Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Towards Safe Neural Network Supported Model Predictive Control

T. Zieger [*,**] **A. Savchenko** [*] **T. Oehlschlägel** [**] **R. Findeisen** [*]

[*] *Laboratory for Systems Theory and Automatic Control, Otto-von-Guericke University Magdeburg, Germany (e-mail: tim.zieger@ovgu.de, anton.savchenko@ovgu.de, rolf.findeisen@ovgu.de)*
[**] *Department of Powertrain Mechatronics, iav GmbH, Gifhorn, Germany, (e-mail: tim.zieger@iav.de, thimo.oehlschlaegel@iav.de)*

Abstract: Model Predictive Control has proven to be a universal and flexible method to control complex nonlinear system with guaranteed constraint satisfaction. However, high dependency on model quality often renders it inappropriate for hard to model systems. On the other hand, machine learning methods show great performance when approximating functions based on data. This capability for learning with poor a priori knowledge, however, comes at the cost of low predictability and lack of safety guarantees. To overcome these drawbacks we illustrate how a neural network can be setup as a nonlinear feedforward control that augments the MPC control signal to approximate a desired control behaviour. For instance, it could aim to mimic the control behaviour of a human driver, while the underlying MPC exploits prior knowledge. Moreover, to preserve constraint satisfaction, we suggest to restrict the range of neural network outputs such that it intrinsically satisfies control input constraints. Subsequently, we represent the neural network control signal as a disturbance which enables the application of tube MPC to retain state constraints satisfaction at the cost of introducing some conservatism. We demonstrate these concepts via simulation, test and highlight both the advantages and the drawbacks of the proposed control structure.

Keywords: Model predictive control; Machine learning; Neural networks; Constraint satisfaction; Tube model predictive control.

1. INTRODUCTION

Continuous progress, great results in certain fields and lowering barriers to entry for sophisticated machine learning techniques have boosted their application and attracted widespread interest. Thus, control research community considers to incorporate established machine learning methods such as neural networks (NN) due to their profound capabilities of approximating functions. Examples such as (Bojarski et al. (2016); Silver et al. (2016); Levine et al. (2015)) confirm the potential of using machine learning in control. However, these concepts often lack approximation guarantees or worst case bounds. In fact, NN can be vulnerable against attacks originating from adversarial NNs that are trained to provoke failure of the target NN with minimal effort (Huang et al. (2017)). Thus, many researchers tackle this challenge combining NN with advanced methods of classical control theory, which enable guarantees on performance and stability. In particular, model predictive control (MPC) is frequently combined with machine learning approaches, as it allows to handle constraints, well established theoretical results, as well as its theoretical similarity to machine learning approaches. For example, in (Bouffard et al. (2012)) a quadratic program has been solved to control a quadrotor in real time. The basic idea is to use a learned model to predict the optimal control trajectory minimizing an unconstrained objective function, while guaranteeing constraints using a nominal model. The use of a safety net, separating the learned model for optimal performance and the nominal model allows to provide such guarantees. It was further expanded in (Bethge et al. (2018)) for the case of multiple models to learn. Another approach has been proposed by (Gros and Zanon (2019); Zanon et al. (2019)), using Reinforcement Learning (RL) to tune parameters of the MPC and safe scenario tree theory to assure constraint satisfaction despite on-line adaptation of MPC parameters. Thus, the learning instance can be viewed as a meta instance affecting the system indirectly by updating admitted MPC parameters. In contrast, in (Wabersich and Zeilinger (2018)) using a model predictive controller as a filter has been proposed, which the control input signal, given by a machine learning algorithm, has to pass before being applied. This filter verifies its admissibility based on a model to predict the resulting state trajectory and updates the signal if needed. It allows for flexibility in choice of the learning method used while constraint satisfaction is guaranteed. However, similar to (Bouffard et al. (2012)), it comes at the cost of increased conservatism. It also only uses prior knowledge to filter unsafe control action and therefore misses out possible enhancements utilizing prior knowledge since the control performance is mainly determined by the learned controller. This as well as approaches proposed in (Li and Bastani (2019); Zhang et al. (2019)) can be classified as *post-posed shielding* (Alshiekh et al.

⋆ This work is supported by IAV GmbH

(2017)) whereas the approach presented in the current work is characterized as *preemptive shielding*. Instead of filtering the NN output signal, we already restrict the neural network output set to inputs which are considered to be safe. Hereby, the learning instance acts as a feedforward control modifying the MPC reference input signal. Such a setup can be motivated by learning a controller that is too time consuming to apply in real time or to approximate a highly complex and unmodeled controller such as a human driver, while certain optimality as well as safety constraints are supposed to be incorporated. Our main contribution is an approach to combine an a priori unsafe neural network with a model predictive controller while preserving safety guarantees at the expense of a robust thus conservative Tube MPC. The remainder of the paper is structured as follows: In Section 2 we formulate our considered problem and continue in Section 3 describing how to incorporate constraints. This is done in two steps: First, we assure control input constraint satisfaction restricting the NN output layer. Secondly, we employ Tube MPC to deal with uncertainty caused by the NN. Subsequently we discuss both steps using an illustrative example and complete by summarizing the findings as well as give an outlook in Section 5.

