

# Direct Data-Driven Trajectory Tracking and Stabilization of an Autonomous Bicycle

Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos

**Abstract**—The control of autonomous riderless bicycles remains a challenging problem in nonlinear control research. Traditional model-based controllers, such as Feedback Linearization (FL), can achieve stabilization but rely on simplified dynamic models that may not capture unmodeled or time-varying effects. Recent data-driven control methods, including Data-enabled Predictive Control (DeePC) and Data-enabled Policy Optimization (DeePO), allow closed-loop optimal control directly from measured input-output data, bypassing explicit system identification. This paper presents a cascaded control architecture that integrates FL, DeePO, and DeePC for simultaneous stabilization, velocity regulation, and trajectory tracking of an autonomous bicycle. First, a multi-input-multi-output (MIMO) FL controller is derived to regulate the lean angle and forward velocity. Then, DeePO is employed to adaptively enhance the FL control law using closed-loop data generated under persistently exciting inputs. As a supplementary contribution, a new forgetting-factor integration scheme was introduced into the standard DeePO algorithm to enhance its performance under nonlinear dynamics and time-varying system parameters. Finally, DeePC operates in the outer control loop, generating reference trajectories for the FL+DeePO controller via regularized data-driven optimization. The proposed cascaded FL-DeePO-DeePC approach is validated through high-fidelity simulations in Simscape.

## I. INTRODUCTION

Autonomous riderless bicycles have emerged as lightweight and cost-effective platforms for research in nonlinear control, robotics, and autonomous mobility. Their dynamic behavior, however, remains a formidable challenge: the system is *nonlinear*, *underactuated*, *non-minimum-phase*, and exhibits *velocity-dependent* stability characteristics [1]. These features make the autonomous bicycle an excellent but difficult testbed for advanced control methodologies. Applications include autonomous fleet reallocation in bike-sharing systems [2], mobility assistance for users with reduced physical ability [3], and as realistic Vulnerable Road Users (VRUs) in the verification of autonomous vehicle safety [4].

Early research has focused on classical control strategies for balance and steering, such as Proportional-Integral-Derivative (PID) control design, [5] and Linear Quadratic Regulator (LQR) designs [4], or nonlinear controllers such as Sliding Mode Control (SMC) [6] and Feedback Linearization (FL) [7]. While these model-based approaches can stabilize the system under nominal conditions, their performance deteriorates under disturbances, unmodeled effects, or parameter

variations. Robust extensions such as Active Disturbance Rejection Control (ADRC) have been experimentally validated for bicycle balance [8]. Nonetheless, the approach required approximate models with as many as 25 parameters to be estimated from the system.

To circumvent the aforementioned limitations of model-based control approaches, reinforcement learning has been investigated for autonomous bicycle control [9]. Despite their success in simulation, such methods often suffer from poor sample efficiency [10]. More recently, *direct data-driven control* has emerged as a principled alternative, leveraging input-output data collected under persistently exciting inputs from a single experiment to synthesize optimal feedback policies without explicit system identification. [11].

Within this paradigm, *Data-enabled Predictive Control* (DeePC) [12] provides a model-free analogue of Model Predictive Control (MPC), while *Data-enabled Policy Optimization* (DeePO) [13], [14] allows adaptive LQR synthesis directly from data. These methods have demonstrated strong performance on various physical systems, including quadrotors [15], power networks [16], and legged robots [17]. Experimental work further shows that DeePO can adaptively stabilize an autonomous bicycle relying on data from the system [18].

This paper proposes a *cascaded data-driven control architecture* that combines FL, DeePO, and DeePC to exploit their complementary advantages.

- We first derive a MIMO FL controller that jointly regulates the lean angle and forward velocity, achieving partial input-output linearization despite the system's underactuation.
- We then augment the FL control law with an adaptive DeePO layer that refines the input commands. We introduce two extensions to the baseline algorithm: (i) an exponential *forgetting factor* to handle time-varying dynamics, and (ii) *state normalization* to improve numerical conditioning and convergence.
- Finally, we deploy DeePC in the outer loop to achieve trajectory tracking via real-time data-driven predictive optimization, thus forming a cascaded FL-DeePO-DeePC control structure.

### A. Statement of Contributions

In summary, this paper's main contributions include:

- A novel cascaded FL-DeePO-DeePC control architecture is proposed for simultaneous stabilization, velocity regulation, and trajectory tracking of an autonomous

The authors are with the Division of Intelligent Future Technologies, Mälardalen University, 721 23 Västerås, Sweden. (e-mails: niklas.persson@mdu.se, mojtaba.kaheni@mdu.se, alessandro.papadopoulos@mdu.se)

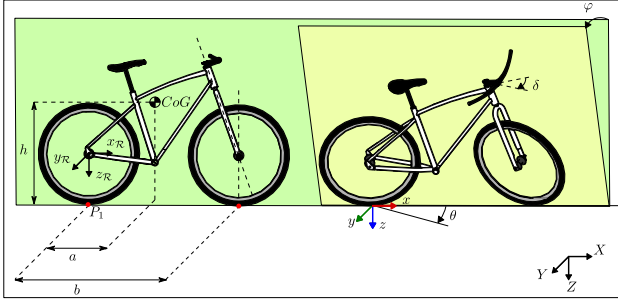


Fig. 1. Illustration of the parameters used in the bicycle model.

bicycle, designed to handle unmodeled dynamics and time-varying parameters.

- A novel forgetting-factor integration scheme is introduced into the conventional DeePO algorithm to improve its adaptability to nonlinear dynamics and time-varying system parameters.

