

Mälardalen University Press Dissertations
No. 453

**DIRECT DATA-DRIVEN AND MODEL-BASED
CONTROL DESIGN FOR AN AUTONOMOUS BICYCLE**

Niklas Persson

2026



Department of Engineering Sciences

Copyright © Niklas Persson, 2026
ISBN 978-91-7485-737-5
ISSN 1651-4238
Cover photo by Henrik Falk
Printed by E-Print AB, Stockholm, Sweden

Till mina superhjältar, Billy och Frans

"Niklas är alltid bra att ha"
- Johan Hjorth

Acknowledgements

During the summer breaks as a teenager, I used to work as a caretaker at the local high school, where my father worked. This was probably the best work in the world, with almost daily Swedish "fika" and a lot of variation in the work you did. I never landed a full-time job there, though. Instead, about 12 and a half years ago, I set my first foot inside Mälardalen University (at that time Mälardalens Högskola). The plan was to do a technical preparatory year, and then three years of study to hopefully become an engineer. Luckily, plans can change. When I'm writing this, I'm about to defend my third thesis, and no one is more surprised than me. However, it was never a one-man job, it has involved lots of people during the years, some more than others.

Without a single doubt, I wouldn't be here if it weren't for Alessandro Vitorio Papadopoulos. You have kept me stable, or at least marginally stable, despite disturbances and noise. I have always felt that you have my back. You have been there through hard times and through fun times, you have always treated me with respect and patience, and you are always very humble. But most importantly, you are a fantastic human being. You are an optimal supervisor, a global optimum if you like, so from the bottom of my heart, Thank you!

However, stabilizing me could never be a one-man job. Martin has been there from start to finish, always there if I needed something, and Micke, who got me into robotics in the first place. Thank you for all the support, discussion, "fika", laughs, supervision, and for being great friends. Finally, Mojtaba, I would not have completed my thesis without you. You provided me with feedback even before you became my supervisor. You always have time to explain things so that I understand, even though my learning rate is not always that high. I would also like to thank Jacob and Nikola for reviewing my proposals.

A huge thank you to all my friends at C2, Jonas, for still putting up with me in the office despite the bicycle and the occasional disturbances. Lennie, the

calmest stressed guy around the office, is always there if I need something. Even though we run at different sample rates, Carl, you are a great friend and always have time to discuss the smallest details. Johan, my work-wife, I will probably never agree with you on anything; however, you are free to have the wrong opinion. Many of the photos in the thesis and through countless presentations are the work of Henrik Falk; additionally, he is always there with a solution if I need something. I would not have survived the spring of 2025 without the help of Emil. You always have time for my stupid questions on ROS, see you at lunch (or probably not). To my German friends, Christoph and Lukas, for the interesting discussions. To the best boss in the business, the master chief, Fredrik, I couldn't include a GIF, nor will I use a thousand words, however, I will use two. Thank you!

I would never have done this without the people close to me, mum and dad, for always supporting me, laughing with me and, perhaps, most importantly, teaching me to never give up. To my brother and his family, who have always been looking out for me, through thick and thin. Lina, you are my best friend, have always supported me, and you are one hell of a mum. Johanna, for putting sunshine in my life and supporting me on the last stretch.

Last, but not least, to my two superheroes, for whom this thesis is dedicated. Billy and Frans, you are the most beautiful boys in the world. I'm forever thankful for every moment I spend with you, the greatest privilege in my life is to be your dad. I will always love you.

In the end, it turns out that almost daily "fika" and variations of work tasks can be found at other jobs...

Niklas Persson, dad, PhD (hopefully)
Västerås, January 2026

Abstract

Autonomous bicycles constitute challenging benchmark systems for control, due to their nonlinear, non-holonomic, and in general, underactuated, open-loop unstable dynamics. Traditional model-based controllers such as proportional-integral-derivative (PID) controllers and linear quadratic regulators (LQRs) can stabilize the bicycle, but rely on simplified models that may not capture unmodelled and time-varying effects. In contrast, recent direct data-driven control methods based on Willems’ fundamental lemma bypass explicit modelling, yet typically assume linear time-invariant dynamic systems and require persistently exciting inputs that are difficult to apply safely on unstable systems.