1.1 Definitions and Nomenclature

In the following we denote the set of non-negative integers by $\mathbb{N}_{\geq 0}$ as well as the set of non-negative rational numbers by $\mathbb{R}_{\geq 0}$. Furthermore we define the Minkowski set addition given two sets $\mathcal{A} \subset \mathbb{R}^{\mathrm{n}}, \mathcal{B} \subset \mathbb{R}^{\mathrm{n}}$ by $\mathcal{A} \oplus \mathcal{B} := \{a + b : a \in \mathcal{A}, b \in \mathcal{B}\}$ and $\mathcal{C} := \mathcal{A} \ominus \mathcal{B}$ if $\mathcal{C} \oplus \mathcal{B} = \mathcal{A}$. The term $\bigoplus_{j=0}^{i-1}$ denotes a series of Minkowski set additions. Furthermore, we use the following definitions of (Borrelli et al. (2017)): Consider a polyhedron $\mathcal{P} = \{z \in \mathbb{R}^{\mathrm{n_z}} | Hz \leq g\}$, with $H \in \mathbb{R}^{\mathrm{n_p} \times \mathrm{n_z}}, g \in \mathbb{R}^{\mathrm{n_p}}$ and an affine mapping $f : z \in \mathbb{R}^{\mathrm{n_z}} \mapsto Az+b,\ A \in \mathbb{R}^{\mathrm{n_A} \times \mathrm{n_z}},\ b \in \mathbb{R}^{\mathrm{n_A}}$. We define $f \circ \mathcal{P} := \{y \in \mathbb{R}^{\mathrm{n_A}} : y = Az+b\ \forall z \in \mathbb{R}^{\mathrm{n_z}}, Hz \leq g\}$ and refer to it using $f\mathcal{P}$. Hence, the expression $F\mathcal{A}$, with F being a real matrix of compatible dimension, denotes the image of the set $\mathcal{A}$ under F which is defined by $F\mathcal{A} := \{Fa : a \in \mathcal{A}\}$. Moreover, with $a \in \mathbb{R}^{\mathrm{n_a}}$ and $Q \in \mathbb{R}^{\mathrm{n_a} \times \mathrm{n_a}}$ $\|a\|_{\mathrm{Q}}$ denotes the seminorm $||a||_{\mathrm{Q}} := a^T Q a$. In context of intervals we use $(\cdot, \cdot)$ for denoting open intervals and $[\cdot, \cdot]$ are understood as closed interval limits. The expression $\{v\}$ with $v \in \mathbb{R}^{\mathrm{n}}$ denotes the set consisting of vector v. Lastly, a positive invariant set Φ is a set such that for a given dynamical system $x^+ = f(x)$ with constraints $x \in \mathcal{X} \supseteq \Phi$ the following holds: $\forall x \in \Phi \Rightarrow x \in \mathcal{X} \wedge f(x) \in \Phi$.

2. PROBLEM FORMULATION

The basic setup of our approach is depicted in Fig. 1. The system consists of a suboptimal controller K and a nonlinear plant Σ. Furthermore, it includes a NN which receives some input data, $x(k), x_{\mathrm{r}}(k), u_{\mathrm{c}}(k), u_{\mathrm{c}}(k-1)$, and updates the input signal of the normal controller respectively to improve performance. The overall nonlinear system including all subsystems is given by:

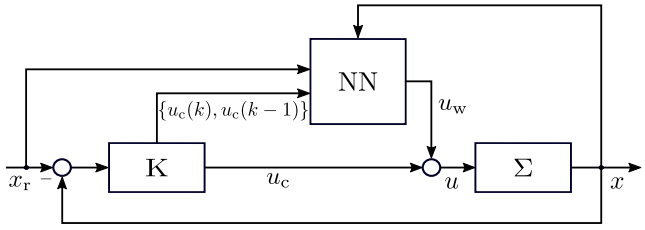


Fig. 1. Illustration of the considered control loop augmented by a neural network acting as a dynamical nonlinear feedforward controller.

$$\text{control input}: \quad u(k) = u_{\mathrm{c}}(k) + u_{\mathrm{w}}(k), \tag{1a}$$
$$\text{baseline controller K}: \quad u_{\mathrm{c}}(k) = \kappa(x_{\mathrm{r}}(k) - x(k)), \tag{1b}$$
$$\text{NN}: \quad u_{\mathrm{w}}(k) = h(x(k), x_{\mathrm{r}}(k), u_{\mathrm{c}}(k), u_{\mathrm{c}}(k-1)), \tag{1c}$$
$$\Sigma: \quad x(k+1) = f(x(k), u(k)), \quad x(0) = x_0. \tag{1d}$$

