2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Quantized Deep Neural Network Based Optimal Control of Greenhouses on a Microcontroller

Kiran Kumar Sathyanarayanan[1], Philipp Sauerteig[1], Pablo Zometa[3] and Stefan Streif[1,2]

Abstract— **Growing crops in Controlled-Environment Agriculture (CEA) farms, such as greenhouses and vertical farms, can help in meeting the demands of urban centers and achieving climate goals. Recently, many advanced control techniques like Model Predictive Control (MPC) and its variants have been developed for energy-efficient operation and minimization of resource utilization. However, real-time implementation of these advanced strategies come along with certain computational hardware requirements, thus, increasing the operating costs. In this work, we propose to learn the MPC policy of a greenhouse control by means of a Deep Neural Network (DNN) in order to be implemented on a low-cost microcontroller. Additionally, we use a feedback law to reduce undesired quantization effects. The efficiency of our approach is exemplified by means of a simulation study for greenhouse control.**

I. Introduction

Energy-efficient and sustainable operation of CEA farms requires automatic control of the climate influencing the crop growth. This includes good control performance with respect to crop growth on the one hand, but also minimal energy usage on the other hand. While many control strategies have been proposed by researchers for greenhouse control, MPC performs best, resulting in energy reduction [1]. Hierarchical Model Predictive Control (HMPC) in the crop production process based on timescale decomposition eases the implementation of receding horizon predictive control [2], [3].

The requirement of high initial investments for fully automated CEA systems and rising energy expenses, increase the total costs of crop production [4]. Additionally, the implementation of advanced control strategies like MPC involves solving complex optimization problems in real time placing a strain on computational resources [5]. Cloud-based MPC eases such shortcomings by solving the optimization problem upon the reception of states from clients and returning the inputs [6]. Such techniques led to the development of Internet of Things (IoT) based control of greenhouses with the help of wireless sensors and actuators [4]. However, the implementation of cloud-based computation requires initial investment or availing Control as a Service (CaaS).

Alternatively, the usage of DNNs to approximate the solution of the predictive control problem is discussed, e.g. in [7], [8]. Moreover, in [9] the authors approximate a robust Nonlinear Model Predictive Controll (NMPC) scheme and implement the DNN based controller on an MicroController Unit (MCU) using single-precision floating-point arithmetic during network inference. The efficiency of DNNs can be further increased with the help of quantization. Quantization is a method of store the network parameters using a fixed-point representation instead of floating points [10]. The quantized DNN executes faster, requires less memory and energy, but the price to pay is a loss of numerical accuracy.

In this work, a two-level hierarchical control is implemented for a semi-closed solar collector greenhouse. The upper level generates suitable reference trajectories on solving an economic optimization problem based on day-ahead weather forecasts. On the lower level, a reference-tracking NMPC is employed to track the generated references on satisfying the desired bounds.

The present paper investigates the use of quantized DNN for the lower level NMPC. Correspondingly, the HMPC problem is solved for around eight months to generate the lower level NMPC state, control input, reference and disturbance data pairs for network training. Then, the reference tracking in the HMPC is implemented by the approximate DNN controller instead of the exact NMPC controller. Thereafter, we extend the DNN controller with a simple LQR controller to account for the errors caused by the quantization approximation of the DNN. Based on hardware-in-the-loop simulations of the proposed approach on an MCU, we show that the quantized DNN along with LQR requires less memory while achieving a better tracking performance and being several orders of magnitude faster than NMPC. The MCU implementaion of the proposed approach can be deployed in small food production units as discussed in [11], which is more desirable in refugee camps and famine affected areas.

Section II recalls the greenhouse dynamics and the HMPC scheme. The quantization of the DNN is discussed in Section III. In Section IV, we devise an approach to efficiently approximate the reference tracking problem using a quantized DNN. Section V is dedicated to review the performance of the proposed method, before we conclude the paper in Section VI.

II. Greenhouse Modeling and Control

This section summarizes the Hierarchical Model Predictive Control (HMPC) of the greenhouse using a mathematical model of the greenhouse climate and the biomass yield.

A. System Description

We consider a Venlo-type semi-closed greenhouse equipped with mechanical and electrical systems for climate control. The greenhouse climate is defined in terms of

The authors are with [1]Technische Universität Chemnitz, Chemnitz, Germany, Automatic Control and System Dynamics Lab, [2]Fraunhofer Institute for Molecular Biology and Applied Ecology, Department of Bioresources, Giessen, Germany, and [3]Faculty of Engineering, German International University, Berlin, Germany. E-mail: {kiran.sathyanarayanan, philipp.sauerteig, stefan.streif}@etit.tu-chemnitz.de; pablo.zometa@giu-berlin.de.

temperature T, carbon dioxide concentration C, and absolute humidity H of the air inside the greenhouse. To develop an HMPC framework for greenhouse control, we recall the model of the greenhouse climate and the crop growth developed in [12]. States, control inputs, and disturbances are listed in Table I. Inputs are normalized to $[0, 1]$ and unitless.

