2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

High-Precision Trajectory Generation Based on Data-Driven Model Predictive Control With Weight Optimization

Junya Fukui[1] and Takashi Nammoto[1]

Abstract— **This paper presents a comprehensive framework for trajectory generation based on data-driven model predictive control (MPC) for industrial applications. The proposed method effectively combines a transfer function model, which is identified from operational data, with a neural network to address and mitigate modeling errors. In contrast to conventional online MPC approaches, the method formulated in this paper treats trajectory generation as an offline optimization problem, wherein the position sequence is directly optimized. Multiple performance metrics are jointly optimized, with the cost function weights being automatically tuned through machine learning-based optimization techniques, all while adhering to explicit tracking error constraints. Experimental validation conducted on a linear guide system demonstrates that the proposed method achieves both high tracking performance and smooth, practical trajectories, eliminating the need for manual parameter adjustments. This framework offers a robust and adaptable solution for advanced trajectory design applicable across a variety of industrial applications.**

I. INTRODUCTION

In recent years, demand for advanced motion-control technologies has grown substantially in autonomous systems such as machine tools, industrial robots, unmanned aerial vehicles (UAVs), and autonomous vehicles. This trend is driven by demands for higher precision, improved quality and efficiency, enhanced safety, and the capability to perform increasingly complex tasks, all of which require high-speed, high-precision trajectory tracking.

Trajectory tracking control strategies can be broadly categorized into two primary approaches. The first approach focuses on optimizing the control system itself to enhance tracking performance. Two-degree-of-freedom control architectures and adaptive control schemes have been extensively studied in traditional research contexts [1], [2]. Recently, data-driven control design methodologies, such as virtual reference feedback tuning (VRFT), have garnered attention. VRFT constructs an inverse model based on results from system identification, thereby facilitating automated tuning of controller gains [3]. Concurrently, advancements in computational resources have led to an increased application of model predictive control (MPC) for trajectory tracking across various fields [4], [5], [6]. MPC computes optimal control inputs at each control interval by utilizing a pre-identified system model, which allows for the simultaneous consideration of physical constraints. Applications of MPC include robust control methods that adaptively update the model to ensure system robustness [7], trajectory smoothness enhancement in UAVs through online optimization of cost function weights [8], disturbance attenuation via frequency-domain weight adjustments [9], and approaches that balance safety and ride comfort in autonomous vehicles [10], [11]. However, the computational complexity associated with MPC poses significant challenges for real-time optimization in industrial robots and machine tools, where rapid control cycles are essential. While strategies such as prediction horizon shortening and high-speed optimization algorithms have been proposed [12], [13], a fundamental solution to this issue remains elusive.

The second approach designs or modifies the reference trajectory to match the control system dynamics. In the context of machine tools and conveyance devices, adequate tracking performance is often hindered by the inherent dynamics of the system. Techniques such as spline interpolation and inverse model compensation have been widely adopted to create smoother trajectories that facilitate easier tracking [14], [15]. New methods have also emerged, including real-time projectile path correction and FIR filtering [16]. However, this approach faces challenges in accounting for variations in system dynamics and external disturbances, often necessitating trajectory redesign or adaptation for specific applications.

To address these challenges, this paper introduces a novel trajectory generation method based on the MPC framework. The main contributions of this work include:

- Proposed a novel trajectory generation method based on the MPC framework.
- Implemented a data-driven identification approach using a high-capacity model to more accurately reflect the characteristics of the equipment
- Formulation of a constrained optimization problem that takes into account the operating conditions of the control system, along with automatic trajectory generation using this formulation.

The remainder of this paper is organized as follows. Section II provides a detailed description of the proposed method. Section III presents experimental results obtained using a linear guide system, which includes linear guides and a servo amplifier, to validate the effectiveness of the proposed method. Section IV summarizes the findings' importance and outlines future research directions.