Here $x(k) \in \mathbb{R}^{\mathrm{n_x}}$ denotes the current state, $x(k+1)$ is the successor and x_0 is the initial state. Furthermore, $u_{\mathrm{c}}(k) \in \mathcal{U} \subset \mathbb{R}^{\mathrm{n_u}}$ is the control input and $u_{\mathrm{w}}(k) \in \mathcal{U}_w \subseteq \mathbb{R}^{\mathrm{n_u}}$ the NN update which is restricted to the set $\mathcal{U}_{\mathrm{w}}$ (cf. Section 3.1). The control policy $\kappa(x_{\mathrm{r}}(k) - x(k))$ is considered to be given by a model predictive controller employing a linear model. We assume the function $f : \mathbb{R}^{\mathrm{n_x}} \times \mathbb{R}^{\mathrm{n_u}} \to \mathbb{R}^{\mathrm{n_x}}$ to be a discretized form of a locally Lipschitz continuous function. Consequently, we can obtain a linear approximation of (1d) via a first-order Taylor polynomial around a local operating point $(x_{\mathrm{s}}, u_{\mathrm{c,s}}, u_{\mathrm{w,s}})$.

$$\Sigma_l: \quad x(k+1) = Ax(k) + Bu_{\mathrm{c}}(k), \tag{2a}$$
$$x(0) = x_0 \tag{2b}$$

Since this approximation leads, in general, to model mismatch and suboptimal control, the NN is injected to update the MPC control input signal in order to compensate for model inaccuracies. However, while the MPC is able to consider state and input constraints $x \in \mathcal{X}$ and $u \in \mathcal{U}$, applying arbitrary updates negates such guarantees. Thus, the following sections outline how to regard constraints in such a control structure, which we refer to as NN supported MPC. The basic idea is to limit the range of the NN by constraining the maximal correction and viewing the resulting influence as disturbance in a tube-based control sense.

3. INCORPORATING CONSTRAINTS TO NN SUPPORTED MPC

3.1 Guaranteeing input constraints and feasibility despite NN based learning

The additional input $u_{\mathrm{w}}(k)$ on Figure 1 is the outcome of the NN that maps the tuple $(x(k), x_{\mathrm{r}}(k), u_{\mathrm{c}}(k), u_{\mathrm{c}}(k-1))$ to a possible high dimensional latent space $\mathcal{S}$ and finally passes it through an output layer onto a control input set $\mathcal{U}_{\mathrm{w}} \subseteq \mathbb{R}^{\mathrm{n_u}}$. How to choose this subset is essential to the performance of the NN and thus for bounding its effect on the state and control trajectory. The choice directly affects closed loop guarantees. We introduce the parameters $p \in \mathcal{U}$, $r \in \mathbb{R}_{\geq 0}$ to define the set M with the metric $d(\cdot, \cdot)$ as follows:

$$\mathcal{U}_{\mathrm{w}} := \underbrace{\{\tilde{u} \in M | d(\tilde{u}, p) \leq r\}}_{\Delta_{\mathrm{w}}} \cap \big(\mathcal{U} \ominus \{u_{\mathrm{c}}(k)\}\big). \tag{3}$$

The set Δ_{w} is designed to limit the range of admissible inputs produced by the NN to a neighborhood of the

point p (up to a ball with a radius r), while the set M can incorporate other types of input constraints. Since NN is supposed to provide an auxiliary control action, one can additionally ensure that the set Δ_w includes the origin, as in this case the network is allowed to be inactive. We formalize the link between these parameters and satisfaction of the input constraints in the following theorem.

Theorem 1. Suppose $\forall k \in N_{\geq 0}$

i) $u_\mathrm{w}(k) \in \mathcal{U}_\mathrm{w}$,
ii) $u_\mathrm{c}(k) \in \mathcal{U}$,
iii) $0 \in \Delta_\mathrm{w}$, $r \in \mathbb{R}_{\geq 0}$, $p \in \mathcal{U} \supseteq M$,

then $u_\mathrm{c}(k) + u_\mathrm{w}(k) \in \mathcal{U}$.

Proof. From i) follows $u_\mathrm{c}(k) + u_\mathrm{w}(k) \in \{u_\mathrm{c}(k)\} \oplus \Big(\Delta_\mathrm{w} \cap \big(\mathcal{U} \ominus \{u_\mathrm{c}(k)\}\big)\Big) = \big(\Delta_\mathrm{w} \oplus \{u_\mathrm{c}(k)\}\big) \cap \mathcal{U}$. From ii) it follows that neither of these sets are empty, thus it only leaves to show that their intersection is not empty. Following iii), $\Delta_\mathrm{w} \oplus \{u_\mathrm{c}(k)\} \ni 0 + u_\mathrm{c}(k) = u_\mathrm{c}(k) \in \mathcal{U}$. □

Thus far we augmented the MPC controlled system with a feedforward control given by a NN such that input constraints are satisfied. Guaranteeing state constraints will require the use of prior knowledge about the NN in the MPC.

