ONLINE LEARNING OF CYLINDER VELOCITY CONTROLLERS FOR EXCAVATOR ASSISTANCE FUNCTIONS USING LOCAL MODEL NETWORKS

Dr. Ozan Demir[1]*, Dr. Benjamin Hartmann[1], Naresh Mandipalli[1],
Dr. Frank Bender[2], Benjamin Ehlers[2], Michael Mehren[2]

[1] Robert Bosch GmbH, Stuttgart, Germany
[2] Bosch Rexroth AG, Lohr am Main, Stuttgart, Germany

** Corresponding author: E-mail address: ozan.demir@de.bosch.com*

ABSTRACT

This contribution introduces a data-based modeling approach using Local Model Networks for the online learning of cylinder velocity controllers that are applied for the realization of excavator assistance functions like reference tracking of the tool center point (TCP).

Even without any individual machine data, just using available data from a similar machine or expert knowledge, we can design an initial controller that is adapted during operation to improve the controller performance and to allow for automatic controller calibration. This allows for a significant reduction of manual machine commissioning efforts while ensuring the required accuracy of the assistance functions. In general, changes in the system behavior over machine lifetime could be compensated with our approach.

To show the effectiveness of the proposed strategy, we have applied the proposed machine learning method to a hydraulic excavator. The data-based controllers are adapted online using a rapid-prototyping system and are sufficiently fast to be implemented on a standard control unit. The control performance is comparable to traditional approaches while drastically reducing the time and effort for calibration.

Keywords: Excavator assistance functions, learning-based control, local model networks, online learning

1. INTRODUCTION

Recently, there has been an increasing demand on assistance functions for working machines, which is also triggered with the recent advances in the field of electro-hydraulics and applied robotics. Such functions can simplify machine usage, increase the overall productivity, and improve machine safety during operation [1].

One of the core technological requirements for assistance functions is the ability of an accurate path tracking for the tool center point (TCP) which is located at the tip of the bucket. Since an experienced operator can follow a desired path with a high accuracy, tracking accuracy requirements for assistance functions are also similarly high. Simple empirical controllers fail to provide the required accuracy since the kinematics and the hydraulic system of the excavator is complex and highly non-linear. Deriving first-principles models for the model-based controller design requires lots of time and effort. Hydraulic excavators are typically high-mix / low-volume products, which makes the development of a model-based controller for each excavator economically not feasible. In the course of this paper, we will apply a learning-based control approach for this control problem. A presentation of the target assistance function for leveling and sloping operations can be also found in the following video: [LINK (https://youtu.be/6xE5ZYCYxtw?si=DyCSdjKmNeBXOdOY)](https://youtu.be/6xE5ZYCYxtw?si=DyCSdjKmNeBXOdOY).

For some time, speed control of hydraulic cylinders for autonomous and semi-autonomous working machines has been in the focus of the control and robotics community. While Bender et. al. [2] and Kalmari et. al. [3] introduce model-based controller structures, recently, pure data-based and hybrid controller structures have been presented. In [4], feedforward Neural Networks (NNs) were used as data-based models, while a Gaussian Process (GP) regression framework has been deployed for the same control problem in [5]. In [6], the authors present a reinforcement learning framework for control, which is trained offline, i.e., without interactions with the real machine using a data-based model. A similar control problem has been solved using local GP models for smaller robots in [7]. In all mentioned papers, the main focus has been the controller design rather than the efficiency of the data-collection process. An efficient data collection method is developed in [8] for the same application using active learning. The importance of an efficient design of experiments (DoE) grows if environmental factors and aging of the system behavior cannot be neglected. Another interesting challenge for data-based controllers is the handling of production tolerances, which can make it necessary to repeat the data collection process for each individual machine.

The main contribution of this work is an efficient online adaptation of data-based controllers for excavator assistance functions, running embedded on the excavator control unit. We apply the concept of local model networks to learn the inverse system behavior so that the obtained model can be used as an adaptive feed-forward controller. Using measurements on a real excavator, we will show that the controller can ensure the necessary accuracy during typical leveling and sloping movements. Moreover, in the case of a deviation from the optimal performance, for example caused by a lack of a rich data sets for the training of the initial model or aging effects, the initial controller can be adapted during regular operation under limited computational and storage resources.

This work is structured as follows. In the next section, we will present the test vehicle and the control problem which will be solved using adaptive data-based controllers. Section 3 introduces the fundamentals of local model networks for non-linear system identification. In Section 4, our proposed online adaptation concept is presented in detail. Vehicle measurements in Section 5 are used to evaluate the effectiveness of the introduced learning-based control strategy. Concluding remarks are given in Section 6.