[1]Junya Fukui and Takashi Nammmoto with Information Technology R&D Center, Mitsubishi Electric Corporation, Japan. Junya.Fukui@cw.MitsubishiElectric.co.jp, Nammoto.Takashi@ds.MitsubishiElectric.co.jp

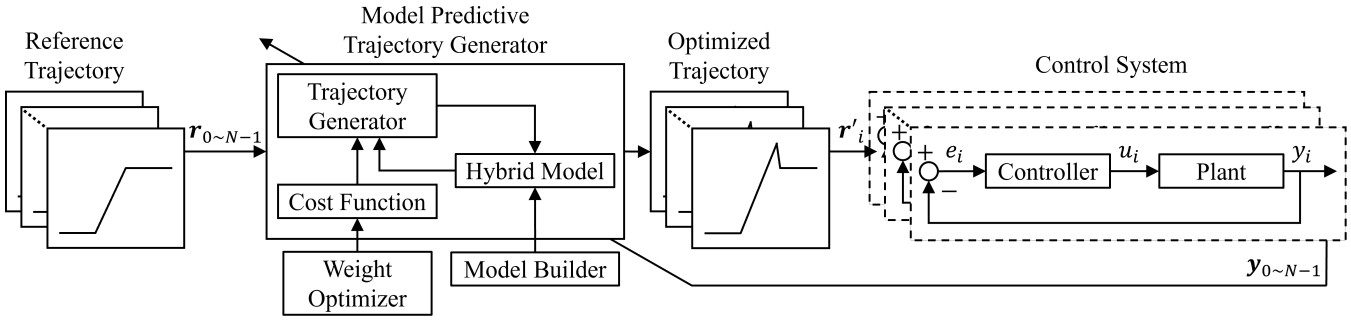


Fig. 1: Architecture and processing flow

II. PROPOSED METHOD

A. Architecture and Processing Flow

Fig. 1 illustrates the architecture and processing flow of the proposed method. As depicted, when a reference trajectory is provided, the model predictive trajectory generator takes into account the dynamic characteristics of the control system and generates an optimized trajectory through iterative trials and evaluations (Sec. II-B). This process ensures that the control system accurately follows the reference trajectory. In this context, the trial results are simulated using a hybrid model that combines a transfer function model with a neural network, both of which have been pre-identified based on the operational data of the control system (Sec. II-C). This integration enhances the accuracy of the simulations and the overall trajectory generation process. Furthermore, the cost function employed for trajectory optimization is a weighted sum of multiple evaluation criteria. The weights for these criteria are determined in advance through simulations conducted with the hybrid model, allowing for a tailored approach to optimization (Sec. II-D). The proposed method facilitates trajectory generation with high adaptability and versatility across various applications by introducing a hybrid model. Additionally, the flexibility in the structure of the cost function and the optimization of the weights for the evaluation criteria enable the achievement of desired levels of tracking accuracy, smoothness, and compliance with constraints. Overall, this approach effectively addresses critical issues such as the deterioration of tracking performance due to model discrepancies, the significant burden of manual parameter tuning, and the lack of transparency inherent in conventional methods. By leveraging these advancements, the model predictive trajectory generator offers a robust solution for dynamic control systems.

B. Trajectory Generator

The details of the trajectory generator are presented. While conventional MPC typically uses torque or velocity sequences as optimization variables, the proposed approach directly optimizes discrete position sequences as trajectories. This design enables simultaneous multi-objective optimization of various performance indices.

The optimized trajectory is defined as a sequence of positions that minimizes the cost function, as shown in the following equation.

$$\mathbf{r}'_i = \arg\min_{\mathbf{r}'_i} J_{\mathrm{TG}}(i) \qquad (i = 0, \ldots, N-1) \tag{1}$$

Here, N represents the number of positions included in the reference trajectory. The local cost function $J_{\mathrm{TG}}(i)$, which evaluates the quality of candidate trajectory corrections starting from time step i, is defined as follows:

$$J_{\mathrm{TG}}(i) = \sum_{j=i}^{i+H-1} \big(w_p \|\hat{\mathbf{y}}_j(\mathbf{r}'_j) - \mathbf{r}_j\|_2 + w_v \|\Delta\hat{\mathbf{y}}_j\|_2 + w_u \|\Delta\mathbf{r}'_j\|_2 \big) \tag{2}$$