TABLE I: List of states, inputs, and disturbances

Symbol	Description	Unit
T	temperature of air inside greenhouse	$^\circ$C
C	CO_2 of air inside greenhouse	$\mathrm{g\,m^{-3}}$
H	absolute humidity of air inside greenhouse	$\mathrm{g\,m^{-3}}$
B	fresh biomass weight of the crop	$\mathrm{kg\,m^{-2}}$
u_{V}	ventilation input	-
u_{C}	CO_2 injection input	-
$u_{\mathrm{Q_h}}$	heater input	-
$u_{\mathrm{Q_c}}$	cooling input	-
T_{out}	temperature of outdoor air	$^\circ$C
C_{out}	CO_2 concentration of outdoor air	$\mathrm{g\,m^{-3}}$
H_{out}	absolute humidity of outdoor air	$\mathrm{g\,m^{-3}}$
Q_{rad}	outdoor shortwave solar radiation	$\mathrm{W\,m^{-2}}$

We briefly recap the dynamic equations for greenhouse climate and tomato growth, for details we refer to [12]. The greenhouse air temperature is affected by various heat fluxes and the energy balance equation is given by

$$k_{\mathrm{C,gh}}\dot{T} = Q_{\mathrm{sun}} + Q_{\mathrm{vent}} + Q_{\mathrm{cov}} + Q_{\mathrm{trans}} + Q_{\mathrm{heat}} - Q_{\mathrm{cool}}. \quad (1)$$

Here, $k_{\mathrm{C,gh}}$ is the total heat capacity of the greenhouse per unit area, Q_{sun} is the incoming solar radiation, Q_{vent} is the heat flux due to vent opening, Q_{cov} is the convective heat loss through the cover, Q_{trans} is heat energy absorbed due to crop evapotranspiration, Q_{heat} and Q_{cool} are the heat fluxes due to heating and cooling devices, respectively.

The absolute humidity of the air influenced by various vapor fluxes is modeled via the mass balance equation

$$k_{\mathrm{h}}\dot{H} = H_{\mathrm{trans}} - H_{\mathrm{vent}} - H_{\mathrm{cov}} + H_{\mathrm{heat}} - H_{\mathrm{cool}}, \quad (2)$$

where k_{h} is the ratio of greenhouse volume to area, H_{trans} is the vapor flux due to crop transpiration, H_{vent} is the humidity change due to air exchange via vent opening, H_{cov} is the vapor condensation to the cover, H_{heat} and H_{cool} are the vapor fluxes due to the heating and cooling devices, respectively. Similarly, denoting the net CO_2 consumption due to photosynthesis by C_{phot}, the mass flux equation of the CO_2 concentration inside the greenhouse is

$$k_{\mathrm{h}}\dot{C} = C_{\mathrm{inj}} + C_{\mathrm{vent}} - C_{\mathrm{phot}}. \quad (3)$$

The CO_2 exchange due to ventilation and the industrial CO_2 supply is described respectively as

$$C_{\mathrm{vent}} = k_{\mathrm{u,vent}} \cdot u_{\mathrm{V}}(C_{\mathrm{out}} - C), \quad C_{\mathrm{inj}} = u_{\mathrm{C}} \cdot C_{\max}, \quad (4)$$

where $C_{\max}$ is the maximum injection rate.

Finally, the biomass production is given by

$$k_{\mathrm{A,s}}\dot{B} = k_{\mathrm{B,CO_2}} \cdot \phi_{\mathrm{CO_2}}, \quad (5)$$

where $\phi_{\mathrm{CO_2}}$ is the net photosynthesis rate of the canopy and $k_{\mathrm{B,CO_2}}$ is the conversion factor to convert consumed CO_2 into fresh-weight biomass.

The resulting dynamical system is of input-affine form and is summarized as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{d}(t)) + g(\mathbf{x}(t), \mathbf{d}(t))\mathbf{u}(t) \quad (6)$$

with state vector $\mathbf{x} = \begin{bmatrix} T & C & H & B \end{bmatrix}^\top$, input vector $\mathbf{u} = \begin{bmatrix} u_{\mathrm{V}} & u_{\mathrm{C}} & u_{\mathrm{Q_h}} & u_{\mathrm{Q_c}} \end{bmatrix}^\top$, and disturbance vector $\mathbf{d} = \begin{bmatrix} T_{\mathrm{out}} & C_{\mathrm{out}} & H_{\mathrm{out}} & Q_{\mathrm{rad}} \end{bmatrix}^\top$.

The greenhouse climate directly affects the crop production. Crop growth may be inhibited by a very high or low greenhouse air temperature [13]. Furthermore, low humidity ceases the photosynthesis process due to stomatal closure and high humidity causes molds to grow on the plants [14]. Studies by, e.g. [15] show that elevation of the CO_2 concentration can compensate for low radiation intensities. The constraints of the states and inputs due to actuators is expressed as $\mathbf{x}(t) \in \mathscr{X} \subset \mathbb{R}^4$ and $\mathbf{u}(t) \in \mathscr{U} \subset \mathbb{R}^4$, respectively.