3.2 Regarding State Constraints in NN Supported MPC

As seen in the previous section, input constraints can easily be met since both inputs, u_c and u_w, do not have any dynamics and can be chosen freely within the input constraint set $\mathcal{U}$. On the other hand, the states have to adhere to their dynamics, which need to be modeled. Thus, we employ tube based MPC to ensure satisfaction of state constraints $x(k) \in \mathcal{X}\, \forall k \in N_{\geq 0}$.

3.3 Rigid Tube MPC

Tube based MPC is a collection of methods that enables consideration of persistent but bounded disturbances. Though its theoretical foundations are thoroughly covered in the literature, we outline next its basic idea. Interested readers are referred to (Mayne et al. (2005, 2006); Rakovic et al. (2003, 2016); Langson et al. (2004)). In order to represent our problem in the tube MPC setting we treat the additional input signal u_w produced by the NN as a purposeful disturbance.
The basic idea is to provide a safety margin for state x and control input u, enabling compensation of any possible "disturbance" $w \in \mathbb{W}$ from the NN. Fortunately the disturbance depends on the choice of $\mathcal{U}_\mathrm{w}$ since $\mathbb{W} = B\mathcal{U}_\mathrm{w}$. Thus, not only can we determine the set $\mathbb{W}$, but additionally we can decide about its extend: First we define the nominal and disturbed linear system (4),(5)

$$\Sigma_\mathrm{N}\colon z(k+1) = Az(k) + Bv(k),\ z(0) = z_0, \tag{4}$$
$$\Sigma_\mathrm{w}\colon x(k+1) = Ax(k) + Bu_\mathrm{c}(k) + \underbrace{Bu_\mathrm{w}(k)}_{w \in \mathbb{W}},\ x(0) = x_0, \tag{5}$$

where $v \in \mathcal{V}$, $z(k)$ denote the nominal control input as well as nominal state governed by the nominal system (4). We define an error coordinate $e(k) = x(k) - z(k)$ and subtract the nominal system (4) from the disturbed system (5) which yields a difference equation governing the error dynamics

$$e(k+1) = Ae(k) + Bu_\mathrm{c}(k) - Bv(k) + w(k). \tag{6}$$

For the purpose of preventing unstable and therefore unbounded error dynamics, we define $u_c(k) = v(k) + K_z(x(k) - z(k))$ and design an additional controller stabilizing the error between the nominal and the actual state. This yields

$$e(k+1) = \underbrace{(A + BK_\mathrm{z})}_{A_\mathrm{K}} e(k) + w(k). \tag{7}$$

For $w(k) \in \mathbb{W}\, \forall k \in \mathbb{N}_{\geq 0}$ the reachable set can be computed via

$$\Phi(i) = \bigoplus_{j=0}^{i-1} A_\mathrm{K}^j . \mathbb{W} \tag{8}$$

assuming $e(0) = 0$. From Theorem 1, iii) this is always admissible. Hence, for $i \to \infty$ it turns into the minimal positive invariant set of $e(k+1)$. However, deriving an infinite Minkowski-sum is not practical. Hence, we employ an upper bound proposed by (Rakovic et al. (2003)):

$$\Phi_\infty \leq (1 - \hat{\alpha})^{-1} \Phi_\mathrm{N}, \text{with } \Phi_\mathrm{N} = \bigoplus_{j=0}^{N-1} A_\mathrm{K}^j \mathbb{W}. \tag{9}$$

Where $\hat{\alpha} \in (0, 1)$ can be chosen to be arbitrarily small. Given the set Φ_N, the nominal state constraint set can be adjusted by $\mathcal{Z} = \mathcal{X} \ominus \Phi_\mathrm{N}$ and since the additional effort of the linear controller K_z also has to be taken into account, the tighter nominal input constraint set is given by $\mathcal{V} = \mathcal{U} \ominus K_\mathrm{z}\Phi_\mathrm{N}$. Finally in order to compute Φ_N offline we omit the intersection and simplify

$$B\Big(\Delta_\mathrm{w} \cap \big(\mathcal{U} \ominus \{u_\mathrm{c}(k)\}\big)\Big) \subseteq B\Delta_\mathrm{w} = \mathbb{W}. \tag{10}$$

Thus, conservatism is introduced in two places: First, from the approximation in (9), which can mostly be countered by choosing parameter $\hat{\alpha}$ sufficiently close to zero. And second, the simplification in (10), which shrinks the admissible state space $\mathcal{Z}$ of the nominal MPC.

The underlying optimization problem therefore takes the following form:

$$\min_{z(k), v(k)} \sum_{k=0}^{N} \|x_\mathrm{r}(k) - z(k)\|_\mathrm{Q} + \|v(k) - v(k-1)\|_\mathrm{R} \tag{11a}$$

subject to

$$z(k+1) = Az(k) + Bv(k), \tag{11b}$$
$$z(k) \in \mathcal{Z} \subseteq \mathcal{X}, \tag{11c}$$
$$v(k) \in \mathcal{V} \subseteq \mathcal{U}. \tag{11d}$$

Note, that since we employ an MPC with a sufficiently accurate linear model to guarantee state and input constraints, we can either preserve these guarantees by restricting the range of NN output space (cf. Section 3.1) or by enforcing state constraints using Tube MPC (TMPC).

