2019 18th European Control Conference (ECC)
Napoli, Italy, June 25-28, 2019

Learning-based approximation of robust nonlinear predictive control with state estimation applied to a towing kite

Benjamin Karg and Sergio Lucia

Abstract— **Handling uncertainties is one of the most important challenges in nonlinear model predictive control (NMPC). While several robust NMPC methods have been recently presented, their implementation on embedded systems is usually difficult due to the necessary conservative assumptions or because of the required computational complexity.**

In this work, we use a complex robust NMPC approach to generate data pairs that are used to learn an approximate robust controller which is robust to model uncertainties. We propose to use deep neural networks to learn the approximate controller based on recent results that prove the powerful representation capabilities of such networks over traditional shallow ones. The approximate controller, which just requires the simple forward evaluation of neural network, can be easily combined with an Extended Kalman Filter to obtain an efficient embedded implementation of an output-feedback robust NMPC scheme. We propose a statistical verification strategy to compute backoffs that lead to the satisfaction of important constraints despite the presence of estimation, measurement and approximation errors. The potential of the approach is illustrated with numerical results for the embedded robust control of a towing kite.

I. Introduction

Model predictive control is one of the most popular advanced control techniques mainly because of its ability to deal with constraints, nonlinear systems and general control goals other than traditional set-point tracking tasks. Arguably, the two most important challenges when designing and implementing a reliable nonlinear model predictive controller are the presence of the uncertainty and the computational complexity of the resulting controller which often prevents an embedded implementation.

Different robust MPC schemes have been developed during the last years to deal with uncertainty. Beyond traditional min-max approaches [1] that do not consider the presence of feedback in the predictions, most of the recent robust MPC methods lead to a different trade-off between complexity and optimality. Tube-based approaches [2] decompose the robust MPC problem in two parts, a nominal MPC controller with tightened constraints and an ancillary controller that tries to steer the real uncertain system as close as possible to its nominal trajectory. In the simplest case, the complexity of the controller is the same as that of standard MPC. However, for an increased performance, the complexity grows as presented in [3] or [4]. Scenario tree-based [5], [6] or multi-stage MPC represents the evolution of the uncertainty using a tree of discrete uncertainty realizations. Since the structure of the feedback in the predictions is not restricted, as usually done in tube-based MPC, increased performance is often achieved in practice [7]. Formulating a multi-stage MPC scheme with stability and robust constraint satisfaction guarantees is also possible [8], and it can be easily designed to achieve good results in practice. However, its computational complexity grows exponentially with the number of uncertainties.

The challenge of the resulting complexity of any NMPC implementation is thus significantly aggravated because of the consideration of uncertainty. A significant progress has been recently achieved towards the efficient solution and implemenation of NMPC controllers, based on advances in hardware, algorithms and tailored implementations. For example, code generation tools [9], [10] enable efficient implementations of linear and nonlinear MPC on embedded hardware, including low-cost microcontrollers [11] and high-performance FPGAs [12].

Another alternative to achieve embedded nonlinear model predictive control is the use of approximate explicit nonlinear model predictive control [13], [14] based on approximating the multi-parametric nonlinear program using similar ideas as for explicit MPC of linear systems [15]. Instead, we propose to use deep neural networks to approximate a robust multi-stage NMPC control law. The idea of approximating the NMPC law using neural networks was already proposed by [16] but only very recently [17], [18] deep neural networks have been proposed as function approximators based on recent theoretical results that suggest the exponentially better approximation capabilities of deep neural networks in comparison to shallow networks [19].

This paper extends the results from [20] by taking into account the case of output feedback when all the states cannot be measured. We also present an overview of existing learning-based control methods and propose an algorithm to achieve constraint satisfaction despite of approximation and estimation errors based on pre-computed backoffs. We present results for an embedded implementation of an output-feedback robust NMPC for a highly nonlinear towing kite case study.

II. Multi-stage nonlinear model predictive control

We shortly introduce here the multi-stage NMPC approach presented in [21], [22] that will be learned using a deep neural network. The main assumption of multi-stage NMPC is that the uncertainty can be modeled by a tree of discrete scenarios that form the branches in the tree shown in Fig. 1.

B. Karg and S. Lucia are with the Laboratory of Internet of Things for Smart Buildings, Technische Universität Berlin, Ernst-Reuter-Platz 7, 10587 Berlin, Germany and Einstein Center Digital Future. (e-mail: sergio.lucia@tu-berlin.de)

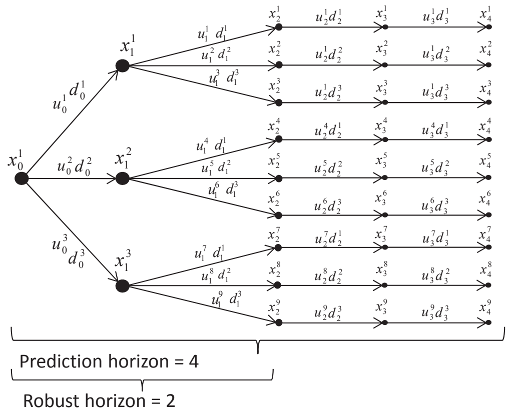

Fig. 1: Scenario tree representation of the evolution of the uncertainty for multi-stage NMPC.