This thesis investigates how traditional model-based and direct data-driven control methods can be used, and combined, to balance and guide an autonomous bicycle using mainly steering actuation as input. First, PID, LQR, fuzzy controller, feedback linearization (FL), and direct data-driven controllers are designed and compared in high-fidelity simulations and experiments on an autonomous bicycle. The results show that classical model-based controllers provide strong baselines, while direct data-driven controllers can enhance performance when combined with classical controllers. Second, a unified framework is proposed in which an inner-loop FL controller stabilizes and partially linearizes the bicycle, and an outer-loop direct data-driven controller operates on the FL-stabilized system. Two different types of direct data-driven methods are evaluated in this setting: a static, nonlinear controller and the Data-enabled Policy Optimization (DeePO) algorithm. Third, the DeePO algorithm is analyzed and modified to mitigate state perturbations, leading to a perturbation-free variant studied on LTI systems. Finally, a model-based PID–MPC trajectory tracking scheme is compared with a data-driven framework relying on Data-enabled Predictive Control (DeePC) for trajectory tracking, combined in a cascade architecture with the FL-DeePO setup. Simulations show that while PID–MPC achieves better tracking accuracy, the data-driven cascade attains successful trajectory tracking without relying on

an explicit dynamic model.

Sammandrag

Autonoma cyklar utgör utmanande riktmärkessystem för reglering, på grund av sin icke-linjära, icke-holonom och i allmänhet underaktuerade, öppen-slinga-instabila dynamik. Traditionella modellbaserade regulatorer, såsom proportional–integral–derivata-regulatorer (PID) och linjärt kvadratiske regulatorer (LQR), kan stabilisera cykeln, men bygger på förenklade modeller som inte nödvändigtvis fångar omodellerade och tidsvarierande effekter. I kontrast till detta kringgår nyare direkta data-drivna regleringsmetoder baserade på Willems fundamentallemma explicit modellering, men antar typiskt linjära tidsinvarianta dynamiska system och kräver ständigt exciterande insignaler som är svåra att tillämpa på ett säkert sätt på instabila system.

Denna avhandling undersöker hur traditionella modellbaserade och direkta data-drivna regleringsmetoder kan användas, och kombineras, för att balansera och guida en autonom cykel med huvudsakligen styraktivering som indata. För det första konstrueras PID-, LQR-, fuzzyregulatorer, feedbacklineariseringsregulatorer (FL) samt direkta data-drivna regulatorer, vilka jämförs i högfidelitets-simuleringar och experiment på en autonom cykel. Resultaten visar att klassiska modellbaserade regulatorer utgör starka riktmärken, medan direkta data-drivna regulatorer kan förbättra prestandan när de kombineras med klassiska regulatorer.

För det andra föreslås ett enhetligt ramverk där en innerloop-FL-regulator stabiliserar och delvis lineariserar cykeln, medan en ytterloop-regulator baserad på direkt data-driven reglering verkar på det FL-stabiliserade systemet. Två olika typer av direkta data-drivna metoder utvärderas i denna konfiguration: en statisk, icke-linjär regulator och algoritmen Data-enabled Policy Optimization (DeePO). För det tredje analyseras DeePO-algoritmen och modifieras för att motverka tillståndsstörningar, vilket leder till en perturbationsfri variant som studeras på linjära tidsinvarianta (LTI) system.

Slutligen jämförs ett modellbaserat PID–MPC-upplägg för banföljning med ett data-drivet ramverk som bygger på Data-enabled Predictive Control (DeePC) för banföljning, kombinerat i en kaskadarkitektur med

FL–DeePO-strukturen. Simuleringar visar att PID–MPC uppnår bättre uppföljningsnoggrannhet, medan den data-drivna kaskaden uppnår lyckad banföljning utan att förlita sig på en explicit dynamisk modell.

