates from the nature of the servo drive combined with the use of a gearbox, and causes the limit-cycle phenomenon visible around the setpoints, which was obviously not captured by the simple viscous friction model employed so far, pointing to the requirement of a higher fidelity model of this real-world system.

4.2.2. Assuming the Nonlinear Friction Model and Input Rate Penalization

In the following, we therefore present experimental results obtained when assuming the nonlinear friction model (3). In (4a) and (4b), one may notice that the inclusion of the dynamic friction model introduced additional nonlinearities into the already fairly complex nominal system dynamics. This challenging mathematical model is henceforth central to both the nonlinear MPC and MHE scheme. It should be noted that since the

friction coefficients γ_i are not a priori known, they are chosen to be identified from the experimentally obtained data and subsequently verified, leading to the following values: $\gamma_1 = 0.476$, $\gamma_2 = 2.098$, $\gamma_3 = 0.649$, $\gamma_4 = 0.006$, $\gamma_5 = 0.727$, $\gamma_6 = 0.076$. The corresponding nonlinear friction characteristics are depicted in Figure 7. One may notice that its viscous dissipation term is very similar to the formerly employed simple viscous friction model, illustrated by gray color.

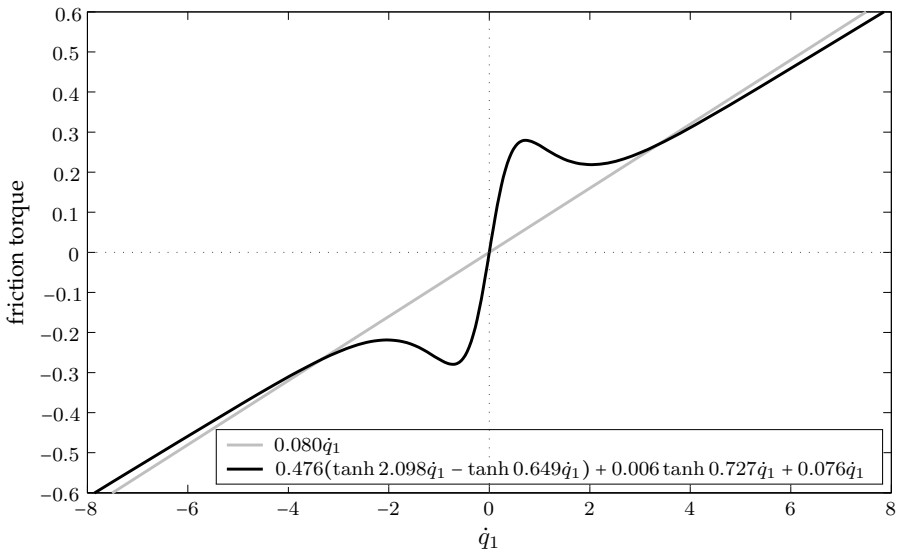

Figure 7. Identified nonlinear friction characteristics compared to the initially employed viscous friction model shown in gray color.

Practical experience also shows that the controller often used to run into two major kinds of problems. Firstly, the transitions between setpoints or even the start of a swing-up used to fail occasionally, as the controller weights, in particular the input ones, were changed stepwise following the proposed weights' switching strategy. This caused an unexpected increase or decrease in the aggressiveness of the controller causing it to react excessively or vice versa. Secondly, a common consequence was an eventual failure of the solver due to infeasibility or other numerical problems. This was the motivation to incorporate the control rate variable (given by torque slew-rate), u_R, into the model and problem formulation. The nonlinear ODE model (7) is hence augmented by $\dot{u} = u_\mathrm{R}$ to define a new state vector $x = [q_1, \dot{q}_1, q_2, \dot{q}_2, u]^\mathsf{T}$ and a new input u_R, with the corresponding nonlinear state-space model given as