4. ILLUSTRATIVE EXAMPLE

We provide two examples of NN augmented model predictive control to highlight possible advantages of these approaches. We focus on ensuring constraint satisfaction to show the effectiveness of applying TMPC. For both types of controllers, MPC and TMPC, we consider the problem of trajectory tracking. The controller brings a Continuous Stirred Tank Reactor (CSTR) to time-varying

state reference $x_{\mathrm{r}}(k)$. The nonlinear plant is given in the following form (Ozkan et al. (2002)):

$$\frac{\partial x_1}{\partial t} = -\alpha x_1 h(x_2) + q(x_{1,f} - x_1), \tag{12a}$$
$$\frac{\partial x_2}{\partial t} = \beta\alpha x_1 h(x_2) - (q+\sigma)x_2 + \sigma u + q x_{2,f}, \tag{12b}$$
$$y = x, \tag{12c}$$
$$h(x_2) = exp\Big(\frac{x_2}{1+x_2/\gamma}\Big). \tag{12d}$$

Here the states x_1, x_2 and input u denote the concentration, the temperature and the cooling jacket temperature. The remaining parameters are constant and their values are provided in Table 1. For solving the MPC, or NMPC problem we use IPOPT (Wächter and Biegler (2006)) in combination with the CasADi framework (Andersson et al. (2019)) implemented in Python. Furthermore, for set operations we used the Mulit-Parametric Toolbox 3 (Herceg et al. (2013)).

Table 1. CSTR model constants

α	β	γ	σ	$x_{1,f}$	$x_{2,f}$	q
0.072	8.0	20.0	0.3	1.0	0.0	1.0

4.1 NN Supported MPC

We first show the advantage of using a NN as a feedforward control in combination with a MPC. The linear model predictive controller solves an optimal control problem with input and state constraints given by (13a) at each sampling point, reacting to possible disturbances. Choosing $x_s = \begin{pmatrix}0.553\\2.752\end{pmatrix}$, $u_{\mathrm{c,s}} = 0$, $u_{\mathrm{w,s}} = 0$, $Q = \begin{pmatrix}0 & 0\\0 & 10\end{pmatrix}$, $R = (0.1)$, $N = 5$ and the sampling time $T_{\mathrm{s}} = 0.05s$, the MPC problem takes the following form:

$$\min_{u_{\mathrm{c}}(k)} \sum_{k=1}^{5} \|x_{\mathrm{r}}(k) - x(k)\|_{\mathrm{Q}} + \sum_{k=0}^{5} \|u_{\mathrm{c}}(k) - u_{\mathrm{c}}(k-1)\|_{\mathrm{R}} \tag{13a}$$

subject to

$$x(k+1) = Ax(k) + Bu_{\mathrm{c}}(k), \tag{13b}$$
$$x(k) \in \mathcal{X}, \tag{13c}$$
$$u_{\mathrm{c}}(k) \in \mathcal{U}. \tag{13d}$$

While the described concept generally admits on-line learning, we choose to pre-train the NN to simplify the example. The objective is to learn the difference between the linear MPC (13a) based on the model (13b) and a desired input $u^*(k)$ which is derived from a Nonlinear Model Predictive Controller (NMPC) $\kappa^*(x_{\mathrm{r}}(k) - x(k)) = u^*(k)$ based on (12). As a loss function we utilize the mean squared error loss

$$MSE = \frac{\sum_{i=1}^{N_{\mathrm{b}}} (u_i^* - u_i)^2}{N_{\mathrm{b}}} \tag{14}$$