B. Optimal Greenhouse Control

We recap the HMPC framework of greenhouse control in [12]. The control framework consists of an open-loop economic optimization on the upper level and a reference tracking NMPC on the lower level. With an objective to maximize yield at economic costs, the upper level generates optimal reference input $\mathbf{u}^{\mathrm{ref}}$ and state $\mathbf{x}^{\mathrm{ref}}$ trajectories for a single day based on the forecast data of $\mathbf{d}$. The optimization problem (7) is considered with sampling time $\delta t > 0$, and the prediction horizon $t_{\mathrm{f}} = N\delta t$ with $N \in \mathbb{N}^+$. The stage and terminal costs for the reference-tracking NMPC are given as

$$l(\mathbf{x}(t), \mathbf{u}(t)) = \|\mathbf{x}(t) - \mathbf{x}^{\mathrm{ref}}(t)\|_Q^2 + \|\mathbf{u}(t) - \mathbf{u}^{\mathrm{ref}}(t)\|_R^2,$$
$$V(\mathbf{x}(t_{\mathrm{f}})) = \|\mathbf{x}(t_{\mathrm{f}}) - \mathbf{x}^{\mathrm{ref}}(t_{\mathrm{f}})\|_P^2,$$

respectively, with weighting matrices $Q \succeq 0$, $R \succ 0$, and $P \succeq 0$. The Optimal Control Problem (OCP) solved repeatedly within the NMPC scheme reads as

$$\begin{aligned} \arg\min_{\mathbf{u}} \quad & \int_{t_0}^{t_{\mathrm{f}}} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\mathrm{d}\tau + V(\mathbf{x}(t_{\mathrm{f}})) \\ \text{s.t.} \quad & \dot{\mathbf{x}}(\tau) = f(\mathbf{x}(\tau), \mathbf{d}(\tau)) + g(\mathbf{x}(\tau), \mathbf{d}(\tau))\mathbf{u}(\tau) \\ & \mathbf{x}(t_0) = \mathbf{x}_0,\ \mathbf{u}(\tau) \in \mathscr{U},\ \mathbf{x}(\tau) \in \mathscr{X}, \end{aligned} \quad (7)$$

where x_0 is the initial value. Although the OCP is formulated in continuous time, our NMPC implementation is done in discrete time. The discretization of the OCP (7) results in $\mathbf{u}^* = (\mathbf{u}_0^*, \cdots, \mathbf{u}_{N-1}^*)^\top$, with $\mathbf{u}_0^*$ being applied to (6). The NMPC feedback controller based on (7) can be expressed as

$$\mathbf{u} = f_{\mathrm{MPC}}(\mathbf{w}), \quad (8)$$

with $\mathbf{w} = \begin{bmatrix} \mathbf{x} & \mathbf{x}^{\mathrm{ref}} & \mathbf{u}^{\mathrm{ref}} & \mathbf{d} \end{bmatrix}^\top$, and $\mathbf{w} \in \mathbb{R}^{16}$.

III. Quantized Deep Neural Network Approach

This section shortly recaps the fundamental concepts of neural networks with the aim to approximate the lower level reference-tracking NMPC. Following that, the process of quantization to convert and store the network parameters as fixed-point arithmetic is discussed.

A. Feedforward Neural Network

The feedforward Neural Network (NN), also known as multi-layer perceptron, consists of an input layer, an output layer, and L hidden layers that can approximate any function mapping with arbitrarily small error [16]. For $L \geq 2$ one speaks of a Deep NN (DNN). The function mapping $f_{\text{NN}} : \mathbb{R}^{n_{\mathbf{w}}} \to \mathbb{R}^{n_{\mathbf{u}}}$ of a feedforward NN is of the form

$$f_{\text{NN}}(\mathbf{w}, \theta, L, n_l) = H_{L+1} \circ g_L \circ H_L \circ \cdots \circ g_1 \circ H_1(\mathbf{w}), \quad (9)$$

where $\mathbf{w}$ is the NN input, n_l is the number of neurons in each hidden layer l, and θ is the vector containing the unknown NN parameters. The component H_l of the NN consists of an affine function $H_l = W_l H_{l-1} + b_l$, where H_{l-1} is the output of the previous hidden layer with $H_0 = \mathbf{w}$, b_l is the bias vector, and W_l is the weight matrix of layer l. Moreover, g_l is the activation function of the layer l. With the greenhouse system being highly nonlinear, we choose the most commonly used nonlinear activation function ReLU, which computes the elementwise maximum between 0 and H_l, i.e., $g_l = \max(0, H_l)$.

Our aim is to approximate the control policy (8) by defining the mapping $\mathbf{u}^D = f_{\text{NN}}(\mathbf{w}, \theta)$. After defining an architecture, the NN is trained by minimizing the specified loss function to determine the optimal W_l and b_l. Note that the length N_θ of the parameter vector $\theta = (b_1, W_1, \cdots, b_{L+1}, W_{L+1})^\top$ determines the memory footprint of the network.

