Received: 30 October 2019 | Revised: 29 June 2021 | Accepted: 29 June 2021

DOI: 10.1002/rnc.5696

SPECIAL ISSUE ARTICLE

WILEY

Probabilistic performance validation of deep learning-based robust NMPC controllers

Benjamin Karg[1] | **Teodoro Alamo**[2] | **Sergio Lucia**[1]

[1]Laboratory of Process Automation Systems, Technische Universität Dortmund, Dortmund, Germany

[2]Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Sevilla, Spain

Correspondence
Sergio Lucia, Laboratory of Process Automation Systems, Technische Universität Dortmund, Emil-Figge-Str. 70, Dortmund, 44227, Germany.
Email: sergio.lucia@tu-dortmund.de

Funding information
Agencia Estatal de Investigación AEI-Spain, Grant/Award Number: PID2019-106212RB-C41/AEI/10.13039/501100011033; Deutsche Forschungsgemeinschaft, Grant/Award Number: 423857295

Abstract

Solving nonlinear model predictive control problems in real time is still an important challenge despite of recent advances in computing hardware, optimization algorithms and tailored implementations. This challenge is even greater when uncertainty is present due to disturbances, unknown parameters or measurement and estimation errors. To enable the application of advanced control schemes to fast systems and on low-cost embedded hardware, we propose to approximate a robust nonlinear model controller using deep learning and to verify its quality using probabilistic validation techniques. We propose a probabilistic validation technique based on finite families, combined with the idea of generalized maximum and constraint backoff to enable statistically valid conclusions related to general performance indicators. The potential of the proposed approach is demonstrated with simulation results of an uncertain nonlinear system.

KEYWORDS

machine learning, model predictive control, probabilistic validation, robust control

1 | INTRODUCTION

Nonlinear model predictive control (NMPC) is a popular advanced control technique that can deal with nonlinear systems and constraints while considering general control goals that go beyond conventional set-point tracking tasks. Two of the main obstacles that one faces when implementing and designing a nonlinear model predictive controller are the accuracy of the model and the computational complexity needed to solve a nonconvex optimization problem online, which often renders its implementation too slow for fast systems or impossible to be deployed on resource-constrained embedded platforms.

Handling uncertainty in the context of model predictive control is the main goal of robust MPC. Traditional min-max approaches[1] do not explicitly consider the fact that new information will be available in the future, which leads to over-conservative solutions. Closed-loop robust MPC avoids the problem of conservativeness by optimizing over control policies instead of optimizing over control inputs,[2] leading however to intractable formulations in the general case. Most of the recent robust MPC methods focus on achieving a good trade-off between complexity and performance. Tube-based approaches[3] decompose the robust MPC problem into a nominal MPC and an ancillary controller. The ancillary controller makes sure that the real uncertain system stays close to the trajectory planned by the nominal MPC. By tightening the constraints of the nominal MPC, robust constraint satisfaction can be achieved. In the simplest version, the complexity of tube-based MPC is the same as that of standard MPC. However, if an increased performance is desired,

the complexity grows as presented in Reference 4 or 5. Scenario tree-based[6-8] or multistage MPC[9] represents the evolution of the uncertainty using a tree of discrete uncertainty realizations. An improved performance can be often seen in practice[10] because the feedback structure is not restricted to be affine, as usually done in tube-based MPC and in other robust approaches.[11] While it is also possible to achieve stability and robust constraint satisfaction guarantees for a multistage MPC formulation,[7,12,13] its computational complexity grows exponentially with the dimension of the uncertainty space. The presence of uncertainty significantly increases the computational complexity of any NMPC implementation if a nonconservative performance is desired.

The last decade has witnessed an important progress on hardware, algorithms and tailored implementations that enable the efficient solution and implementation of NMPC controllers based, for example, on code generation tools[14,15] that provide efficient implementations of linear and nonlinear MPC on embedded hardware, including low-cost microcontrollers[16] and high-performance field programmable gate array (FPGAs).[17]

A different possibility to achieve embedded nonlinear model predictive control is the use of approximate explicit nonlinear model predictive control[18,19] based on approximating the multiparametric nonlinear program using similar ideas as for explicit MPC of linear systems.[20] We propose in this work to use deep neural networks (DNNs) to approximate a robust multistage NMPC control law. The idea of using a neural network as function approximator for an NMPC feedback law was already proposed by Reference 21 back in 1995, but only very recently[22,23] DNNs (neural networks with several hidden layers) have been proposed as function approximators. The use of DNNs is motivated by recent theoretical results that suggest the exponentially better approximation capabilities of DNNs in comparison to shallow networks.[24]

Assessing the closed-loop performance of approximate controllers, or any other controller subject to further random disturbances or estimation errors, is particularly challenging in the case of complex nonlinear systems. The theory of randomized algorithms[25,26] provides different schemes capable of addressing this issue. For example, statistical learning techniques can be used to design stochastic model predictive controllers with probabilistic guarantees.[27-29] Also, under a convexity assumption, convex scenario approaches[30] can be used in the context of chance constrained MPC.[31-33] The main limitation of the aforementioned approaches based on statistical learning results,[25,34] and scenario-based ones[30] is that the number of random scenarios that have to be generated (sample complexity) grows with the dimension of the problem.

Probabilistic validation[35,36] allows one to determine if a given controller satisfies, with a prespecified probability of violation and confidence, the control constraints. The sample complexity in this case does not depend on the dimension of the problem, but only of the required guaranteed probability of violation and confidence. Examples of probabilistic verification approaches in the context of control of nonlinear uncertain systems can be found, for example, in References 26,37,38. These techniques have also been used for the probabilistic certification of the behaviour of off-line approximations of nonlinear control laws.[39,40]

The main contribution of this paper, which extends the results from Reference 41, is the formulation of general closed-loop performance indicators that are not restricted to binary functions as in Reference 39 and can be computed simulating the closed-loop system with any given controller. We also provide sample complexity bounds that do not grow with the size of the problem for the case of a finite family of design parameters and general performance indicators. Our approach allows to discard a finite number of worst-case closed-loop simulations, improving significantly the applicability of the probabilistic validation scheme compared to existing works. The potential of the presented approach is illustrated for a highly nonlinear towing kite system including a real-time capable embedded implementation of an approximate, but probabilistically safe, robust nonlinear model predictive controller on a low-cost microcontroller.

The paper is organized as follows. The closed-loop performance indicators are introduced in Section 2 which are used in a novel probabilistic validation methodology for arbitrary controllers in Section 3. The mathematical framework for the output feedback robust NMPC problem considered in this work is presented in Section 4 and the use of deep learning to obtain approximate robust NMPC controllers is summarized in Section 5. The case study is detailed in Section 6, the results in Section 7 and the paper is concluded in Section 8.

2 | CLOSED-LOOP PERFORMANCE INDICATORS

2.1 | System description and constraints

We are interested in optimally controlling the following class of nonlinear discrete time systems:

$$x(k+1) = f(x(k), u(k), d(k)), \tag{1}$$

$$y(k) = h(x(k), u(k), d(k)), \tag{2}$$

where $x(k) \in \mathcal{R}^{n_x}$ is the state vector, $u(k) \in \mathcal{R}^{n_u}$ is the control input, and $d(k) \in \mathcal{R}^{n_d}$ is the disturbance vector. In general, not all states can be measured, and a state estimate $\hat{x}(k)$ should be computed based on the past measurements $y(k) \in \mathcal{R}^{n_y}$. It is assumed that the disturbances $d(k)$ take values, with high probability, from a known set $\mathcal{D}$.

Assumption 1. The nonlinear discrete-time system (1) and (2) is observable and controllable.

The closed-loop trajectory should satisfy general nonlinear input and state constraints defined by

$$g_l(x(k), u(k), d(k)) \leq 0,\ l = 1, \ \dots\ , n_g, \tag{3}$$

where n_g is the number of constraints.

2.2 | Closed-loop behavior

The goal of a controller $\kappa : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ is that the closed-loop trajectory of the uncertain nonlinear system defined by

$$x(k+1) = f(x(k), \kappa(\hat{x}(k)), d(k)), \tag{4}$$

obtains a desired performance level, for example, does not violate the predefined constraints, despite the presence of uncertainty, for any initial state $x(0)$ in the set $\mathcal{X}_0$ of feasible initial conditions, for any admissible initial estimation error $x(0) - \hat{x}(0)$ and for any sequence of uncertainty realizations $\{d(0), d(1), \ \dots\ , d(\infty)\}$.

Determining if a given controller provides admissible closed-loop trajectories, under the presence of nonlinearity and uncertainty, is in general an intractable problem.[42] Instead, we focus on the use of finite-time closed-loop performance indicators that can be obtained by simulating the closed-loop system. The underlying assumption is that models which can be run a large number of times are available so that statistical guarantees can be obtained. A closed-loop performance indicator is defined as follows.

Definition 1 (Closed-loop finite-time performance indicator). Let $w = \{x(0), \hat{x}(0), d(0), \ \dots\ , d(N_{\text{sim}} - 1)\}$ denote the variables that uniquely define the closed loop trajectories $z(w; N_{\text{sim}}, \kappa) = \{x(0), \hat{x}(0), \kappa(\hat{x}(0)), d(0), x(1), \kappa(\hat{x}(1)), d(1), \ \dots\ , x(N_{\text{sim}})\}$ given an initial condition $x(0)$, an initial estimate $\hat{x}(0)$, a sequence of uncertainty realizations $d(0), \ \dots\ , d(N_{\text{sim}} - 1)$ that also include the measurement noise, a controller κ and a finite number of simulation steps N_{sim}. A closed-loop finite-time performance indicator is a measurable function $\phi(w; N_{\text{sim}}, \kappa) : \mathcal{W} = \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_d} \times \ \dots\ \times \mathbb{R}^{n_d} \to \mathbb{R}$ that takes as input all variables defining the closed-loop trajectories for a controller κ until time N_{sim} and gives a scalar as a measure of closed-loop performance:

$$\phi(w; N_{\text{sim}}, \kappa) = \phi(x(0), \hat{x}(0), d(0), d(1), \ \dots\ , d(N_{\text{sim}} - 1)). \tag{5}$$

Assumption 2. There exists a simulator that is able to compute closed-loop trajectories defined by (4). In addition, there exists a known operator Φ_k that provides the state estimation $\hat{x}(k+1)$ from $\hat{x}(k)$, $y(k)$ and $u(k)$. That is,

$$\hat{x}(k+1) = \Phi_k(\hat{x}(k), y(k), u(k)). \tag{6}$$

Assumption 2 implies that given N_{sim} and the controller κ, the closed-loop trajectories are completely determined by w. Probabilistic validation normally relies on a binary performance indicator that determines if the closed-loop is admissible or not. That is,

$$\phi(w; N_{\text{sim}}, \kappa) = \begin{cases} 0 & \text{if the closed-loop trajectory is admissible for } w, \\ 1 & \text{otherwise.} \end{cases}$$

For this particular setting, one can resort to well-known results to obtain probabilistic guarantees about the probability $\Pr_{\mathcal{W}}(\phi(w, N_{\text{sim}}, \kappa))$. For a review on these results, see Reference 36. See also Reference 39, where Hoeffding's inequality[43] is used to derive probabilistic guarantees in the context of learning an approximate model predictive controller.