H denotes the prediction horizon over which the cost function is calculated. $\hat{\mathbf{y}}_j$ is the estimated position of the plant when controlled with the reference position $\mathbf{r}'_j$, utilizing the hybrid model. $\Delta\hat{\mathbf{y}}_j$ denotes the discrete-time velocity obtained from the difference of successive estimated position, while $\Delta\mathbf{r}'_j = \mathbf{r}'_j - \mathbf{r}'_{j-1}$ signifies the reference position deviation. Furthermore, w_p, w_v, and w_u are the weighting parameters for position error, velocity, and input variation, respectively.

A notable feature of this framework is that adjusting the weights w_p and w_v allows for trajectory design focused on minimizing total displacement or ensuring smooth acceleration and deceleration. The input variation term w_u facilitates trajectory shaping to accommodate the controlled system's response delay and oscillatory characteristics. This optimization framework also integrates various practical constraints, including physical input limitations and geometric accuracy requirements. The optimal trajectory is obtained by appropriately adjusting the cost function weights, including the use of automatic weight optimization.

C. Hybrid Model and Model Builder

The hybrid model is defined as follows:

$$\hat{\mathbf{y}}_j = \hat{\mathbf{y}}_j^{\mathrm{TF}} + \hat{\mathbf{e}}_j^{\mathrm{NN}} \tag{3}$$

As shown in (3), $\hat{\mathbf{y}}_j$ is the sum of $\hat{\mathbf{y}}_j^{\mathrm{TF}}$, which estimates the position after control using the transfer function model, and

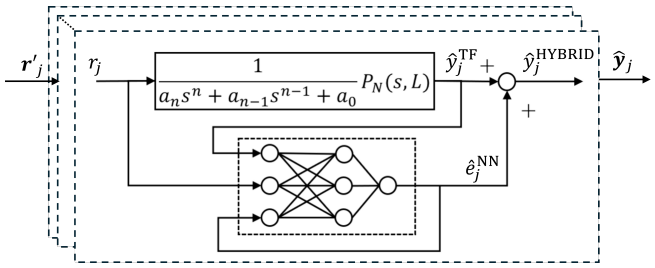


Fig. 2: Hybrid model

$\hat{\mathbf{e}}_j^{\mathrm{NN}}$, which estimates the error that is likely to occur at that time.

The transfer function model can reproduce most aspects of the actual control system behavior. To dynamically compensate for residual modeling errors, a neural network, which is widely used for system modeling [17], is introduced to augment the transfer function model.

The neural network is trained to learn and compensate for the identification error on a sequential basis. The neural network receives the reference position, the simulated output from the transfer function, and its estimation error from the previous time step as inputs. It predicts and compensates for the difference between the measured position after control and the output of the transfer function model. The structure described above is illustrated in Fig. 2.

The control system is modeled by an n-th-order transfer function, where the pure time delay term e^{-Ls} is approximated by the Padé approximation $P_N(s, L)$.

$$G(s) = \frac{1}{a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0} P_N(s, L) \tag{4}$$

This model is discretized, and parameter fitting is based on the difference between the transfer function model output $\hat{\mathbf{y}}_k^{\mathrm{TF}}$ corresponding to the input $\mathbf{r}'_k$ and the measured output $\mathbf{y}_k^{\mathrm{meas}}$. The fitting cost function is defined as:

$$J_{\mathrm{TF}} = \sum_{k=0}^{M-1} \left\| \mathbf{y}_k^{\mathrm{meas}} - \hat{\mathbf{y}}_k^{\mathrm{TF}} \right\|_2 \tag{5}$$

The parameter estimation problem is formulated as the minimization of J_{TF}, wherein $a_n, \ldots, a_0, L$ are identified using parameter optimization methods such as black-box optimization.