List of Publications

Papers Included in This Thesis¹

Paper A: *A Comparative Analysis and Design of Controllers for Autonomous Bicycles.*

Niklas Persson, Tom Andersson, Anas Fattouh, Martin Ekström, Alessandro V. Papadopoulos.

In European Control Conference (ECC), 2021.

Paper B: *Trajectory Tracking and Stabilisation of a Riderless Bicycle.*

Niklas Persson, Martin Ekström, Mikael Ekström, Alessandro V. Papadopoulos.

In International Conference on Intelligent Transportation Systems (ITSC), 2021.

Paper C: *A Direct Data-Driven Control Design for Autonomous Bicycles.*

Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.

In International Conference on Automation Science and Engineering (CASE), 2024

Paper D: *An Adaptive Data-Enabled Policy Optimization Approach for Autonomous Bicycle Control.*

Niklas Persson, Feiran Zhao, Mojtaba Kaheni, Florian Dörfler, Alessandro V. Papadopoulos.

Submitted to Transactions on Control Systems Technology (Feb 2025)

¹The included papers have been reformatted to comply with the thesis layout.

Paper E: *A Modified Adaptive Data-Enabled Policy Optimization Control to Resolve State Perturbations.*

Mojtaba Kaheni, **Niklas Persson**, Vittorio De Iuliis, Costanzo Manes, Alessandro V. Papadopoulos.

Accepted to the Conference on Decision and Control(CDC), 2025

Paper F: *Direct Data-Driven Trajectory Tracking and Stabilization of an Autonomous Bicycle.*

Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.

Submitted to the European Control Conference (ECC) (Nov 2025)

Papers Not Included in This Thesis

- *A loop shaping method for stabilising a riderless bicycle.*
Tom Andersson, Niklas Persson, Anas Fattouh, Martin Ekström.
European Conference on Mobile Robots (ECMR), 2019.
- *On the initialization problem for timed-elastic bands.*
Niklas Persson, Martin Ekström, Mikael Ekström, Alessandro V. Papadopoulos.
IFAC-PapersOnLine, 2023.
- *Resilient Direct Data-Driven Control Design under Poisoned Input-Output Samples*
Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.
Submitted to the European Control Conference (ECC) (Nov 2025)

Contents

I	Thesis	1
1	Introduction	3
2	Background	7
2.1	Data-Driven Control	11
2.2	Human Control	14
3	Related Work	17
3.1	Model-based control	17
3.2	Trajectory tracking	19
3.3	Model-free control	21
4	Research Overview	25
4.1	Problem Formulation	25
4.2	Research Method	27
4.3	Experimental platform	29
5	Thesis Contributions	31
5.1	Research Contribution	31
5.2	Overview of included papers	38
5.2.1	Paper A	39
5.2.2	Paper B	40
5.2.3	Paper C	40
5.2.4	Paper D	41
5.2.5	Paper E	42
5.2.6	Paper F	43
6	Conclusions and Future Work	45
6.1	Future Work	46

Bibliography 48

II Included Papers 57

7 Paper A

A Comparative Analysis and Design of Controllers for Autonomous Bicycles 59

7.1	Introduction	61
7.2	Related Work	62
7.3	Modeling of instrumented bicycle	63
7.3.1	Experimental platform	63
7.3.2	Linear model	65
7.3.3	Nonlinear model	68
7.4	Control strategies	68
7.4.1	PID controllers	69
7.4.2	Linear quadratic regulator	70
7.4.3	Fuzzy controller	71
7.5	Results	72
7.5.1	Simulation setup	72
7.5.2	Simulation results	73
7.5.3	Experimental setup	73
7.5.4	Experimental results	74
7.6	Conclusion and Future Work	76
7.7	Acknowledgements	77