$$\dot{x} = f(x, u_\mathrm{R}), \tag{10a}$$
$$y = h(x). \tag{10b}$$

This indeed implies that the decision variable in the NMPC formulation henceforth is u_R. This definition of the system input however stays merely formal as the control action is further determined by the current fifth state $x_5 = u$. This fact has to be properly taken into account within the real-time software implementation, where everything is executed in the same fashion except for the state u, which is at each time instant selected, adequately utilized within the NMPC–NMHE scheme, and—once converted to voltage—sent to the actuator. Nevertheless, for ease and clarity of further presentation, we continue to refer to u as control input and to u_R as control rate. Note also that the NMPC reference vector has to be changed accordingly, i.e., $[\cos x_1^\mathrm{ref}, \sin x_1^\mathrm{ref}, 0, \cos x_2^\mathrm{ref}, \sin x_2^\mathrm{ref}, 0, 0]^\mathsf{T}$ and $u_\mathrm{R}^\mathrm{ref} = 0$ (valid for the unforced equilibria tracking). The augmented controller weights read as $\mathrm{diag}(Q_{\mathrm{C}1}) = [2 \cdot 10^2, 2 \cdot 10^2, 1, 2 \cdot 10^2, 2 \cdot 10^2, 1, 1.5]$ and $R_{\mathrm{C}1} = 0.75$, for the swing-up part of the experiment, and $\mathrm{diag}(Q_{\mathrm{C}2}) = [2 \cdot 10^3, 2 \cdot 10^3, 1, 10^2, 10^2, 1, 8 \cdot 10^2]$ and $R_{\mathrm{C}2} = 0$ for stabilization in the unstable unforced $\downarrow\uparrow$ equilibrium.

The following experimental results were obtained after incorporating the nonlinear dynamic friction model (3) and the control rate u_R into the system description and thus the NMPC–NMHE problem formulation, while assuming the identical control scenarios as investigated in Figures 4–6. In particular, Figures 8–10 present the experimental results obtained for the task of controlling the system to the ↓↑, ↑↑, and multiple unforced equilibria, respectively.

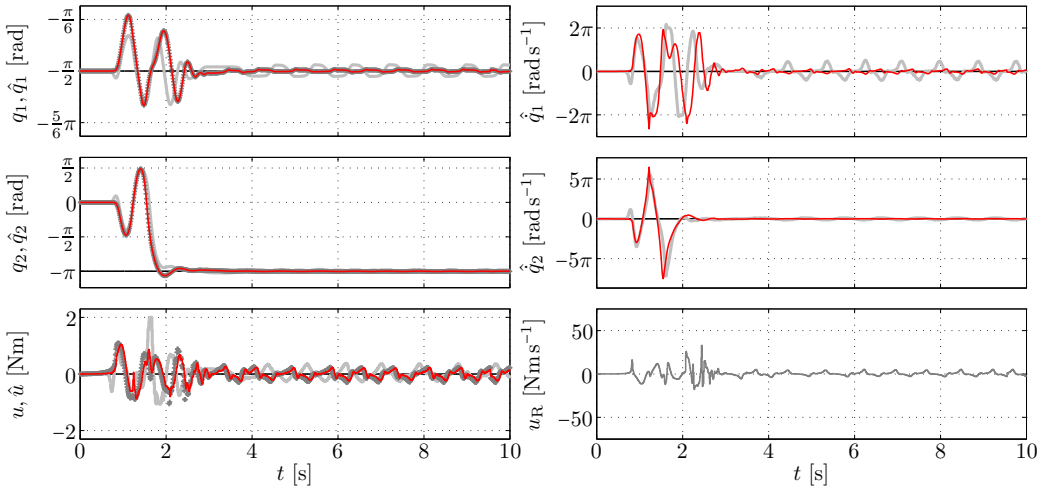

Figure 8. Experimental results for the transition from ↓↓ to ↓↑ unforced equilibrium configuration: red line—state and control input estimates, gray '+' markers—measurements, black line—references. The thick light gray curves in the background stand for the same control scenario assuming the simple viscous friction model cf. Figure 4). The same holds for Figures 9 and 10.

The last graph in each figure corresponds to the control (torque) rate calculated by the NMPC controller. In addition, note that each graph, except the last one, is always appended in the background by the evolution of the respective variable taken from the same control experiment, yet assuming the simple viscous friction model, allowing visual inspection of the effect of the nonlinear friction model. Clearly, its incorporation into the system dynamics greatly contributes to the control performance in all shown control scenarios, while the limit-cycle oscillation is largely suppressed, leading to a fairly improved performance of the controlled system. This is naturally of the utmost importance in precise positioning applications, as demonstrated here by Pendubot control.