### B. Organization

The remainder of this paper is organized as follows. Section II details the derivation of the MIMO FL controller. Section III presents the DeePO and DeePC algorithms, including the proposed modifications. Simulation results and comparative evaluations are provided in Section IV, followed by concluding remarks in Section V.

### C. Notation

Vectors and matrices are denoted by lowercase bold letters and uppercase letters, respectively. The identity matrix of order  $n$  is denoted by  $I_n$ , and  $A^\top$ ,  $A^\dagger$ , and  $\|A\|$  denote transpose, pseudoinverse, and the 2-norm of  $A$ , respectively. The operator  $\text{diag}(\cdot)$  forms a diagonal matrix. Weighted norms are written as  $\|x\|_Q^2 = x^\top Q x$  for  $Q \succeq 0$ . Continuous-time signals are  $x(t)$ , while discrete-time signals are  $x_k$ . Normalized quantities are indicated by a tilde (e.g.,  $\tilde{x}$ ).

## II. BICYCLE STABILIZATION

Balancing an autonomous bicycle poses a challenge due to its nonlinear dynamics and its unstable nature at lower velocities. While persistently exciting input signals are critical for data-driven control design, directly applying such inputs to an open-loop bicycle leads to loss of balance. To overcome this limitation, we employ an inner-loop FL controller, based on a simple point-mass model of a bicycle. Moreover, the FL controller partially cancels the system's nonlinearities, making the system, composed of FL and the bicycle, more suitable for direct data-driven control methods, as these methods are in general derived for linear systems [12], [13]. The proposed control design has shown promising results in our previous work using a constant velocity [18]. However, in this paper, the goal is to control a MIMO system, where both steering velocity  $\dot{\delta}$  and the acceleration of the rear wheel,  $\dot{v}$ , are the inputs to the system.

With respect to Fig. 1, the point-mass model of a bicycle is formulated as

$$\ddot{\varphi}(t) = \frac{g}{h} \sin(\varphi(t)) + \frac{a}{bh} \cos(\varphi(t)) v(t) \dot{\delta}(t) - \left( \frac{1}{bh} - \frac{1}{b^2} \tan(\delta(t)) \tan(\varphi(t)) \right) \tan(\delta(t)) v(t)^2, \quad (1)$$

where  $b$  represents the wheel base of the bicycle. The horizontal and vertical distances from the rear wheel's contact point with the ground ( $P_1$ ) to the center of gravity (CoG) of the bicycle are denoted  $a$  and  $h$ , respectively. Moreover,  $g$  denotes the gravitational constant. The state vector

$$\mathbf{x}(t) := [x_1(t), x_2(t), x_3(t), x_4(t)]^\top = [\varphi(t), \dot{\varphi}(t), \delta(t), v(t)]^\top$$

represents the lean angle, lean rate, steer angle, and forward velocity of the bicycle while  $\dot{\delta}(t)$  represents the input to the system. We can rewrite the model with two inputs,  $\mathbf{u}^{FL}(t) = [u_1(t), u_2(t)]^\top = [\dot{\delta}(t), \dot{v}(t)]^\top$ , as

$$\dot{\mathbf{x}} = \begin{bmatrix} \frac{g}{h} \sin(x_1) - \left( \frac{1}{bh} - \frac{1}{b^2} \tan(x_3) \tan(x_1) \right) \tan(x_3) x_4^2 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{a}{bh} \cos(x_1) x_4 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad (2)$$

and more compact as

$$\dot{\mathbf{x}} = f(\mathbf{x}) + G(\mathbf{x}) \mathbf{u}^{FL}, \quad \mathbf{y}^{FL} = h(\mathbf{x}), \quad (3)$$

where

$$\begin{aligned} y_1^{FL} &= h_1(\mathbf{x}) = x_1 = \varphi \\ y_2^{FL} &= h_2(\mathbf{x}) = x_4 = v, \end{aligned} \quad (4)$$

are the outputs we are interested in controlling using FL. Since the relative degree,  $r$ , of  $y_1^{FL}$  and  $y_2^{FL}$  is  $r_1 = 2$  and  $r_2 = 1$  respectively, the total relative degree of the system is 3, i.e., we have one state which we can not directly control. Thus, we can only achieve partial input-output linearization of the system.

The design procedure of the FL controller for the MIMO system follows the approach in the Slotine and Li book [19]. First, the Lie derivatives for the decoupling matrix  $E(\mathbf{x})$  are computed as

$$E(\mathbf{x}) = \begin{bmatrix} L_{g_1} L_f^{r_1-1} h_1 & L_{g_2} L_f^{r_1-1} h_1 \\ L_{g_1} L_f^{r_2-1} h_2 & L_{g_2} L_f^{r_2-1} h_2 \end{bmatrix} = \begin{bmatrix} \frac{a}{bh} \cos(x_1) x_4 & 0 \\ 0 & 1 \end{bmatrix} \quad (5)$$

and since  $\det(E(\mathbf{x})) = \frac{a}{bh} \cos(x_1) x_4$  for  $v \neq 0$  and lean angles between  $\pm \frac{\pi}{2}$  the system is locally input-output linearizable. The FL control law is chosen as

$$\mathbf{u}_t^{FL} = E(\mathbf{x})^{-1}(\mathcal{L} - \alpha(\mathbf{x})), \quad (6)$$