In this paper we address a more general setting in which we do not circumscribe the performance indicator to the class of binary functions. For example, we consider the closed-loop finite-time performance indicator given by the largest

value of any component of g_l along the closed-loop simulation:

$$\phi(w; N_{\text{sim}}, \kappa) = \max_{\substack{k=0,\dots,N_{\text{sim}}-1 \\ l=1,\dots,n_g}} g_l(x(k), \kappa(\hat{x}(k)), d(k)). \tag{7}$$

Another possibility is to consider the average constraint violation as a performance indicator. That is,

$$\phi(w; N_{\text{sim}}, \kappa) = \frac{1}{N_{\text{sim}} n_{\text{g}}} \sum_{k=0}^{N_{\text{sim-1}}} \sum_{l=1}^{n_g} \max\{0, g_l(x(k), \kappa(\hat{x}(k)), d(k))\}. \tag{8}$$

Moreover, in many applications it is relevant to consider indicators related to the closed-loop cost, such as an average cost:

$$\phi(w; N_{\text{sim}}, \kappa) = \frac{1}{N_{\text{sim}}} \sum_{k=0}^{N_{\text{sim-1}}} \ell(x(k), \kappa(\hat{x}(k))), \tag{9}$$

or any other combination. In the following section we address how to obtain probabilistic guarantees on the random variable $\phi(w; N_{\text{sim}}, \kappa)$.

3 | PROBABILISTIC VALIDATION

The derived closed-loop performance indicators can be used in the framework of probabilistic validation[26,36] to obtain probabilistic guarantees regarding the satisfaction of a given set of control specifications. In this section we present a novel result that allows us to address the probabilistic validation of arbitrary control schemes where the performance is influenced by hyper-parameters, for example, backoff parameters or the control sampling time.

3.1 | Probabilistic performance indicator levels

We consider a finite family of controllers

$$\kappa_i(\hat{x}),\ i = 1, \dots, M,$$

corresponding to M different combinations of hyper-parameter values, for example, for constraint backoffs or control sampling times. The objective of this section is to provide a probabilistic validation scheme that allows us to choose, from the M possible controllers, the one with the best probabilistic certification for any given closed-loop finite-time performance indicator $\phi(w; N_{\text{sim}}, \kappa_i)$. For simplicity in the notation, we denote the closed-loop finite-time performance indicator obtained with the controller κ_i with N_{sim} simulation steps as $\phi_i(w)$.

Remark 1. The stochastic variable w that defines the closed-loop trajectories follows the probability distribution $\mathcal{W}$ from which it is possible to obtain independent identically distributed (i.i.d.) samples.

Remark 1 only requires knowledge of the probability distributions of the uncertainty.

Definition 2 (Probabilistic performance indicator level). We say that $\gamma \in \mathbb{R}$ is a probabilistic performance indicator level with violation probability $\epsilon \in (0, 1)$ for a sample w drawn from $\mathcal{W}$ for the measurable function $\phi : \mathcal{W} \to \mathbb{R}$ if the probability of violation $\Pr_{\mathcal{W}}\{\cdot\}$ satisfies

$$\Pr_{\mathcal{W}}\{\phi(w) > \gamma\} \leq \epsilon.$$

To obtain probabilistic performance indicator levels for the considered controllers κ_i, $i = 1, \dots, M$, we generate N i.i.d. scenarios

$$w^{(j)} = \{x^{(j)}(0), \hat{x}^{(j)}(0), d^{(j)}(0), \dots, d^{(j)}(N_{\text{sim}-1})\},\ j = 1, \dots, N.$$

For a given controller κ_i, with $1 \leq i \leq M$, and the uncertain realizations $w^{(j)}$, $j = 1, \dots, N$, one could simulate the closed-loop dynamics and obtain the performance indicator corresponding to each uncertain realization. That is, one could obtain

$$\mathbf{v}_i = [\phi_i(w^{(1)}), \phi_i(w^{(2)}), \dots, \phi_i(w^{(N)})]^\top \in \mathbb{R}^N.$$

It is clear that the largest value of the components of $\mathbf{v}_i$ could serve as an empirical performance level for the controller κ_i provided that N is large enough.[35] Another possibility is to discard the $r-1$ largest components of $\mathbf{v}_i$ and consider the largest of the remaining components as a (less conservative) empirical performance indicator level (r equal to one corresponds to not discarding any component).[44] In the following section we show how to choose N such that the obtained empirical performance indicator levels are, with high confidence $1-\delta$, probabilistic performance indicator levels with probability of violation ϵ.

3.2 | Sample complexity

We first present a generalization of the notion of the maximum of a collection of scalars. This generalization is borrowed from the field of order statistics,[45,46] and will allow us to reduce the conservativeness that follows from the use of the standard notion of max function. See also section 3 of Reference 44.

Definition 3 (Generalized max function). Given the vector

$$\mathbf{v} = [v^{(1)}, v^{(2)}, \dots, v^{(N)}]^\top \in \mathbb{R}^N,$$

and the integer r with $1 \leq r \leq N$ we denote

$$\psi(\mathbf{v}, r) = v_+^{(r)},$$

where the vector $\mathbf{v}_+ = [v_+^{(1)}, v_+^{(2)}, \dots, v_+^{(N)}]^\top \in \mathbb{R}^N$ is obtained by rearranging the values of the components of $\mathbf{v}$ in a nonincreasing order. That is,

$$v_+^{(1)} \geq v_+^{(2)} \geq \cdots \geq v_+^{(N-1)} \geq v_+^{(N)}.$$

Clearly, given $\mathbf{v} = [v^{(1)}, \dots, v^{(N)}]^\top \in \mathbb{R}^N$, we have

$$\psi(\mathbf{v}, 1) = \max_{1 \leq i \leq N} v^{(i)}, \quad \psi(\mathbf{v}, N) = \min_{1 \leq i \leq N} v^{(i)}.$$

Furthermore, $\psi(\mathbf{v}, 2)$ denotes the second largest value in $\mathbf{v}$, $\psi(\mathbf{v}, 3)$ the third largest one, etc. We notice that the notation $\psi(\mathbf{v}, r)$ does not need to make explicit N and the number of components of $\mathbf{v}$.

The next theorem provides a way to compute probabilistic performance levels for a family of M controllers. The theorem constitutes a generalization of a similar result, presented in Reference 44 for the particular case $M = 1$. See also the seminal paper[35] for the particularization of the result to the case $r = 1, M = 1$.

Theorem 1. *Given the controllers κ_i, $i = 1, \dots, M$, and the integer $r \geq 1$, suppose that N i.i.d. scenarios*

$$w^{(j)} = \{x^{(j)}(0), \hat{x}^{(j)}(0), d^{(j)}(0), \dots, d^{(j)}(N_{\text{sim}} - 1)\}, \; j = 1, \dots, N,$$

are generated. We denote with $\mathbf{v}_i$, $i = 1, \dots, M$, the vector of performance indicators corresponding to the controller κ_i. That is,

$$\mathbf{v}_i = [\phi_i(w^{(1)}), \phi_i(w^{(2)}), \dots, \phi_i(w^{(N)})]^\top \in \mathbb{R}^N, \; i = 1, \dots, M.$$

Then, with probability no smaller than $1-\delta$, we have a probability of violation

$$\Pr_{\mathcal{W}}\{\phi_i(w) > \psi(\mathbf{v}_i, r)\} \leq \epsilon, \; i = 1, \dots, M,$$

provided that $1 \leq r \leq N$ *and*

$$\sum_{\zeta=0}^{r-1} \binom{N}{\zeta} \epsilon^{\zeta}(1-\epsilon)^{N-\zeta} \leq \frac{\delta}{M}. \tag{10}$$

In addition, (10) is satisfied if:

$$N \geq \frac{1}{\epsilon}\left(r - 1 + \ln\frac{M}{\delta} + \sqrt{2(r-1)\ln\frac{M}{\delta}}\right). \tag{11}$$

Proof. Given the controller κ_i and $\gamma \in \mathbb{R}$, we denote $E_i(\gamma)$ the probability of the event $\phi_i(w) > \gamma$. That is,

$$E_i(\gamma) := \Pr_{\mathcal{W}}\{\phi_i(w) > \gamma\}.$$

We denote probability of asymptotic failure the probability of generating N i.i.d scenarios and obtaining an empirical probabilistic performance level that does not meet the probabilistic specification on probability of violation. We now make use of property 3 in Reference 44, which states that, with probability no smaller than

$$1 - \sum_{\zeta=0}^{r-1} \binom{N}{\zeta} \epsilon^{\zeta}(1-\epsilon)^{N-\zeta},$$

we have

$$E_i(\psi(\mathbf{v}_i, r)) = \Pr_{\mathcal{W}}\{\phi_i(w) > \psi(\mathbf{v}_i, r)\} \leq \epsilon.$$

This means that the probability of asymptotic failure $\Pr_{\mathcal{W}^N}\{\cdot\}$ for N samples $w^{(i)}, \ldots, w^{(N)}$ drawn from $\mathcal{W}$ satisfies

$$\Pr_{\mathcal{W}^N}\{E_i(\psi(\mathbf{v}_i, r)) > \epsilon\} \leq \sum_{\zeta=0}^{r-1} \binom{N}{\zeta} \epsilon^{\zeta}(1-\epsilon)^{N-\zeta} := B(N, \epsilon, r-1).$$

Consider now the probability δ_F that, after drawing N i.i.d. samples $w^{(j)}$, $j = 1, \ldots, N$, one or more of the empirical performance indicator levels

$$\gamma_i = \psi(\mathbf{v}_i, r),\ i = 1, \ldots, M,$$

are not probabilistic performance indicator levels with violation probability ϵ. We have

$$\begin{aligned}
\delta_F &= \Pr_{\mathcal{W}^N}\{\max_{i=1,\ldots,M} E_i(\psi(\mathbf{v}_i, r)) > \epsilon\} \\
&\leq \sum_{i=1}^{M} \Pr_{\mathcal{W}^N}\{E_i(\psi(\mathbf{v}_i, r)) > \epsilon\} \\
&\leq M \sum_{\zeta=0}^{r-1} \binom{N}{\zeta} \epsilon^{\zeta}(1-\epsilon)^{N-\zeta} \leq \delta.
\end{aligned}$$

That is, δ_F is smaller or equal than δ provided that (10) holds. This proves the first claim of the property. The second one follows directly from corollary 1 of Reference 36, which provides an explicit number N of samples that guarantees that a binomial expression $B(N, \epsilon, r-1)$ is smaller than a given constant. ∎

The major advantage of Theorem 1 is that a family of controllers can be evaluated for the same N samples. This is beneficial when the family of controllers can be evaluated in parallel or when drawing samples is expensive, for example,

in experimental setups. The number of required samples for the same probabilistic statement is significantly smaller as when all controllers would be evaluated in a sequential approach as in References 36,47,48.

Remark 2. Given a family of controllers κ_i, $i = 1, \dots, M$, one does not need to compute all the empirical performance indicator levels $\psi(\mathbf{v}_i, r)$, $i = 1, \dots, M$. It is sufficient to find one that meets the desired performance indicator levels. For example, if the performance indicator $\phi_i(w)$ is defined as the average constraint violation along the trajectory (see (8)), then the controller κ_i provides an admissible closed-loop trajectory for w if and only if $\phi_i(w) = 0$. In this case, the empirical performance indicator $\psi(\mathbf{v}_i, r)$ corresponding to N i.i.d. scenarios is equal to 0 if no more than $r - 1$ trajectories are nonadmissible when applying the controller κ_i to the scenarios. If N is chosen according to (10) then Theorem 1 implies that with probability no smaller than $1 - \delta$, all the controllers κ_i, $i = 1, \dots, M$, providing $\psi(\mathbf{v}_i, r) = 0$ are such that

$$\Pr_{\mathcal{W}}(\phi_i(w) > 0) \leq \epsilon.$$

It is also important to remark that the cardinality M of the family of proposed controllers has little effect on the sample complexity N because it appears into a logarithm. See also subsection 4.2 in Reference 36 for other randomized approaches based on a design space of finite cardinality.