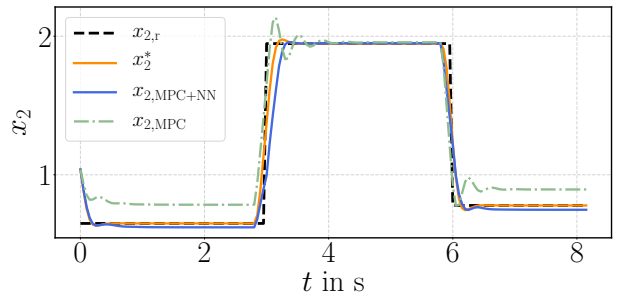
with $N_{\mathrm{b}} = 512$ denoting the size of a batch containing randomly sampled data points from a data set of $N_{\mathrm{D}} = 2^{12}$ values. This data consists of the current state x_k, a reference state $x_{\mathrm{r},k}$ as well as previous control inputs u_{k-1}, with the index k indicating time dependent variable that this data element represents after the training. For example, $x_{\mathrm{r},k}$ is replaced by $x_{\mathrm{r}}(k)$ after training. The data is generated by solving both the linear and the nonlinear optimization problems for tuples $(x_{k,j}, x_{\mathrm{r},k,j}, u_{k-1,j})_j$ to determine the corresponding labels $(u^*_{k,j} - u_{k,j})$ for $j = 1, 2, ...N_{\mathrm{D}}$. The NN is composed of fully-connected layers, where an input with 6 neurons is followed by $N_{\mathrm{l}} = 3$ hidden layers with 16 neurons each and a single neuron for the output layer. For all but the last layer a *ReLU*-function is chosen as an activation function. For training we employ the *ADADELTA* algorithm proposed by (Zeiler (2012)). To guarantee that assumption i) of Theorem 1 holds, we choose as output activation function similar as in (Hausknecht and Stone (2018)) a saturating *tanh*-function. The parameter p is set to zero and the input set is $\mathcal{U} = [-4, 4]$. Moreover, we choose $\Delta_{\mathrm{w}} = \{u \in M | \ \|u, 0\|_\infty < 2.4\}$ and the intersected set in (3) is simplified to $\mathcal{U}_{\mathrm{w}}(k) = [\underline{u}_{\mathrm{w}}(k), \overline{u}_{\mathrm{w}}(k)]$, where $\underline{u}_{\mathrm{w}}(k), \overline{u}_{\mathrm{w}}(k)$ denote lower and upper set boundaries. As a result, the scaled NN output is given by (15) with the neural network latent state, weights and biases $s_{\mathrm{N}}, W_{\mathrm{N}}, b_{\mathrm{N}}$ of the last hidden layer:

$$\gamma = \frac{1}{2}(\overline{u}_{\mathrm{w}}(k) - \underline{u}_{\mathrm{w}}(k)), \tag{15a}$$
$$c = \gamma + \underline{u}_{\mathrm{w}}(k), \tag{15b}$$
$$\sigma(s_N) = \gamma \tanh(W_{\mathrm{N}} s_{\mathrm{N}} + b_{\mathrm{N}}) + c, \tag{15c}$$

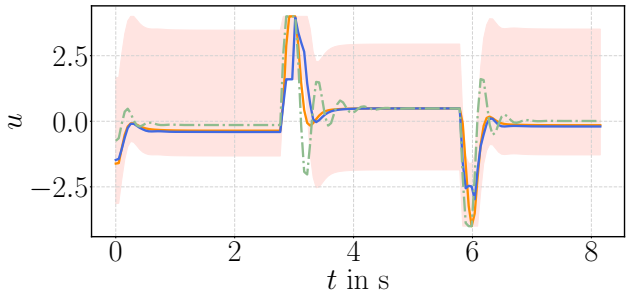
Simulation Results and Discussion We simulated a tracking experiment with three steps to illustrate the varying performances at different points in state space as shown in Fig. 2. Since the NN learned the difference between the optimal control input u^* and the input u_{MPC}, we first compare different control trajectories as shown in Fig. 2b. Comparing the stand-alone MPC (depicted as green dashed line) to the NN supported MPC (represented as a blue line), we notice that our approach succeeds in approximating the optimal NMPC control trajectory (orange line) as steady state deviations as well as largest oscillation are dampened. Furthermore, the red coloured area in Fig. 2b (not to be confused with tubes of TMPC) visualizes the set where the control trajectory $u_{\mathrm{MPC+NN}}$ can possibly lie due to the output scaling described in Section 4.1. Note that this area is included within the input constraints and therefore preserves them (cf. also (3)). However, we also note that in particular for the beginning of transients the performance of NN supported MPC is worse since the NN does not receive future reference signals. Even though this does not cause state constraint violation, it shows the necessity for further coupling MPC and NN to retain state constraint satisfaction. This leads us to the second example – NN supported Tube MPC.

4.2 Tracking with NN supported Tube MPC

We now incorporate the knowledge about the output constraint NN into the controller using Tube MPC. In order to apply (11) we augment our nominal controller by the error $e(k) = x(k) - z(k)$ stabilizing controller K_{z}. To do so a linear quadratic regulator is chosen with the objective matrices $Q_{\mathrm{z}} = \begin{pmatrix}0 & 0\\0 & 10\end{pmatrix}$, $R_{\mathrm{z}} = (1)$. Given this and setting $r = 0.6$, $p = 0$ we compute the set $\mathbb{W}$ using (9) which we utilize to compute the tighter sets $\mathcal{Z}, \mathcal{V}$. Considering the NN we replace u_{c} with $\tilde{u}_{\mathrm{c}} = u_{\mathrm{MPC}} + u_{\mathrm{K_z}}$ so the NN receives additional information and learns to compensate suboptimality of the MPC as well as the



(a) Tracking experiment: State trajectories of different control approaches



(b) Tracking experiment: Control and input trajectories for different set points

Fig. 2. Performance comparison: Step tracking experiment to compare performance of NN supported MPC versus MPC

additional controller K_z. Moreover we substitute $u_\mathrm{c}(k)$ in (3) with $\tilde{u}_\mathrm{c}(k)$ to ensure input constraint satisfaction using Theorem 1.