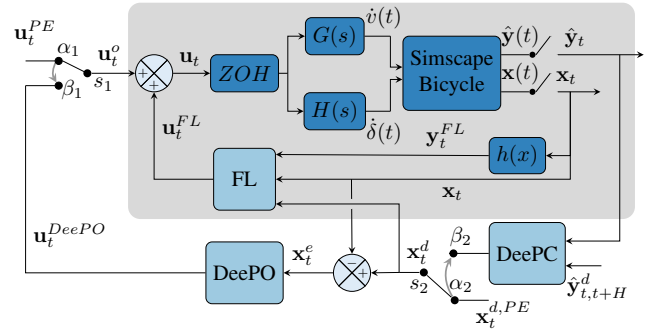
$$\begin{aligned} \alpha(\mathbf{x}) &= \begin{bmatrix} L_f^2 h_1 \\ L_f h_2 \end{bmatrix} = \\ &= \begin{bmatrix} \frac{g}{h} \sin(x_1) - \left( \frac{1}{bh} - \frac{1}{b^2} \tan(x_3) \tan(x_1) \right) \tan(x_3) x_4^2 \\ 0 \end{bmatrix} \end{aligned} \quad (7)$$

### III. DATA-DRIVEN CONTROL

### A. DeePO

$$\begin{aligned} X_{0,t} &:= [\mathbf{x}_0^e \quad \mathbf{x}_1^e \quad \dots \quad \mathbf{x}_{t-1}^e] \in \mathbb{R}^{n \times t}, \\ U_{0,t} &:= [\mathbf{u}_0^o \quad \mathbf{u}_1^o \quad \dots \quad \mathbf{u}_{t-1}^o] \in \mathbb{R}^{m \times t}, \\ X_{1,t} &:= [\mathbf{x}_1^e \quad \mathbf{x}_2^e \quad \dots \quad \mathbf{x}_t^e] \in \mathbb{R}^{n \times t}, \end{aligned} \quad (8)$$
$$D_{0,t} := \begin{bmatrix} U_{0,t} \\ X_{0,t} \end{bmatrix} \quad (9)$$

| Parameter           | Symbol | Value | Unit             |
|---------------------|--------|-------|------------------|
| CoG w.r.t $p_1$ (x) | $a$    | 0.550 | m                |
| CoG w.r.t $p_1$ (z) | $h$    | 0.700 | m                |
| Wheelbase           | $b$    | 1.200 | m                |
| Gravity             | $g$    | 9.82  | m/s <sup>2</sup> |
| $k_{1\varphi}$      | -      | 2     | -                |
| $k_{2\varphi}$      | -      | 10    | -                |
| $k_{3\varphi}$      | -      | 0.1   | -                |
| $k_{1v}$            | -      | 25    | -                |



has full row rank  $\text{rank}(D_{0,t}) = m+n$ . The sample covariance of the data is then [13]:

$$\Phi_t := \frac{1}{t} D_{0,t} D_{0,t}^\top = \begin{bmatrix} U_0 D_{0,t}/t \\ X_0 D_{0,t}/t \end{bmatrix} = \begin{bmatrix} \bar{U}_0 \\ \bar{X}_0 \end{bmatrix}, \quad (10)$$

$$\begin{bmatrix} K \\ I \end{bmatrix} = \Phi_t V. \quad (11)$$

To obtain an initial stabilizing policy for DeePO, we solve the regularized covariance-parameterized LQR problem [18, Eq. (27)]

$$\begin{aligned}
& \min_{W, S, L, Z \succeq 0} \text{Tr}(Q_{\text{DeepPO}} S) + \text{Tr}(R_{\text{DeepPO}} L) + \gamma \text{Tr}(Z \Phi_t), \\
& \text{s.t. } S = \bar{X}_{0,t} W, \\
& \quad [S - I, \quad \bar{X}_{1,t} W; \quad W^\top \bar{X}_{1,t}^\top, \quad S] \succeq 0, \\
& \quad [L, \quad \bar{U}_{0,t} W; \quad W^\top \bar{U}_{0,t}^\top, \quad S] \succeq 0, \\
& \quad [Z, \quad W; \quad W^\top, \quad S] \succeq 0,
\end{aligned} \tag{12}$$

$$K_0 = U_{0,t} W S^{-1}, \quad V_{\text{prim}} = \Phi_t^{-1} \begin{bmatrix} K_{t_0} \\ I \end{bmatrix}, \quad (13)$$

### B. Forgetting factor

By using rank-one updates of the data matrices, it is possible to run the algorithm in an efficient way without the need to solve any optimization problem online [13]. To improve adaptation to time-varying or nonlinear dynamics,

we introduce a recursive weighting of the covariance matrix that favors newer information through a time-varying weight,  $w_t$ . Let  $\lambda_f \in (0, 1]$  be the forgetting factor and define

$$w_{t+1} = \lambda_f w_t + 1, \quad w_0 = 0. \quad (14)$$

The exponentially weighted covariance update then becomes

$$\Phi_{t+1} = \left( \frac{w_{t+1} - 1}{w_{t+1}} \right) \Phi_t + \frac{1}{w_{t+1}} \phi_t \phi_t^\top, \quad (15)$$

where  $\phi_t = [\mathbf{u}_t^\top \mathbf{x}_t^\top]^\top$ . When  $\lambda_f = 1$ , the recursion in (15) reduces to the standard unweighted form used in [14]. Furthermore, the inverse of  $\Phi_t$  can be efficiently updated via the Sherman–Morrison formula [20],

$$\Phi_{t+1}^{-1} = \frac{w_{t+1}}{w_{t+1} - 1} \left( \Phi_t^{-1} - \frac{\Phi_t^{-1} \phi_t \phi_t^\top \Phi_t^{-1}}{w_{t+1} - 1 + \phi_t^\top \Phi_t^{-1} \phi_t} \right), \quad (16)$$

and finally, the parameterized policy itself can be updated in a rank-one fashion [14]

$$V_{t+1} = V_t + \frac{1}{w_{t+1}} \Phi_{t+1}^{-1} \phi_t \phi_t^\top (I - \Phi_t) V_t. \quad (17)$$

enabling a fully recursive online implementation. The use of  $\lambda_f < 1$  makes recent data more important in the online update of the data matrices.