To train our network, we rely on a training data matrix $\mathcal{D} \in \mathbb{R}^{n_{\mathbf{w}} \times N_D}$, which is obtained by running the NMPC problem for various initial conditions. Here, N_D represents the number of training data points and $n_{\mathbf{w}}$ is the number of components of a data point $\mathbf{w}$. Normalizing the training data helps to enhance the numerical properties of the network. The normalization operation on the data matrix $\mathcal{D}$ at row i and column j of the network $\mathbf{w}_i^j$ is defined by

$$\tilde{\mathbf{w}}_i^j = \mathcal{N}(\mathbf{w}_i^j, \mu_i, \sigma_i) \coloneqq (\mathbf{w}_i^j - \mu_i)\sigma_i^{-1},$$

where μ_i is the mean and σ_i is the standard deviation of the row (parameter) i. The above transformation results in a normalized data matrix $\mathcal{D}_N$ with $\tilde{\mu}_i = 0$ and $\tilde{\sigma}_i = 1$, i.e., each parameter of $\mathbf{w}$ is normalized. Similarly, inverse transformation $\mathrm{w}_i^j = \mathcal{N}^{-1}(\tilde{\mathrm{w}}_i^j, \mu_i, \sigma_i) \coloneqq \tilde{\mathrm{w}}_i^j \sigma_i + \mu_i$, is applied during the NN inference to recover the output in the range of the training data.

B. Quantized DNN (QDNN)

Quantization is the process of reducing the precision of parameters θ and activations g_l of a NN such that they consume less memory. The main advantages of quantization are reduced memory, lower network latency, and better power efficiency. However, the QDNN suffers from precision loss, underflow and overflow during the inference [10].

The trained neural networks generally use single-precision floating point to store the parameter vector θ. In general, quantization involves storing the θ in 8-bit scaled integer data types (i8) to support the NN inference using a low-power microcontrollers or FPGA. Out of many available quantization methods, we used uniform asymmetric quantization [10]. In this method, the normalized inputs $\tilde{\mathbf{w}}$ of the network are transformed from a floating point number to an integer via

$$\hat{\mathbf{w}} = \mathcal{Q}(\tilde{\mathbf{w}}, S, Z) \coloneqq \text{i8}(\tilde{\mathbf{w}}/S) - Z, \quad (10)$$

where S is a real-valued scaling factor, Z is an integer zero point or zero offset, and the i8 function maps a floating point to an 8-bit integer representation through a rounding operation. In asymmetric quantization, the min/max of the real-valued signal is used as clipping range for finding the scaling factor S. Similar to quantization, the output of the QDNN $\hat{\mathbf{u}}$ must be dequantized to a floating-point normalized output $\tilde{\mathbf{u}}$ using

$$\tilde{\mathbf{u}} = \mathcal{Q}^{-1}(\hat{\mathbf{u}}, S, Z) \coloneqq S(\hat{\mathbf{u}} + Z). \quad (11)$$

However, the output of $\mathcal{Q}^{-1}$ suffers from precision loss resulting in inference error. Here, the scaling and offset parameters in (10) are determined using the normalized data $\mathcal{D}_N$. In the consecutive section, we provide a linear-quadratic regulator to account for the precision loss.

IV. Quantized Deep Neural Network based NMPC

Now, the QDNN described in Section III is used to approximate the solution to the NMPC problem presented in Section II. Furthermore, the augmentation of an online feedback linear-quadratic regulator (LQR) to improve the accuracy of the reference tracking is discussed. We denote the approach as QDNN+LQR.

A. Feedback Compensation

The main motivation for using a QDNN is that it can be deployed on platforms with limited computational capabilities, like microcontrollers. To improve the closed-loop performance of such networks without significantly increasing the computational demands, a simple online feedback compensation term can be added. In [17], two proportional controllers are used to reduce the error introduced by a QDNN in a robotic path-following application. Although it may be possible to similarly apply a combination of several independent single-input single-output (SISO) controllers (e.g., proportional integral) to compensate for errors in the QDNN that approximates (8), it may be difficult to tune such independent controllers due to the interdependencies of the states in the system dynamics (6). To overcome such limitations, here we propose to use a linear multi-input multi-output (MIMO) controller, specifically a linear-quadratic regulator.

B. Linear-Quadratic Regulator (LQR)

An LQR is a MIMO control method based on the same optimal control principles as MPC, however, without state and control constraints. The LQR problem is stated as:

$$\begin{aligned} \mathbf{u}^* &= \arg\min_{\mathbf{u}} \int_0^\infty \ell(\mathbf{x}(\tau), \mathbf{u}(\tau)) \mathrm{d}\tau \\ \ell(\mathbf{x}, \mathbf{u}) &= \mathbf{x}^\top \underline{Q} \mathbf{x} + \mathbf{u}^\top \underline{R} \mathbf{u}, \\ \dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u}. \end{aligned} \quad (12)$$

The optimal control input that minimizes the value of the cost above is given by the feedback law $\mathbf{u}^* = -K\mathbf{x}$, where $K = \underline{R}^{-1}B^\top \underline{P}$, and $\underline{P}$ is the solution to the continuous-time algebraic Riccati equation. For stabilizable (A, B), and $\underline{Q} \succeq 0$ and $\underline{R} \succ 0$, the closed-loop system $\dot{\mathbf{x}} = (A - BK)\mathbf{x}$ is guaranteed to be globally asymptotically stable [18]. Note that in contrast to NMPC (7), in the LQR problem (12):

1) input and state constraints are not considered,
2) the system must be linear,
3) we stabilize around the origin (no reference tracking),
4) an explicit expression for the control law is obtained,
5) an infinite horizon is used, which guarantees stability.