Because of the tree structure, feedback is explicitly considered in the predictions, which results in less conservative solutions when compared to classical open-loop approaches. In the case that the uncertainty is indeed discrete-valued, multi-stage NMPC provides the best possible solution for a given prediction horizon, and for continuous-valued uncertainty it provides an approximation.

In the multi-stage NMPC framework, the following discrete-time nonlinear system is considered:

$$x_{k+1}^j = f\left(x_k^{p(j)}, u_k^j, d_k^{r(j)}\right), \tag{1a}$$

where each state vector $x_{k+1}^j \in \mathbb{R}^{n_x}$ at stage $k+1$ and position j depends on the parent state $x_k^{p(j)}$ at stage k, the vector of control inputs $u_k^j \in \mathbb{R}^{n_u}$ and the corresponding realization r of the uncertainty $d_k^{r(j)} \in \mathbb{R}^{n_d}$. The uncertainty at the stage k is defined by $d_k^{r(j)} \in \{d_k^1, d_k^2, \ldots, d_k^s\}$ for s different possible combinations of values of the uncertainty. The set of indices (j,k) in the scenario tree is denoted by I and S_i denotes the i-th scenario which is the path from the root node x_0 to one of the leaf nodes.

To design a multi-stage NMPC controller, a scenario tree should be built. In the linear case, using all possible combinations of the extreme values of the uncertainty as branches guarantees constraint satisfaction. While in the nonlinear case constraint satisfaction cannot be guaranteed, good results are obtained in practice for challenging nonlinear systems [21], [22]. Rigorous guarantees for the nonlinear case can be achieved by combining it with reachability analysis tools as shown in [23]. To avoid the exponential growth of the tree with the prediction horizon, we consider that the uncertainty remains constant after a certain stage (called robust horizon N_r) as shown in Fig. 1.

The optimization problem to be solved at each sampling instant is:

$$\min_{x_{k+1}^j, u_k^j, \forall (j,k) \in I} \quad \sum_{i=1}^{N} \omega_i J_i(X_i, U_i) \tag{2a}$$

subject to:

$$x_{k+1}^j = f\left(x_k^{p(j)}, u_k^j, d_k^{r(j)}\right), \quad \forall (j, k+1) \in I, \tag{2b}$$

$$0 \geq g\left(x_{k+1}^j, u_k^j, d_k^{r(j)}\right), \quad \forall (j,k) \in I, \tag{2c}$$

$$u_k^j = u_k^l \text{ if } x_k^{p(j)} = x_k^{p(l)}, \quad \forall (j,k), (l,k) \in I, \tag{2d}$$

$$x_0 = x_{\text{init}}, \tag{2e}$$

where X_i, U_i are contain all the states and control inputs that belong to the scenario S_i with a weight occurrence ω_i. The general constraints on inputs and states are denoted by $g(\cdot)$ and the cost of each scenario is denoted by $J_i(\cdot)$ and can be written as:

$$J_i(X_i, U_i) := \sum_{k=0}^{N_p - 1} L\left(x_{k+1}^j, u_k^j\right), \; \forall x_{k+1}^j, u_k^j \in S_i. \tag{3}$$

The constraints in (2d) are called non-anticipativity constraints which enforce causality of the control policy.

III. Learning-based approximate robust NMPC

Approximating an NMPC controller with a neural network was already proposed by [16] 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 [24]. Based on the universal approximation theory, most works avoided the use of deeper networks because it made training more difficult and it was theoretically not necessary to achieve a good approximation accuracy.

In this work, we use deep neural networks which have been shown to achieve excellent results in the field of computer science [25]. Recent works have studied the reasons for the good performance of such networks and have shown that they can approximate certain functions exponentially faster than networks with only one layer [19]. Based on these recent findings, the works [18], [17] proposed the use of deep neural networks to approximate explicit MPC solutions. [17] showed that DNNs of a determined size can exactly represent the piecewise affine function defined by an MPC problem with linear time-invariant systems.