Simulation Results and Discussion In order to illustrate the correct consideration of the potentially disturbing NN influence using Tube MPC approach Fig. 3 shows an open loop stabilization of the linearized plant. Here, we set the horizon to $N = 15$, sampling time is $T_\mathrm{s} = 0.45s$ and the initial state as well as the reference point are set to

$$x_0 = \begin{pmatrix} 0.925 \\ 2.156 \end{pmatrix}, \quad x_\mathrm{r} = \begin{pmatrix} 0.906 \\ 0.902 \end{pmatrix}, \quad u_{k-1} = (2.639)\,.$$

Moreover, instead of the previous feedforward NN we use a residual NN with the same structure of neurons and layers. Since we only want to show robustness against disturbances originating in a malfunctioning NN, we simulate the trajectory with the simplified model, that is used by the linear MPC. Hence, we cannot expect to exceed the performance of the system controlled solely by the MPC. This explains why z_k as well as x^*, which use the exact model, converge to the reference state x_r, while the actual state x_k converges to another state induced by the NN. However, the states stay within the admissible tube at all times. Furthermore, the nominal state trajectory clearly shows that constraints set $z_k \in \mathcal{Z}$ (highlighted in grey), as well as $x_k \in \mathcal{X}$ are respected during the optimization, since neither the nominal state nor the tube leave $\mathcal{Z}$ or $\mathcal{X}$. Thus, at the cost of conservatism from the Tube MPC approach, we retain constraint satisfaction.

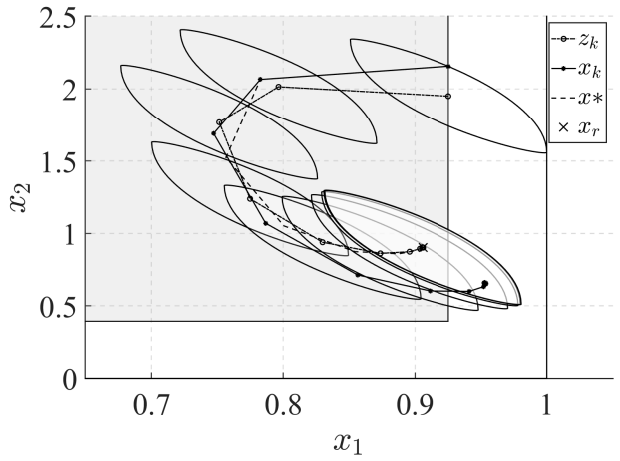


Fig. 3. Phase diagram showing MPC as well as Tube MPC controlled state trajectory and the corresponding tubes. It illustrates state constraint satisfaction while stabilizing around reference point x_r.

Next, Fig. 4 shows the same tracking experiment as in Fig. 2 with MPC substituted for TMPC. While we observe a slower convergence in the beginning of Fig. 4a, the NN augmented Tube MPC outperforms the simple MPC during the steady phase. This can be explained by the input trajectories depicted in Fig. 4b. The allowed range is smaller than in the previous experiment because the tolerated interference is down scaled to $r = 0.6$. This results in the optimal trajectory u^* being located outside of set $\mathcal{U}$ in the beginning. However, as soon as u^* enters $\mathcal{U}_\mathrm{w} + u_\mathrm{c}$, the NN augmented TMPC approximates the optimal trajectory well and therefore the state approaches the desired reference.

5. CONCLUSION AND OUTLOOK

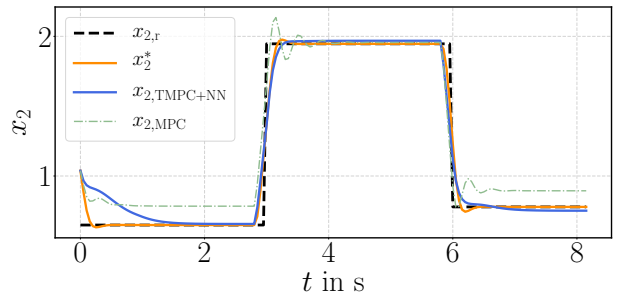
We proposed a safe way of augmenting a model predictive controller with a neural network by using latter as a nonlinear feedforward control such that the modified control input signal approximates a desired input signal. To do so we proposed to restrict the last layer of a neural network in order to limit the range of its inputs. To satisfy state constraints we suggested employing tube MPC. Finally, we tested both control structures in simulation for tracking performance in comparison to the nominal controller. In future work we will investigate adaptable tubes whose extends are dynamically given by the MPC. Moreover, we aim to use this approach to safely approximate more challenging controllers such as human drivers.