Remark 3. Theorem 1 can also be applied in the case when the performance indicators only take binary values. This has been presented in a similar form in Reference 49 and was used for control design problems. See, for example, References 37 and 38.

4 | ROBUST OUTPUT-FEEDBACK NONLINEAR MODEL PREDICTIVE CONTROL

In this work our goal is to design an NMPC scheme that is able to control the uncertain nonlinear system (1) in an output-feedback setting where not all the states can be measured as described by the output Equation (2). While the novel probabilistic validation scheme described in Section 3 can be applied using any controller, we believe that because of the complexity of the general robust output-feedback problem, it is a promising idea to use an approximate robust NMPC scheme that is validated a posteriori using probabilistic validation.

There exist many different robust model predictive control schemes, but there are four important characteristics that differentiate one approach from the other: the choice of cost function, the propagation of the uncertainty, robust constraint satisfaction, and the characterization of feedback information.

The *cost function* can be chosen following a min-max approach, where the worst-case realization of the uncertainty $d(k)$ at each step in the prediction is chosen.[1] Tube-based methods usually choose the cost incurred by the closed-loop system driven by the nominal realization of the uncertainty.[50] Scenario-tree based methods use a weighted sum of a set of discrete scenarios[9] and stochastic MPC schemes[7] make use of, for example, the expectation operator. In this work, we consider a general cost function $V(\mathcal{X}(0), \mathcal{D}; N_\mathrm{p}, \kappa)$ that depends on an initial state set $\mathcal{X}(0)$, the uncertainty set $\mathcal{D}$, the prediction horizon N_p and the control policy κ.

The *propagation of the uncertainty* is one of the key elements of any robust NMPC scheme. A general framework, which is used in this work, is the definition of reachable sets at each sampling time in the prediction based on a current initial condition, the system model, the applied input and the uncertainty set $\mathcal{D}$. The reachable set at sampling time $k + 1$ can be thus denoted as:

$$\mathcal{X}(k+1) = \{f(x(k), u(k), d(k)) : x(k) \in \mathcal{X}(k), d(k) \in \mathcal{D}\}. \tag{12}$$

There are several methods to compute such reachable sets. In the linear case, the consideration of the vertices of the uncertain set and their propagation along the prediction horizon is enough to compute an exact reachable set. In the nonlinear case, linearization techniques[51] or ordinary differential equation bounding techniques[52] can be used to obtain guaranteed over-approximations, which can be then used in robust optimal control schemes. To maintain the notation independent of the method used to obtain an (over-) approximation of the reachable sets at each sampling time, the *bounding* operator denoted as $\diamond f(\cdot)$ is used, which is defined as:

$$\mathcal{X}(k+1) = \diamond f(\mathcal{X}(k), u(k), \mathcal{D}). \tag{13}$$

Another possibility for the propagation of uncertainty is to resort to probabilistic reachable sets as performed in Reference 53.

Robust constraint satisfaction is often one of the main motivations for the use of a robust NMPC approach. It means that the requirements of the closed-loop system in form of input and state constraints should be satisfied for all possible outcomes of the uncertainty and it is usually enforced by embedding the reachable sets (13) into the constraints of an optimization problem.

The *characterization of feedback* that is employed is another key property of any robust MPC scheme. It is well known that considering a sequence of optimal control inputs in the prediction under uncertainty can result in very conservative performance of the closed-loop because it is ignored that new information about the future will be available in the form of measurements and thus future actions can be adapted accordingly. To avoid this conservatism, closed-loop approaches can be used, in which one optimizes over a sequence of control policies κ and can be formulated as the receding horizon solution of the following optimization problem:

$$\mathbb{P}_{\text{ideal}}: \quad \underset{\kappa(\cdot)}{\text{minimize}} \quad V(\mathcal{X}(0), \mathcal{D}; N_{\text{p}}, \kappa), \tag{14a}$$

$$\text{subject to} \quad \mathcal{X}(k+1) = \diamond f(\mathcal{X}(k), \kappa(\mathcal{X}(k)), \mathcal{D}), \quad \text{for } k = 0, \dots, N_{\text{p}} - 1, \tag{14b}$$

$$g_l(\mathcal{X}(k), \kappa(\mathcal{X}(k)), \mathcal{D}) \leq 0, \; l = 1, \dots, n_g, \quad \text{for } k = 0, \dots, N_{\text{p}} - 1, \tag{14c}$$

$$\mathcal{X}(N_{\text{p}}) \subseteq \mathcal{X}_f, \tag{14d}$$

$$\mathcal{X}(0) = \{\hat{x}(0)\} \oplus \mathcal{E}_{\text{est}}, \tag{14e}$$

where the constraints (14c) denote that $g_l(x, \kappa(x), d) \leq 0$ should be satisfied for all $x \in \mathcal{X}(k)$ and for all $d \in \mathcal{D}$.

Solving the ideal robust NMPC problem $\mathbb{P}_{\text{ideal}}$ defined in (14), one obtains a receding horizon policy $\kappa_{\text{ideal}}(\hat{x}(0))$ which is a function of the initial state estimate $\hat{x}(0)$ that has been obtained with a certain estimation error bounded by $\mathcal{E}_{\text{est}}$.

Obtaining an exact solution of $\mathbb{P}_{\text{ideal}}$ is usually intractable mainly because of the bounding operator $\diamond f(\cdot)$ and the general feedback law $\kappa(\cdot)$. There are different alternative solutions to obtain approximations of this problem. A common simplifying assumption is to restrict the search to affine policies on the state or on the disturbances.[11] A different alternative is the use of a scenario tree[6,7,9] in a so-called multistage NMPC approach. A multistage NMPC scheme is based on the representation of the uncertainty via a scenario tree (see Figure 1), which branches at each sampling time. This means that the uncertainty set is approximated by a discrete number of uncertainty realizations:

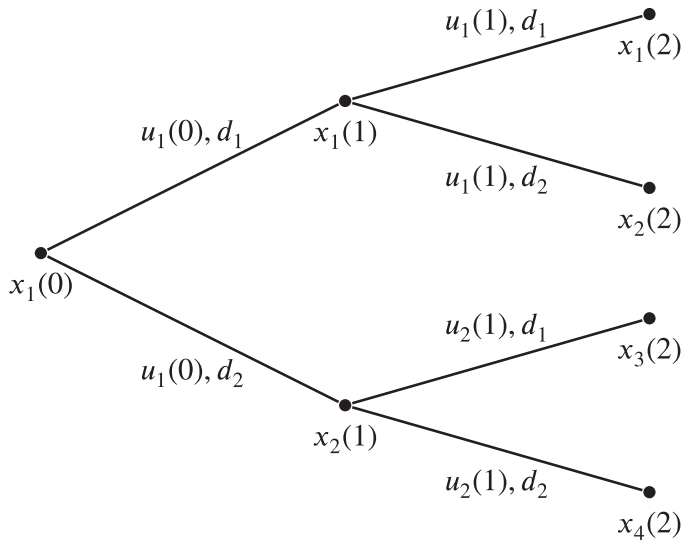
$$\mathcal{D} \approx \tilde{\mathcal{D}} = \{d_1, \dots, d_s\}, \tag{15}$$

where s is the number of possible realizations of the uncertainty that are considered in the tree. The considered realizations mean that each node branches s times which results in s^k nodes at stage k. Using a scenario tree formulation, an approximation of the reachable set can be obtained as the convex hull of the set of all the nodes at a given stage, that is:

$$\mathcal{X}(k) \approx \text{Conv}(\tilde{\mathcal{X}}(k)) = \text{Conv}\left(\bigcup_{i=1}^{s^k} x_i(k)\right), \tag{16}$$

where $\text{Conv}(\cdot)$ denotes the convex hull of a set and $x_i(k)$ denotes the node i of the tree at stage k as depicted in Figure 1. In the linear case with polytopic uncertainty, including the extreme values of the uncertainty in $\tilde{\mathcal{D}}$ guarantees an exact representation of the actual reachable set. In the nonlinear case considered in this paper it is only an approximation and therefore we focus on the point-wise approximation $\tilde{\mathcal{X}}$. Following the same notation, the bounding operator used to propagate the point-wise uncertainty description can be denoted as:

$$\tilde{\mathcal{X}}(k+1) = \diamond f(\tilde{\mathcal{X}}(k), \kappa(\tilde{\mathcal{X}}(k)), \tilde{\mathcal{D}}) \approx \bigcup_{i=1}^{s^k} \bigcup_{j=1}^{s} f(x_i(k), u_i(k), d_j). \tag{17}$$



$$\tilde{\mathcal{X}}(0) = \{x(0)\}; \quad \tilde{\mathcal{X}}(1) = \{x_1(1), x_2(1)\}; \quad \tilde{\mathcal{X}}(2) = \{x_1(2), \dots, x_4(2)\};$$

FIGURE 1 Scenario tree representation

The cost function is usually chosen as a weighted sum of the stage cost for each node in the scenario tree:

$$V(x_1(0), \tilde{\mathcal{D}}; N_\mathrm{p}, \kappa) = \sum_{k=0}^{N_\mathrm{p}-1} \sum_{i=1}^{s^k} \ell(x_i(k), \kappa(x_i(k))) + \sum_{i=1}^{s^{N_\mathrm{p}}} V_f(x_i(N_\mathrm{p})). \tag{18}$$

Introducing (18) and (17) in the ideal formulation of robust NMPC $\mathbb{P}_{\text{ideal}}$ we obtain the optimization problem that should be solved at each sampling time:

$$\mathbb{P}_{\text{ms}}: \quad \underset{\kappa(\cdot)}{\text{minimize}} \quad \sum_{k=0}^{N_\mathrm{p}-1} \sum_{i=1}^{s^k} \ell(x_i(k), \kappa(x_i(k))) + \sum_{i=1}^{s^{N_\mathrm{p}}} V_f(x_i(N_\mathrm{p})), \tag{19a}$$

$$\text{subject to} \quad \tilde{\mathcal{X}}(k+1) = \bigcup_{i=1}^{s^k} \bigcup_{j=1}^{s} f(x_i(k), \kappa(x_i(k)), d_j), \quad \text{for } k = 0, \dots, N_\mathrm{p}-1, \tag{19b}$$

$$g_l(\tilde{\mathcal{X}}(k), \kappa(\tilde{\mathcal{X}}(k)), \tilde{\mathcal{D}}) \le 0, \ l = 1, \dots, n_g, \quad \text{for } k = 0, \dots, N_\mathrm{p}-1, \tag{19c}$$

$$\tilde{\mathcal{X}}(N_\mathrm{p}) \subseteq \mathcal{X}_f, \tag{19d}$$

$$\tilde{\mathcal{X}}(0) = \{\hat{x}(0)\}, \tag{19e}$$

where the constraints (19c) denote that $g_l(x, \kappa(x), d) \le 0$ should be satisfied for all $x \in \tilde{\mathcal{X}}(k)$ and for all $d \in \tilde{\mathcal{D}}$. The optimal solution of (19) is denoted as the multistage NMPC feedback policy κ_{ms}.

To avoid the exponential growth of the tree with the prediction horizon, a usual additional simplifying assumption is to consider that the tree branches only up to a given stage (usually called robust horizon). While this simplification introduces further errors in the approximation of the reachable sets at each stage, it achieves good results in practice.[10] The current estimation error as well as the presence of future estimation errors should be also included in the problem formulation to achieve stability and recursive feasibility guarantees. This can be done in a multistage framework as shown in Reference 54, but additional uncertainties should be included in the scenario tree. To mitigate the exponential growth of the scenario tree with the number of considered uncertainties, we do not consider the estimation error directly in the formulation of the tree. Following ideas from tube-based MPC, these additional errors will be taken into account by means of constraint tightening as explained in Section 5.