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

$$\mathcal{N}(x;\theta) = \begin{cases} f_{L+1} \circ g_L \circ f_L \circ \cdots \circ g_1 \circ f_1(x) & \text{for} \quad L \geq 2, \\ f_{L+1} \circ g_1 \circ f_1(x), & \text{for} \quad L = 1, \end{cases} \tag{4}$$

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 number of neurons in each hidden layer M is usually denoted as width of the network and the number of hidden layers, also known as depth, is denoted by L. If $L \geq 2$, the network $\mathcal{N}$ is a *deep* neural network and if $L = 1$ a *shallow* neural network. Each hidden layer consists of an affine function:

$$f_l(\xi_{l-1}) = W_l \xi_{l-1} + b_l, \tag{5}$$

where $\xi_{l-1} \in \mathbb{R}^M$ is the output of the previous layer with $\xi_0 = x$. The second element of the neural network is a non-linear activation function g_l. Typical activation functions include the sigmoid *tanh* functions or the rectifier linear units (ReLU). The parameter $\theta = \{\theta_1, \ldots, \theta_{L+1}\}$ of the neural network contains all the weights and biases that define the affine functions of each layer

$$\theta_l = \{W_l, b_l\} \quad \forall l = 1, \ldots, L+1. \tag{6}$$

The proposed approximate multi-stage NMPC tries to learn the solution of the optimization problem (2) as a function of the current state x_{init}. Training a network with a predefined depth L and width M consists in finding the optimal parameters $\theta_l = \{W_l, b_l\}$ that minimize the following cost function:

$$\min_{\theta_l} \frac{1}{N_s} \sum_{p=1}^{N_s} (u^*(x_{\text{init}}^{[p]}) - \mathcal{N}(x_{\text{init}}{}^{[p]}, \theta_l))^2, \tag{7}$$

where N_s is the total amount of the data pairs considered in the training. The training data is formed by the input to the network, which is the current state of the system x_{init} and the output of the network, which is the resulting input signal obtained from the multi-stage formulation, denoted as $u^*(x_{\text{init}}^{[p]})$ for the training point p.

IV. Constraint satisfaction of learning-based robust NMPC

A. *Overview of existing approaches*

One of the main challenges in the design of a learning-based controller is the satisfaction of constraints despite the unavoidable approximation errors introduced by the learning component. Due to the increased recent popularity of learning-based control, different methods have been proposed to tackle this challenge. Since the term learning-based control is used to refer to different approaches, we present here a short overview of some of the methods that have been proposed to achieve constraint satisfaction.

Traditionally, learning-based control has referred to the use of system identification techniques [26] and the design of a controller for the identified system. Recent results show that constraint satisfaction can be achieved based on different assumptions. As an example, the work presented in [27] is based on Lipschitz continuity to achieve guarantees, [28] uses set-membership approaches, [29] is based on non-asymptotic statistical theory and [30] estimates the uncertain part of the dynamic system using gaussian processes. The underlying common idea of these works is the identification of a dynamic system with a quantification of the uncertainty and the design of a robust controller.

Learning-based control is also used to denote controllers that learn and improve performance over time based on data. The approach presented in [31] takes advantage of the repetitive nature of many control tasks by using information about previous repetitions to improve the performance of MPC with guarantees.

Another set of works deals with guarantees for controllers that have been obtained based on some learning strategy. The work from [32] uses safe sets where a learning-based controller can be applied. Such a controller might have a high performance but may not always be safe in the whole state space. Whenever the system leaves the safe set, a backup controller should be activated to guarantee constraint satisfaction of the closed-loop system. The works in [17], [18] propose the use of deep learning to learn an approximate MPC controller based on many offline solutions of the MPC problem. Constraint satisfaction guarantees are recovered by projecting the approximate input onto a control invariant set for the controlled system. The work in [33] proposes to robustify a priori an approximate controller and to verify a posteriori with statistical learning theory that the error of the approximate controller is indeed smaller than the one that had been assumed.

B. *Statistical verification*

In this work, we consider a nonlinear MPC scheme in the output feedback setting, which significantly complicates the analysis of robust constraint satisfaction of the closed-loop. For this reason, we resort to a stochastic verification scheme to compute an appropriate backoff that can be used within the multi-stage MPC to achieve constraint satisfaction despite of the measurement noise, estimation errors as well as the errors incurred in the approximation by a deep neural network.

We propose the solution of the following modified multi-stage MPC problem

$$\min_{x_{k+1}^j, u_k^j, \forall (j,k) \in I} \quad \sum_{i=1}^{N} \omega_i J_i(X_i, U_i) \tag{8a}$$

subject to:

$$\begin{aligned}
x_{k+1}^j &= f\left(x_k^{p(j)}, u_k^j, d_k^{r(j)}\right), & \forall (j, k+1) \in I, \quad &(8b)\\
0 &\geq g\left(x_{k+1}^j, u_k^j, d_k^{r(j)}\right) + \eta, & \forall (j,k) \in I, \quad &(8c)\\
u_k^j &= u_k^l \text{ if } x_k^{p(j)} = x_k^{p(l)}, & \forall (j,k), (l,k) \in I, \quad &(8d)\\
x_0 &= \hat{x}_{\text{e}}, & &(8e)
\end{aligned}$$

in which the only differences with the problem in (2) are the introduction of a time-invariant parameter η, which is a constraint backoff, and the consideration of the estimated states $\hat{x}_{\text{e}}$ as initial condition.