REFERENCES

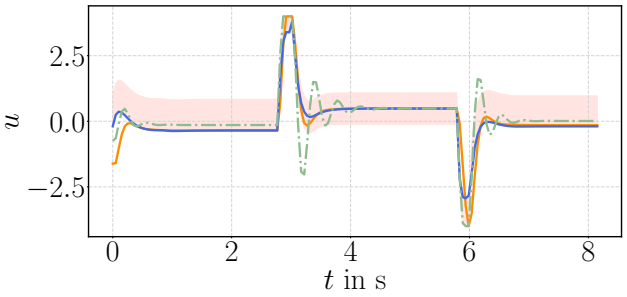
Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., and Topcu, U. (2017). Safe reinforcement learning via shielding. *CoRR*, abs/1708.08611.

Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., and Diehl, M. (2019). CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1), 1–36. doi: 10.1007/s12532-018-0139-4.

Bethge, J., Morabito, B., Matschek, J., and Findeisen, R. (2018). Multi-mode learning supported model pre-



(a) Tracking experiment: State trajectories of differently controlled systems



(b) Tracking experiment: Control trajectories of different control approaches

Fig. 4. Performance comparison: Step tracking experiment to compare performance of NN augmented Tube MPC versus MPC

dictive control with guarantees. *IFAC-PapersOnLine*, 51(20), 517–522. doi:10.1016/j.ifacol.2018.11.037. URL `http://www.sciencedirect.com/science/article/pii/S2405896318326946`.

Bojarski, M., Testa, D.D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). End to end learning for self-driving cars. *CoRR*, abs/1604.07316.

Borrelli, F., Bemporad, A., and Morari, M. (2017). *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press. doi:10.1017/9781139061759.

Bouffard, P., Aswani, A., and Tomlin, C. (2012). Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *2012 IEEE International Conference on Robotics and Automation*, 279–284. doi:10.1109/ICRA.2012.6225035.

Gros, S. and Zanon, M. (2019). Towards safe reinforcement learning using nmpc and policy gradients: Part i - stochastic case. *CoRR*, abs/1906.04057.

Hausknecht, M. and Stone, P. (2018). Deep reinforcement learning in parameterized action space. In *2018 IEEE International Symposium on Multimedia (ISM)*, 101–104.

Herceg, M., Kvasnica, M., Jones, C., and Morari, M. (2013). Multi-Parametric Toolbox 3.0. In *Proc. of the European Control Conference*, 502–510. Zürich, Switzerland. `http://control.ee.ethz.ch/ mpt`.

Huang, S.H., Papernot, N., Goodfellow, I.J., Duan, Y., and Abbeel, P. (2017). Adversarial attacks on neural network policies. *CoRR*, abs/1702.02284.

Langson, W., Chryssochoos, I., Rakovic, S.V., and Mayne, D.Q. (2004). Robust model predictive control using tubes. *Automatica*, 40, 125–133.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2015). End-to-end training of deep visuomotor policies. *CoRR*, abs/1504.00702.

Li, S. and Bastani, O. (2019). Robust model predictive shielding for safe reinforcement learning with stochastic dynamics. *arXiv preprint arXiv:1910.10885*.

Mayne, D.Q., Raković, S.V., Findeisen, R., and Allgöwer, F. (2006). Robust output feedback model predictive control of constrained linear systems. *Automatica*, 42(7), 1217–1222. doi:10.1016/j.automatica.2006.03.005. URL `http://dx.doi.org/10.1016/j.automatica.2006.03.005`.

Mayne, D.Q., Seron, M.M., and Raković, S.V. (2005). Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41(2), 219–224. doi:10.1016/j.automatica.2004.08.019.

Ozkan, L., Kothare, M.V., and Georgakis, C. (2002). Control of a solution copolymerization reactor using piecewise linear models. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, volume 5, 3864–3869 vol.5. doi:10.1109/ACC.2002.1024531.

Rakovic, S.V., Kerrigan, E.C., Kouramas, K., and Mayne, D.Q. (2003). Approximation of the minimal robustly positively invariant set for discrete-time lti systems with persistent state disturbances. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, volume 4, 3917–3918 vol.4. doi:10.1109/CDC.2003.1271761.

Rakovic, S.V., Levine, W.S., and Açikmese, B. (2016). Elastic tube model predictive control. In *2016 American Control Conference (ACC)*, 3594–3599. doi:10.1109/ACC.2016.7525471.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529, 484–503. URL `http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html`.

Wabersich, K.P. and Zeilinger, M.N. (2018). Safe exploration of nonlinear dynamical systems: A predictive safety filter for reinforcement learning. *CoRR*, abs/1812.05506.

Wächter, A. and Biegler, L.T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1), 25–57. doi:10.1007/s10107-004-0559-y.

Zanon, M., Gros, S., and Bemporad, A. (2019). Practical reinforcement learning of stabilizing economic mpc. *CoRR*, abs/1904.04614.

Zeiler, M.D. (2012). Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701.

Zhang, W., Bastani, O., and Kumar, V. (2019). MAMPS: Safe multi-agent reinforcement learning via model predictive shielding.