8 Paper B

Trajectory tracking and stabilisation of a riderless bicycle 81

8.1	Introduction	83
8.2	Related Work	84
8.2.1	Modelling	84
8.2.2	Path tracking	85
8.3	Bicycle model	86
8.4	Control design	89
8.4.1	Balance controller	89
8.4.2	Path tracking	92
8.5	Results	94
8.5.1	Simulation setup	95
8.5.2	Simulation results	96
8.5.3	Discussion	97

8.6	Conclusion	98
8.7	Acknowledgements	100
9	Paper C	
	Direct Data-Driven Control Design for Autonomous Bicycles	103
9.1	Introduction	105
9.1.1	Statement of Contributions	106
9.1.2	Organization	107
9.1.3	Notation	107
9.2	Direct Data-Driven Control	107
9.2.1	DPT Control for Nonlinear Systems	109
9.3	Autonomous Bicycle Control Design	110
9.3.1	Control Design	111
9.3.2	Simulation setup	113
9.3.3	Data collection	115
9.4	Numerical Results	116
9.4.1	Discussion	118
9.5	Conclusion	120
10	Paper D	
	An Adaptive Data-Enabled Policy Optimization Approach for Autonomous Bicycle Control	123
10.1	Introduction	125
10.1.1	Statement of Contributions	128
10.1.2	Organization	129
10.1.3	Notation	129
10.2	Autonomous bicycle control	129
10.2.1	Bicycle Dynamics and Mathematical Modeling	129
10.2.2	Control overview	131
10.3	Data-enabled policy optimization for autonomous bicycle control	133
10.3.1	The linear quadratic regulator	133
10.3.2	Data-driven covariance parametrization of the LQR with exponential weighted data	134
10.3.3	Data-enabled policy optimization for adaptive LQR control with exponentially weighted data	136
10.3.4	Learning an initial stabilizing policy using robustness promoting regularization	138
10.3.5	Control gain update rate	139
10.4	Simulations and experiments	139
10.4.1	Instrumented Bicycle	140

10.4.2	Simulation setup	140
10.4.3	Simulation results	142
10.4.4	Experimental setup	145
10.4.5	Experimental results	147
10.5	Conclusion	148

11 Paper E

A Modified Adaptive Data-Enabled Policy Optimization Control to Resolve State Perturbations		155
11.1	Introduction	157
11.1.1	Notation	158
11.2	Data-Driven Policy Optimization for LQR Learning	159
11.2.1	Background	159
11.2.2	Why is probing noise added to $\mathbf{u}_i$ in Algorithm 11.1? .	162
11.2.3	Undesired State Perturbations by Adding Probing Noise to $\mathbf{u}_i$	165
11.3	Perturbations-free DeePO (PFDeePO)	165
11.4	Simulations	170
11.5	Conclusion	171

12 Paper F

Direct Data-Driven Trajectory Tracking and Stabilization of an Autonomous Bicycle		177
12.1	Introduction	179
12.1.1	Statement of Contributions	180
12.1.2	Organization	180
12.1.3	Notation	180
12.2	Bicycle stablization	181
12.3	Data-driven control	183
12.3.1	DeePO	183
12.3.2	Forgetting factor	185
12.3.3	Normalization	186
12.3.4	DeePC	187
12.4	Results	189
12.4.1	Simulation setup	189
12.4.2	Simulation results	191
12.4.3	Discussion	191
12.5	Conclusion	193