The policy obtained by solving (8) is denoted as $\kappa_{\text{MPC}}(\hat{x}_{\text{e}})$. The closed-loop trajectory of a system controlled by such policy is $x(k) = \Phi(x(0), \kappa_{\text{MPC}}(\hat{x}_{\text{e}}), d_{[0,k]}, \eta)$. The state of the system $x(k)$ at time k is a function of the initial state $x(0)$, the multi-stage policy $\kappa_{\text{MPC}}(\hat{x}_{\text{e}})$ based on the estimated state $\hat{x}_{\text{e}}$, the sequence of actual realizations of the uncertain parameters $d_{[0,k]}$ from time 0 until k and the time-invariant parameter η.

We use the ideas from [34] and define requirements using temporal logic [35]. In this manner, the requirements of the closed-loop system for each trajectory can be formulated as

$$\gamma(\eta) = \Box_{[k_1,k_2]}(g(x(k), \kappa_{\text{MPC}}(\hat{x}_{\text{e}}), d_{[0,k]}) + \eta \geq 0), \tag{9}$$

which means that $g(\cdot) \leq 0$ has to be "always" true between the time steps k_1 and k_2. Replacing $\Box$ in (9) with $\Diamond$ would shift the meaning from "always" to "eventually". If the temporal logic is satisfied for a given trajectory, the requirement function is $\gamma(\eta) = +1$ and it is $\gamma(\eta) = -1$ in case it is not satisfied.

To verify properties of a given closed-loop system, [34] proposes to evaluate the requirements $\gamma(\cdot)$ for N_v state trajectories for N_v random realizations of the time-invariant parameter η. The goal of the evaluation of N_v different trajectories is to compute

$$p_{\text{sat}(\eta)} = \mathbb{P}(\gamma(\eta) > 0), \tag{10}$$

which is the probability of satisfaction of the requirements as a function of the time-invariant parameter η. For each parameter η, the satisfaction of the requirements is not deterministic because it depends on the actual realization of the uncertainties $d_{[0,k]}$ and therefore in general only a probability of satisfaction can be computed. Instead of just performing a validation of the closed-loop system, we propose in this work to consider the backoff η introduced in the multi-stage NMPC problem (8) as a time-invariant parameter following ideas from tube-based MPC. Based on the data obtained by the N_v simulations, the probability (10) can be estimated using support vector machines, gaussian processes, or other methods in a similar fashion as done in [34]. Once such a probability is computed or estimated, an appropriate backoff can be chosen depending on the amount of violations that can be allowed for a given application. Taking directly into account the design of the scenario tree as part of the verification strategy is out of the scope of this work.

V. Kite Model and Control Problem

We consider as case study the optimal control of a towing kite that pulls a boat to reduce fuel consumption. Different kite models have been proposed in the literature. While there exist very detailed models, simple ones [36], [37] are more suitable for its use in online optimal control schemes like MPC. In this work, we rely on the model proposed in [38] that consists of three ordinary differential equations. For the sake of brevity the reader is referred to [38] for a deeper explanation of the modeling assumptions and the derivation of the equations. The dynamics of the kite are described by the following set of differential equations:

$$\dot{\theta} = \frac{v_\text{a}}{L_\text{T}}(\cos\psi - \frac{\tan\theta}{E}), \tag{11a}$$

$$\dot{\phi} = -\frac{v_\text{a}}{L_\text{T}\sin\theta}\sin\psi, \tag{11b}$$

$$\dot{\psi} = \frac{v_\text{a}}{L_\text{T}}\tilde{u} + \dot{\phi}\cos\theta, \tag{11c}$$

where

$$v_\text{a} = v_0 E\cos\theta, \tag{11d}$$

$$v_0 = v_\text{m} + v_\text{A}\cdot\sin(2\pi\cdot v_\text{f}\cdot t + v_\text{off}), \tag{11e}$$

$$E = E_0 - \tilde{c}\tilde{u}^2, \tag{11f}$$

$$P_\text{D} = \rho v_0^2/2, \tag{11g}$$

$$\begin{aligned} T_\text{F} = {} & P_\text{D} A\cos^2\theta(E+1)\sqrt{E^2+1} \\ & \cdot(\cos\theta\cos\beta + \sin\theta\sin\beta\sin\phi), \end{aligned} \tag{11h}$$

and θ, ϕ and ψ represent the three angles of the spherical coordinate system. Hereby, θ is the zenith angle (between wind and tether), ϕ is the azimuth angle (between the vertical and the plane whose normal is the wind direction) and ψ is the angle denoting the orientation of the kite. The area of the kite is denoted by A while L_T is the length of the tether. The angle β gives the relation between the orientation of the boat and the direction of the wind. The glide ratio E is an aerodynamic coefficient, ρ is the air density and the wind speed v_0 is modeled in our work by a general sine wave composed of the mean wind speed v_m, the amplitude v_A, the frequency v_f and the offset v_off. The manipulated variable is the steering deflection command $\tilde{u}$ and $\tilde{c}$ denotes the related correction coefficient. The presented model from [38] was slightly costumized in [39] by making E a time-varying parameter depending on the base glide ratio E_0 and the influence of the steering deflection command as described in (11f). The parameter values used in this work are given in Table I.