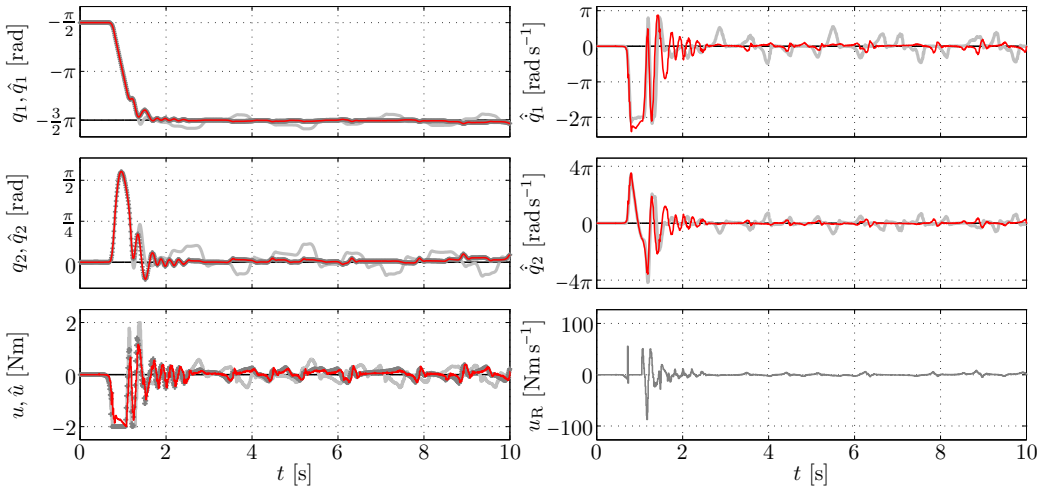

Figure 9. Experimental results for the transition from ↓↓ to ↑↑ unforced equilibrium configuration.

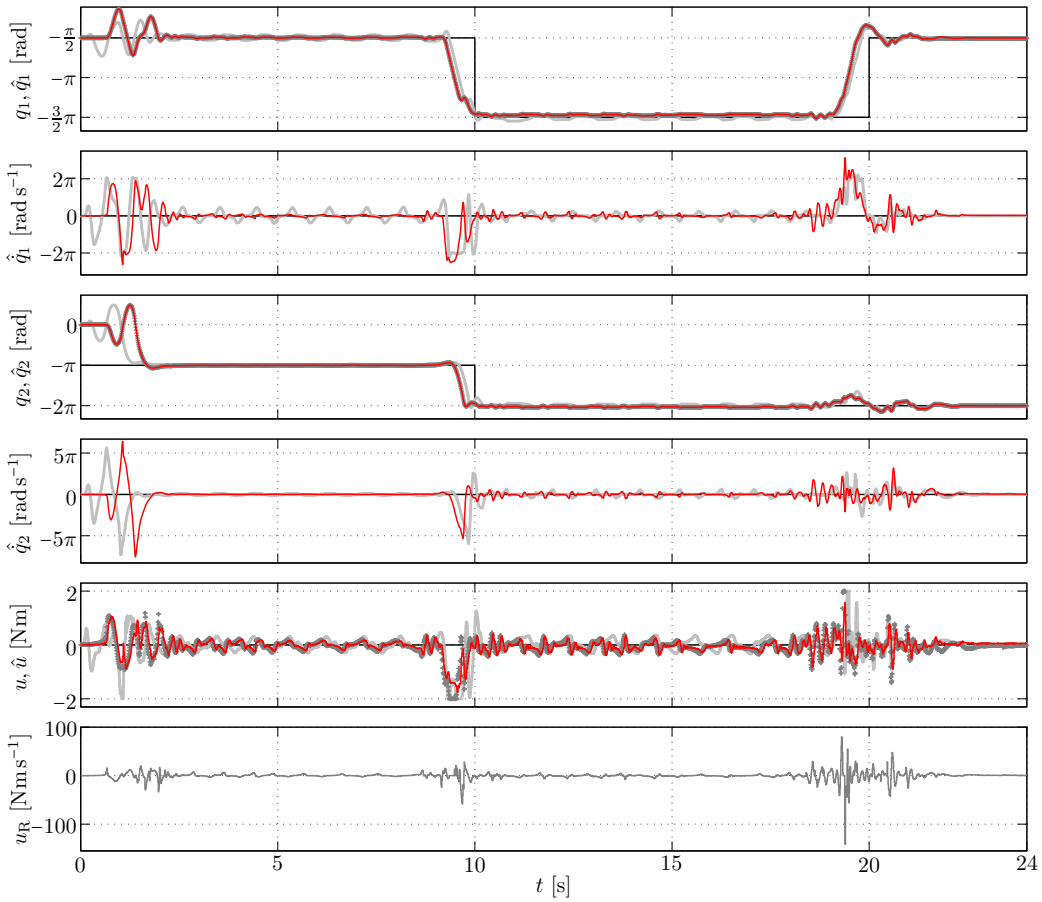

Figure 10. Experimental results for the ↓↓–↓↑–↑↑–↓↓ unforced equilibria tracking task.