2. PROBLEM SETUP

2.1. Test Vehicle

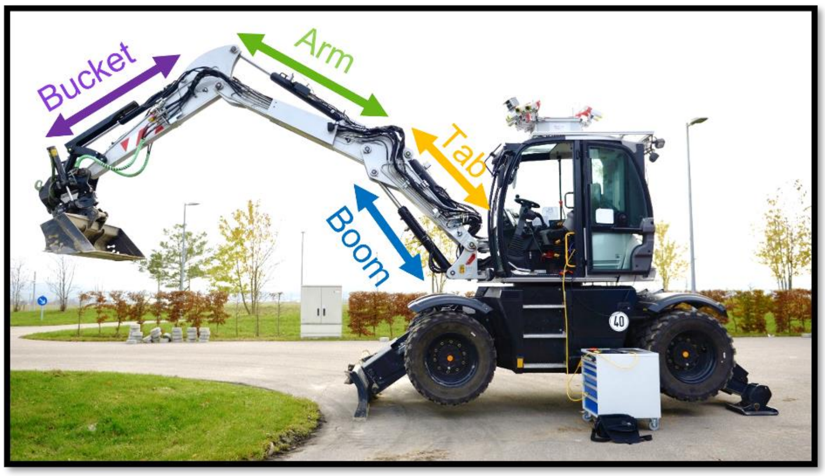

Figure 1: Test Vehicle: JCB Hydradig hydraulic excavator

The test vehicle as shown in Fig. 1 is a JCB Hydradig mobile hydraulic excavator which is used to validate the effectiveness of the proposed control strategy. The working arm of the excavator consists of four links actuated with four cylinders, called boom, tab, arm and bucket cylinder. Three of the cylinders are controlled using our proposed controller while the tab cylinder remains at a constant position during the excavator movement. The control input to the test vehicle is the modified output of the joystick signals. The operational software of the working machine is not modified.

The test vehicle is equipped with a *dSpace MicroAutoBox II* rapid prototyping system. The proposed functionality is implemented in Matlab & Simulink which allows for efficient code generation for the dSpace system.

2.2. Control Problem

The structure of the overall system for the path tracking problem is presented in Fig. 2. As stated in the introduction, focus of this paper will be the learning control for hydraulic cylinders. It is important to notice that the controller structure has also been used in our recent papers [4, 5].

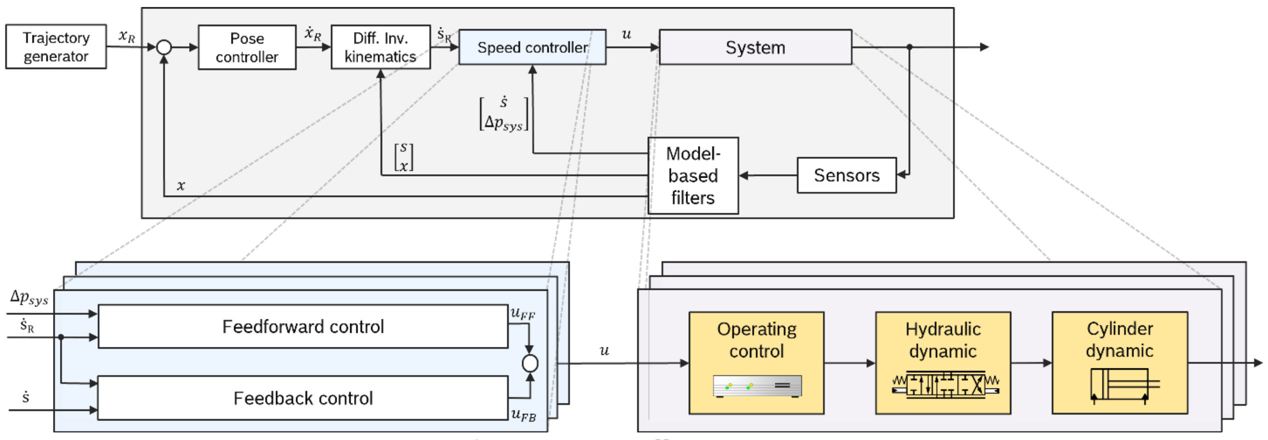

Figure 2: Controller structure

It is assumed that the kinematics of the excavator arm are exactly known so that differential inverse kinematics can be calculated and have not to be learned from data. Details on the optimization-based solution of the inverse kinematics problem, as it has been applied here, can be found in Bender et. al. [2]. Hence, the tracking problem can be transformed from TCP coordinates to joint coordinates. The feedforward controller is learned from data and reflects the inverse system behavior. The control variables are the three joystick signals.