We assume that not all states can be properly measured online. The azimuth angle and the zenith angle of the kite are obtained via the measurement function

$$y(x) = [\theta + w_\theta,\, \phi + w_\phi]^T, \tag{12}$$

where $x = [\theta,\, \phi,\, \psi]^T$ is the state vector and the noises $w_\theta = \mathcal{N}(0, 0.01)$ and $w_\phi = \mathcal{N}(0, 0.01)$ are zero-mean gaussian. We use an Extended Kalman Filter (EKF) to estimate the orientation ψ and the uncertain parameters glide ratio E_0 and wind speed v_0. This results in the augmented state $\hat{x}_\text{aug} = [\theta, \phi, \psi, E_0, v_0]^T$. The EKF is evaluated with a sampling time of $t_e = 0.05\,\text{s}$. The matrix defining the covariance of the process noise is chosen as $Q = \text{diag}([0.001,\, 0.001,\, 0.05,\, 0.005,\, 1.0])$, the covariance of the observation noise is $R = \text{diag}([0.01\; 0.01])$ and the initial estimation covariance is $P = \text{diag}([0.01, 0.01, 0.01, 0.1, 0.2])$. The estimated values of the uncertain parameters are not included in the control since those are incorporated in the scenario-tree approach of the robust control scheme to account for possible variations.

The critical constraint of the optimal control problem is that the height of the kite

$$h(x) = L_\text{T}\sin\theta\cos\phi, \tag{13}$$

should not be lower than $h_\text{min} = 100\,\text{m}$.

We group all the uncertainties related to the wind model into one uncertain value $v_0 \in [7, 13]$. To design a multi-stage NMPC, we consider the extreme values of the wind

TABLE I: Overview of the model states and parameters. The possible values of the uncertain parameters and the state and control constraints are given as intervals.

Symbol	Type	Values / Constraints	Units
E_0	Uncertain parameter	$[4, 6]$	-
v_{m}	Uncertain parameter	$[8, 12]$	$\mathrm{m\,s^{-1}}$
v_{A}	Uncertain parameter	$[0, 1]$	$\mathrm{m\,s^{-1}}$
v_{f}	Uncertain parameter	$[0.05, 0.15]$	$\mathrm{s^{-1}}$
v_{off}	Uncertain parameter	$[0, 2\pi]$	rad
θ	State	$[0, \frac{\pi}{2}]$	rad
ϕ	State	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	rad
ψ	State	$[0, 2\pi]$	rad
$\tilde{u}$	Control input	$[-10, 10]$	N
$\tilde{c}$	Known parameter	0.028	-
β	Known parameter	0	rad
ρ	Known parameter	1	$\mathrm{kg\,m^{-3}}$

speed v_0 and the glide ratio E_0, leading to four different branches in the scenario tree. In the case that all states can be perfectly measured, a multi-stage NMPC controller that directly maximizes the obtained thrust leads to robust satisfaction of the constraints, as it was shown in [7]. We consider in this work estimation and measurement errors, and therefore the height constraints are implemented as soft constraint to account for possible violations. Thus, the stage cost used at each prediction step is defined by:

$$\ell(x, u) = -w_{\mathrm{F}} T_{\mathrm{F}} + w_{\mathrm{u}}(u - u_{\mathrm{old}})^2 + w_{\mathrm{s}}\epsilon^2, \tag{14}$$

where $w_{\mathrm{F}} = 10^{-4}$ is the weight for the tension, and $w_{\mathrm{u}} = 0.5$ punishes changes in the control input to obtain a smooth behavior. The weight $w_{\mathrm{s}} = 10^3$ is a penalty for the violation of the height constraint:

$$h_{\min} + \eta - h(x) \leq \epsilon, \tag{15}$$

where $\epsilon \geq 0$ is a slack variable.

VI. Results

A. Output-feedback multi-stage NMPC

Using a scenario tree with robust horizon equal to one step and four branches in the scenario tree based on the extreme values of v_0 and E_0, the multi-stage NMPC defined in (8) is solved for $\eta = 0$, i.e. with no additional backoff and a prediction horizon of $N = 80$ steps. Due to the estimation errors introduced by the Extended Kalman Filter, the measurement noise and the highly non-linear system, the height constraint is violated for some realizations of the uncertain parameters. The average performance of the controller is summarized in the second column of Table II, which indicates a maximum violation of almost 2.5 m.

B. Training of the deep learning-based robust controller

To design an approximate robust NMPC controller that can be implemented on a low-cost embedded platform in real-time, we ran 100 simulations with random parameter settings for 70 s with a controller sampling time $t_c = 0.15$ s for the same problem (8) generating a total of $N_s = 45000$ data points.