The functional structure of the neural network is given by:

$$\hat{\mathbf{e}}_j^{\mathrm{NN}} = \mathbf{f}_{\mathrm{NN}} \left(\mathbf{r}'_j, \hat{\mathbf{y}}_j^{\mathrm{TF}}, \hat{\mathbf{e}}_{j-1}^{\mathrm{NN}}; \boldsymbol{\theta} \right) \tag{6}$$

$\boldsymbol{\theta}$ represents the parameters of the neural network, and their values are adjusted through training. The neural network is trained using the difference between the actual measured outputs $\mathbf{y}_k^{\mathrm{meas}}$ and the outputs of the transfer function model $\hat{\mathbf{y}}_k^{\mathrm{TF}}$, with the teaching signals defined as $\mathbf{e}_k^{\mathrm{TS}} = \mathbf{y}_k^{\mathrm{meas}} - \hat{\mathbf{y}}_k^{\mathrm{TF}}$. The following loss function is minimized during training:

$$J_{\mathrm{NN}} = \sum_{k=0}^{M-1} ||\mathbf{e}_k^{\mathrm{TS}} - \hat{\mathbf{e}}_k^{\mathrm{NN}}||_1 \tag{7}$$

M denotes the number of training data samples. In this work, the nominal axis dynamics are described using a stable transfer function model, and a neural network is introduced to compensating for residual modeling errors. This hybrid structure is applied independently to each axis of the system.

D. Cost Function and Weight Optimizer

The optimization method for the weights of the cost function, w_p, w_v, and w_u, which are utilized to generate optimal trajectories. Properly tuning these weights according to the characteristics of the controlled plant, operational objectives, and allowable tracking error is crucial. Traditionally, this adjustment has been performed manually, relying on skilled operators, which leads to a heavy dependence on empirical knowledge and trial-and-error methodologies. The proposed method treats the weights w_p, w_v, and w_u as optimization variables. By minimizing the total travel distance as the objective and imposing constraints on both the mean and maximum tracking errors, this approach effectively suppresses unnecessary high-frequency oscillations and excessive tracking actions, resulting in practical and smooth trajectory generation. The optimization problem is formulated as follows:

$$\begin{aligned} \min_{w_p, w_v, w_u} \quad & \sum_{i=0}^{N-1} \|\Delta \mathbf{r}'_i(w_p, w_v, w_u)\|_2 \\ \text{s.t.} \quad & \frac{1}{N} \sum_{i=0}^{N-1} \|\hat{\mathbf{y}}_i - \mathbf{r}_i\|_2 \leq \epsilon_{\mathrm{mean}} \\ & \max_i \|\hat{\mathbf{y}}_i - \mathbf{r}_i\|_2 \leq \epsilon_{\max} \end{aligned} \tag{8}$$

As shown in (8), the total reference displacement is minimized while satisfying the constraints. The constraints set upper limits ϵ_{mean} and $\epsilon_{\max}$ on both the average error and the maximum error between the reference trajectory and the estimated position of the plant after control. In (2), the reference position deviation $||\Delta \mathbf{r}'_i||_2$ serves to locally suppress excessive inputs within the prediction horizon. In contrast, (8) utilizes this deviation to determine the weights that shape the global characteristics of the entire trajectory. To solve this constrained optimization problem efficiently, machine learning-based methods such as evolutionary strategies and Bayesian optimization are employed for weight tuning. A significant advantage of this approach is that the constraint values are explicitly specified as quantitative parameters, making it straightforward to accommodate changes in system specifications or the addition of new operational objectives.

III. EXPERIMENTS

The effectiveness of the proposed method is demonstrated through experimental validation on the actual machine.

A. Experimental Setup

The experiments were conducted using a control system that consists of a servo amplifier and a linear guide, as shown in Fig. 3. This linear guide is equipped with