I

Thesis

Chapter 1

Introduction

A bicycle, with its two inline wheels, is a naturally unstable system. The rider uses a combination of steering, moving of the body mass, and forward speed to stabilize the bicycle in everyday cycling. To balance, the center of mass must be kept over the base of support, i.e., the wheels. By steering the bicycle in the direction of the fall, the center of mass of the bicycle is brought back over the base of support, a fact that was known by Rankine already in 1869 [1]. A bicycle is a nonlinear, underactuated, non-holonomic system, and at lower velocities it is also naturally unstable [2], thus, it serves as an excellent benchmark in control. The same strategy that was known by Rankine in the 19th century is now used for stabilizing autonomous bicycles [3, 4, 5]. When the bicycle leans to the left, the handlebars steer to the left to maintain balance. This requires a sensor for measuring the lean angle, typically an inertia measurement unit (IMU), and two motors, one for forward movement and one for actuation of the handlebars. Other strategies for balancing autonomous bicycles include directly regulating the lean angle by employing flywheels [6], control moment gyroscopes [7], a moving mass [8], or changing the mechanical structure of the bicycle by adding extra wheels [9]. Moreover, some work utilizes a combination of different strategies as they have their own advantages and disadvantages [10]. On the one hand, balancing using direct regulation of the lean angle shows promising performance at lower velocities, but struggles at higher velocities [5]. On the other hand, stabilizing the bicycle by only steering regulation is not possible at zero velocity. However, using steering actuation requires a minor alteration of the bicycle's appearance, compared to mounting a flywheel or a moving mass, and is also more energy efficient [11].

Beyond its role as a control benchmark, an autonomous bicycle can serve as a maneuverable test object for assessing Autonomous Emergency Braking (AEB) and Autonomous Emergency Steering (AES) systems' performance in

vehicles at a cyclist encounter. Vulnerable Road Users (VRUs), including cyclists, are rather unprotected in urban traffic environments where they are often forced to share road segments with other vehicles [12]. In fact, recent reports shows that the number of accidents with VRUs are increasing ¹. To improve the safety of the VRUs, autonomous systems, such as AEB and AES systems, are developed for motorized vehicles. These systems are tested and evaluated by the European New Car Assessment Programme (EuroNCAP), where a bicycle, with a dummy cyclist, is placed on a sled ². The bike is constrained by the linear motion of the sled. Thus, an autonomous bicycle that could maneuver similarly to a real bicycle would improve the testing area. Another potential application for autonomous bicycles is to improve the availability in bike-sharing systems [13]. A common problem in bike-sharing systems is that the bikes end up in a few popular places and have to be redistributed by hand [9]. Instead, an autonomous bicycle could potentially re-localize itself and navigate to charging stations, streamlining fleet management in bike-sharing systems, similar to what has already been proposed for scooters [14]. Finally, it could also be used as a message delivery service or act as a steering assistant for individuals with limited physical capabilities [15].

Common for the majority of the controllers that have been applied to stabilizing bicycles is that they rely on first-principles models, which are derived from the geometric and physical parameters of the bike [2, 16]. Even though some of the models of bicycles have been validated in experiments [17] and successfully used for balancing bicycles [3], they may require up to 25 parameters to be measured and/or estimated. Moreover, in general, models do not fully leverage the extensive data collected from onboard sensors. This data could instead be utilized in a system identification process to estimate a model of the bicycle, rather than depending solely on first-principles modeling [18]. However, regardless of first-order principle modeling or system identification techniques, the modeling of a system is a time-consuming and complex task. In fact, it is estimated that about 75% of the time put into control design is devoted to designing the model [19]. Moreover, in general, a model does not perfectly represent the actual system due to unmodeled dynamics, parametric uncertainty, various disturbances, and time-varying dynamics. To address these limitations and as an alternative to model-based control design, direct data-driven control techniques, based on Willems' fundamental lemma [20], have emerged in recent years [21, 22, 23]. The fundamental lemma establishes

¹<https://www.mynewsdesk.com/se/dina-forsakringar/pressreleases/trafikolyckorna-med-oskyddade-trafikanter-oekar-saa-ser-det-ut-regionalt-3409613>

²<https://www.euroncap.com/en/vehicle-safety/the-ratings-explained/vulnerable-road-user-vru-protection/aeb-cyclist/>

that, under persistency of excitation, sampled trajectories are enough to parameterize all future trajectories over a finite horizon, eliminating the need for an explicit parametric model. Thus, direct data-driven control bypasses the often time-consuming task of modeling the system and instead, it identifies a control law directly by using samples of the persistently excited input and the corresponding output data from the system.