In the context of CEA, the last two points may be considered advantageous, however, points 1), 2), and 3) are important limitations. Note that we do not propose to replace the NMPC setup described in Section II with an LQR, but rather to use the LQR to improve the QDNN approximation of the NMPC presented in Section III. In the following, we discuss the limitations mentioned above in more detail and how they relate to the problem described in Section II.

1) Input and state constraints: In our setup, the control input is subject to box constraints. These constraints can be enforced by saturating the input. The theoretical global stability guarantees of the LQR are no longer valid, however. In practice, with a suitable selection of matrices $\underline{Q}$ and $\underline{R}$, the closed-loop system remains locally stable (see Section V). The state constraints cannot be enforced by an LQR, but by careful selection of the matrices $\underline{Q}$ and $\underline{R}$, together with the QDNN base behaviour, an acceptable closed-loop response can be achieved that *mostly* respects the state constraints.

2) Linearization: Linearization of the nonlinear system (6) is done around an equilibrium point, i.e. a point $\bar{\mathbf{x}}, \bar{\mathbf{u}}, \bar{\mathbf{d}}$ such that $\dot{\mathbf{x}} = f(\bar{\mathbf{x}}, \bar{\mathbf{d}}) + g(\bar{\mathbf{x}}, \bar{\mathbf{d}})\bar{\mathbf{u}} = 0$. Due to the nonlinear interdependencies among the states, in particular, the biomass (5), it was not possible to find an equilibrium point for system (6). Observe, however, that the biomass does not directly depend on the inputs. In fact, the inputs have a direct and faster effect on the rate of change of the other state variables, which in turn affect the rate of change of the biomass. Thus, in the following we consider that $\dot{B} = 0$, and we reduce our nonlinear system to only consider the variables temperature T, CO_2 concentration C, and absolute humidity H as in (1)–(3). Note that, $\dot{B} = 0$ implies $C_{\text{phot}} = 0$, thus, (3) reduces to

$$k_{\text{V,gh}}\dot{\underline{C}} = C_{\text{inj}} + C_{\text{vent}}. \tag{13}$$

We introduce a new state variable $\underline{\mathbf{x}} = [T \ \underline{C} \ H]^\top \in \mathbb{R}^3$, and accordingly a new reference $\underline{\mathbf{x}}^{\text{ref}} \in \mathbb{R}^3$ (i.e., without the biomass reference). The control input vector $\mathbf{u}$, its reference $\mathbf{u}^{\text{ref}}$, and the disturbance $\mathbf{d}$ remain unchanged. Thus, the reduced nonlinear system is

$$\dot{\underline{\mathbf{x}}} = h(\underline{\mathbf{x}}, \mathbf{d}, \mathbf{u}) = \underline{f}(\underline{\mathbf{x}}, \mathbf{d}) + \underline{g}(\underline{\mathbf{x}}, \mathbf{d})\mathbf{u}. \tag{14}$$

The reduced system (14) has infinitely many equilibrium points. Furthermore, it still contains nonlinear interdependencies among the new states variables. Here, to simplify the search for a suitable equilibrium point among infinite possibilities, we propose to use average values of the disturbance variables for a specified period in the past (i.e., using historical weather data): outdoor temperature T_{out}, outdoor CO_2 concentration C_{out}, outdoor solar radiation Q_{rad}, and outdoor humidity H_{out}. We denote the respective average values of these variables as $(\cdot)^{\text{avg}}$. Furthermore, to minimize the energy costs associated to the inputs, we would like to use $\bar{\mathbf{u}} = \mathbf{0}$. However, that input together with the $\bar{\mathbf{d}} = \mathbf{d}^{\text{avg}}$ makes it again impossible to find an equilibrium point of (14). Thus, here we use the control input variables $\bar{u}_V = \bar{u}_C = \bar{u}_{Q_h} = 0$, and leave $\bar{u}_{Q_c}$ as a free variable. To further simplify the equations at the equilibrium point, we set the variables $T = T_{\text{out}}^{\text{avg}}$, $\underline{C} = C_{\text{out}}^{\text{avg}}$. Thus, an equilibrium point of (14), where $\bar{u}_{Q_c}$ and $\bar{H}$ are unknown, can be computed as

$$\begin{aligned}
\underline{\bar{\mathbf{x}}} &= [T_{\text{out}}^{\text{avg}} \quad C_{\text{out}}^{\text{avg}} \quad \bar{H}]^\top, \\
\bar{\mathbf{u}} &= [0 \quad 0 \quad 0 \quad \bar{u}_{Q_c}]^\top, \\
\bar{\mathbf{d}} &= [T_{\text{out}}^{\text{avg}} \quad C_{\text{out}}^{\text{avg}} \quad H_{\text{out}}^{\text{avg}} \quad Q_{\text{rad}}^{\text{avg}}]^\top.
\end{aligned} \tag{15}$$

By replacing the equilibrium points (15) in the nonlinear system (14), and equating to zero, we obtain a system of nonlinear equations. Note that $\bar{u}_V = 0$ and $\bar{u}_C = 0$ in (4) implies $C_{\text{vent}} = 0$ and $C_{\text{inj}} = 0$ respectively, which together make $\dot{\underline{C}} = 0$ in (13). Thus, system (14) reduces to following system of two nonlinear equations with two unknowns: $\dot{T}(\bar{u}_{Q_c}, \bar{H}) = 0$, $\dot{H}(\bar{u}_{Q_c}, \bar{H}) = 0$, which can be readily solved by a nonlinear root-finding algorithm.