5 | DEEP LEARNING-BASED APPROXIMATE ROBUST NMPC

Despite recent advances in algorithms and hardware, solving the simplified output-feedback robust NMPC problem defined in (19) in real time can be challenging. To avoid the need for the real-time solution of nonconvex optimization problems, this work considers the data-based approximation of the implicit feedback law defined by (19) following the same ideas as explicit model predictive control. Approximating an NMPC controller with a neural network was already proposed by Reference 21 back in 1995, where the use of shallow networks (with only one hidden layer) was proposed. This suggestion is based on the universal approximation theory that shows that a neural network with only one layer can approximate any function with any desired accuracy under mild conditions.[55]

5.1 | Deep neural networks

The function approximators chosen for this work are DNNs. This is motivated by recent theoretical results that support the increased representation power of neural networks with several hidden layers as opposed to classical shallow networks.[24] For the approximation of MPC laws via DNNs, good results were obtained in References 22,23,39,40 among other recent works. In the case of linear time-invariant systems, it was shown in Reference 22 that a DNN with a given size can exactly represent the explicit MPC law. The robust NMPC problem (19) is a parametric optimization problem that depends on the current (estimated) state and on the uncertainty values used to define the scenario tree. To perform a deep learning-based approximation, a finite amount of N_s samples $x^{(i)}$ of the state space are chosen and then N_s different optimization problems are solved to obtain the corresponding optimal inputs $\kappa_\mathrm{ms}(x^{(i)})$.

A standard deep feed-forward neural network with fully connected layers is defined as a sequence of layers which determines a function $\mathcal{N} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ of the form

$$\mathcal{N}(x;\theta) = \begin{cases} \alpha_{L+1} \circ \beta_L \circ \alpha_L \circ \cdots \circ \beta_1 \circ \alpha_1(x) & \text{for} \quad L \geq 2, \\ \alpha_{L+1} \circ \beta_1 \circ \alpha_1(x), & \text{for} \quad L = 1, \end{cases} \tag{20}$$

where the input of the network is $x \in \mathbb{R}^{n_x}$ and the output of the network is $u \in \mathbb{R}^{n_u}$. The dimensions of the network are defined by the number of hidden layers L and the number of neurons H per hidden layer, also denoted as the width of the hidden layer, when equal width for all hidden layers is assumed. In contrast to *shallow* neural networks with $L = 1$ hidden layers, *deep* neural networks have $L \geq 2$ hidden layers. The complexity of a neural network can be defined either by the number of weights

$$N_\mathrm{weights} = n_x \cdot (H+1) + (L-1) \cdot (H+1) \cdot H + H \cdot (n_u + 1), \tag{21}$$

or the number of neurons

$$N_\mathrm{neurons} = L \cdot H, \tag{22}$$

that form a given network. The number of weights defines the necessary memory that is needed to store a neural network while the number of neurons determines the maximum possible amount of nonlinear functions present in the approximation. Each hidden layer connects a preceding affine function:

$$\alpha_l(\xi_{l-1}) = W_l \xi_{l-1} + b_l, \tag{23}$$

where $\xi_{l-1} \in \mathbb{R}^H$ is the output of the previous layer and $\xi_0 = x$, with a nonlinear activation function β_l. Common choices for the activation function are rectifier linear units (*ReLU*) and the sigmoid function or the hyperbolic tangent (*tanh*):

$$\beta_l(\alpha_l) = \frac{e^{\alpha_l} - e^{-\alpha_l}}{e^{\alpha_l} + e^{-\alpha_l}}, \tag{24}$$

which will be used throughout this work. The parameters of all layers are summarized in $\lambda = \{\lambda_1, \ldots, \lambda_{L+1}\}$ with

$$\lambda_l = \{W_l, b_l\} \quad \forall l = 1, \ldots, L+1, \tag{25}$$

where W_l are the weights and b_l are the biases describing the corresponding affine functions. The best data-based approximation of the exact multistage NMPC (19) with a neural network for a given training data set $\mathcal{T} = \{(x^{(1)}, \kappa_{\text{ms}}(x^{(1)})), \dots, (x^{(N_\text{s})}, \kappa_{\text{ms}}(x^{(N_\text{s})}))\}$ with N_s elements and fixed dimensions L and H is achieved for:

$$\lambda^* = \arg\min_{\lambda} \quad \frac{1}{N_s} \sum_{i=1}^{N_\text{s}} (\kappa_{\text{ms}}(x^{(i)}) - \mathcal{N}(x^{(i)}; \lambda))^2. \tag{26}$$

The resulting deep learning-based controller is denoted as $\kappa_{\text{dnn}}(x) = \mathcal{N}(x; \lambda^*)$.

5.2 | Constraint tightening

We propose to use a robust NMPC scheme to take explicitly into account the most important uncertainties that affect the system. Still, it is virtually impossible to account for all possible uncertainties and to obtain exact state estimates which in comparison to the ideal, robust NMPC feedback law κ_{ideal} results in two sources of error:

$$||\kappa_{\text{ideal}}(x(k)) - \kappa_{\text{ms}}(\hat{x}(k))|| \leq \epsilon_{\text{est}} + \epsilon_{\text{ms}}, \tag{27}$$

where ϵ_{est} is the estimation and measurement error and ϵ_{ms} is the error caused by the approximation of the reachable set by a set of discrete scenarios. Because solving the multistage NMPC problem (19) online is challenging, our goal is to determine a candidate neural network controller by generating input-output data pairs via the solution of the multistage NMPC problem (19) and approximating its solution via a DNN solving (26). This means that the closed-loop will be controlled using the feedback law κ_{dnn} that approximates the behavior of κ_{ms} which introduces an error ϵ_{approx} on top of those described in (27):

$$\begin{aligned} ||\kappa_{\text{ideal}}(x(k)) - \kappa_{\text{dnn}}(\hat{x}(k))|| &= ||\kappa_{\text{ideal}}(x(k)) - \kappa_{\text{ms}}(\hat{x}(k)) + \kappa_{\text{ms}}(\hat{x}(k)) - \kappa_{\text{dnn}}(\hat{x}(k))|| \\ &\leq ||\kappa_{\text{ideal}}(x(k)) - \kappa_{\text{ms}}(\hat{x}(k))|| + ||\kappa_{\text{ms}}(\hat{x}(k)) - \kappa_{\text{dnn}}(\hat{x}(k))|| \\ &\leq \epsilon_{\text{est}} + \epsilon_{\text{ms}} + \epsilon_{\text{approx}}. \end{aligned} \tag{28}$$

Finding upper-bounds for each one of the errors to apply traditional robust NMPC schemes is not possible for the general nonlinear case.

To counteract the possible errors ϵ_{est}, ϵ_{ms}, and ϵ_{approx}, and following ideas from tube-based MPC, an additional backoff η is used to tighten the original constraints of the robust NMPC problem that is solved to generate input–output data for training:

$$\begin{aligned} & \underset{\kappa(\cdot)}{\text{minimize}} && \sum_{k=0}^{N_\text{p}-1} \sum_{i=1}^{s^k} \ell(x_i(k), \kappa(x_i(k))) + \sum_{i=1}^{s^{N_\text{p}}} V_f(x_i(N_\text{p})), && \text{(29a)} \\ \mathbb{P}_{\text{ms},\eta}: \quad & \text{subject to} && \tilde{\mathcal{X}}(k+1) = \bigcup_{i=1}^{s^k} \bigcup_{j=1}^{s} f(x_i(k), \kappa(x_i(k)), d_j), \quad \text{for } k = 0, \dots, N_\text{p}-1, && \text{(29b)} \\ & && g_l(\tilde{\mathcal{X}}(k), \kappa(\tilde{\mathcal{X}}(k)), \tilde{\mathcal{D}}) \leq -\eta, \; l = 1, \dots, n_g, \quad \text{for } k = 0, \dots, N_\text{p}-1, && \text{(29c)} \\ & && \tilde{\mathcal{X}}(0) = \{\hat{x}(0)\}. && \text{(29d)} \end{aligned}$$

Solving (29) online would lead to the feedback controller $\kappa_{\text{ms}}(\hat{x}, \eta)$. We are however interested in the proposed approximate robust NMPC $\kappa_{\text{dnn}}(\hat{x}, \eta)$ that is obtained by training a DNN via (26) based on input–output data generated by solving (29) for many different initial conditions. Introducing a backoff η does not guarantee in general that the closed-loop satisfies the constraints. For this reason, closed-loop constraint satisfaction is also not ensured a priori with a terminal set. The probabilistic design scheme presented in the previous sections is employed to select the backoff parameter η. The proposed methodology provides probabilistic guarantees on the performance indicators of the closed-loop uncertain system.

6 | CASE STUDY

We investigate the optimal control of a kite which is used to tow a boat. The stable and safe operation of the kite is challenging due to the highly nonlinear system dynamics, uncertain parameters, strong influence from disturbances like wind speed and noisy measurements. To develop optimal control schemes of a kite system, typically models with moderate complexity such as References 31,56 are considered because of the required short sampling times. Although for our proposed strategy, also a high-fidelity model could be considered since the majority of the computational load is shifted offline, we consider a popular three-state model as presented in Reference 57 to facilitate the comparison of the results with previous works. We derive an approximate deep learning-based controller from a robust NMPC formulation, which enables a very fast and easy evaluation of the controller even on computationally limited hardware. The idea of learning a controller for a kite has already been exploited in Reference 58, where polynomial basis functions were used to approximate the behavior of a human pilot based on measurements.

6.1 | Kite model

In the context of NMPC, we focus on the model presented in Reference 57 which consists of three states, one control input and two uncertain parameters. The state evolution is given by the ordinary differential equations of the three angles θ_{kite}, ϕ_{kite}, and ψ_{kite} of the spherical coordinate system describing the position of the kite:

$$\dot{\theta}_{\text{kite}} = \frac{v_{\text{a}}}{L_{\text{T}}}\left(\cos\psi_{\text{kite}} - \frac{\tan\theta_{\text{kite}}}{E}\right), \tag{30a}$$

$$\dot{\phi}_{\text{kite}} = -\frac{v_{\text{a}}}{L_{\text{T}}\sin\theta_{\text{kite}}}\sin\psi_{\text{kite}}, \tag{30b}$$

$$\dot{\psi}_{\text{kite}} = \frac{v_{\text{a}}}{L_{\text{T}}}\tilde{u} + \dot{\phi}_{\text{kite}}\cos\theta_{\text{kite}}, \tag{30c}$$

where

$$v_{\text{a}} = v_0 E\cos\theta_{\text{kite}}, \tag{30d}$$

$$E = E_0 - \tilde{c}\tilde{u}^2. \tag{30e}$$

The angle between wind and tether (zenith angle) is described by θ_{kite}, the angle between the vertical and the plane is denoted by ϕ_{kite} and ψ_{kite} represents the orientation of the kite. The three states can be manipulated via the steering deflection $\tilde{u}$. The area of the kite is denoted as A, and L_{T} is the length of the tether. The effect of the wind is denoted as v_{a}, which is strongly influenced by the wind speed v_0, the first uncertain parameter. The glide ratio E is dependent on the base glide ratio E_0, the second uncertain parameter, and the magnitude of the steering deflection $\tilde{u}$.[59] The parameters of the kite model are shown in the upper part of Table 1.