Ultimately, the proposed real-time NMHE-based NMPC strategy building upon the full nonlinear friction-exploiting model allows the Pendubot the achievement of any of its unforced equilibrium configurations, and moreover smooth transfer between them while respecting the imposed input and state constraints. At this point, it should be noted that the concept of control rate penalization accomplishes its objective as well. This may be observed in Figures 4 and 6, namely in terms of the control action, which is adequately tuned to allow for the smooth transitions between setpoints. In particular, one may observe that it makes the input less aggressive and thereby avoids hitting its limits. The estimator plays an important role as well. Its moving horizon concept helps it to deliver reliable and accurate estimates, further used by the controller to determine adequate control action. In summary, the present estimation-control scheme demonstrates its excellent control performance in all investigated scenarios, including the ones not shown here. This makes it a powerful strategy that proves its potential for use in fast mechatronic applications.

4.3. Tracking of Forced Equilibrium Positions

Based on the excellent controller performance presented in Section 4.2.2, we can proceed to the task of point-to-point transition between multiple reference equilibrium configurations including the forced ones. Starting and finishing in the downward resting position, the control scenario was to sequentially achieve and transfer between selected setpoints in 5 s intervals. Reference vectors had to be set and switched accordingly, including one stable (↓↓) and two main unstable (↓↑,↑↑) unforced equilibrium positions, and, in addition, four different forced equilibrium configuration positions (↙↑,↖↑,↗↑,↘↑) with the angular deflection of the active link from the vertical axis chosen as $\pm\pi/4$ in all four of them. Note that in the latter case, the nonzero reference values of the control input had to be set as well in order to effectively compensate for the gravity phenomena. Following the

mathematical description of the system, it was found that $u^{\mathrm{ref},i} = \pm 0.52$ Nm with polarity depending on the respective forced equilibrium configuration of the Pendubot.

As evidenced by the system's behavior illustrated in Figure 11, when tuned properly, the controller is able to perform the swinging maneuvers and balancing in the reference positions smoothly and precisely. Notice that the depicted 5 s time windows are shifted 1 s backwards due to the predictive capability of the controller ($T_{\mathrm{C}} = 1$ s), which allows it anticipation of the changes in reference and thus reaction in advance. Note also that any other controllable forced equilibrium setpoints may be assumed in a similar fashion.