### C. Normalization

As the state measurements of DeePO inherently have different units and magnitudes (e.g. lean angle in degrees, forward velocity in m/s), we normalize the data prior to computing the initial policy and before updating any data matrices in the online phase. For each row,  $\mathbf{z}$ , in  $D_{0,t}$ , the mean  $\mu_z$  and standard deviation  $\sigma_z$  are computed, and  $z$ -score normalization is then applied to the data as

$$\tilde{\mathbf{z}} = \frac{\mathbf{z} - \mu_z}{\sigma_z}. \quad (18)$$

The DeePO recursion is then carried out entirely on normalized data with corresponding scaled weighting matrices

$$\tilde{Q} = S_x^{-\top} Q_{\text{DeePO}} S_x^{-1}, \quad \tilde{R} = S_u^{-\top} R_{\text{DeePO}} S_u^{-1}, \quad (19)$$

where  $S_x = \text{diag}(\sigma_x)$ ,  $S_u = \text{diag}(\sigma_u)$ . After updating the normalized policy  $\tilde{K}$ , the gain is mapped back to physical coordinates as

$$K = S_u^{-1} \tilde{K} S_x. \quad (20)$$

This ensures that the controller is applied to the unnormalized physical states of the bicycle. The complete algorithm is summarized in Algorithm 1, where the gradient of the covariance-parameterized LQR cost  $J_t(V_t)$  is computed as

$$\nabla J_t(V_t) = 2(\tilde{U}_{0,t}^\top \tilde{R} \tilde{U}_{0,t} + \tilde{X}_{1,t}^\top P_t \tilde{X}_{1,t}) V_t \Sigma_t, \quad (21)$$

where  $P_t$  and  $\Sigma_t$  are obtained recursively as in [14]. The projection operator  $\Pi_{\tilde{X}_{0,t}} = I_n - \tilde{X}_{0,t}^\dagger \tilde{X}_{0,t}$  enforces  $\tilde{X}_{0,t} V_t = I_n$  after each gradient step. Finally, the learning rate  $\eta$  is normalized by  $\|\tilde{U}_{0,t} \Pi_{\tilde{X}_{0,t}} \tilde{U}_{0,t}^\top\|^{-1}$  following the normalization strategy in [18].

---

### Algorithm 1 DeePO (normalized, with forgetting factor)

---

**Input:** Offline data  $(X_{0,t_0}, U_{0,t_0}, X_{1,t_0})$ , scaling  $(\mu_x, S_x, \mu_u, S_u)$ , initial  $\tilde{K}_{t_0}$ , stepsize  $\eta, \lambda_f$ .  
1: **init**  $w_{t_0} = 0$ ,  $\tilde{\Phi}_{t_0}^{-1}$ , and  $\tilde{X}_{0,n,t_0}, \tilde{X}_{1,n,t_0}, \tilde{U}_{0,n,t_0}$ .  
2: **for**  $t = t_0, t_0 + 1, \dots$  **do**  
3:   **Apply**  $\mathbf{u}_t^{\text{DeePO}} = S_u^{-1} \tilde{K}_t S_x \mathbf{x}_t^e$ , measure  $\mathbf{x}_{t+1}^e$ .  
4:   **Normalize**  $\tilde{\mathbf{x}}_t^e = S_x^{-1}(\mathbf{x}_t^e - \mu_x)$ ,  $\tilde{\mathbf{u}}_t = S_u^{-1}(\mathbf{u}_t - \mu_u)$ ,  $\tilde{\phi}_t = [\tilde{\mathbf{u}}_t^\top \tilde{\mathbf{x}}_t^\top]^\top$ .  
5:   **Update**  $w_{t+1}$  via (14); update  $\tilde{\Phi}_{t+1}^{-1}$  via (16).  
6:   **Update data matrices with normalized data**  
 $\tilde{X}_{1,t+1}, \tilde{X}_{0,t+1}, \tilde{U}_{0,t+1}$  using the weights of (15).  
7:   **Policy parameterization:**  $V_{t+1} = \tilde{\Phi}_{t+1}^{-1} \begin{bmatrix} \tilde{K}_t \\ I_n \end{bmatrix}$ .  
8:   **One step projected gradient descent**  
 $V'_{t+1} = V_{t+1} - \eta_t \Pi_{\tilde{X}_{0,n,t+1}}(\nabla J_{t+1}(V_{t+1}))$ .  
9:   **Gain update:**  $\tilde{K}_{t+1} = \tilde{U}_{0,t+1} V'_{t+1}$ .  
10: **end for**