6.2 | Wind model

The wind speed v_0 is considered as a single uncertainty in (29), but the realizations of the values are computed based on a simulation model. The underlying wind model was presented in Reference 60 and is described by:

$$v_0 = v_{\text{m}} + \bar{v}_{\text{N}} + \sigma_v c_v p_v, \tag{31a}$$

where

$$\sigma_v = k_{\sigma_v} v_{\text{m}}, \tag{31b}$$

$$\bar{v}_{\text{N}} = -\sigma_v/(2v_{\text{m}}), \tag{31c}$$

TABLE 1 Overview of the model states and parameters and as which variable they are considered in (29)

	Symbol	Type	Values/constraints	Units	Variable
kite model	θ_{kite}	State	$[0, \frac{\pi}{2}]$	rad	x
	ϕ_{kite}	State	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	rad	x
	ψ_{kite}	State	$[0, 2\pi]$	rad	x
	$\tilde{u}$	Control input	$[-10, 10]$	N	u
	$\tilde{c}$	Known parameter	0.028	—	—
	β	Known parameter	0	rad	—
	ρ	Known parameter	1	kg m^{-3}	—
	h_{min}	Known parameter	100	m	—
	E_0	Uncertain parameter	unif(4, 6)	—	d
wind model	p_v	State	—	s	d via v_0
	k_{σ_v}	Known parameter	0.14	—	
	L_v	Known parameter	100	m	
	T_v	Known parameter	0.15	s	
	v_{m}	Uncertain parameter	unif(7, 9)	m s^{-1}	
	w_{tb}	Uncertain parameter	normal(0, 0.25)	—	

$$\tau_{\text{F}} = L_v / v_{\text{m}}, \tag{31d}$$

$$K_{\text{F}} = \sqrt{1.49 \tau_{\text{F}} / T_v}, \tag{31e}$$

$$c_v = K_F / \tau_{\text{F}}, \tag{31f}$$

$$\dot{p}_v = -p_v / \tau_{\text{F}} + w_{\text{tb}}, \tag{31g}$$

when the wind shear is neglected. The term w_{m} gives the current average wind speed, w_{tb} is introduced as a white noise generator to model the short term turbulence and $p_v(0) = \text{normal}(0, 0.25)$ is the initial state of the turbulence, where $x_{\text{normal}} = \text{normal}(\mu_{\text{normal}}, \sigma_{\text{normal}})$ denotes that the variable x_{normal} follows a normal distribution with mean μ_{normal} and standard deviation σ_{normal}. In a similar manner, $x_{\text{unif}} = \text{unif}(a_{\text{unif}}, b_{\text{unif}})$ means that the variable x_{unif} follows a uniform distribution between a_{unif} and b_{unif}. An overview of the parameters for the wind model is given in the lower part of Table 1. For further details on modeling assumptions and the choice of parameters the reader is referred to Reference 60.

6.3 | Extended Kalman filter

We assume that we can measure the two angles θ_{kite} and ϕ_{kite} and the wind speed v_0. An Extended Kalman filter (EKF) is used to obtain an estimate of the augmented state $x_{\text{aug}} = [\theta_{\text{kite}}, \phi_{\text{kite}}, \psi_{\text{kite}}, E_0, v_0]^T$ in each control instance from the measurements:

$$y(x_{\text{aug}}) = [\theta_{\text{kite}} + w_{\theta_{\text{kite}}}, \phi_{\text{kite}} + w_{\phi_{\text{kite}}}, v_0 + w_{v_0}]^T, \tag{32}$$

with the zero-mean gaussian noises $w_{\theta_{\text{kite}}} = \text{normal}(0, 0.01)$, $w_{\phi_{\text{kite}}} = \text{normal}(0, 0.01)$ and $w_{v_0} = \text{normal}(0, 0.05)$. The augmented state is initialized for all simulations as $x_{\text{aug}}(0) = [\theta_{\text{kite}}(0) \cdot \delta_{\theta_{\text{kite}}}, \phi_{\text{kite}}(0) \cdot \delta_{\phi_{\text{kite}}}, \psi_{\text{kite}}(0) \cdot \delta_{\psi_{\text{kite}}}, E_0 \cdot \delta_{E_0}, v_0(0) \cdot \delta_{v_0}]^T$, where all noises $\delta_{(\cdot)}$ are drawn from normal(1, 0.05). Neither the estimates of the uncertain parameters nor the measurement of the wind speed are used in the computations of the controller, because their possible values are considered in the robust NMPC approach. The initial covariance matrix is given by $P_{\text{EKF}} = \text{diag}([1 \times 10^{-2}, 1 \times 10^{-2}, 1 \times 10^{-2}, 1.0, 2 \times 10^{-1}]$,

the estimate of the process noise by $Q_{\mathrm{EKF}} = \mathrm{diag}([1 \times 10^{-5}, 1 \times 10^{-5}, 1 \times 10^{-4}, 1 \times 10^{-5}, 3 \times 10^{-3}])$, the measurement noise matrix by $R_{\mathrm{EKF}} = \mathrm{diag}([1 \times 10^{-2}, 1 \times 10^{-2}, 5 \times 10^{-2}])$ and the observer has a sampling time of $t_{\mathrm{EKF}} = 0.05\,\mathrm{s}$.

6.4 | Objective, constraints, and control settings

The goal of the control is to maximize the thrust of the tether defined by:

$$T_{\mathrm{F}} = \frac{1}{2}\rho v_0^2 A \cos^2\theta_{\mathrm{kite}}(E+1)\sqrt{E^2+1}\cdot(\cos\theta_{\mathrm{kite}}\cos\beta + \sin\theta_{\mathrm{kite}}\sin\beta\sin\phi_{\mathrm{kite}}), \tag{33}$$

while maintaining a smooth control performance and satisfying the constraints. The desired behavior is enforced in the stage cost:

$$\ell(x,u) = -w_{\mathrm{F}}T_{\mathrm{F}} + w_u(\tilde{u} - \tilde{u}_{\mathrm{prev}})^2, \tag{34}$$

where $w_{\mathrm{F}} = 1e-4$ and $w_u = 0.5$ are weights and $\tilde{u}_{\mathrm{prev}}$ is the previous control input and sampling time of the controller $t_{\mathrm{c}} = 0.15\,\mathrm{s}$ with a prediction horizon of $N_{\mathrm{P}} = 40$ steps.

Throughout the operation of the kite it has to be ensured that the height of the kite:

$$h(x) = L_{\mathrm{T}}\sin\theta_{\mathrm{kite}}\cos\phi_{\mathrm{kite}}, \tag{35}$$

never falls below $h_{\mathrm{min}} = 100\,\mathrm{m}$. The height constraint is a critical constraint of the control task since the best performance is obtained when the kite is operated close to h_{min}. Because of the error ϵ_{ms} caused by the approximation of the reachable sets in the multistage NMPC formulation, the errors due to a deep learning-based approximation $\epsilon_{\mathrm{approx}}$ as well as the errors related to estimation and measurement errors ϵ_{est}, constraint satisfaction can not be guaranteed. To cover the effect of the errors, the backoff parameter $\eta > 0$ m is introduced and the height constraint:

$$h(x) > h_{\mathrm{min}} + \eta, \tag{36}$$

is formulated as a soft constraint to avoid numerical problems.

To build a multistage NMPC controller, we consider the combinations of the extreme values of the base glide ratio $E_0 \in [4, 6]$ and the wind speed $v_0 \in [6\ \mathrm{m\ s^{-1}}, 10\ \mathrm{m\ s^{-1}}]$ and a one-step robust horizon resulting in a total of four scenarios. The interval for the wind speed is obtained by summarizing the possible effects of the uncertain wind model parameters v_{m}, $p_v(0)$ and w_{tb} into the single uncertain variable v_0.

6.5 | Simulation

For the simulation of the system, it is assumed that the uncertain parameters E_0 and w_{m} are constant over a given closed-loop simulation and that w_{tb} changes every $t_{\mathrm{c}} = 0.15\,\mathrm{s}$. The values of the uncertain parameters are drawn from the probability distribution described in Table 1.

7 | RESULTS

The proposed method for the probabilistic verification of controllers is analyzed for the towing kite case study. The baseline controller for our investigations, which is also used for the training data generation for the corresponding approximate neural network controller $\kappa_{\mathrm{dnn},\eta}$, is the exact multistage NMPC controller $\kappa_{\mathrm{ms}}(\hat{x},\eta)$ (29) that derives its initial state estimate $\hat{x}$ from the EKF based on the current measurement (32). This means that the baseline controller is affected by the estimation error ϵ_{est} and the error ϵ_{ms} caused by the discrete representation of the uncertainties in the scenario tree and hence no formal guarantees on constraint satisfaction can be given. To avoid numerical problems for the solver in case of violations, the critical height constraint is implemented as a soft constraint.

TABLE 2 Overview of the the parameter sampling via uniform distribution, normal distribution, beta(2,5) and pareto(5) distribution and results of evaluating the approximate controller κ_{dnn} with $\eta = 4$ m for 1388 randomly drawn scenarios w. The measurement noise $w_{\text{meas}} = [w_{\theta_{\text{kite}}}, w_{\phi_{\text{kite}}}, w_{\psi_{\text{kite}}}]^T$, the initial state of the turbulence $p_v(0) = \text{normal}(0, 0.25)$, the white noise modeling the short-term turbulences $w_{\text{tb}} = \text{normal}(0, 0.25)$ and the initialization of the estimation vector $x_{\text{aug}}(0)$ is for all scenario spaces identical

Distribution	$\theta_{\text{kite}}(0)$ (°)	$\phi_{\text{kite}}(0)$ (°)	$\psi_{\text{kite}}(0)$ (°)	E_0 (-)	v_{m} (m s^{-1})	Feasible trajectories	$\psi(\text{v}, 4)$ (m)
Uniform	(28.0, 30.0)	(−10.0, 10.0)	(−2.0, 2.0)	(4.0, 6.0)	(7.0, 9.0)	1385/1388	−0.316
Normal	(29.0, 0.35)	(0.0, 3.5)	(0.0, 0.7)	(5.0, 0.35)	(8.0, 0.35)	1387/1388	−0.739
Beta	(2.0, 28.0)	(20.0, −10.0)	(4.0, −2.0)	(2.0, 4.0)	(2.0, 7.0)	1387/1388	−0.556
Pareto	(5.0, 28.0)	(5.0, −10.0)	(5.0, 2.0)	(5.0, 4.5)	(5.0, 7.5)	1385/1388	−0.037

7.1 | Learning an approximate output-feedback robust NMPC controller

The training process of a neural network is determined by the quality of the data and the chosen hyperparameters like activation function of the hidden layers and network size (21), (22). In the following, we discuss how the training data can be generated in a way that reduces the number of samples that are needed to achieve a satisfactory approximation in comparison to a random sampling. For the training of the neural networks we used the toolbox Keras[61] with the backend TensorFlow[62] and Adam[63] as the optimization algorithm. The weights were initialized based on the glorot uniform distribution[64] and the biases were set to zero. All considered networks use hyperbolic tangent (tanh) as activation function in the hidden layers and a linear output layer. As the focus of this work is the verification of a given approximate controller and not the training process or the choice of the optimal network architecture, we refrained from applying methods such as Bayesian Optimization to obtain an optimal structure of the underlying network.[65]