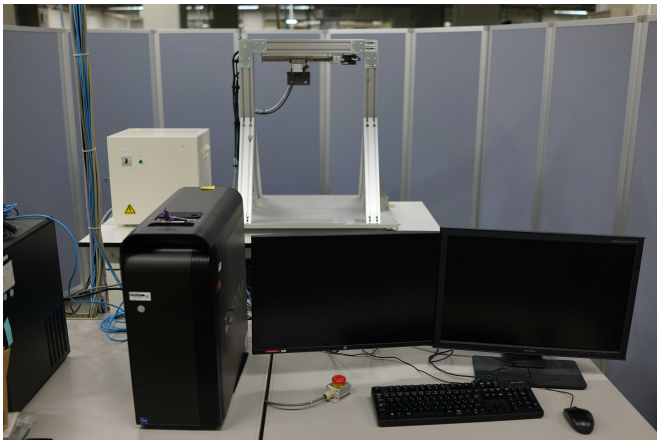


Fig. 3: Linear guide system

TABLE I: Experimental trajectories

Shape	Feed rate [mm/min]	Size [µm]
Circle, Star, Square	1000, 5000	1000

two axes, X and Y, with each axis driven by a servo motor connected via a coupling, which converts the motor's rotational motion into precise linear motion of the table. This configuration enables high-precision linear positioning of the table. The control system is equipped with several pre-installed control loops that provide reference positions for each axis, allowing for automatic positioning. However, due to the inherent dynamics of the linear guide, mechanical delays and residual errors can arise when the system executes high-speed commands or abrupt trajectory changes. In such cases, the proposed method becomes essential for maintaining accurate motion. In the experimental setup, a servo amplifier (MR-J4W2-22B, Mitsubishi Electric) is connected to a PC via a position control board (MR-MC341, Mitsubishi Electric). The reference position is updated for each control cycle. By loading trajectory files generated offline using the proposed method into the PC, the reference position can be updated in real-time during operation. The gains of the servo amplifier are set in advance and remain constant throughout the evaluation.

B. Experimental Trajectories

The experimental validation involved evaluating multiple shapes with different characteristics. All combinations listed in Table I were tested, and several verification patterns were established, such as with or without trajectory optimization and with different simulation models.

C. System Identification Results

The system identification results are presented for a hybrid model, validated on the linear guide setup. As described earlier, the hybrid model consists of a transfer function model that accounts for pure time delay and a neural network to compensate residual modeling errors. Although linear guides are commonly represented by a second-order transfer