Finally, linearizing (14) around (15), results in

$$\dot{\underline{\mathbf{x}}} = A(\underline{\mathbf{x}} - \underline{\bar{\mathbf{x}}}) + B(\mathbf{u} - \bar{\mathbf{u}}), \quad \text{with} \tag{16}$$

$$A = \frac{\partial h}{\partial \underline{\mathbf{x}}}\Big|_{\underline{\bar{\mathbf{x}}}, \bar{\mathbf{d}}, \bar{\mathbf{u}}}, \quad B = \frac{\partial h}{\partial \mathbf{u}}\Big|_{\underline{\bar{\mathbf{x}}}, \bar{\mathbf{d}}, \bar{\mathbf{u}}}, \tag{17}$$

which are used to find K from (12) for a given $\underline{Q}$ and $\underline{R}$.

3) Stabilization around the origin: To obtain the familiar linear state space form $\dot{\mathbf{z}} = A\mathbf{z} + B\mathbf{v}$, we perform a change of coordinates in (16), i.e., $\mathbf{z} = \underline{\mathbf{x}} - \underline{\bar{\mathbf{x}}}$, and $\mathbf{v} = \mathbf{u} - \bar{\mathbf{u}}$. A regulator like the LQR, with $\mathbf{v} = -K\mathbf{z}$, attempts to stabilize the system around the origin, i.e. around $\mathbf{z} = 0$ and $\mathbf{v} = 0$.

Note that the origin of system $\dot{\mathbf{z}} = A\mathbf{z} + B\mathbf{v}$ is the equilibrium point $\underline{\bar{\mathbf{x}}}, \bar{\mathbf{u}}$ of system (16) given in (15). In other words, the LQR control action $\mathbf{u} - \bar{\mathbf{u}} = -K(\underline{\mathbf{x}} - \underline{\bar{\mathbf{x}}})$ moves the system state towards the equilibrium point. However, as described in Section II, we want to follow the reference trajectories $\mathbf{x}^{\text{ref}}$ and $\mathbf{u}^{\text{ref}}$, respectively.

As stated at the beginning of this section, we are only interested in adding a minor correction to the control input $\tilde{\mathbf{u}}$ inferred by the QDNN network, that is:

$$\mathbf{u} = \tilde{\mathbf{u}} + \mathbf{u}^{\text{lqr}}, \quad \text{with} \quad \mathbf{u}^{\text{lqr}} = -K(\underline{\mathbf{x}} - \underline{\mathbf{x}}^{\text{ref}}). \tag{18}$$

The rationale for this is that we only want to add a minor compensation $\mathbf{u}^{\text{lqr}}$ if we are far from our state reference. The selected equilibrium point is based on average values of the current environment conditions and, thus, the Jacobian matrices A and B in (17) contain information of how the states *approximately* change for any state, disturbance, and a given input close to the selected equilibrium point. In

our results, we see that this practical approach is able to improve the performance of the QDNN. Further theoretical considerations are beyond the scope of this work.

V. Simulation and Results

In this section, we show the potential of the approach developed in the previous sections by means of a numerical case study. In our HMPC setup, the upper level OCP is solved subject to input constraints $0 \leq \mathbf{u} \leq 1$ and soft constraints on states $18 \leq T \leq 26$ °C, $500 \leq C \leq 900$ ppm, and $9.219 \leq H \leq 21.928$ g m^{-3}. The constraint on the state C is generally mentioned in ppm. Additionally,

$$\mathcal{X} = [14, 30] \times [300, 1000] \times [1.206, 30.356] \times [0, 100]$$

defines a hard box constraint set [12]. The DNN is trained with 285,000 data pairs $\mathcal{D}$, of which 80% used for training and the remainder for testing the network accuracy. Also, the infeasible solutions of the MPC are excluded from $\mathcal{D}$. The weighing matrices for the lower level NMPC are chosen as $Q = 100 \cdot \mathrm{I}_4$, $R = \mathrm{I}_4$, and $P = Q$. For the LQR we selected the diagonal matrices $\underline{Q} = \mathrm{diag}(2e2,\ 1e3,\ 1e2)$ and $\underline{R} = \mathrm{diag}(1e3,\ 1e0,\ 1e6,\ 1e8)$.

For the linearization of system (14), an equilibrium point (15) is found using $T_{\text{out}}^{\text{avg}} = 21$, $C_{\text{out}}^{\text{avg}} = 0.5833$, $H_{\text{out}}^{\text{avg}} = 16$, $Q_{\text{rad}}^{\text{avg}} = 170$, which correspond to a three-month average (i.e., one season) using historic weather data near the greenhouse location. Averaging over shorter periods (e.g., a month) may improve the closed-loop performance, at the expense of higher use of computational resources.

A. Reference Tracking via QDNN+LQR

Figure 1 shows the state evolution and Figure 2 presents the control inputs during a single day ($N_{\text{sim}} = 24 \cdot 60 = 1440$ steps) for different implementations. In all figures, we show the open-loop economic MPC reference (ref) together with the closed-loop NMPC (mpc) with $N = 5$, $\delta t = 60$s from the HMPC framework in Section II. Additionally, simulations using the quantized deep neural network (qdnn) and the same network with a LQR compensation (qdnn+lqr) are shown.