---

### D. DeePC

In our setting, the DeePC algorithm predicts the system's future trajectories without requiring an explicit model. Given a dataset of length  $T$ , containing the measured inputs  $\hat{\mathbf{u}}_t \in \mathbb{R}^m$  and outputs  $\hat{\mathbf{y}}_t \in \mathbb{R}^p$  [12], we construct the corresponding Hankel matrices  $\mathcal{H}_\ell$  of depth  $\ell$  as

$$\mathcal{H}_\ell(\mathbf{u}^{\text{DeePC}}) := \begin{bmatrix} \hat{\mathbf{u}}_0 & \hat{\mathbf{u}}_1 & \cdots & \hat{\mathbf{u}}_{T-\ell} \\ \hat{\mathbf{u}}_1 & \hat{\mathbf{u}}_2 & \cdots & \hat{\mathbf{u}}_{T-\ell+1} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\mathbf{u}}_{\ell-1} & \hat{\mathbf{u}}_\ell & \cdots & \hat{\mathbf{u}}_{T-1} \end{bmatrix}, \quad (22)$$

$\mathcal{H}_\ell(\hat{\mathbf{y}})$  is defined analogously. Here

$$\hat{\mathbf{y}}_t = [x_t, y_t, \theta_t], \quad (23)$$

represents the pose of the bicycle and

$$\hat{\mathbf{u}}_t = [\varphi_t^d, v_t^d]^\top, \quad (24)$$

provides the desired lean angle and forward velocity for the inner control loop. By selecting an initialization horizon  $T_{\text{ini}}$  and a prediction horizon  $H$ , the input and output Hankel matrices are partitioned as

$$\mathcal{H}_{T_{\text{ini}}+H}(\hat{\mathbf{u}}) = \begin{bmatrix} U_q \\ U_f \end{bmatrix}, \quad \mathcal{H}_{T_{\text{ini}}+H}(\hat{\mathbf{y}}) = \begin{bmatrix} Y_q \\ Y_f \end{bmatrix},$$

where  $(U_q, Y_q)$  correspond to the past data and  $(U_f, Y_f)$  to the future trajectories. At each sampling instant, DeePC solves the following finite-horizon optimization problem:

$$\begin{aligned} \min_{g, u, y} \quad & \|\hat{\mathbf{y}} - \hat{\mathbf{y}}_d\|_{\hat{Q}}^2 + \|\hat{\mathbf{u}} - \hat{\mathbf{u}}_d\|_{\hat{R}}^2 + \lambda_{\text{gd}} \|\mathbf{g} - \mathbf{g}_d\|_2^2 \\ & + \lambda_{\text{ini}} \|Y_q \mathbf{g} - \hat{\mathbf{y}}_{\text{ini}}\|_2^2 + (\hat{\mathbf{y}}_H - \hat{\mathbf{y}}_H^d)^\top P (\hat{\mathbf{y}}_H - \hat{\mathbf{y}}_H^d), \\ \text{s.t.} \quad & \begin{bmatrix} U_q \\ Y_q \\ U_f \\ Y_f \end{bmatrix} \mathbf{g} = \begin{bmatrix} \hat{\mathbf{u}}_{\text{ini}} \\ \hat{\mathbf{y}}_{\text{ini}} \\ \hat{\mathbf{u}} \\ \hat{\mathbf{y}} \end{bmatrix}, \end{aligned} \quad (25)$$

where  $\hat{\mathbf{y}}_{t,t+H}^d = [\hat{\mathbf{y}}_t^d, \hat{\mathbf{y}}_{t+1}^d, \dots, \hat{\mathbf{y}}_{t+H}^d]$  represents the desired trajectory and  $\hat{\mathbf{u}}_d = [0, 3.33\text{m/s}]$  denotes the nominal input. Before the input is passed to the inner FL-DeePO controller, the discrete-time derivatives  $\dot{\varphi}_t^d$  and  $\dot{v}_t^d$  are computed, yielding the complete desired state vector  $\mathbf{x}_t^d = [\varphi_t^d, \dot{\varphi}_t^d, v_t^d, \dot{v}_t^d]^\top$ . Moreover,  $\mathbf{g}$  is the coefficient vector describing a feasible trajectory,  $\lambda_{\text{gd}}$  and  $\lambda_{\text{ini}}$  are regularization parameters [12]. The weighting matrices  $\hat{Q}$  and  $\hat{R}$  penalize the deviation from the desired outputs and the control effort, respectively. The vector  $\mathbf{g}_d$  represents the steady-state trajectory that best aligns with the desired signals by linearly combining the columns of the Hankel matrices. Among all possible combinations consistent with the desired signals,  $\mathbf{g}_d$  is the minimum-norm solution, providing a compact representation of the desired steady-state behavior [15]. To further promote stability and smooth convergence toward the terminal desired pose, a terminal quadratic penalty is added on the predicted output at the end of the horizon. The corresponding terminal weighting matrix is  $P = I_p \times 10^4$ .

In this paper, DeePC is placed in the outer loop of the cascade architecture, as illustrated in Fig 2. The optimization problem in (25) is solved at every sampling interval using the OSQP quadratic programming solver at a sampling rate of 0.1s [21].