Each pair of state estimates and optimal inputs of the trajectories was used as a training point. The controller is learned by solving the training problem defined in (7) with stage cost (14) based on the states estimated by the EKF, i.e, $x_{\mathrm{init}} = \hat{x}_{\mathrm{e}}$, and the solution of the multi-stage NMPC problem.

For the design, training and evaluation of the network we used Keras [40] with Tensorflow [41] as the back-end. The deep neural network has $L = 6$ layers and $M = 10$ neurons per layer and we use the *tanh* function as activation function.

The trained deep-learning based robust NMPC has a very similar performance when compared to exact multi-stage NMPC as shown in the third column of Table II. Since the exact multi-stage NMPC lead to constraint violations, its approximation also violates the constraints. Because the deep-learning based MPC has a very small complexity, it could be executed at a faster rate to potentially obtain an increased performance. However, the uncertainty present due to the estimation cannot be removed. As it can be seen in Table II, reducing the sampling time of controller does not lead to any improvements in the performance.

C. Obtaining an appropiate backoff

To achieve satisfaction of the height constraints despite of the estimation, measurement and approximation errors, we introduce the time-invariant parameter $\eta \in [0, 10]$ in (8) as a backoff for the height constraint. We do not consider backoffs for the state box-constraints because they are not active during the optimal operation of the kite.

We ran 100 solutions of (8) with random realizations of η within its bounds during 20 s. The choice of the duration is justified by the quasi-periodic behavior of the kite, where the completion of one cycle corresponds to approximately 20 s.

The requirements of the closed-loop system are defined using the following temporal logic requirement:

$$\gamma_{\mathrm{h}}(x) = \square_{[0,20]}(h(x) - 100 \leq 0). \tag{16}$$

That is, the original height constraint should not be violated. We consider the simplified case in which we want to estimate the required η_{sat} that leads to no false positives, meaning that all trajectories generated with $\eta > \eta_{\mathrm{sat}}$ were indicated with a $\gamma_{\mathrm{h}} = +1$. This occurred for a value $\eta_{\mathrm{sat}} = 4.7$. We choose a backoff of $\eta = 5.0$ for our exact multi-stage NMPC, and perform again 60 different trajectories for different values of the uncertain parameters that are summarized in the sixth column of Table II. As it can be seen, because of the considered backoff, no constraint violations occur. The tighter height constraint of the modified problem reduces the scope of the kite movements leading to a reduced thrust T_{F} compared to the original problem, but it ensures a safe operation (see Fig. 2).

We use this multi-stage NMPC controller to generate data points for a deep-learning based NMPC with exactly the same tuning parameters as previously described. The performance of the approximate controller is summarized in the last columns of Table II.

The results show that the proposed approximate output-feedback robust NMPC is able to achieve robust constraint satisfaction for all the trajectories, and it obtains a very similar performance compared to the online solution of large nonlinear programming problems, as shown by the achieved thrust and Fig. 3. Even though there are no constraint violations for the controllers with backoff, we cannot gaurantee constraint satisfaction because in general, only a probability of constraint satisfaction (10) can be computed with data-based methods.

To analyze the robustness of the proposed controller against measurement noise, we considered a scenario with less noise $w_\theta = w_\phi = \mathcal{N}(0, 0.001)$, and a scenario with more noise $w_\theta = w_\phi = \mathcal{N}(0, 0.03)$. In both scenarios all trajectories satisfy the height constraint for different realizations of the disturbances. While the thrust of the tether T_F is $0.4\,\%$ higher with smaller noise, it is reduced by $2.7\,\%$ for the noisier scenario.

D. Embedded implementation

Solving the approximated robust NMPC only requires to the evaluation of a neural network which consists of matrix-vector multiplications and the evaluation of the tangent hyperbolic function. The results for the two deep learning-based MPC controllers presented in this work correspond to an embedded implementation. They have been implemented on a low-cost 32-bit microcontroller (ARM Cortex-M3) with 96 kB of RAM, 512 kB of Flash and a frequency of 89 MHz and been test in hardware-in-the-loop fashion.

The whole code, including the EKF, the neural network, and the communication with the real system simulated on a computer had a memory footprint of $51.32\,\mathrm{kB}$. The evaluation time for the EKF was $23.84\,\mathrm{ms}$ and for the neural network $6.02\,\mathrm{ms}$. Due to this short evaluation times we could further reduce the sampling time t_c of our approximated controller in case it is necessary. We chose $t_\mathrm{c} = \{0.05\,\mathrm{s}, 0.1\,\mathrm{s}, 0.15\,\mathrm{s}\}$ which leaves for the fastest sampling rate $t_\mathrm{c} = 0.05\,\mathrm{s}$ a time buffer of $20.14\,\mathrm{ms}$ to handle other possible tasks.