3. LEARNING-BASED FEEDFORWARD CONTROL USING LOCAL MODEL NETWORKS

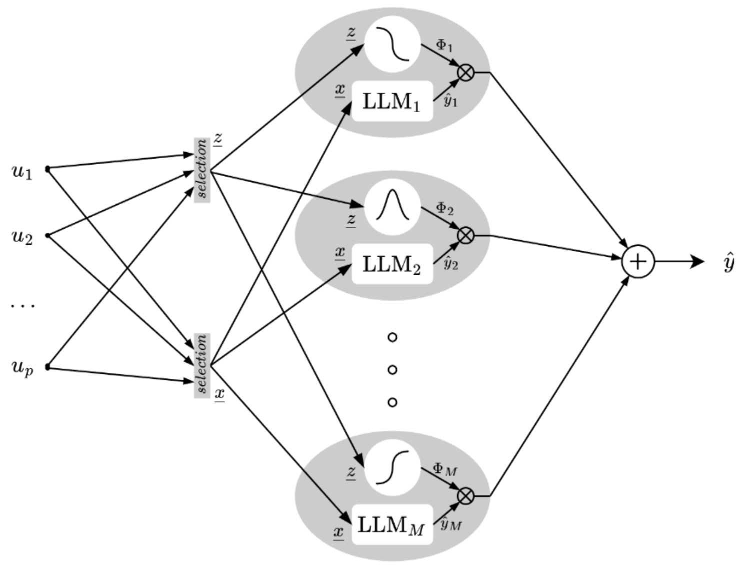

Figure 3: Illustration of a Local Model Network, see [10].

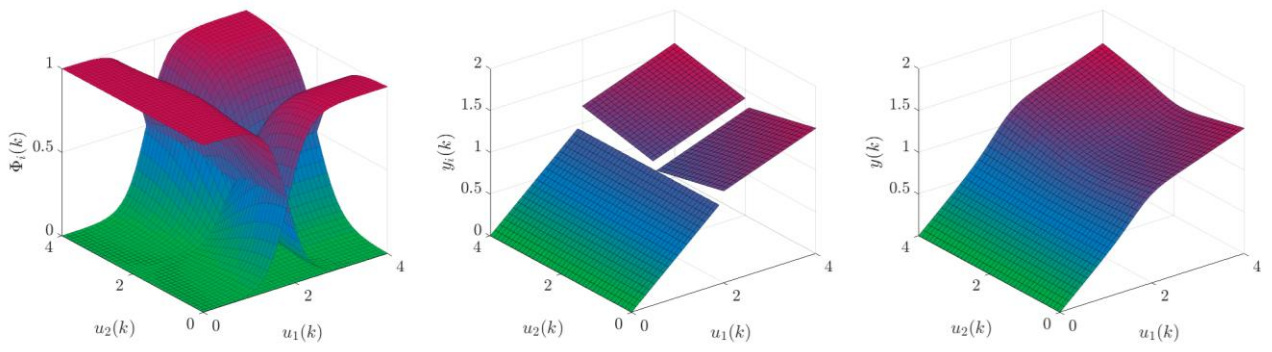

Figure 4: Local model network with two inputs. The model output (right) is calculated as the weighted interpolation of local linear models (middle) using the corresponding validity functions (left).

3.1. Local Model Networks (LoMoNet)

Local Model Networks (Fig. 3) are a special type of neural networks which were developed in the context of nonlinear system identification [10]. The main idea is to calculate the model output $\hat{y}$ as a weighted sum of local models $\hat{y}_i$. Here, the weightings are so-called *validity functions* Φ_i, and the local models are usually chosen to be of *linear* type:

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i(\underline{x})\Phi_i(\underline{z}) = \sum_{i=1}^{M} (w_{i0} + w_{i1}x_1 + w_{i2}x_2 + \cdots + w_{i,nx}x_{nx})\Phi_i(\underline{z}) \,. \tag{1}$$

Using this model construction, it is possible to describe a nonlinear process behavior while having simple linear equations locally, see Fig. 4.

An important aspect of LoMoNet models is that the inputs to calculate the local models as well as the inputs to calculate the validities can be treated individually. Hence, it can be distinguished between the validity function inputs $\underline{z}$ that are influencing the model in a nonlinear way and the local model inputs $\underline{x}$ that affect the model output linearly. This is especially important, e.g., when dealing with dynamic processes where delayed versions of physical inputs have high correlation and should only be incorporated in $\underline{x}$, but are omitted in $\underline{z}$.