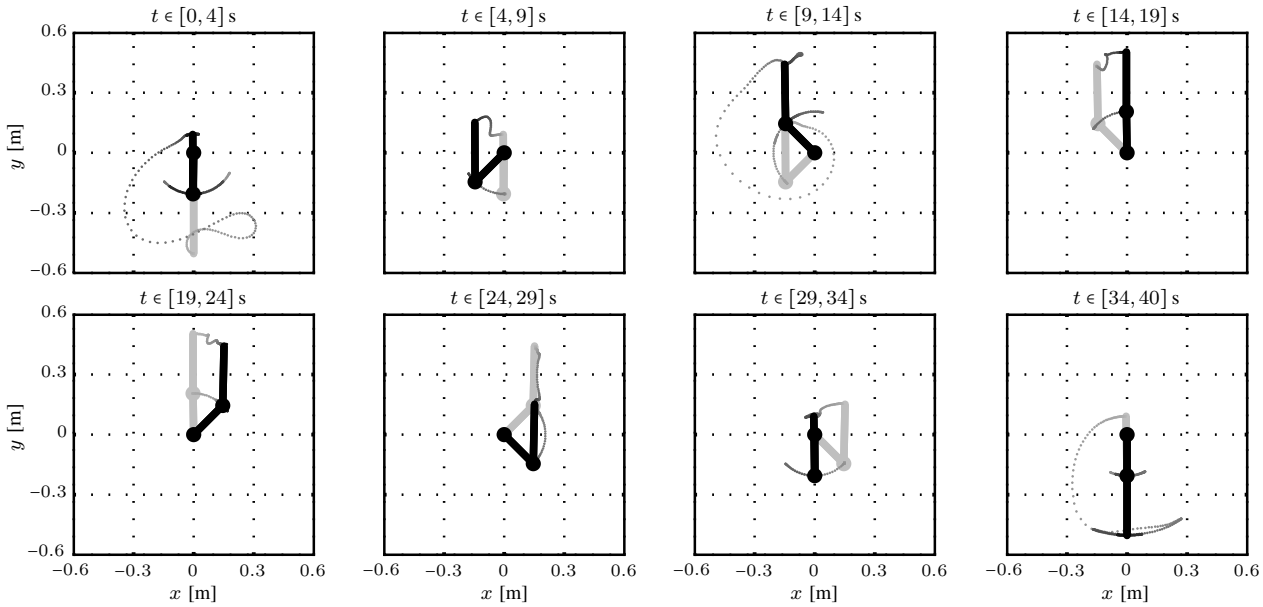

Figure 11. Experimental-data-based snapshots for the real-time control task of point-to-point tracking the forced and unforced equilibrium configurations of the Pendubot, depicted in eight consecutive sequences corresponding to particular setpoints in the order of ↓↓–↓↑–↙↑–↖↑–↑↑–↗↑–↘↑–↓↑–↓↓. The light gray and black colored snapshots depict configurations of the laboratory system at the beginning and at the end of given time frame, respectively. Video of one such experiment can be viewed at [28].

4.4. Evaluation of Computational Complexity

To assess the computational complexity of this implementation, Table 1 reports execution times of its main algorithmic ingredients, relating to the experiment in Figure 11. In particular, it treats the iterations of the RTI scheme for both NMHE and NMPC problem separately and splits the computational effort of each into preparation phase (PP) and a feedback/estimation phase (FP/EP). As outlined earlier, for solving these nonlinear optimization problems with SQP, we used the condensing-based qpOASES solver as well as the sparsity-exploiting qpDUNES solver. Both approaches clearly evidence that the RTI execution time is dominated by the PP, where most of the effort is spent on linearization of the NLP. The FP and EP are devoted to the solution of a single dense or sparse QP subproblem. The timing results indicate lower per-iteration complexity of the sparse QP strategy that scales better with the problem size and is therefore better suited for control problems requiring longer horizons, which is also the case here.

As further evidenced in Table 1, the computational performance can be also improved by exploiting the parallelization of particular routines. While the EP and FP steps need to be executed sequentially, given the multicore processor, the NMHE PP can be triggered simultaneously with the NMPC FP. This idea is also illustrated in Figure 12. After a new measurement $\bar{y}(k)$ is ready at time $t = 4.53$ s, the NMHE EP is triggered. Once the state estimate $\hat{x}_0$ becomes available, it is processed in the NMPC FP to determine the control action $u_0^\star$, without waiting for the NMHE PP to finish.

Table 1. Real-time performance in terms of execution times (ms) of the NMHE–NMPC iterations †.

Execution Mode		**Sequential**							**Parallelized**						
(See Figure 12)		**NMHE**			**NMPC**			**TET**	**NMHE**			**NMPC**			**TET**
Solver		**EP**	**PP**	**RTI**	**FP**	**PP**	**RTI**		**EP**	**PP**	**RTI**	**FP**	**PP**	**RTI**	
qpOASES	t_{avg}	0.148	1.259	1.407	0.923	4.418	5.339	6.750	0.163	1.301	1.464	0.908	4.446	5.354	5.591
	t_{max}	0.339	1.902	2.071	3.010	6.519	7.540	9.017	0.455	1.763	2.111	2.758	5.110	7.257	7.462
qpDUNES	t_{avg}	0.205	1.029	1.254	0.446	2.119	2.567	3.751	0.202	0.983	1.180	0.501	2.144	2.643	2.877
	t_{max}	1.109	1.345	2.048	0.833	2.711	3.200	4.968	0.345	1.582	1.904	0.739	3.557	4.338	4.630

† t_{avg} / t_{max}—average / maximum computation time over the entire experimental run depicted in Figure 11, TET—total execution time of the NMHE-NMPC iterations, EP—estimation phase, FP—feedback phase, PP—preparation phase, RTI—full real-time iteration.