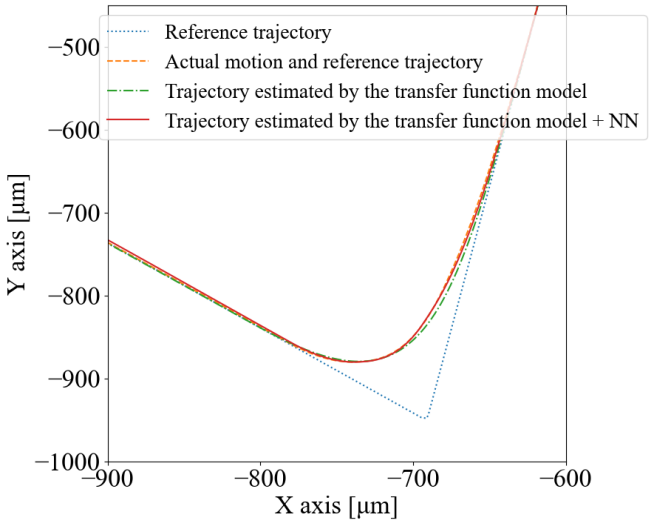


Fig. 4: Trajectory reproducibility

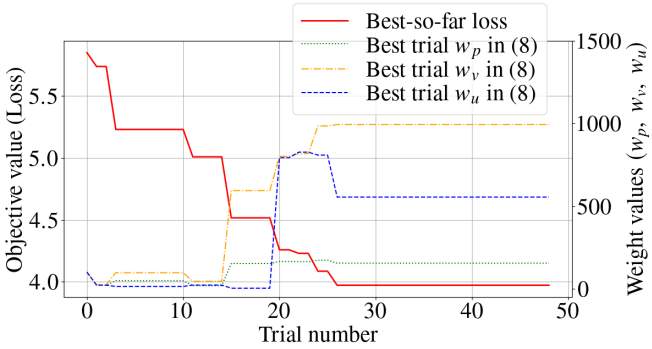
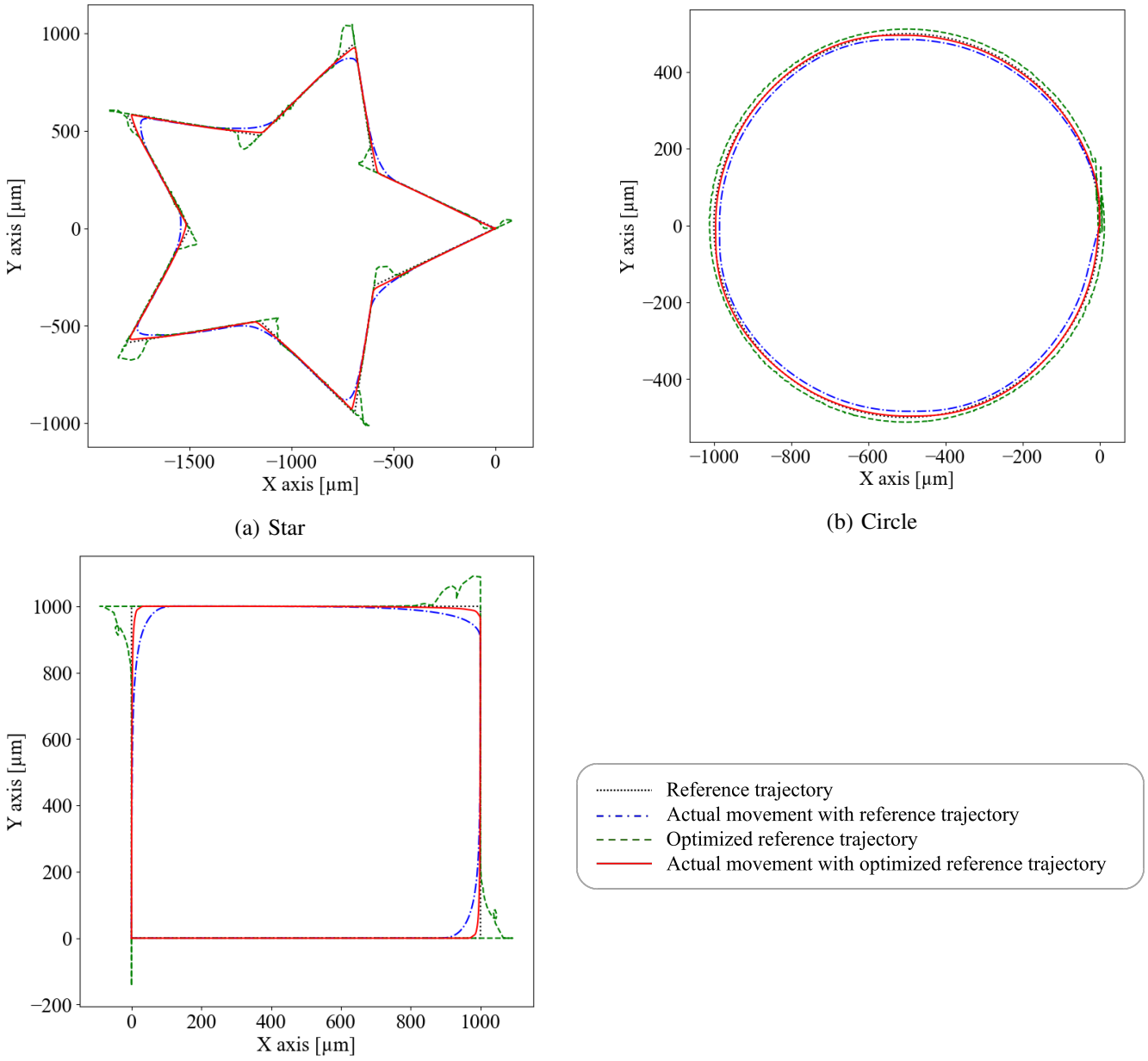


Fig. 5: Weight optimization result

function model, the identification setup in this study includes a servo amplifier, justifying the use of a fourth-order transfer function model for the system identification. The parameters of the transfer function model were estimated using the covariance matrix adaptation evolution strategy (CMA-ES), a widely used evolutionary strategy for parameter search. The neural network consists of three fully connected layers; the hidden layers have 256 and 64 units, respectively. Sigmoid activation functions were used, and the network was trained with the Smooth L1 loss. Batch normalization and dropout (p = 0.2) were optionally applied during training to mitigate overfitting. These hyperparameters were determined experimentally, and both N and M were set to 2000. The results for trajectory reproducibility under the conditions of star-shaped trajectory, with a feed rate of 5000 mm/min and a size of 1000 µm, are shown in Fig. 4. Visually, accuracy improves when neural-network-based error compensation is incorporated into the transfer function model.