As shown in (1), the *i*-th local linear model (LLM) $\hat{y}_\mathrm{i}$ is described with:

$$\hat{y}_i(\underline{x}) = w_{i0} + w_{i1}x_1 + w_{i2}x_2 + \cdots + w_{i,nx}x_{nx} = \underline{\tilde{x}}^T \underline{w_i} \,, \tag{2}$$

where $\underline{w_i} = \left[w_{i0}\; w_{i1} \ldots w_{i,nx}\right]^T$ are the local model parameters, i.e., the offset and slopes in each input dimension and $\underline{\tilde{x}} = [1\; x_1 \ldots x_{nx}]^T$ the input vector for the local model. The regressor “1” in $\underline{\tilde{x}}$ extends the input vector $\underline{x}$ for the estimation of the offsets w_{i0}. The linear formulation has the benefit that standard (weighted) Least Squares methods can be applied for estimation. Furthermore, especially in the context of the underlying application, this is an essential feature, because it allows to use *recursive* algorithms for updating the parameters online.

The validity functions determine in which input region a local model is valid. The determination of the position and shape of these validity functions or the so-called partitioning, respectively, is a complex nonlinear optimization problem and, therefore, often is solved heuristically in practice.

The underlying work uses the Local Linear Model Tree (LoLiMoT) algorithm which is described in detail in [10]. Here, the validity functions are calculated as normalized Gaussian membership functions. The partitioning is achieved with orthogonal splits which lead to Gaussians, described with diagonal covariance matrices. Hence, the calculation of the membership functions simplifies to

$$\mu_i(\underline{z}) = \exp\left(-\frac{1}{2}\left(\frac{(z_1 - c_{i1})^2}{\sigma_{i1}^2} + \cdots + \frac{\left(z_{nz} - c_{i,nz}\right)^2}{\sigma_{i,nz}^2}\right)\right), \tag{3}$$

where σ_{ij} are the standard deviations and c_{ij} the centers of each local model. The membership functions are then normalized to achieve a partition of unity. Hence, for each input sample $\underline{z}$ all validity functions sum up to one:

$$\Phi_i(\underline{z}) = \frac{\mu_i(\underline{z})}{\sum_{j=1}^{M} \mu_j(\underline{z})}, \; \sum_{i=1}^{M} \Phi_i(\underline{z}) = 1 \,. \tag{4}$$

Please, refer to [10] for more details on the functioning and characteristics of the LoLiMoT algorithm. However, important to note is that the LoLiMoT algorithm has two main hyperparameters: the maximum number of local models M and the smoothness value which proportionally affects the calculation of the standard deviations σ_{ij} in (3). Hence, the interpolation smoothness between the local models is influenced. The higher the smoothness value is chosen the smoother the interpolation between local models becomes. Note that with higher interpolation smoothness the number of *effective* model parameters is reduced due to the local estimation of the LLM parameters. This leads to an implicit regularization effect, see [10] for details.

For the underlying application, LoMoNet showed similar performance compared to, e.g., Gaussian Process models or neural networks (MLP). However, the architecture of LoMoNet models is especially useful for our use case, because the model meets the memory and computation requirements of the embedded controller and, furthermore, can be adapted in real-time on the target device.

3.2. Online Learning using LoMoNet

As shown in Fig. 2, the speed controller contains a feed-forward part that captures the inverse behavior of a hydraulic actuation and determines a corresponding joystick position based on the desired trajectories, like described in chapter 2.2. In the following, the inverse plant behavior is modeled with the LoMoNet approach.

The learning of LoMoNet consists of two parts:

1. Learning structural parameters (σ_{ij}, c_{ij}) that capture nonlinear characteristics by partitioning the input space.
2. Learning the parameters of local linear model parameters $\underline{w}_i$.

The first part of learning is challenging due to the nonlinear formulation of the partitioning. In our work, the incremental, axis-orthogonal tree construction algorithm LoLiMoT was used to train the partitioning parameters from offline data. We assume that for similar excavators the partitioning does not change significantly, because the nonlinear behavior is comparable. Hence, the partitioning is trained offline based on data of a “golden sample” excavator and is then transferred to a new, similar excavator. Next, using the pre-trained partitioning, only the local model parameters are adapted to the new system. Therefore, it is possible to separate the complex offline training of the validity functions from the online learning of the local model weights. Note that it is even possible to manually define the input space partitioning based on physical system knowledge when the first commissioning of the machine is done with only a low number of inputs and initially no golden sample data is available.