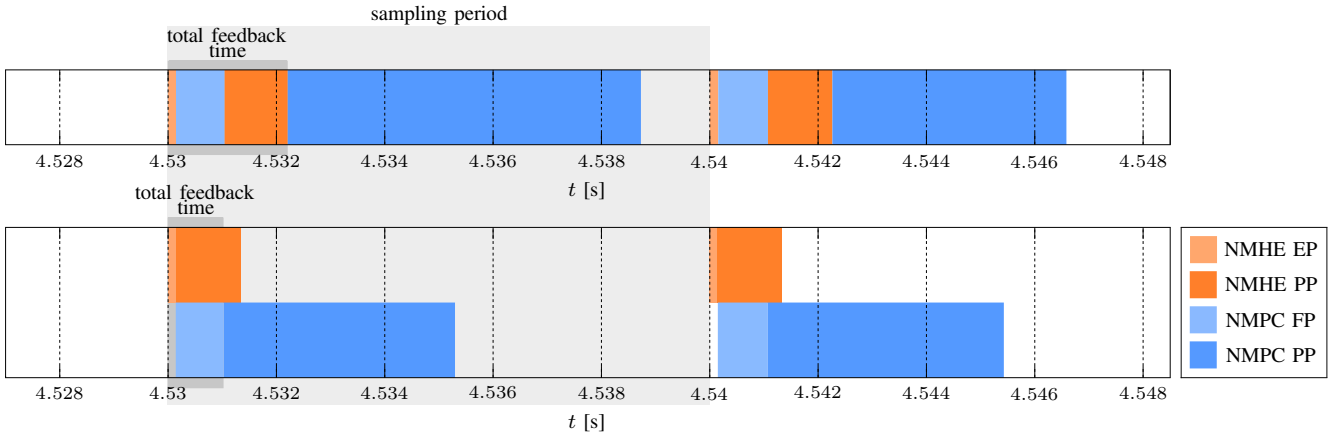

Figure 12. Illustration of timing performance for the NMHE-NMPC real-time iteration scheme when executed in the standard—sequential (**upper** graph) and the parallelized (**lower** graph) configuration, taken during the ↙↑–↖↑ transition of the Pendubot.

Note that the achieved computation times still leave enough headroom for other potential procedures and auxiliary tasks, eventually allowing the increase in the NMPC and NMHE horizon lengths or the number of iterations of the QP solver. This also suggests that by employing more efficient state-of-the-art software tools, such as acados [29], the very recent successor of ACADO Toolkit, the proposed NMHE–NMPC scheme may be deployed even on much less powerful hardware, such as embedded microcontrollers. In terms of real-time software implementation for the Pendubot control, the above concept of closed-loop parallelization was carried out by means of the openMP API.

The presented experimental results obtained within various Pendubot control scenarios clearly validate the theoretical assumptions and objectives, and demonstrate the applicability of real-time nonlinear model predictive control not only for underactuated mechanical systems, but for fast mechatronic systems in general. Fast sampling is not only necessary to capture the system dynamics, but it also allows for a fast reaction to external disturbances. The control scheme is moreover augmented by a nonlinear moving horizon observer, exploiting essentially the same concepts from the field of numerical optimization. In fact, the entire NMHE-based NMPC framework is shown to perform very well, with feasible execution times well below the 10 ms sampling time. With regards to the main objective, the proposed scheme enables the Pendubot to promptly achieve the desired unstable equilibria, and to smoothly transition between them upon request.

5. Conclusions

The paper presented a practical implementation of the NMHE–NMPC framework for the real-time control of the Pendubot system. Within a unified, nonlinear optimization-based control strategy, it enables to swing up, balance, and transition this underactuated

robot between multiple unstable configurations, including not only the typically assumed unforced equilibria, but also, and mainly, the most challenging—forced equilibrium configurations.

In addition, in order to improve the model fidelity and hence control performance, an advanced nonlinear dynamic friction model was considered. The control course was also smoothened by incorporating the input rate into the control scheme and its appropriate penalization. Apart from control performance, an effort was also put into the improvement of computational efficiency by replacing the dense QP solver with a sparse one, as well as by introducing parallelization into the proposed NMHE–NMPC real-time-iteration-based framework. Its performance and computational efficiency were experimentally tested.