Observe that the NMPC follows the open-loop reference rather closely, in most cases obeying the hard constraints $\mathcal{X}$. The QDNN implementation shows large deviations in the states, with CO_2 violating the hard constraint. However, in most cases, the LQR brings the states closer to the reference without hard constraint violation. The saturated control input variables show a similar behavior on reference tracking.

The closed-loop performance S is given by the selling cost of yield S_s minus the production costs S_p (energy cost + injected CO_2 cost), i.e., $S = S_s - S_p$ and

$$S_s = q^\top(\mathbf{x}(N_{\text{sim}}) - \mathbf{x}(0)), \quad S_p = \sum_{i=0}^{N_{\text{sim}}-1} r^\top u(i),$$

with $q^\top = [0\ 0\ 0\ 600]$ and $r^\top = [0.0061\ \ 0.041\ \ 0.56\ \ 0.56]$.

Table II summarizes the performance of the different implementations, where higher values of S indicate higher profit. Note that in Figure 1, although the QDNN has the highest yield ($S_s = 175.95$), it does so with much higher production costs ($S_p = 55.27$) as seen in the Figure 2. The LQR compensation, although it does not consider the biomass dynamics (5), it improves the overall performance S of the QDNN by around 20%.

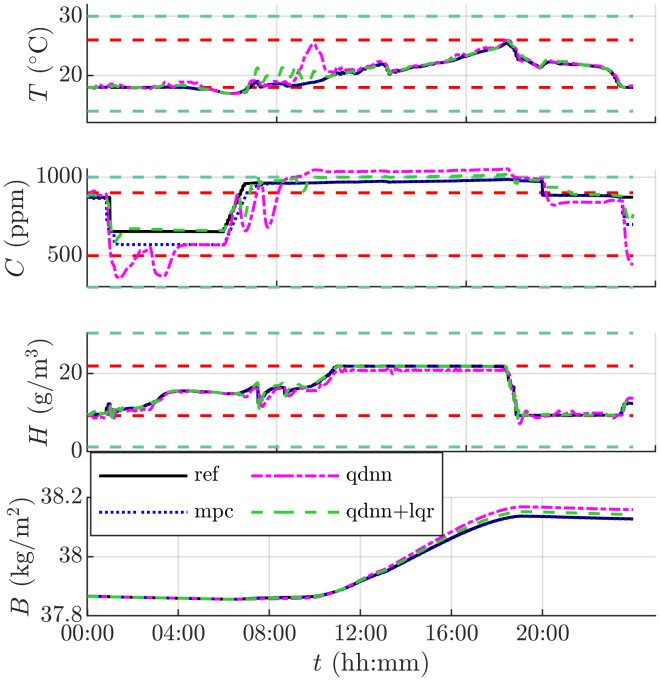

Fig. 1: Comparison of the convergence of states using NMPC, QDNN and QDNN+LQR tracking the reference trajectories. The red and green dashed lines indicate the soft and hard constraints on the states, repectively.

TABLE II: Closed-loop performance

Implementation	S_s	S_p	S
Reference	156.94	2.38	154.56
MPC	156.80	5.36	151.44
QDNN	175.95	55.27	120.68
LQR	166.17	26.26	139.91

B. Implementation on a Microcontroller

The QDNN+LQR scheme is deployed on an STM32F407 MCU, which runs a Cortex-M4F processor core with 168 MHz clock frequency. The Cortex-M4F includes a single-precision floating-point unit and 1 MB nonvolatile memory.

The QDNN consists of 5 layers with roughly 3800 parameters, with 1 parameter requiring 1 byte (8-bit) of storage. Additional information about the network, like the parameters of the uniform asymmetric quantization, requires around 400 bytes of flash memory. Thus, a quantized network uses less than 5 kB of the MCU's flash memory. In comparison, a non-quantized network requires 4 bytes (32-bit for single precision float) to store each parameter. The inference time of the quantized network on the MCU is on average approximately 200 microseconds. Currently, and to the best of our knowledge, there are no NMPC solvers that can be

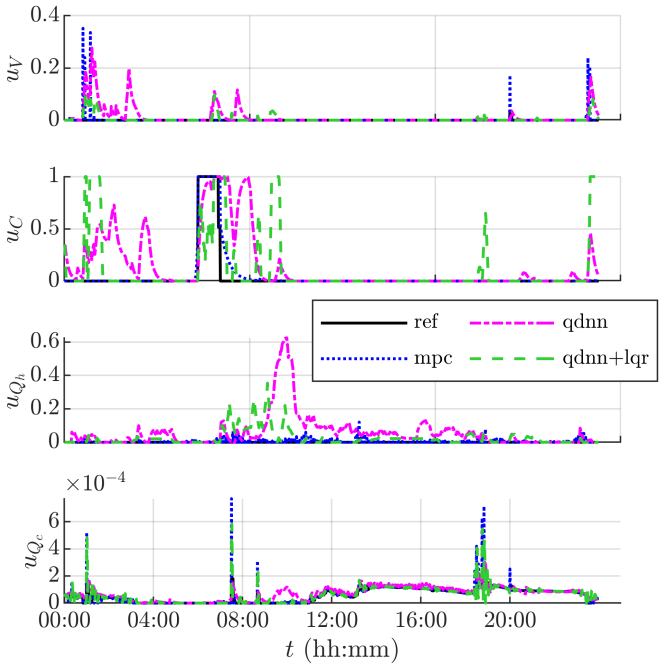

Fig. 2: Comparison of the control inputs (ventilation, CO_2 injection, heating, cooling) using NMPC, QDNN and QDNN+LQR with respect to the reference trajectories.