We consider two training data sets $\mathcal{T}_{\text{feas}}$ and $\mathcal{T}_{\text{opt}}$, and two validation data sets $\mathcal{V}_{\text{feas}}$ and $\mathcal{V}_{\text{opt}}$. Each dataset contains samples $(x^{(i)}, \kappa_{\text{ms}}(x^{(i)}))$ corresponding to the numerical solution of the multistage problem (29) at state $x^{(i)}$. The subscript *opt* indicates that the data was derived from optimal closed-loop trajectories, for example, $\mathcal{T}_{\text{opt}} = \{(x^{(j)}, \kappa_{\text{ms}}(x^{(j)})), \dots, (x^{(N_{\text{sim}} \cdot N_{\text{traj}})}, \kappa_{\text{ms}}(x^{(N_{\text{sim}} \cdot N_{\text{traj}})}))\}$ is composed of N_{traj} state-feedback closed-loop simulations of length N_{sim} using the exact multistage NMPC (29) under the dynamics presented in (30), where the uncertain parameters of the model and the initial conditions are drawn according to the distributions given in Table 1 and first row of Table 2, respectively. The subscript *feas* means that the data was obtained at randomly sampled states, for example, $\mathcal{T}_{\text{feas}} = \{(x^{(i)}, \kappa_{\text{ms}}(x^{(i)})), \dots, (x^{(N_{\text{s}})}, \kappa_{\text{ms}}(x^{(N_{\text{s}})}))\}$ is obtained by sampling $x^{(i)}$ uniformly from the feasible state space and solving (29). Since the training data is generated based on simulations, the application of output-feedback via EKF is not necessary and not used for the data generation. Each trajectory consists of $N_{\text{sim}} = 400$ simulation steps which results in a total simulation time of $t_{\text{sim}} = N_{\text{sim}} \cdot t_{\text{c}} = 60$ s. For $\mathcal{T}_{\text{opt}}$, $N_{\text{traj}} = 200$ closed loop runs were simulated leading to $N_{\text{traj}} \cdot N_{\text{sim}} = 80,000$ data pairs and for the validation $N_{\text{traj}} = 50$ simulations were rolled out, resulting in $N_{\text{traj}} \cdot N_{\text{sim}} = 20,000$ samples in $\mathcal{V}_{\text{opt}}$. For the datasets $\mathcal{T}_{\text{feas}}$ and $\mathcal{V}_{\text{feas}}$, $N_{\text{s}} = 80,000$ and $N_{\text{s}} = 20,000$ random samples were drawn, respectively.

For the following investigations, we trained five deep networks with $L = 6$ layers and $H = 30$ neurons per layer on each training set and evaluated all five obtained networks on each validation set. By averaging the results over five networks the impact of the stochastic learning is reduced. Training a DNN with the data pairs $\mathcal{T}_{\text{opt}}$ leads to a significantly smaller average mean squared error (MSE) when compared to the training using the training data $\mathcal{T}_{\text{feas}}$, as Figure 2 shows, because the sampled space of optimal trajectories is smaller in comparison to the feasible space. To investigate the impact of the training data set on the actual performance, the networks are tested on the validation sets $\mathcal{V}_{\text{feas}}$ and $\mathcal{V}_{\text{opt}}$. The networks trained on $\mathcal{T}_{\text{feas}}$ perform better when evaluated on whole feasible space with an average MSE of 0.0048 in comparison to the networks trained on $\mathcal{T}_{\text{opt}}$ with an average MSE of 0.2105. But when the networks are evaluated on the space of optimal closed-loop trajectories via $\mathcal{V}_{\text{opt}}$, the networks trained on $\mathcal{T}_{\text{opt}}$ have a significantly smaller average MSE of 0.0087 than networks trained on $\mathcal{T}_{\text{feas}}$ with an average MSE of 0.1642. The fact that controllers trained on $\mathcal{T}_{\text{opt}}$ do not cover the whole feasible space is not critical since the learning-based controller will only operate in the neighborhood of optimal trajectories where a close approximation of the exact multistage NMPC is achieved. Additionally, the controllers will be probabilistically validated, and this validation is completely independent of the data used for training. Our experience shows that extracting training data from closed loop trajectories can significantly reduce the necessary number of training samples N_{s} and the dimensions L and H of the neural network to obtain a desired approximation error of the deep learning-based controller in the critical domain.

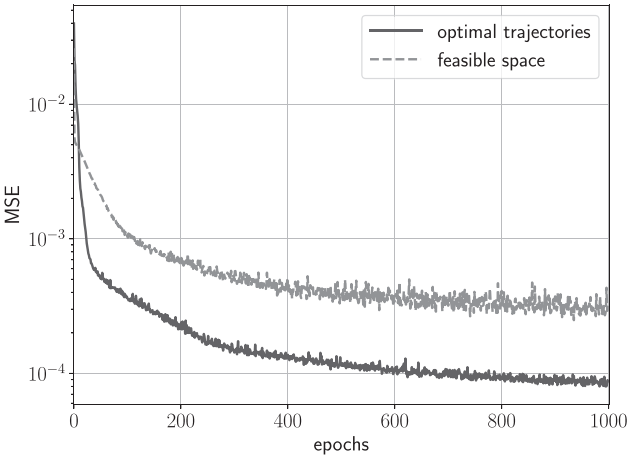


FIGURE 2 Mean squared error obtained when training a deep neural network using the space of optimal trajectories $\mathcal{T}_{\text{opt}}$ or the full feasible space $\mathcal{T}_{\text{feas}}$ as training data

For all the results presented in the remainder of the paper, we use DNNs with $L = 6$ and $H = 30$ which were trained on the space of optimal trajectories $\mathcal{T}_{\text{opt}}$ due to the observed superior approximation quality in the crucial regions of the state space.

7.2 | Verification of a deep learning-based embedded output-feedback robust NMPC

Because of the approximation errors, measurement and estimation errors as well as the errors derived from the multistage formulation, we refrain from a worst-case deterministic analysis and resort to the probabilistic verification scheme based on closed-loop trajectories presented in Section 3. We consider four possible values for the backoff hyper-parameter η, that is, $\eta \in \{0, 2, 4, 6\ \text{m}\}$. This leads to a family of $M = 4$ deep learning-based approximate controllers $\kappa_{\text{dnn},\eta}$. Each one of these controllers was obtained training on data sets $\mathcal{T}_{\text{opt},\eta}$ containing 80,000 data pairs each. The resulting controllers were analyzed for N i.i.d. scenarios $w^{(j)}$, $j = 1, \dots, N$ corresponding to N closed-loop simulations under the dynamics presented in (30), where the uncertain parameters of the model and the initial conditions are drawn according to the distributions given in Table 1 and first row of Table 2, respectively. Since the height constraint (36) is the most critical constraint, we define the performance indicator:

$$\phi(w; N_{\text{sim}}, \kappa_{\text{dnn},\eta}) = \max_{j=0,\dots,N_{\text{sim}}} (h_{\min} - h(x(j, w))), \tag{37}$$

where $x(j, w)$ is the state trajectory at sampling time j caused by scenario w using controller $\kappa_{\text{dnn},\eta}$. The performance indicator (37) extracts the largest violation of the minimum height $h_{\min}$, if a violation occurs, or the closest value to $h_{\min}$ throughout one scenario. Each scenario has a duration of 60 s which means $N_{\text{sim}} = 400$. To consider a controller probabilistically safe, we require that the probabilistic performance indicator satisfies:

$$\Pr_{\mathcal{W}}(\phi(w; N_{\text{sim}}, \kappa_{\text{dnn},\eta}) > 0) \leq \epsilon, \tag{38}$$

with confidence $1 - \delta$ for a randomly sampled scenario w according to $\Pr_{\mathcal{W}}$. Following the notation of Theorem 1, the performance indicators corresponding to backoff parameters $\{0, 2, 4, 6\ \text{m}\}$ are collected into vectors $\{\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}, \mathbf{v_4}\}$, respectively. We consider a value of the discarding parameter $r = 4$. That is, a controller is probabilistically validated if no more than 3 simulations violate the height constraint. For these specifications ($\epsilon = 0.02$, $\delta = 1 \times 10^{-6}$, and $r = 4$), $N = 1388$ samples are required (see (11)). The family of controllers was evaluated for 1388 i.i.d. scenarios $w^{(j)}$ and the results are summarized in Table 3. If no backoff is considered ($\eta = 0$ m) the exact multistage NMPC operates often at the constraint bound which leads to small violations of the height constraint as ϵ_{ms} and ϵ_{est} are ignored. The corresponding approximate controller $\kappa_{\text{dnn},0}$ is additionally affected by ϵ_{approx} (28) which leads to violations of the height constraint in more

TABLE 3 Comparison of the members of a deep learning-based based family of controllers defined by $M = 4$ different choices of the backoff parameter $\eta = \{0\text{ m}, 2\text{ m}, 4\text{ m}, 6\text{ m}\}$. The parameters for the probabilistic safety certificate were chosen to $\epsilon = 0.02$ and $\delta = 1 \times 10^{-6}$. The necessary number of samples for 3 discarded worst-case runs ($r = 4$) is $N = 1388$ and computed via (11)

Controller	$\kappa_{\text{dnn},0}$	$\kappa_{\text{dnn},2}$	$\kappa_{\text{dnn},4}$	$\kappa_{\text{dnn},6}$
Feasible trajectories	660/1388	1380/1388	1385/1388	1387/1388
$\psi(\mathbf{v}, 4)$ (m)	1.682	0.273	−0.316	−1.818
T_{F} (avg.) (kN)	227.516	225.997	224.185	222.179
Probabilistically safe	No	No	Yes	Yes

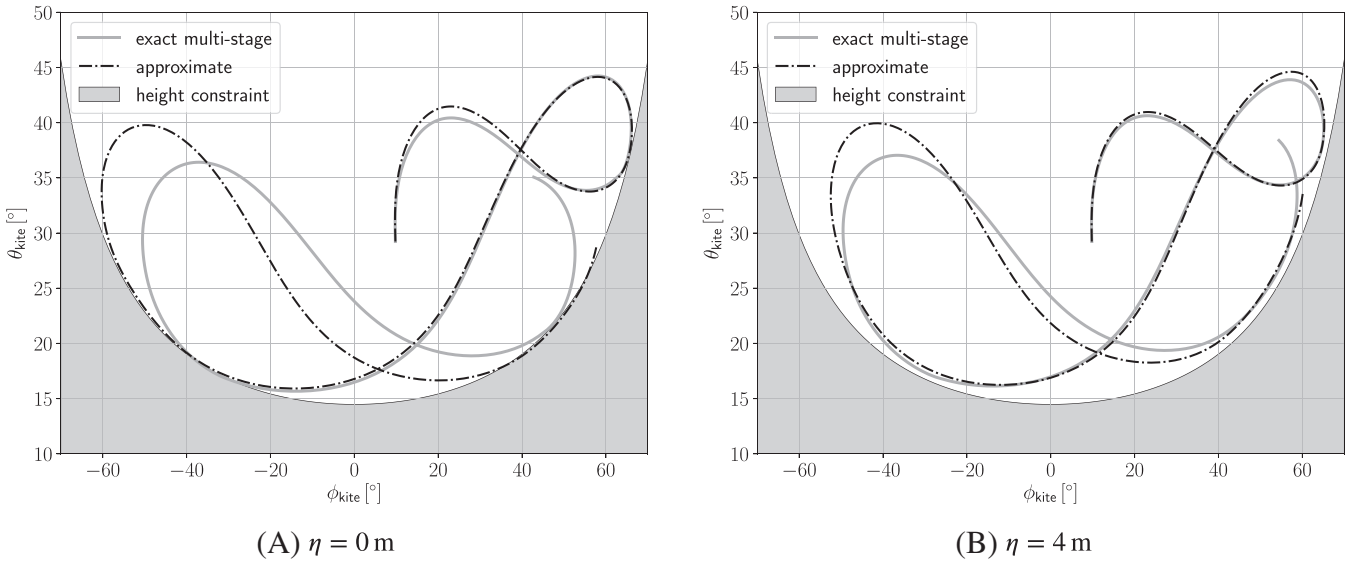


FIGURE 3 Comparison of the exact multistage NMPC and its deep learning-based approximation for one sample w and two different choices of the back-off parameter η. By choosing no backoff ($\eta = 0$ m) the kite is often operated at the bound which leads to frequent constraint violations due to the estimation and uncertainty discretization errors ϵ_{est} and ϵ_{ms}. The constraint violations by the neural network controller $\kappa_{\text{dnn},0}$ are more significant as it is additionally affected by the approximation error ϵ_{approx} (A). By introducing a backoff of $\eta = 4$ m the impact of the three error sources is mitigated which enables the probabilistically safe operation of the kite with both controllers (B)