D. Weight Optimization Results

The optimization results of the trajectory generation weights are presented. The optimization was performed using operational results obtained from the experimental system with the objective of minimizing the cost function.



(a) Star

(b) Circle

(c) Square

Fig. 6: Experimental results at a feed rate of 5000 mm/min.

In this study, the tree-structured parzen estimator (TPE) was employed as the optimization method [18]. TPE is a widely used black-box optimization algorithm that possesses the ability to flexibly incorporate constraints into the exploration process, making it a suitable choice in this context. As this experiment represents a preliminary trial, the default settings of the TPE were employed. The results are shown in Fig. 5. The solid line indicates the minimum value of the cost function achieved among all previously explored solutions, while the dotted, dash-dotted, and dashed lines represent the respective weights associated with the best-performing solution found so far. It can be observed that the position weight, w_p, is minimized within the range allowed by the imposed constraints, which helps to mitigate excessive responses. In this experiment, the weight optimization was performed using a simplified model without the neural network component, in order to efficiently explore the optimal weight configuration.

E. Trajectory Generation Results

The results of trajectory generation are presented. The prediction horizon H was determined experimentally as $H = 5$. Fig. 6 illustrates the trajectories for the star-shaped, circular, and square shapes, at a feed rate of 5000 mm/min. Each figure includes the reference trajectory, the actual trajectory obtained by directly inputting the reference

TABLE II: Experimental conditions

No.	Description
(i)	Not optimization
(ii)	Optimization of reference trajectory without weight optimization
(iii)	Optimization of reference trajectory with weight optimization

TABLE III: Experimental results (evaluation values in μm)

Shape	No.	1000 [mm/min] RMSE	1000 [mm/min] Max Error	5000 [mm/min] RMSE	5000 [mm/min] Max Error
(a) Star	(i)	1.65	8.04	7.19	39.93
	(ii)	1.56	10.30	3.11	30.37
	(iii)	1.11	7.29	4.47	18.29
(b) Circle	(i)	1.48	6.64	7.95	16.16
	(ii)	1.65	6.72	2.20	5.36
	(iii)	1.49	6.76	2.08	4.86
(c) Square	(i)	0.75	7.90	8.76	44.15
	(ii)	0.41	5.68	1.54	12.74
	(iii)	0.39	4.66	2.22	13.81

trajectory into the control system without optimization, the optimized reference trajectory, and the actual trajectory obtained by inputting the optimized reference trajectory into the control system. The figures show that the actual trajectory before optimization deviates significantly from the reference trajectory, as it passes through the interior of the shape. After optimization, the actual trajectory closely follows the reference trajectory.

Table III summarizes the trajectory generation results. As outlined in Table II, we conducted a comparative analysis under three distinct scenarios. The upper row presents the trajectory configurations, with the corresponding feed rates shown in parentheses. These tables collectively illustrate the individual and combined effects of trajectory and weight optimization on system performance. Except for a few outliers, the scenario incorporating both weight and trajectory optimization consistently demonstrated superior performance.

IV. CONCLUSIONS AND FUTURE WORKS

In this study, we propose a trajectory generation method based on the MPC framework, which incorporates a model that captures the characteristics of the experimental system. To perform system identification on the linear guide testbed, we implemented a hybrid model that combines a fourth-order transfer function model with a pure time delay and a neural network. This model exhibits high representational capacity and practical applicability. Furthermore, the hybrid-model-based trajectory generation maintained accurate tracking and contour performance. An approach for optimizing the trajectory generation weights was implemented and validated through experiments. The overall framework was validated through comprehensive experimental evaluation. Future work will include a detailed analysis of each component, experimental validation of the proposed method on other machines such as machine tools and industrial robots, and an investigation of optimization algorithms that are better aligned with the characteristics of this problem than the commonly used method adopted for verification.