run on an MCU out of the box. Thus it is not possible to directly compare the computation times of an optimization-based NMPC on a MCU. However, simulations on PC typically report around 3 orders of magnitude reduction on the computation time of a DNN-based NMPC approximation compared to an optimization-based NMPC [9], [17].

Regarding the LQR, note that the Riccati equation is only solved once, and can be done offline. The LQR online compensation consists only of the operations in (18), where the most demanding computation is the matrix vector operation $K(\underline{\mathbf{x}} - \underline{\mathbf{x}}^{\text{ref}})$, with $K \in \mathbb{R}^{4\times 3}$ constant. The computational requirements of the LQR are, therefore, negligible compared to the neural network inference. Note that the performance of the QDNN+LQR can be improved by adaptive LQRs at the expense of memory usage and setup complexity.

VI. Conclusions and Outlook

The paper proposed a method to implement hierarchical control of greenhouse climate on a low-cost microcontroller. We showed a method to train the quantized DNN to mimic the lower level nonlinear tracking MPC and to design the error compensation controller. A numerical case study with the hardware-in-the-loop implementation showed the potential of our approach to reduce the computational burden and memory requirement while maintaining a reasonable overall performance. Although the tracking accuracy is slightly compromised, the proposed approach may still be good for heterogeneous microclimate control in small CEA farms, where the external disturbances can be well maintained. Future research will address the effectiveness of season-based linearizations or similar approach to further improve the performance of the LQR. Additionally, investigating nonlinear approaches to improve the performance of QDNN should be considered.

References

[1] M. Zhang, T. Yan, W. Wang, X. Jia, J. Wang, and J. J. Klemeš, "Energy-saving design and control strategy towards modern sustainable greenhouse: A review," *Renewable and Sustainable Energy Reviews*, vol. 164, p. 112602, 2022.

[2] E. Van Henten and J. Bontsema, "Time-scale decomposition of an optimal control problem in greenhouse climate management," *Control Engineering Practice*, vol. 17, no. 1, pp. 88–96, 2009.

[3] R. van Ooteghem, "Optimal control design for a solar greenhouse," Ph.D. dissertation, 2007, wageningen University, Wageningen, The Netherlands. 304 p.

[4] M. Ahamed, M. Sultan, R. Shamshiri, M. Rahman, M. Aleem, and S. Balasundram, "Present status and challenges of fodder production in controlled environments: A review," *Smart Agricultural Technology*, vol. 3, p. 100080, 2023.

[5] A. Badji, A. Benseddik, H. Bensaha, A. Boukhelifa, and I. Hasrane, "Design, technology, and management of greenhouse: A review," *Journal of Cleaner Production*, vol. 373, p. 133753, 2022.

[6] A. Alexandru, M. Morari, and G. Pappas, "Cloud-based MPC with encrypted data," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5014–5019.

[7] 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.

[8] S. Pon Kumar, A. Tulsyan, B. Gopaluni, and P. Loewen, "A deep learning architecture for predictive control," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 512–517, 2018, 10th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2018.

[9] S. Lucia and B. Karg, "A deep learning-based approach to robust nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 511–516, 2018, 6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018.

[10] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," in *Low-Power Computer Vision*. Chapman and Hall/CRC, 2022, pp. 291–326.

[11] M. Padmanabha, L. Beckenbach, and S. Streif, "Model predictive control of a food production unit: A case study for lettuce production," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 771–15 776, 2020, 21st IFAC World Congress.

[12] K. Sathyanarayanan, P. Sauerteig, and S. Streif, "Deep neural network based optimal control of greenhouses," *arXiv preprint arXiv:2311.04077*, 2023.

[13] M. Ahamed, H. Guo, L. Taylor, and K. Tanino, "Heating demand and economic feasibility analysis for year-round vegetable production in canadian prairies greenhouses," *Information Processing in Agriculture*, vol. 6, no. 1, pp. 81–90, 2019.

[14] R. Shamshiri, J. Jones, K. Thorp, D. Ahmad, H. Man, and S. Taheri, "Review of optimum temperature, humidity, and vapour pressure deficit for microclimate evaluation and control in greenhouse cultivation of tomato: a review," *International agrophysics*, vol. 32, no. 2, pp. 287–302, 2018.

[15] D. Dannehl, H. Kläring, and U. Schmidt, "Light-mediated reduction in photosynthesis in closed greenhouses can be compensated for by CO_2 enrichment in tomato production," *Plants*, vol. 10, no. 12, p. 2808, 2021.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[17] P. Zometa and T. Faulwasser, "Quantized deep path-following control on a microcontroller," in *2023 European Control Conference (ECC)*, 2023, pp. 1–6.

[18] B. Anderson and J. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.