Once the partitioning is available, the local model weights are adapted online, embedded in the control unit software. The initial values of the local model parameters are taken from the offline training as well, hence, are determined by the data of a similar excavator. For (online) adaptation of local model parameters, a state-of-the-art recursive least squares algorithm can be used.

4. EXPERIMENTAL RESULTS

Databased models only ensure a good model accuracy if the working space is covered with data in a dense way. In this work, we have been using an already available, rich dataset which has been also introduced in [4]. The dataset has been generated using quasi-random (multi-sine and amplitude modulated pseudo random binary signal (APRBS)) excitation of single and multiple cylinders. With this dataset, three different LoMoNet models were trained for the three cylinders with the following input features and model settings, see Table 1. Note that in all experiments the inputs for $\underline{x}$ and $\underline{z}$ are chosen equally.

Table 1: Input features and hyperparameters of LoMoNet models for different cylinders (#LMs = number of local models).

Model	Input features for $\underline{x}$ and $\underline{z}$	#LMs	smoothness
Boom Cylinder	speed, acceleration, Δp_{sys}	110	1.5
Arm Cylinder	speed, acceleration	90	1.2
Bucket Cylinder	speed, Δp_{sys}	51	1.2

4.1. Offline Validation of Online Learning using Vehicle Measurements

First, the obtained models have been validated using test data which wasn't included in the training data. While training data has been generated during random APBRS and multi-sine movements, test data has been collected using typical leveling and sloping operations. Levelling and sloping movements are repeatable excavator tasks with high accuracy requirements and low external load. As expected, all the controllers show a good modeling accuracy as shown in Table 2.

Next, the robustness of the obtained controller against changes in the system behavior has been analyzed. Such deviations in the behavior can arise due to aging or other environmental effects. Another possible reason for data-to-behavior deviations would be production tolerances, if collection of individual training data for each vehicle is omitted and initial controllers have been trained using the same nominal data set. In our experiments, in order to analyze the robustness of the databased controller against changes in the system, we switched at our test machine to a different operation mode (power mode); when not only the speed of the combustion engine and consequently the maximum available power for the hydraulic system but also the characteristics between the joystick position and volume flow demand changes. Please note that the vehicle behavior in power mode stimulates the behavior of a similar but not identical vehicle. Consequently, we observed a clear drop of the model accuracy which is reflected in the model errors shown in Table 2. Please notice that the training data has been collected in normal mode and we didn't repeat the data collection in power mode.

The bad performance obtained with the initial controller in power mode (i.e. at a different working machine) is not really surprising and the typical measure against this situation would be to repeat the data collection stage at the "new vehicle" i.e. at power mode. Consequently, new models can be trained using individual training at the cost of a higher data generation effort. However, even if the system behavior in power mode deviates from the behavior in normal mode, it is still similar so that it is reasonable to expect that the databased controller can achieve again a good performance after a slight adaptation of local model parameters. An offline analysis of vehicle measurements also validates our expectation; using online learning, we obtained a very significant improvement of the system accuracy, see Table 2 vs. Table 3. Table 3 shows that even at the normal mode, after online adaptation, the controller performance can be improved since in this way the model parameters can be optimized more for the leveling operation at the cost of the model generalization.

Table 2: Path tracking performance for different cylinders at normal mode and power mode.

Model	Normal mode		Power mode	
	RMSE [%]	R^2 [-]	RMSE [%]	R^2 [-]
Boom Cylinder	4.03	0.974	6.52	0.861
Arm Cylinder	4.83	0.983	5.84	0.952

Bucket Cylinder	4.29	0.973	4.62	0.948

Table 3: Path tracking performance for different cylinders at normal mode and power mode (online adaptation is enabled).

Model	Normal mode		Power mode	
	RMSE [%]	R^2 [-]	RMSE [%]	R^2 [-]
Boom Cylinder	2.14	0.992	1.99	0.987
Arm Cylinder	1.87	0.997	2.08	0.994
Bucket Cylinder	1.36	0.997	3.21	0.975

4.2. Online Validation at the Working Machine

After the offline analysis, the online adaptation of feed-forward databased controllers has been also validated at the test vehicle. Please note that during validation measurements, the feedback part of the cylinder speed controller is deactivated so that different feed-forward controllers can be directly compared.

In the measurements, first, online adaptation is disabled and the path following accuracy during typical levelling movements at normal mode has been evaluated. Figure 5 shows the tracking accuracy at different hydraulic cylinders.