The autonomous bicycle is a nonlinear, non-holonomic, and in general underactuated system where the dynamics of the system are heavily influenced by the speed; it is a challenging system for any control technique. Even more so in terms of direct data-driven control techniques, which often assume the system to be linear time-invariant (LTI) and stable. However, as more research is conducted, direct data-driven control is used for more challenging control problems [24, 25, 26, 27, 28].

In this thesis, several traditional and direct data-driven control approaches are investigated for balancing an autonomous bicycle through actuation of the rear wheel and the handlebars, i.e., the considered autonomous bicycle is underactuated. Furthermore, following a cascaded control scheme, the autonomous bicycle is enhanced with trajectory tracking capabilities, relying on either model-based or direct data-driven control methods. The direct data-driven control methods evaluated for stabilization of the bicycle include a nonlinear controller [21] and the Data-enabled Policy Optimization (DeePO) algorithm, an adaptive control approach to handle time-varying systems [23]. Moreover, several model-based controllers for stabilizing the autonomous bicycle, such as Proportional-Integral-Derivative (PID), Linear Quadratic Regulator (LQR), and Feedback Linearization (FL), are considered. A unified control framework is also proposed, combining a traditional control method using FL with both direct data-driven control methods. Finally, we implement and compare Model Predictive Control (MPC) and Data-enabled Predictive Control (DeePC) for realizing trajectory tracking in the outer loop. The control approaches are evaluated in high-fidelity simulations and experiments conducted on an instrumented bicycle.

Chapter 2

Background

The bicycle has been the subject of research for over a century. In 1899, both Francis Whipple [29] and Emmanuel Carvallo [30] published their respective work on bicycle dynamics and showed in theory that a bicycle can self-stabilize, given enough forward speed. In their work, the bicycle is modeled using four rigid bodies, including two thin wheels, a main frame, and a front fork with handlebars. Each body has its own associated mass and inertia. Other peripherals, such as pedals, chains, and brakes, are neglected. Moreover, the friction between the joints is disregarded in their models. Since they assume thin wheels, the contact between the ground and the wheels is modelled as a point contact. The friction in the point contact is neglected, and the wheels are assumed to roll without side or longitudinal slip, thus making the system non-holonomic. Finally, the rider is assumed to be a rigid part of the main frame, adding to its mass and inertia. A passive rider model based on this Carvallo–Whipple bicycle model (commonly referred to as the Whipple model) was proposed by Plöchl et al., where the riders’ lean angle was added as an additional degree of freedom [31]. The Whipple model can be linearised around its upright equilibrium point, assuming small steering and lean angles and with a constant forward velocity ($\dot{v} = 0$), resulting in the fourth-order model [2],

$$\mathbf{M}\ddot{\mathbf{q}}(t) + v\mathbf{C}_1\dot{\mathbf{q}}(t) + [g\mathbf{K}_0 + v^2\mathbf{K}_2]\mathbf{q}(t) = \mathbf{f}(t), \quad (2.1)$$

where $\mathbf{q}(t) = [\varphi(t), \delta(t)]^\top$ represents the lean and steering angle respectively. The inertia and mass properties of the bicycle are captured in the symmetric invertible matrix $\mathbf{M}$. The damping-like matrix is denoted $\mathbf{C}$ and is proportional to the forward velocity v . The stiffness matrix is divided into two parts: a constant part $\mathbf{K}_0$ proportional to the gravity constant g and one part, $\mathbf{K}_1$, which is proportional to the square of the forward velocity. The input, $\mathbf{f}(t) = [\tau_\varphi(t), \tau_\delta(t)]^\top$,

is the lean torque and the torque applied to the handlebar, respectively. To construct the constant matrices $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}_0$, and $\mathbf{K}_1$, 25 design parameters need to be measured or estimated from the bicycle [17].