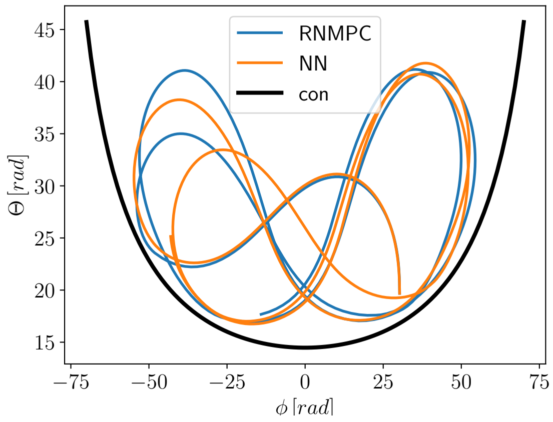

Fig. 2: Trajectory of the kite with a backoff $\eta = 5\,\mathrm{m}$. Both the solutions of the robust NMPC and the NN lead to similar trajectories.

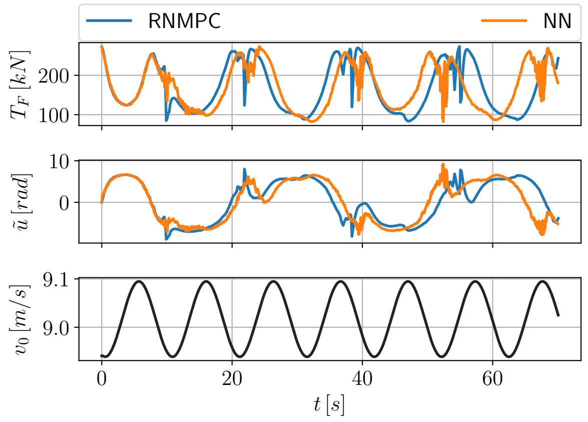

Fig. 3: Thrust T_F, steering deflection command $\tilde{u}$ and wind speed v_0 for one parameter setting of the robust NMPC controller and its deep learning-based approximation for $\eta = 5\,\mathrm{m}$.

VII. Conclusion

We proposed an approximate output-feedback robust NMPC scheme via deep learning that can be easily implemented on low-cost embedded hardware. We propose a multi-stage NMPC controller to deal with important model uncertainties, and use it to generate training data to learn an approximate controller using deep neural networks.

To deal with approximation, estimation and measurement errors, we propose the combination of statistical verification techniques with the use of backoffs in the multi-stage formulation. The potential of the presented approach has been validated for the embedded control of a highly-nonlinear kite model. The simulation results showed that the deep-learning based approximation achieves a very similar performance compared to the exact robust NMPC scheme.

The resulting approximated controller has very fast computation times which renders the real-time control of the kite possible. In combination with its low memory footprint, the approximation can be easily deployed on an embedded device.

Future work includes the use of Gaussian processes to obtain better estimates of appropriate constraint backoffs and the systematic design of scenario trees to improve the verification methodology described in the paper.

References

[1] P. J. Campo and M. Morari, "Robust model predictive control," in *Proc. of the American Control Conference*, 1987, pp. 1021–1026.

[2] D. Q. Mayne, M. M. Seron, and S. V. Rakovic, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, pp. 219 – 224, 2005.

[3] S. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Fully parameterized tube MPC," in *Proc. of the 18th IFAC World Congress Milano*, 2011, pp. 197–202.

TABLE II: Average thrust of the tether T_{F}, average and maximum violation of the height constraint for robust NMPC with and without backoff of $\eta = 5\,\mathrm{m}$, and the corresponding deep learning-based approximations over 60 simulations with random realizations of the disturbances.

Method	RNMPC	NN	NN	NN	RNMPC	NN	NN	NN
Backoff η [m]	0	0	0	0	5	5	5	5
Sampling Time t_c [ms]	150	150	100	50	150	150	100	50
Thrust (avg.) [kN]	224.44	226.85	226.97	227.05	221.59	221.59	222.19	223.05
Violation (avg.) [m]	2.621×10^{-3}	1.001×10^{-3}	1.669×10^{-3}	1.110×10^{-3}	0	0	0	0
Violation (max.) [m]	2.499	1.177	1.799	2.419	0	0	0	0

[4] J. Fleming, B. Kouvaritakis, and M. Cannon, "Robust tube mpc for linear systems with multiplicative uncertainty," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 1087–1092, 2015.

[5] P. Scokaert and D. Mayne, "Min-max feedback model predictive control for constrained linear systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 8, pp. 1136–1142, 1998.

[6] D. Muñoz de la Peña, A. Bemporad, and T. Alamo, "Stochastic programming applied to model predictive control," in *Proc. of the 44th IEEE Conference on Decision and Control*, 2005, pp. 1361–1366.

[7] S. Lucia, A. Tatulea-Codrean, C. Schoppmeyer, and S. Engell, "An environment for the efficient testing and implementation of robust NMPC," in *Proc. of the 2014 IEEE Multi-Conference on Systems and Control*, 2014, pp. 1843–1848.