As described above, we wanted to check the robustness of the databased controller against changes of the system behavior. Figure 6 shows the controller performance at power mode which represents a similar but not identical machine. As expected, we observed a significant drop of the controller performance. Especially, strong oscillations arise at boom cylinder which is not acceptable.

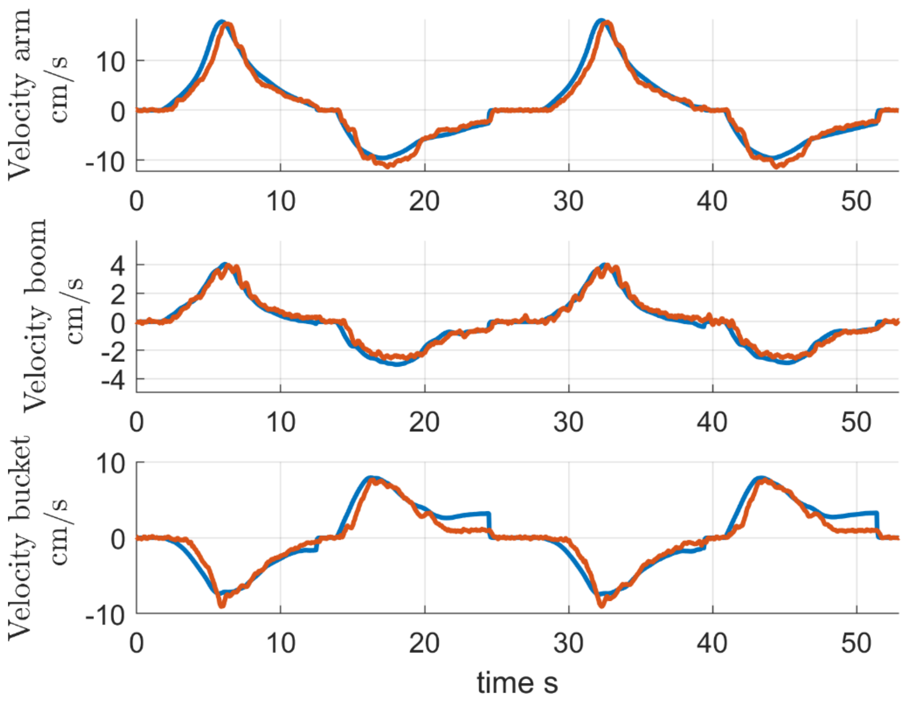

Figure 5: Trajectory tracking performance of databased controllers during leveling at normal mode (nominal machine). The desired cylinder speeds are depicted in blue and the measured speeds in red.

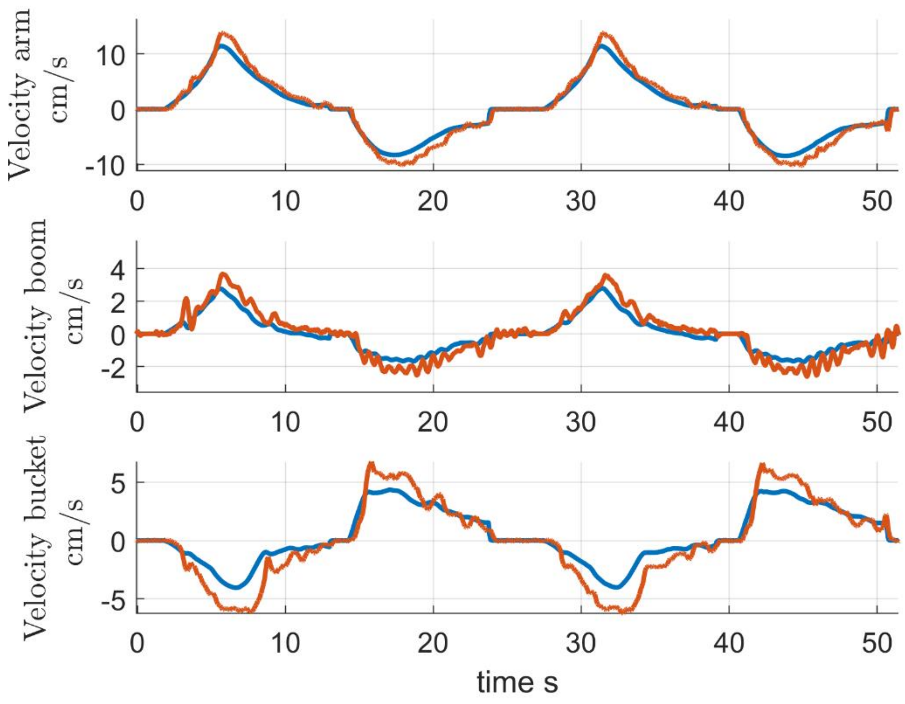

Figure 6: Trajectory tracking performance of databased controllers at power mode (i.e at a similar machine) during leveling. The desired cylinder speeds are depicted in blue and the measured speeds in red.