than half of the scenarios when applied. Exemplary trajectories for the exact multistage NMPC and the approximate controller for one scenario are visualized in Figure 3(A). By considering $\eta = 2$ m, the amount of violations can be significantly reduced to 8 scenarios, which shows the importance of the backoff parameter. However, the performance of $\kappa_{\text{dnn},2}$ is not considered probabilistically safe because after discarding the allowed number of worst-case simulation runs, we get $\psi(\mathbf{v_2}, 4) = 0.273\text{ m} > 0\text{ m}$. With larger backoffs $\eta = 4$ m and $\eta = 6$ m, we obtain two probabilistically safe controllers with performance indicator levels $\psi(\mathbf{v_3}, 4) = -0.316$ m and $\psi(\mathbf{v_4}, 4) = -1.818$ m, respectively. For the same scenario w as in Figure 3(A), the trajectories of exact multi-stage NMPC with $\eta = 4$ m and $\kappa_{\text{dnn},4}$ are depicted in Figure 3(B). Due to the backoff, the kite is keeping a safety distance to the constraint bound and the impact of ϵ_{ms} and ϵ_{est} does not directly lead to constraint violations. Also the trajectory of the approximate controller does not violate the trajectories despite being affected by the additional approximation error ϵ_{approx}. The preferred deep learning-based controller is $\kappa_{\text{dnn},4}$ due to the higher average tether thrust T_{F} provided. By introducing a performance indicator level for the average thrust per simulation run:

$$\phi_{T_{\text{F}}}(w; N_{\text{sim}}, \kappa) = \frac{1}{N_{\text{sim}}} \sum_{k=0}^{N_{\text{sim-1}}} - T_{\text{F}}(k), \tag{39}$$

it is possible to obtain probabilistic statements about the performance in the same fashion as for violation of the height constraint. Using the parameters $\delta = 1 \times 10^{-6}$, $\epsilon = 0.02$, $M = 4$ and $r = 4$ we obtain, for the controller $\kappa_{\text{dnn},4}$, that with confidence $1 - \delta$, the probability that the average thrust for a simulation run of 60 s duration is lower than 111.346 kN is not larger than $\epsilon = 0.02$. A smaller number of samples is required if the discarding parameter r is set equal to 1. However, this leads to more conservative results because violations of the height constraint occur throughout the closed loop simulations used for verification. This is even worse when the performance index is a binary function determining if the trajectories are admissible or not. In this case, the obtained results are often not informative because in a binary setting with $r = 1$, a single violated trajectory out of N determines that the controller does not meet the probabilistic constraints. Larger values for r, along with the consideration of nonbinary violation performance indexes, provide more informative results. One more advantage of the proposed probabilistic method is that a family of controllers can be evaluated in parallel in the closed loop for the same set of sampled scenarios. This can reduce the verification effort significantly, if drawing samples w from $\mathcal{W}$ is costly or the closed-loop experiments have a long duration.

7.3 | Robustness of the probabilistic validation scheme

All obtained probabilistic guarantees are only valid if the assumptions about the probability density functions (PDFs) of $\mathcal{W}$ from which the scenarios w are drawn are correct. For the verification, the N closed-loop simulations were generated using the dynamics presented in (30) and the different $\kappa_{\text{dnn},\eta}$ controllers. The uncertain parameters of the model and the initial conditions were drawn according to the distributions given in Table 1 and first row of Table 2, respectively.

To test the robustness of the probabilistic statements w.r.t. to wrong assumptions about the PDFs, the performance of the approximate controllers $\kappa_{\text{dnn},\eta}$ is evaluated using not the distribution of the first row of Table 2, but the second (normal distribution), the third (beta distribution) and the fourth one (pareto distribution). The first parameter in the description of the beta distribution is the scaling and the second parameter is the offset, for example,

$$\theta_0 = 2.0 \cdot \text{beta}(2, 5) + 28.0.$$

The long-tailed pareto distribution is also described with two parameters, where the first one is the tail index and the second one is the scaling, for example,

$$\theta_0 = \text{pareto}(5.0) + 28.0.$$

The possible extreme values of samples from the space of beta distributions $\mathcal{W}_{\text{beta}}$ are identical with those when sampling from the space of uniform distributions $\mathcal{W}$, see Figure 4. In case of sampling from $\mathcal{W}_{\text{normal}}$ and $\mathcal{W}_{\text{pareto}}$, which have infinite support, the occurrence of values in w which exceed the bounds of the scenarios considered in the robust MPC formulation and the verification scenarios is likely, which highlights the importance of including the discarding parameter r. The four different considered PDFs including the bounds applied in the NMPC formulation are shown in Figure 4 for the example base glade ratio E_0.

The results corresponding to drawing 1388 scenarios from each of the distributions $\mathcal{W}_{\text{normal}}$, $\mathcal{W}_{\text{beta}}$, and $\mathcal{W}_{\text{pareto}}$, and evaluating the approximate controller $\kappa_{\text{dnn},4}$ are given in Table 2. For the case of $\mathcal{W}_{\text{normal}}$ and $\mathcal{W}_{\text{beta}}$ one simulation run violates the height constraint, while three simulation runs violate the constraints for $\mathcal{W}_{\text{pareto}}$. This means that the probabilistic requirements for the safety certificate ($\epsilon = 0.02$, $\delta = 1 \times 10^{-6}$, $M = 4$, $r = 4$) hold for all alternative choices of distributions. This shows that neither the training of the network nor the verification approach fails catastrophically when the statistical assumptions are not exactly fulfilled.

7.4 | Embedded implementation

One of the major advantages of learning the complex optimal control law via DNNs is the reduction of computational load and the fast evaluation. The computation of the control input is reduced from solving an optimization problem to one matrix-vector multiplication per layer and the evaluation of the tanh-function. This enables the implementation of

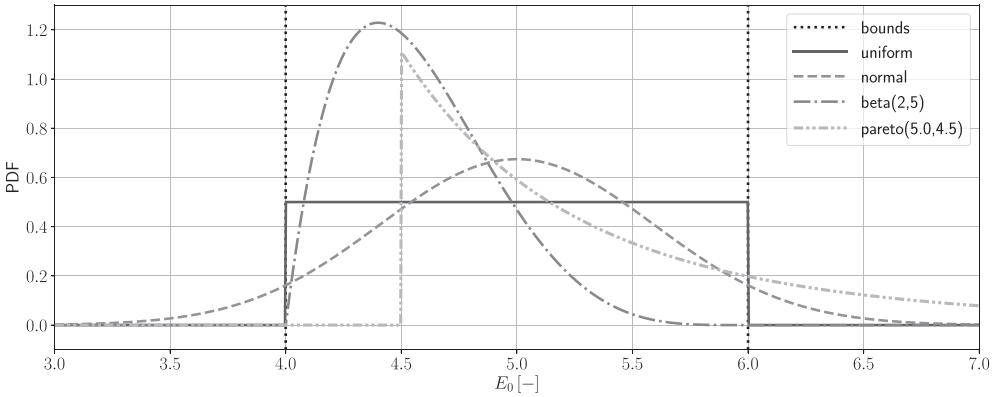


FIGURE 4 Four different considerations of the uncertain parameter base glide ratio E_0 and the considered bounds in the NMPC formulation. The normal and the pareto distribution exceed the considered bounds

a probabilistically validated, approximate robust nonlinear model predictive control on limited hardware such as microcontrollers or FPGAs. We deployed the approximate controller on a low-cost microcontroller (ARM Cortex-M3 32-bit) running with a frequency of 89 MHz with 96 kB RAM. The memory footprint of both the EKF and the neural network that describes the approximate robust NMPC is only 67.0 kB of the 512 kB flash memory. The average time needed to evaluate the neural network was 32.1 ms (max. evaluation time: 33.0 ms) and the average evaluation time for one EKF step was 28.3 ms (max. evaluation time: 30.0 ms), which shows that the proposed controller is real-time capable with a worst-case evaluation time of 63.0 ms. We analyzed the impact of the evaluation time on the safety by simulating the kite for the same 1388 scenarios considered in Table 3 drawn from the uniform distribution, but by applying the computed control inputs with a delay of $t_{\text{delay}} = 65.0$ ms, emulating a hardware-in-the-loop setting. We rounded the time delay up to 65.0 ms to account for possible time measurement errors. To deal with the additional error ϵ_{delay} caused by t_{delay}, we chose $\eta = 6$ m. Out of the 1388 simulated scenarios, 1374 were free of violations. This means the controller violates the height constraint in 0.86% of the cases, which is less than the probabilistically guarantee $\epsilon = 0.02$ chosen in Section 7.2, despite of the additional errors induced through the delay. If the performance needs to be further improved for the hardware-in-the-loop setting, training data for the controller could be generated where the deterministic t_{delay} is incorporated in the NMPC formulation. This is an additional advantage of the proposed approach, because the evaluation time of a given neural network is deterministic and can be known in advance. Additional measures to counteract the impact of delayed application of the control inputs such as advanced-step NMPC[66] or the real-time iteration scheme[67] could be also incorporated in the scheme.

8 | CONCLUSIONS AND FUTURE WORK

The computation complexity related to output-feedback robust NMPC controllers is prohibitive in most cases. Instead of relying on strong assumptions on error bounds and invariant sets that cannot be verified in practice, we propose a probabilistic performance validation scheme that can be used to obtain probabilistic guarantees about the closed-loop performance of approximate robust NMPC controllers based on a tree of discrete scenarios. To enable the implementation of such controllers in real-time even on limited embedded hardware, we used deep learning to approximate the proposed robust NMPC controller.

To deal with errors related to estimation, computation of approximate reachable sets based on scenarios as well as approximation of the resulting optimization problem with a neural network, we tighten the original constraints of the problem using a backoff parameter. The novel probabilistic validation framework lead to less-restrictive results when compared to previous approaches because of the incorporation of a discarding parameter and the consideration of nonbinary performance indicators. Moreover, the required sample complexity does not depend on the dimension of the problem. The promising results for the embedded output-feedback robust NMPC of a towing kite show the potential of the proposed approach. Future work includes the definition of robust margins based on probabilistic validation techniques as well as the learning of controllers that are parameterized, for example, with a backoff parameter.

ACKNOWLEDGEMENTS

This work was supported by the Agencia Estatal de Investigación (AEI)-Spain under Grant PID2019-106212RB-C41/AEI/10.13039/501100011033. The research leading to these results has received funding from from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant agreement number 423857295.

CONFLICT OF INTEREST

The authors declare no potential conflict of interests.

DATA AVAILABILITY STATEMENT

Data available on request.