[8] S. Lucia, *Robust Multi-stage Nonlinear Model Predictive Control*. Shaker, 2014.

[9] B. Houska, H. Ferreau, and M. Diehl, "An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range," *Automatica*, vol. 47, pp. 2279–2285, 2011.

[10] J. Mattingley and S. Boyd, "Cvxgen: A code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[11] P. Zometa, M. Kögel, and R. Findeisen, "muAO-MPC: A free code generation tool for embedded real-time linear model predictive control," in *Proc. of the American Control Conference*, June 2013, pp. 5320–5325.

[12] S. Lucia, D. Navarro, O. Lucia, P. Zometa, and R. Findeisen, "Optimized FPGA implementation of model predictive control using high level synthesis tools," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 1, pp. 137–145, 2018.

[13] T. A. Johansen, "Approximate explicit receding horizon control of constrained nonlinear systems," *Automatica*, vol. 40, no. 2, pp. 293–300, 2004.

[14] ——, "On multi-parametric nonlinear programming and explicit nonlinear model predictive control," in *IEEE conference on decision and control*, vol. 3. IEEE; 1998, 2002, pp. 2768–2773.

[15] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3 – 20, 2002.

[16] T. Parisini and R. Zoppoli, "A Receding-horizon Regulator for Nonlinear Systems and a Neural Approximation," *Automatica*, vol. 31, no. 10, pp. 1443–1451, 1995.

[17] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *arXiv preprint arXiv:1806.10644*, 2018.

[18] S. Chen, K. Saulnier, and N. Atanasov, "Approximating Explicit Model Predictive Control Using Constrained Neural Networks," in *Proc. of the American Control Conference*, 2018, pp. 1520–1527.

[19] I. Safran and O. Shamir, "Depth-width tradeoffs in approximating natural functions with neural networks," in *ICML*, 2017.

[20] S. Lucia and B. Karg, "A deep learning-based approach to robust nonlinear model predictive control," in *Proc. of the IFAC Conference on Nonlinear Model Predictive Control*, 2018, pp. 610–615.

[21] S. Lucia, T. Finkler, and S. Engell, "Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty," *Journal of Process Control*, vol. 23, pp. 1306–1319, 2013.

[22] S. Lucia, J. Andersson, H. Brandt, M. Diehl, and S. Engell, "Handling uncertainty in economic nonlinear model predictive control: a comparative case-study," *Journal of Process Control*, vol. 24, pp. 1247–1259, 2014.

[23] S. Lucia, R. Paulen, and S. Engell, "Multi-stage nonlinear model predictive control with verified robust constraint satisfaction," in *Proc. of the 54th IEEE Conference on Decision and Control*, 2014, pp. 2816–2821.

[24] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[25] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, p. 484, jan 2016.

[26] L. Ljung, "Perspective on System Identification," *Annual Reviews in Control*, vol. 34, no. 1, pp. 1–12, 2010.

[27] D. Limon, J. Calliess, and J. M. Maciejowski, "Learning-based nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7769–7776, 2017.

[28] E. Terzi, L. Fagiano, M. Farina, and R. Scattolini, "Learning-based predictive control for linear systems: a unitary approach," *arXiv preprint arXiv:1810.12584*, 2018.

[29] S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu, "On the Sample Complexity of the Linear Quadratic Regulator," *arXiv preprint arXiv:1710.01688*, 2017. [Online]. Available: http://arxiv.org/abs/1710.01688

[30] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, "Robust Constrained Learning-based NMPC enabling reliable mobile robot path tracking," 2016.

[31] U. Rosolia and F. Borrelli, "Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework." *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2017.

[32] K. P. Wabersich and M. N. Zeilinger, "Linear model predictive safety certification for learning-based control," *arXiv preprint arXiv:1803.08552*, 2018.

[33] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, 2018.

[34] J. F. Quindlen, "Data-driven methods for statistical verification of uncertain nonlinear systems," Ph.D. dissertation, Massachusetts Institute of Technology, 2018.

[35] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[36] B. Houska and M. Diehl, "Optimal control of towing kites," in *Proc. of the 45th IEEE Conference on Decision and Control*, 2006, pp. 2693–2697.

[37] L. Fagiano, M. Milanese, V. Razza, and I. Gerlero, "Control of power kites for naval propulsion," in *Proc. of the American Control Conference*, 2010, pp. 4325–4330.

[38] M. Erhard and H. Strauch, "Control of towing kites for seagoing vessels," *IEEE Transactions on Control Systems Technology*, vol. 21, pp. 1629–1640, 2013.

[39] S. Costello, G. François, and D. Bonvin, "Real-time optimization for kites," in *Proc of the IFAC International Workshop on Periodic Control Systems (PSYCO)*, 2013, pp. 64–69.

[40] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[41] M. A. et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/