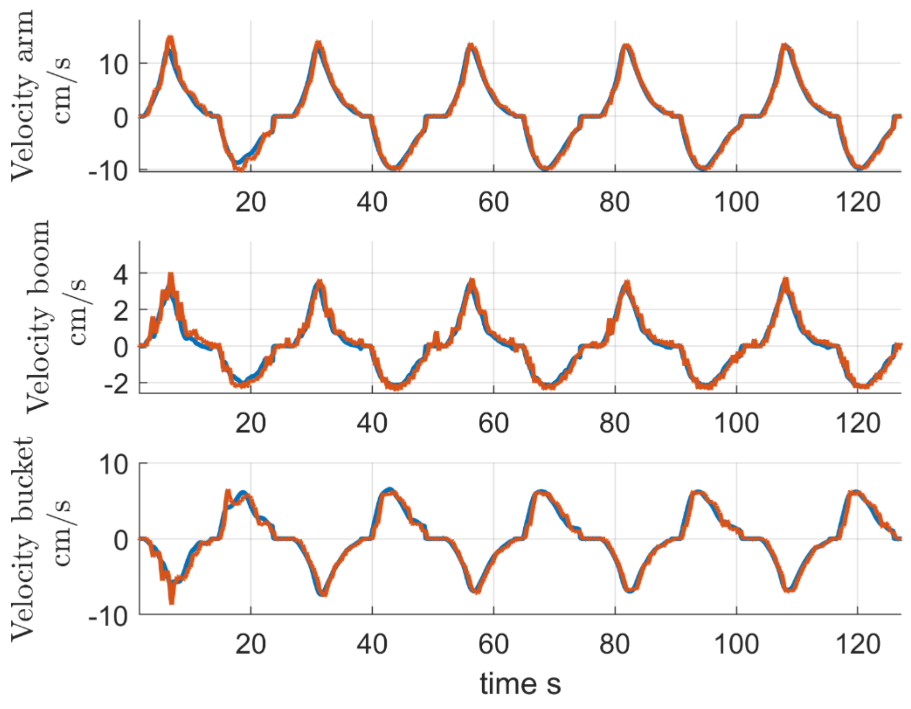

Figure 7: Trajectory tracking performance of databased controllers at power mode (i.e at a similar machine) during leveling. The desired cylinder speeds are depicted in blue and the measured speeds in red. Online adaptation of local model parameters is enabled, which ensured a low tracking error compared with the previous measurement presented in Figure 6.

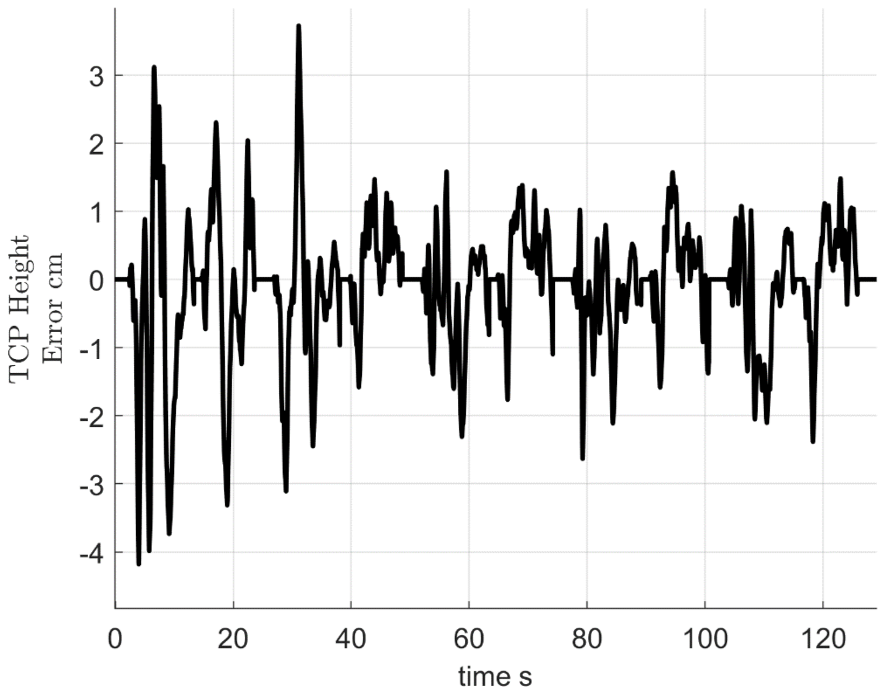

Figure 8: Leveling error (deviation from desired height) during online learning.