Although the intended objectives of this paper were accomplished, there are still promising directions for further research, namely in terms of computational complexity. This issue becomes particularly challenging when targeting low-cost embedded microcontroller-based computing platforms, which is undeniably a massive trend in modern control applications, and therefore also subject to ongoing research of the authors. For appropriately cast NMPC problems, their computational requirements may be reduced even more than shown herein, e.g., by employing structure-exploiting interior-point QP solvers. Another most recent way towards very fast NMPC implementations suggests combining the real-time iteration scheme with first-order methods, which have already attracted much attention for MPC applications on embedded hardware. In addition, the widely known field programmable gate arrays (FPGAs) also offer a lot of room for very efficient and customized implementations, largely exploiting parallelization. It would also be interesting to devise a stabilizing extension for the fast auto-generated NMPC schemes presented in this work. Despite the fact that such attempts have not been proven to be useful in other related works, such an extension would surely find its theoretical justification or use in control scenarios with shorter horizon lengths. Despite the nominal stability under realistic assumptions shown in [30], in general. it constitutes a difficult problem as the RTI scheme does not solve the underlying problems exactly, but rather approximately, hence the classical NMPC stability theory is not trivial to be extended to this scheme, but it may be also tackled as a part of our future work.

Author Contributions: Conceptualization, M.G., M.S. and B.R.-I.; methodology, M.G. and M.S.; software, M.G. and M.S.; validation, M.G. and M.S.; formal analysis, M.G. and M.S.; investigation, M.G. and M.S.; resources, M.G. and M.S.; data curation, M.G.; writing—original draft preparation, M.G.; writing—review and editing, M.G.; visualization, M.G.; supervision, B.R.-I.; project administration, M.G.; funding acquisition, B.R.-I. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Slovak Research and Development Agency under the grants APVV-18-0023 and APVV-22-0436, by the Cultural and Educational Grant Agency of the Ministry of Education of Slovak Republic under the grant KEGA 012STU-4/2021 and by the European Union's Horizon Europe under the grant no. 101079342 (Fostering opportunities towards Slovak excellence in advanced control for smart industries).

Data Availability Statement: Data sharing is not applicable.

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

References

1. Choukchou-Braham, A.; Cherki, B.; Djemaï, M.; Busawon, K. *Analysis and Control of Underactuated Mechanical Systems*; Springer: London, UK, 2014.
2. Olfati-Saber, R. Nonlinear Control of Underactuated Mechanical Systems with Application to Robotics and Aerospace Vehicles. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2001.
3. Fantoni, I.; Lozano, R. *Non-Linear Control for Underactuated Mechanical Systems*; Verlag: London, UK, 2002.
4. Xin, X.; Liu, Y. *Control Design and Analysis for Underactuated Robotic Systems*; Springer: London, UK, 2014.
5. Jiang, J.; Astolfi, A. Stabilization of a class of underactuated nonlinear systems via underactuated back-stepping. *IEEE Trans. Autom. Control* **2021**, *66*, 5429–5435. [CrossRef]