ORCID

Benjamin Karg https://orcid.org/0000-0002-9779-3101
Sergio Lucia https://orcid.org/0000-0002-3347-5593

REFERENCES

1. Campo PJ, Morari M. Robust model predictive control. Paper presented at: Proceedings of the American Control Conference. Minneapolis, MN; 1987:1021-1026.
2. Lee JH, Yu ZH. Worst-case formulations of model predictive control for systems with bounded parameters. *Automatica*. 1997;33(5):763-781.
3. Mayne DQ, Seron MM, Rakovic SV. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*. 2005;41:219-224.
4. Rakovic S, Kouvaritakis B, Cannon M, Panos C, Findeisen R. Fully parameterized tube MPC. Paper presented at: Proceedings of the 18th IFAC World Congress. Milano, Italy; 2011:197-202.
5. Fleming J, Kouvaritakis B, Cannon M. Robust tube MPC for linear systems with multiplicative uncertainty. *IEEE Trans Autom Control*. 2015;60(4):1087-1092.
6. Scokaert P, Mayne D. Min-max feedback model predictive control for constrained linear systems. *IEEE Trans Autom Control*. 1998;43(8):1136-1142.
7. Muñoz de la Peña D, Bemporad A, Alamo T. Stochastic programming applied to model predictive control. Paper presented at: Proceedings of the 44th IEEE Conference on Decision and Control. Sevilla, Spain; 2005:1361-1366.
8. Bernardini D, Bemporad A. Scenario-based model predictive control of stochastic constrained linear systems. Paper presented at: Proceedings of the 48th IEEE Conference on Decision and Control. Shanghai, China; 2009:6333-6338.
9. Lucia S, Finkler T, Engell S. Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *J Process Control*. 2013;23:1306-1319.
10. Lucia S, Andersson J, Brandt H, Diehl M, Engell S. Handling uncertainty in economic nonlinear model predictive control: a comparative case-study. *J Process Control*. 2014;24:1247-1259.
11. Goulart P, Kerrigan EC, Maciejowski JM. Optimization over state feedback policies for robust control with constraints. *Automatica*. 2006;42:523-533.
12. Lucia S. *Robust Multi-Stage Nonlinear Model Predictive Control*. Aachen, Germany: Shaker, Verlag; 2014.
13. Lucia S, Subramanian S, Limon D, Engell S. Stability properties of multi-stage nonlinear model predictive control. *Syst Control Lett*. 2020;143:104743.
14. Houska B, Ferreau H, Diehl M. An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range. *Automatica*. 2011;47:2279-2285.
15. Mattingley J, Boyd S. Cvxgen: a code generator for embedded convex optimization. *Optim Eng*. 2012;13(1):1-27.
16. Zometa P, Kögel M, Findeisen R. muAO-MPC: a free code generation tool for embedded real-time linear model predictive control. Paper presented at: Proceedings of the American Control Conference. Washington, DC, USA; 2013:5320-5325.
17. Lucia S, Navarro D, Lucia O, Zometa P, Findeisen R. Optimized FPGA implementation of model predictive control using high level synthesis tools. *IEEE Trans Ind Inform*. 2018;14(1):137-145.
18. Johansen TA. Approximate explicit receding horizon control of constrained nonlinear systems. *Automatica*. 2004;40(2):293-300.
19. Johansen TA. On multi-parametric nonlinear programming and explicit nonlinear model predictive control. Paper presented at: Proceedings of the IEEE Conference on Decision and Control. Las Vegas, NV, USA; Vol. 3, 2002:2768-2773.
20. Bemporad A, Morari M, Dua V, Pistikopoulos EN. The explicit linear quadratic regulator for constrained systems. *Automatica*. 2002;38(1):3-20.
21. Parisini T, Zoppoli R. A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica*. 1995;31(10):1443-1451.
22. Karg B, Lucia S. Efficient representation and approximation of model predictive control laws via deep learning. *IEEE Trans Cybern*. 2020;50(9):3866-3878.

23. Chen S, Saulnier K, Atanasov N. Approximating explicit model predictive control using constrained neural networks. Paper presented at: Proceedings of the American Control Conference. Milwaukee, WI, USA; 2018:1520-1527.
24. Safran I, Shamir O. Depth-width tradeoffs in approximating natural functions with neural networks. Paper presented at: Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia; 2017:2979-2987.
25. Vidyasagar M. *A Theory of Learning and Generalization*. London, UK: Springer; 1997.
26. Tempo R, Calafiore G, Dabbene F. *Randomized Algorithms for Analysis and Control of Uncertain Systems, with Applications*. 2nd ed. London, UK: Springer-Verlag; 2013.
27. Grammatico S, Zhang X, Margellos K, Goulart P, Lygeros J. A scenario approach for non-convex control design. *IEEE Trans Autom Control*. 2016;61(2):334-345.
28. Lorenzen M, Dabbene F, Tempo R, Allgöwer F. Stochastic MPC with offline uncertainty sampling. *Automatica*. 2017;81:176-183.
29. Mammarella M, Lorenzen M, Capello E, et al. An offline-sampling SMPC framework with application to autonomous space maneuvers. *IEEE Trans Control Syst Technol*. 2018;28(2):1-15.
30. Calafiore G, Campi M. The scenario approach to robust control design. *IEEE Trans Autom Control*. 2006;51(5):742-753.
31. Fagiano L, Milanese M, Razza V, Gerlero I. Control of power kites for naval propulsion. Paper presented at: Proceedings of the American Control Conference. Baltimore, ML, USA; 2010:4325-4330.
32. Deori L, Garatti S, Prandini M. Trading performance for state constraint feasibility in stochastic constrained control: a randomized approach. *J Frankl Inst*. 2017;354(1):501-529.
33. Margellos K, Goulart P, Lygeros J. On the road between robust optimization and the scenario approach for chance constrained optimization problems. *IEEE Trans Autom Control*. 2014;59(8):2258-2263.
34. Alamo T, Tempo R, Camacho E. Randomized strategies for probabilistic solutions of uncertain feasibility and optimization problems. *IEEE Trans Autom Control*. 2009;54(11):2545-2559.
35. Tempo R, Bai E, Dabbene F. Probabilistic robustness analysis: explicit bounds for the minimum number of samples. *Syst Control Lett*. 1997;30:237-242.
36. Alamo T, Tempo R, Luque A, Ramirez D. Randomized methods for design of uncertain systems: sample complexity and sequential algorithms. *Automatica*. 2015;52:160-172.
37. Alamir M, Fiacchini M, Queinnec I, Tarbouriech S, Mazerolles M. Feedback law with probabilistic certification for propofol-based control of BIS during anesthesia. *Int J Robust Nonlinear Control*. 2018;28(18):6254-6266.
38. Alamir M. On probabilistic certification of combined cancer therapies using strongly uncertain models. *J Theor Biol*. 2015;384:59-69.
39. Hertneck M, Köhler J, Trimpe S, Allgöwer F. Learning an approximate model predictive controller with guarantees. *IEEE Control Syst Lett*. 2018;2(3):543-548.
40. Zhang X, Bujarbaruah M, Borrelli F. Safe and near-optimal policy learning for model predictive control using primal-dual neural networks; 2019. arXiv preprint arXiv:1906.08257.
41. Karg B, Lucia S. Learning-based approximation of robust nonlinear predictive control with state estimation applied to a towing kite. Paper presented at: Proceedings of the European Control Conference. Naples, Italy; 2019:16-22.
42. Blondel V, Tsitsiklis J. A survey of computational complexity results in systems and control. *Automatica*. 2000;36(9):1249-1274.
43. Hoeffding W. Probability inequalities for sums of bounded random variables. *J Am Stat Assoc*. 1963;58(301):13-30.
44. T. Alamo, J. Manzano, and E. Camacho, "Robust design through probabilistic maximization," in *Uncertainty in Complex Networked Systems: In Honor of Roberto Tempo*, T. Basar, Ed. Cham, Switzerland: Birkhäuser, 2018, pp. 247–274.
45. Ahsanullah M, Nevzorov V, Shakil M. *An Introduction to Order Statistics*. Paris: Atlantis Press; 2013.
46. Arnold B, Balakrishnan N, Nagaraja H. *A First Course in Order Statistics*. New York, NY: John Wiley and Sons; 1992.
47. Oishi Y. Polynomial-time algorithms for probabilistic solutions of parameter-dependent linear matrix inequalities. *Automatica*. 2007;43(3):538-545.
48. Calafiore GC, Dabbene F, Tempo R. Research on probabilistic methods for control system design. *Automatica*. 2011;47(7):1279-1293.
49. Alamo T, Tempo R, Luque A. On the sample complexity of randomized approaches to the analysis and design under uncertainty. Paper presented at: Proceedings of the American Control Conference; 2010; Baltimore.
50. Rawlings J, Mayne D. *Model Predictive Control: Theory and Design*. Goleta, CA: Nob Hill Publishing; 2009.
51. Althoff M, Krogh B. Reachability analysis of nonlinear differential-algebraic systems. *IEEE Trans Autom Control*. 2014;59:371-383.
52. Sahlodin A, Chachuat B. Convex/concave relaxations of parametric ODEs using Taylor models. *Comput Chem Eng*. 2011;35(5):844-857.
53. Hewing L, Zeilinger MN. Stochastic model predictive control for linear systems using probabilistic reachable sets. Paper presented at: Proceedings of the 2018 IEEE Conference on Decision and Control (CDC); 2018:5182-5188.
54. Subramanian S, Lucia S, Engell S. A synergistic approach to robust output feedback control: tube-based multi-stage NMPC. *IFAC-PapersOnLine*. Shenyang, China; 2018;51(18):500-505.
55. Barron AR. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans Inf Theory*. 1993;39(3):930-945.
56. Houska B, Diehl M. Optimal control of towing kites. Paper presented at: Proceedings of the 45th IEEE Conference on Decision and Control; 2006:2693-2697.
57. Erhard M, Strauch H. Control of towing kites for seagoing vessels. *IEEE Trans Control Syst Technol*. 2013;21:1629-1640.
58. Fagiano L, Novara C. Automatic crosswind flight of tethered wings for airborne wind energy: a direct data-driven approach. *IFAC Proc Vol*. 2014;47(3):4927-4932.
59. Costello S, François G, Bonvin D. Real-time optimization for kites. Paper presented at: Proceedings of the IFAC International Workshop on Periodic Control Systems (PSYCO); 2013:64-69.

60. Costello S, François G, Bonvin D. Crosswind kite control–a benchmark problem for advanced control and dynamic optimization. *Eur J Control*. 2017;35:1-10.
61. Chollet F. Keras; 2015. https://github.com/fchollet/keras.
62. Abadi M, Agarwal A, Barham P, et al. TensorFlow: large-scale machine learning on heterogeneous systems. Software; 2015. tensorflow.org. http://tensorflow.org/.
63. Kingma DP, Ba J. Adam: a method for stochastic optimization. Paper presented at: Proceedings of the 3rd International Conference on Learning Representations (ICLR); 2014.
64. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. Paper presented at: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics. Sardinia, Italy; 2010:249-256.
65. Kandasamy K, Neiswanger W, Schneider J, Poczos B, Xing EP. Neural architecture search with Bayesian optimisation and optimal transport. In: Proceedings of the 32nd International Conference on Neural Information Processing System. Montreal, Canada; 2018;31:2020-2029.
66. Zavala VM, Biegler LT. The advanced-step NMPC controller: optimality, stability and robustness. *Automatica*. 2009;45(1):86-93.
67. Diehl M, Bock HG, Schlöder JP, Findeisen R, Nagy Z, Allgöwer F. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *J Process Control*. 2002;12(4):577-585.

How to cite this article: Karg B, Alamo T, Lucia S. Probabilistic performance validation of deep learning-based robust NMPC controllers. *Int J Robust Nonlinear Control*. 2021;31(18):8855-8876. https://doi.org/10.1002/rnc.5696