The Hausdorff distance  $d_H(A, B)$  [22], provides a measure of similarity between two parametrized curves,  $A$  and  $B$ :

$$d_H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b) \right\}, \quad (26)$$

where  $d(a, b)$  is the Euclidean distance between the points  $a$  and  $b$ . To find the parameters of the DeePC problem in (25), we formulate another optimization problem aiming to minimize the Hausdorff distance between the desired trajectory and the bicycle trajectory:

$$\begin{aligned} \min_{\hat{Q}, \hat{R}, \lambda_{\text{gd}}, \lambda_{\text{ini}}} \quad & d_H(\hat{\mathbf{y}}_d, \hat{\mathbf{y}}(\hat{Q}, \hat{R}, \lambda_{\text{gd}}, \lambda_{\text{ini}})) \\ \text{s.t.} \quad & 100 \leq \hat{Q}_{ii} \leq 600, \\ & 0.001 \leq \hat{R}_{jj} \leq 50, \\ & 10 \leq \lambda_{\text{gd}} \leq 500, \\ & 10^3 \leq \lambda_{\text{ini}} \leq 10^{10}. \end{aligned} \quad (27)$$

where  $\hat{\mathbf{y}}(\cdot)$  denotes the measured trajectory of the bicycle.

#### IV. RESULTS

In this section, we first present the bicycle we used in the simulation. Next, the setup of the controllers is presented, followed by the results from simulations and insights from the results.

##### A. Simulation setup

To evaluate the performance of the proposed method, a CAD model of an instrumented bicycle is imported into MathWorks Simscape and controlled through Simulink. Two revolute joints connect the front and rear wheel to the main frame, and the actuation of the rear wheel is controlled by

$u_2(t) = \dot{v}$ . To model the rear wheel motor, a simple low-pass filter

$$G(s) = \frac{1}{0.5s + 1}, \quad (28)$$

is utilized. Moreover, a third revolute joint connects the front fork and handlebar to the mainframe and is controlled by  $u_1(t) = \dot{\delta}(t)$ . The steering dynamics is modelled using a step response matching procedure [4], and the resulting transfer function from the control signal to the applied steering rate

$$H(s) = \frac{100 + s}{100} \quad (29)$$

is put in series with the bicycle model as illustrated in Fig 2. The steering rate control signal is saturated at  $\pm 4\text{rad/s}$ , and has a resolution of  $0.024\text{rad/s}$ . Zero-mean Gaussian noise was added to all measured and estimated signals in both the inner and outer control loops. Each noise component  $\xi_i$  was drawn from a normal distribution  $\xi_i \sim \mathcal{N}(0, \sigma_i^2)$  with  $\sigma_i = 0.01$ . The noise was applied to  $\varphi_t, \delta_t, x_t, y_t, \psi_t, v_t$  and their respective derivatives.

An initial simulation is conducted where the bicycle is controlled using only the FL controller and a persistently exciting input, i.e.  $\mathbf{u}_t = \mathbf{u}_t^{FL} + \mathbf{u}_t^o = \mathbf{u}_t^{FL} + \mathbf{u}_t^{PE}$ , the input and error state data are sampled at 100Hz. We use 1000 input and output samples to solve (12) and obtain an initial policy for DeePO with  $Q_{\text{DeePO}} = \text{diag}(50, 1, 0.1)$ ,  $R_{\text{DeePO}} = \text{diag}(1, 10)$ ,  $\eta = 10^{-4}$ ,  $\gamma = 0.1$ , and  $\lambda_f = 1 - 10^4$ . Next, the switch ( $s_1$ ) in Fig. 2 is moved from  $\alpha_1$  to  $\beta_1$ , while  $s_2$  is placed in  $\alpha_2$ . Another simulation is conducted using DeePO and FL for stabilizing the bicycle while tracking the desired states  $\mathbf{x}_t^{d, PE}$  which consist of a slow varying sinusoidal with a low amplitude, high frequency pseudorandom binary sequence (PRBS) for  $\varphi_t^d$  and  $v_t^d$ , while  $\dot{\varphi}_t^d$  and  $\dot{v}_t^d$  are obtained by computing the discrete time derivative of  $\varphi_t^d$  and  $v_t^d$ . The input and output data, with length  $T = 331$ , are sampled at 10Hz and later used for design of the DeePC controller. As the performance of DeePC on a nonlinear system is heavily influenced by the data, a grid search is conducted on  $T_{\text{ini}} = [1, 2, \dots, 10]$  and  $H = [10, 15, \dots, 30]$ , where the problem (27) is solved for each combination using Matlabs `patternsearch` routine, and each set of parameters is evaluated by tracking the desired trajectory given in Fig. 4, using the control system given in Fig. 2.

##### B. Simulation results

The results of solving the problem in (27) are given in Fig. 3 over different prediction and initialization horizons. The best set of parameters  $\hat{Q} = \text{diag}(510.7, 423.4, 423.4)$ ,  $\hat{R} = \text{diag}(5 \times 10^{-3}, 42.5)$ ,  $\lambda_{\text{gd}} = 47.24$ ,  $\lambda_{\text{ini}} = 5 \times 10^8$  are obtained with  $H = 30$  and  $T_{\text{ini}} = 1$  resulting in  $d_H = 3.03\text{m}$ . The trajectory tracking performance of the bicycle using the best set of parameters is presented in Fig. 4. The tracking performance of the DeePO and FL controller for the desired states,  $\mathbf{x}_t^d$ , is given in Fig. 5. Moreover, to evaluate the performance of our proposed FL-DeePO-DeePC scheme, we compare it to the results of using a PID-MPC scheme in our previous work [23]. The PID was used in an inner loop for balancing the bicycle through

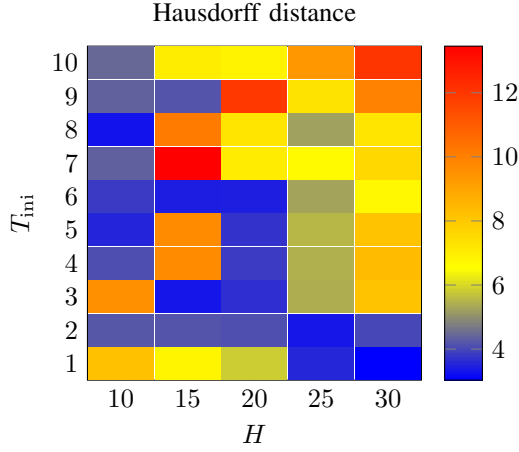


Fig. 3. Optimal value solving (27) for different values of  $H$  and  $T_{\text{ini}}$ .