To be able to compensate for the change of the system behavior, online adaptation is enabled, and the local model parameters are adapted dependent on the obtained prediction error. Figure 7 clearly shows that the controller performance can be improved in a significant way after a very short time. With the help of online adaptation, also the TCP-tracking accuracy can be improved, which is presented in Figure 8. Please notice that online adaptation ensures that the deviation from the desired height remains under 3cm which is a very impressive performance for the size of the excavator.

5. CONCLUSION AND OUTLOOK

In this work, we presented a learning-based control concept for the speed control of hydraulic cylinders using local model networks. The proposed framework models the complex hydraulic behavior with good accuracy. Furthermore, model parameters can be adapted in an efficient way, so that it is even possible to run it on the embedded control unit software. As a result, the robustness of the learning-based controller is significantly improved against changes of the system behavior. This is a very important achievement since such deviations are very typical during regular operation due to environmental effects, aging or part replacements.

In our current research, we are working on the robustness of the online learning algorithms against “bad” data, so that learning can be activated all the time and functionality shall decide which data can be used to adapt databased models. Another interesting research point is to learn databased models of the forward path of the system which can be then used for controller design.

NOMENCLATURE

TCP	Tool center point	
x	Pose of tool center point in horizontal, vertical, and relative bucket angle	m, m, rad
$\underline{x}_R$	Desired pose of tool center point in horizontal, vertical, and relative bucket angle	m, m, rad
s	Measured cylinder position for boom, arm, and bucket cylinder	m
s_R	Desired cylinder position for boom, arm, and bucket cylinder	m
u	Controller output (modified joystick output)	%
Δp_{sys}	System pressure i.e. pressure difference between pump pressure and load sensing pressure	bar
LLM	Local linear model	
w_i	Parameters of the i-th local linear model	
σ_{ij}	Standard deviations of i-th local model inputs	
c_{ij}	Center of i-th local model	
RMSE	Root mean square error	
R^2	Coefficient of determination	

REFERENCES

[1] Bender, F., Kaiser, N., Jayakumar, B., *Industry Spotlight: Automated operations for construction machines*, In: New Horizons – Building the Recovery. Hillhead Digital Conference, March 30-31, 2021

[2] Bender, F., Sonntag, M., Sawodny, O., *Nonlinear model predictive control of a hydraulic excavator using Hammerstein models*, In: Proceedings of the 6th International Conference on Automation,Robotics and Applications, pp. 557–562, 2015.

[3] Kalmari, J., Backman, J., Visala, A., *Nonlinear model predictive control of hydraulic forestry crane with automatic sway damping*, In: Computers and Electronics in Agriculture, vol. 109, pp. 36–45, 2014

[4] Weigand, J., Raible, J., Zantopp, N., Demir, O., Trachte, A., Wagner, A., Ruskowski, M., *Hybrid*

data-driven modelling for inverse control of hydraulic excavators, In: Proceedings of the International Conference on Intelligent Robots and Systems, 2021, pp. 2127–2134

[5] Rabenstein, G., Demir, O., Trachte, A., Graichen, K., *Data-driven feed-forward control of hydraulic cylinders using Gaussian process regression for excavator assistance functions,* In: Proc. of CCTA. Trieste, Italy, 2022, pp. 962–969.

[6] Egli, P., Hutter, M., *Towards RL-based hydraulic excavator automation*, In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020

[7] Nguyen D., Seeger M., Peters J., *Real-time local GP model learning,* In: From Motor Learning to Interaction Learning in Robots, Springer, 2010, pp. 193–207.

[8] Dio M., Demir O., Trachte A., Graichen K., *Safe Active Learning and Probabilistic Design of Experiment for Autonomous Hydraulic Excavators,* In: Proceedings of the International Conference on Intelligent Robots and Systems, 2023 (accepted).

[9] Siciliano, B., Sciavicco, L.,Villani, L., Oriolo, G., *Robotics: modelling, planning and control*. Springer, London, 2010.

[10] Nelles, O., *Nonlinear System Identification*. Springer, 2020.