REFERENCES

[1] R. Miklosovic and Z. Gao, "A Robust Two-Degree-of-Freedom Control Design Technique and its Practical Application," *Proc. 39th IAS Annual Meeting, IEEE Industry Applications Conference*, vol. 3, pp. 1495–1502, 2004.

[2] Q. Sang and G. Tao, "Adaptive control of piecewise linear systems: the state tracking case," *IEEE Transactions on Automatic Control*, vol. 57, no. 2, pp. 522–528, 2012.

[3] F. Previdi, T. Schauer, S. M. Savaresi, and K. J. Hunt, "Data-driven control design for neuroprotheses: a virtual reference feedback tuning (VRFT) approach," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 1, pp. 176–182, 2004.

[4] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Model predictive control with learned vehicle dynamics for autonomous vehicle path tracking," *IEEE Access*, vol. 9, 2021.

[5] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time Neural MPC: Deep Learning Model Predictive Control for Quadrotors and Agile Robotic Platforms," *IEEE Robotics and Automation Letters*, 2023.

[6] Y. Ding, A. Pandala, C. Li, Y.-H. Shin, and H.-W. Park, "Representation-free model predictive control for dynamic motions in quadrupeds," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1154–1171, 2021.

[7] T. Nammoto and T. Mariyama, "Robust Real-Time Model Predictive Control using Inertial Motion Prediction, Bisection, and Self-Tuning," *Proc. IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2024.

[8] D. Kostadinov and D. Scaramuzza, "Online Weight-Adaptive Nonlinear Model Predictive Control," *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–6, 2020.

[9] R. Grandia, F. Farshidian, A. Dosovitskiy, R. Ranftl, and M. Hutter, "Frequency-aware model predictive control," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1517–1524, 2019.

[10] D. He, W. He, and X. Song, "Efficient predictive cruise control of autonomous vehicles with improving ride comfort and safety," *Measurement and Control*, vol. 53, pp. 18–28, 2020.

[11] Y. Liang, Y. Li, A. Khajepour, and L. Zheng, "Holistic Adaptive Multi-Model Predictive Control for the Path Following of 4WID Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, pp. 69–81, 2021.

[12] M. A. Stephens, C. Manzie, and M. C. Good, "Model predictive control for reference tracking on an industrial machine tool servo drive," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 808–816, 2013.

[13] M. Diehl, H. J. Ferreau, and N. Haverbeke, "Efficient numerical methods for nonlinear MPC and moving horizon estimation," in *Nonlinear Model Predictive Control*, Lecture Notes in Control and Information Sciences, vol. 384, pp. 391–417, Springer, 2009.

[14] S. Sun, P. Zhao, T. Zhang, B. Li, D. Yu, "Smoothing interpolation of five-axis tool path with less feedrate fluctuation and higher computation efficiency," *Journal of Manufacturing Processes*, vol. 109, pp. 669-693, 2024.

[15] Y. Li, F. S. Liang, L. Lu, C. Fan, "Improved time-optimal B-spline feedrate scheduling for NURBS tool paths in CNC machining," *Advanced Manufacturing*, vol. 11, pp. 111–129, 2023.

[16] K. Ishizaki and E. Shamoto, "A new real-time trajectory generation method modifying trajectory based on trajectory error and angular speed for high accuracy and short machining time," *International Journal of Precision Engineering and Manufacturing*, vol. 76, pp. 173–189, 2022.

[17] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks, " *IEEE Transactions on Neural Networks*, vol. 1 no.1, pp. 4–27, 1990.

[18] J. Bergstra, R. Bardenet, Y. Bengio and B. Kegl, "Algorithms for hyper-parameter optimization," *Neural Information Processing Systems*, pp. 2546–2554, 2011.