6. Pazderski, D.; Parulski, P.; Bartkowiak, P.; Herman, P. Sub-optimal stabilizers of the Pendubot using various state space representations. *Energies* **2022**, *15*, 5146. [CrossRef]
7. Sætre, C.F.; Shiriaev, A. Orbital stabilization of point-to-point maneuvers in underactuated mechanical systems. *Automatica* **2023**, *147*, 110735. [CrossRef]
8. Spong, M.; Block, D. The Pendubot: A mechatronic system for control research and education. In Proceedings of the 34th IEEE Conference on Decision and Control, New Orleans, LA, USA, 13–15 December 1995; pp. 555–556.
9. He, B.; Wang, S.; Liu, Y. Underactuated robotics: A review. *Intell. Manuf. Robot.* **2019**, *16*. [CrossRef]
10. Gulan, M.; Salaj, M.; Rohal'-Ilkiv, B. Achieving an equilibrium position of Pendubot via swing-up and stabilizing MPC. *J. Electr. Eng.* **2014**, *65*, 356–363.
11. Eom, M.; Chwa, D. Robust swing-up and balancing control using a nonlinear disturbance observer for the Pendubot system with dynamic friction. *IEEE Trans. Robot.* **2015**, *31*, 331–343. [CrossRef]
12. Wei, C.; Chai, T.; Xin, X.; Chen, X.; Wang, L.; Chen, Y.H. A signal compensation-based robust swing-up and balance control method for the Pendubot. *IEEE Trans. Ind. Electron.* **2022**, *69*, 3007–3016. [CrossRef]
13. Wang, Z.; Guo, Y. Unified control for Pendubot at four equilibrium points. *IET Control Theory Appl.* **2011**, *5*, 155–163. [CrossRef]
14. Gulan, M.; Salaj, M.; Rohal'-Ilkiv, B. Nonlinear Model Predictive Control with Moving Horizon Estimation of a Pendubot system. In Proceedings of the 2015 20th International Conference on Process Control, Štrbské Pleso, Slovakia, 9–12 June 2015; pp. 226–231.
15. Turrisi, G.; Carlos, B.B.; Cefalo, M.; Modugno, V.; Lanari, L.; Oriolo, G. Enforcing constraints over learned policies via nonlinear MPC: application to the Pendubot. *IFAC-PapersOnLine* **2020**, *53*, 9502–9507. [CrossRef]
16. Wang, L.; Lai, X.; Zhang, P.; Wu, M. Motion control strategy based on integrated trajectory for the Pendubot. In Proceedings of the 2021 European Control Conference, Delft, The Netherlands, 29 June–2 July 2021; pp. 2151–2156.
17. Xia, D.; Chai, T.; Wang, L. Fuzzy neural-network friction com-pensation-based singularity avoidance energy swing-up to nonequilibrium unstable position control of Pendubot. *IEEE Trans. Control Syst. Technol.* **2014**, *22*, 690–705. [CrossRef]
18. Ramírez-Neria, M.; Sira-Ramírez, H.; Garrido-Moctezuma, R.; Luviano-Juárez, A.; Gao, Z. Active disturbance rejection control for reference trajectory tracking tasks in the Pendubot system. *IEEE Access* **2021**, *9*, 102663–102670. [CrossRef]
19. Diehl, M.; Bock, H.G.; Schlöder, J. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM J. Control Optim.* **2005**, *43*, 1714–1736. [CrossRef]
20. Houska, B.; Ferreau, H.J.; Diehl, M. ACADO toolkit—An open-source framework for automatic control and dynamic optimization. *Optim. Control Appl. Methods* **2011**, *32*, 298–312. [CrossRef]
21. Houska, B.; Ferreau, H.J.; Diehl, M. An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range. *Automatica* **2011**, *47*, 2279–2285. [CrossRef]
22. Eltrabyly, A.; Ichalal, D.; Mammar, S. Quadcopter trajectory tracking in the presence of 4 faulty actuators: A nonlinear MHE and MPC Approach. *IEEE Control Syst. Lett.* **2022**, *6*, 2024–2029. [CrossRef]
23. Salaj, M.; Gulan, M.; Rohal'-Ilkiv, B. Pendubot control scheme based on nonlinear MPC and MHE exploiting parallelization. In Proceedings of the 19th International Conference on Intelligent Engineering Systems, Bratislava, Slovakia, 3–5 September 2015; pp. 353–358.
24. Makkar, C.; Dixon, W.; Sawyer, W.; Hu, G. A new continuously differentiable friction model for control systems design. In Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Monterey, CA, USA, 24–28 July 2005; pp. 600–605.
25. Diehl, M.; Ferreau, H.J.; Haverbeke, N. Efficient numerical methods for nonlinear MPC and moving horizon estimation. In *Nonlinear Model Predictive Control—Towards New Challenging Applications*; Magni, L., Raimondo, D.M., Allgöwer, F., Eds.; Number 384 in Lecture Notes in Control and Information Sciences; Springer: Berlin/Heidelberg, Germany, 2009; pp. 391–417.
26. Ferreau, H.; Kirches, C.; Potschka, A.; Bock, H.; Diehl, M. qpOASES: A parametric active-set algorithm for quadratic programming. *Math. Program. Comput.* **2014**, *6*, 327–363. [CrossRef]
27. Frasch, J.V.; Vukov, M.; Ferreau, H.J.; Diehl, M. A new quadratic programming strategy for efficient sparsity exploitation in SQP-based nonlinear MPC and MHE. In Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, 25–29 August 2014; pp. 2945–2950.
28. Gulan, M.; Salaj, M. NMHE-NMPC Based Tracking of Unforced and Forced Equilibrium Positions of a Pendubot System. Available online: https://youtu.be/YRelFXm5Sso (accessed on 27 June 2023).
29. Verschueren, R.; Frison, G.; Kouzoupis, D.; Frey, J.; van Duijkeren, N.; Zanelli, A.; Novoselnik, B.; Albin, T.; Quirynen, R.; Diehl, M. acados—A modular open-source framework for fast embedded optimal control. *Math. Program. Comput.* **2021**, *14*, 147–183. [CrossRef]
30. Diehl, M.; Findeisen, R.; Allgower, F.; Bock, H.; Schloder, J. Nominal stability of real-time iteration scheme for nonlinear model predictive control. *IEE Proc.—Control Theory Appl.* **2005**, *152*, 296–308. [CrossRef]