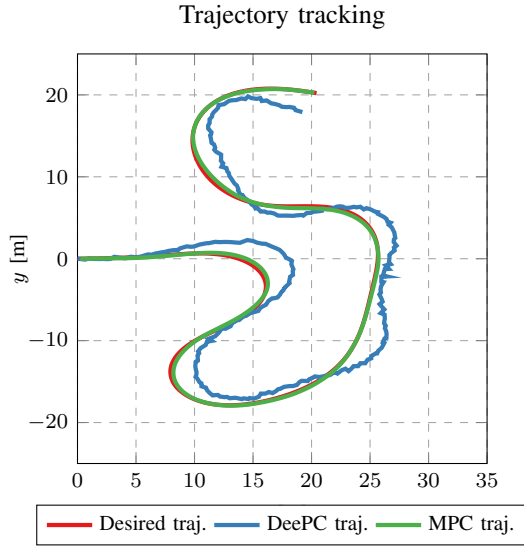


Fig. 4. Trajectory tracking using the proposed FL-DeePO-DeePC scheme, compared to a PID-MPC scheme [23].

actuation of the steer rate, and the MPC in the outer loop tracked a given trajectory and generated references for the PID. The comparison is presented in Fig. 4, where the PID-MPC scheme resulted in a  $d_H = 0.359\text{m}$ .

### C. Discussion

The results demonstrate that the proposed FL-DeePO-DeePC cascade can maintain balance and achieve reliable trajectory tracking of the nonlinear bicycle. The DeePO controller enhances the performance of the FL controller, while the DeePC controller in the outer loop realizes the trajectory tracking based only on nonlinear, noise-cluttered input-output data of the system. Nevertheless, the model-based PID-MPC benchmark clearly outperformed the proposed data-driven approach, yielding a Hausdorff distance of  $d_H = 0.36\text{m}$  compared to  $d_H = 3.03\text{m}$  for the best DeePC configuration. This can be explained by the fact that the MPC uses a given bicycle model of the system and state feedback, while DeePC predictions rely on finite

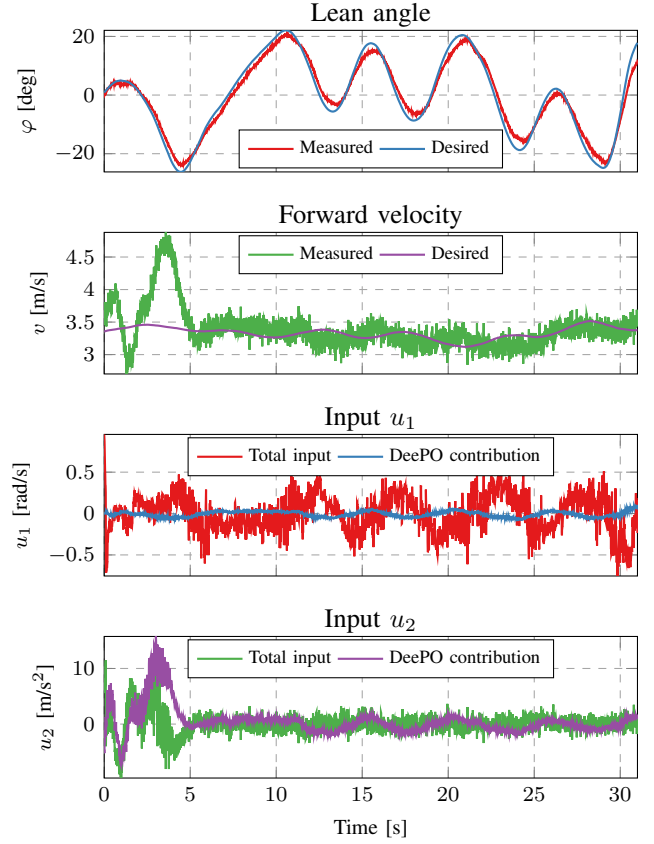


Fig. 5. The tracking performance of the FL-DeePO controller. The first two plots show the tracking of the desired lean angle and velocity, respectively, while the last two plots highlight the DeePO contribution in the total input.

Hankel matrices that approximate the nonlinear dynamics only locally, and are cluttered with noise. To close the gap between MPC and DeePC, there are several options that can be explored, for example, improving the excitation data, refining the regularization, or combining model-based and data-driven elements similar to the FL-DeePO setup in the inner control loop.

The grid search results in Fig. 3 suggest trends in the Hausdorff distance with respect to the initialization and prediction horizons. Smaller  $T_{\text{ini}}$  generally yielded better performance, suggesting that shorter past windows of data are sufficient to capture the relevant system behavior. In contrast, increasing  $H$  improved performance to  $H = 30$ , beyond which the improvement saturated. Thus, a longer horizon enhances prediction but increases optimization complexity.

Despite the performance gap between a model-based and data-driven trajectory tracking controller, the DeePC controller remained stable and feasible across all tested horizons and required no explicit dynamic model, highlighting its robustness and ease of deployment.

## V. CONCLUSION

This paper presented a cascaded control architecture combining FL, DeePO, and DeePC for balance and trajectory tracking of an autonomous bicycle. The scheme achieved

full trajectory tracking in simulation using only measured input–output data.

To improve numerical conditioning for datasets with differing units and magnitudes, a normalization step is incorporated into the DeePO algorithm. Moreover, a forgetting factor is added to DeePO to enhance adaptability to nonlinear and time-varying dynamics in the balance controller. In the DeePC algorithm, regularization is used to cope with nonlinearities, and a terminal constraint is applied to improve stability at the end of the horizon. However, comparison with a model-based PID–MPC approach revealed a significant performance gap in tracking precision.

Possible future research directions include adaptive hybrid strategies that integrate DeePC with approximate models, extend the dataset online, and validate the proposed approach experimentally on the physical platform.

## REFERENCES

- [1] J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, “Linearized dynamics equations for the balance and steer of a bicycle: a benchmark and review,” *Proceedings of the Royal Society A: mathematical, physical and engineering sciences*, vol. 463, no. 2084, pp. 1955–1982, 2007.
- [2] N. C. Sanchez, L. A. Pastor, and K. Larson, “Autonomous bicycles: A new approach to bicycle-sharing systems,” in *International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.
- [3] L. Alizadehsaravi and J. K. Moore, “Bicycle balance assist system reduces roll and steering motion for young and older bicyclists during real-life safety challenges,” *PeerJ*, vol. 11, p. e16206, 2023.
- [4] N. Persson, T. Andersson, A. Fattouh, M. C. Ekström, and A. V. Papadopoulos, “A comparative analysis and design of controllers for autonomous bicycles,” in *European Control Conference (ECC)*, 2021, pp. 1570–1576.
- [5] Y. Tanaka and T. Murakami, “Self sustaining bicycle robot with steering controller,” in *The 8th IEEE International Workshop on Advanced Motion Control*, 2004, pp. 193–197.
- [6] M. Defoort and T. Murakami, “Sliding-mode control scheme for an intelligent bicycle,” *IEEE Transactions on Industrial Electronics*, vol. 56, no. 9, pp. 3357–3368, 2009.
- [7] N. Getz, “Control of balance for a nonlinear nonholonomic non-minimum phase model of a bicycle,” in *Proceedings of 1994 American Control Conference - ACC '94*, vol. 1, 1994, pp. 148–151 vol.1.
- [8] M. Baquero-Suárez, J. Cortés-Romero, J. Arcos-Legarda, and H. Coral-Enriquez, “A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle,” *Multibody System Dynamics*, vol. 45, no. 1, pp. 7–35, 2019.
- [9] S. Choi, T. P. Le, Q. D. Nguyen, M. A. Layek, S. Lee, and T. Chung, “Toward self-driving bicycles using state-of-the-art deep reinforcement learning algorithms,” *Symmetry*, vol. 11, no. 2, p. 290, 2019.
- [10] L. P. Tuyen and T. Chung, “Controlling bicycle using deep deterministic policy gradient algorithm,” in *International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2017, pp. 413–417.
- [11] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. De Moor, “A note on persistency of excitation,” *Systems & Control Letters*, vol. 54, no. 4, pp. 325–329, 2005.
- [12] J. Coulson, J. Lygeros, and F. Dörfler, “Data-enabled predictive control: In the shallows of the DeePC,” in *18th European Control Conference (ECC)*, 2019, pp. 307–312.
- [13] F. Zhao, F. Dörfler, and K. You, “Data-enabled policy optimization for the linear quadratic regulator,” in *62nd IEEE Conference on Decision and Control (CDC)*, 2023, pp. 6160–6165.
- [14] F. Zhao, F. Dörfler, A. Chiuso, and K. You, “Data-enabled policy optimization for direct adaptive learning of the LQR,” *arXiv preprint arXiv:2401.14871*, 2024.
- [15] E. Elokda, J. Coulson, P. N. Beuchat, J. Lygeros, and F. Dörfler, “Data-enabled predictive control for quadcopters,” *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8916–8936, 2021.
- [16] I. Markovsky, L. Huang, and F. Dörfler, “Data-driven control based on the behavioral approach: From theory to applications in power systems,” *IEEE Control Systems Magazine*, vol. 43, no. 5, pp. 28–68, 2023.
- [17] R. T. Fawcett, K. Afsari, A. D. Ames, and K. A. Hamed, “Toward a data-driven template model for quadrupedal locomotion,” *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7636–7643, 2022.
- [18] N. Persson, M. Kaheni, and A. V. Papadopoulos, “A direct data-driven control design for autonomous bicycles,” in *IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2024, pp. 114–120.
- [19] J. Slotine and W. Li, *Applied Nonlinear Control*, ser. Prentice-Hall International Editions. Prentice-Hall, 1991.
- [20] J. Sherman and W. J. Morrison, “Adjustment of an inverse matrix corresponding to a change in one element of a given matrix,” *The Annals of Mathematical Statistics*, vol. 21, no. 1, pp. 124–127, 1950.
- [21] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, “OSQP: an operator splitting solver for quadratic programs,” *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.
- [22] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*. Springer Science & Business Media, 2009, vol. 317.
- [23] N. Persson, M. C. Ekström, M. Ekström, and A. V. Papadopoulos, “Trajectory tracking and stabilisation of a riderless bicycle,” in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 1859–1866.