Given the Whipple model, it is possible to form its characteristic fourth-degree polynomial as

$$\det(\mathbf{M}\lambda^2 + v\mathbf{C}_1\lambda + \mathbf{K}_0 + v^2\mathbf{K}_2) = 0, \quad (2.2)$$

where the eigenvalues λ can be used to determine the velocities at which the bicycle is self-stabilized. When the real part of all eigenvalues is less than zero, i.e., $\text{Re}(\lambda) < 0$, the bicycle is self-stabilized. As can be seen in Figure 2.1, the asymptotically stable range lies between the so-called weave speed, v_c , and capsize speed v_w [2]. To compute the eigenvalues, JBike6¹ was utilized. The 25 parameters used to construct $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}_0$, and $\mathbf{K}_1$ in Figure 2.1 originate from the instrumented bicycle used in experiments in **Paper A** and **Paper D** in this thesis. Using the characteristic equation, the Whipple model was experimentally validated in the work of Kooijman et al. [17], where a human ran beside the bicycle to an adequate speed and then released it into free coasting. While coasting, the roll and steering angle, yaw rate, and the velocity of the bicycle were measured. By means of curve fitting, the eigenvalues of the measurements could be estimated and compared to the eigenvalues of (2.2). They found a good conformity in velocity ranges between 3 and 6 m/s, which suggests that the model is a good approximation of a real bicycle.

A drawback of the Whipple model is its complexity, and the 25 parameters that need to be measured or estimated are a time-consuming task [32, 17]. Getz and Marsden proposed a simple bicycle model that only requires five parameters [33, 34]. Consider the bicycle in Figure 2.2 where the centre of mass of the bike is denoted G , and the contact point between the rear wheel and the ground is denoted P_1 . The five parameters needed are the vertical h and horizontal distance a between G and P_1 , the wheelbase b , the total mass m of the bicycle, and the gravity constant g . The bike is considered to be a point mass, lumped together in the point G . Besides the assumptions made in the Whipple model, this model also neglects the radii and inertial moments of the wheels; it assumes a vertical steering axis, i.e., $\nu = \frac{\pi}{2}$ and, in extension, zero trail, $c = 0$. The input is the steering velocity, $\dot{\delta}(t)$. A similar model to the one presented in the work of Getz and Marsden is found in the work of Limebeer and Sharp [35], where they instead use the steering angle, $\delta(t)$, as control input for balance. An advantage of using $\delta(t)$ or $\dot{\delta}(t)$ as a control input, as compared to the steering

¹http://ruina.tam.cornell.edu/research/topics/bicycle_mechanics/JBike6_web_folder/index.htm

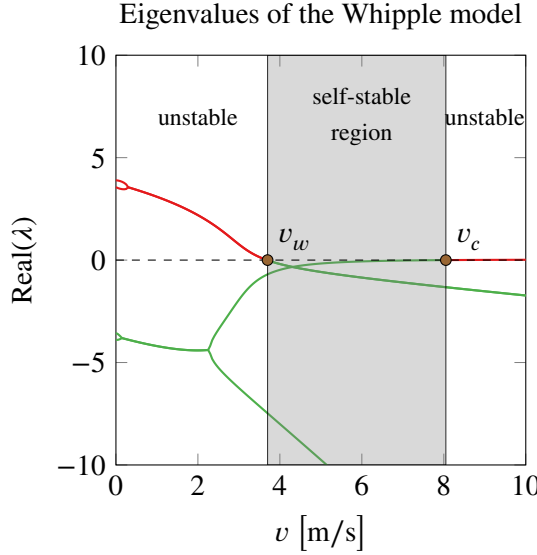


Figure 2.1: The Eigenvalues of the characteristic polynomial of the linearized Whipple model with different forward velocities. All eigenvalues have a negative real part between the weave speed (v_w) and capsize speed (v_c), making the system self-stabilized. The parameters are measured and estimated on the instrumented bicycle considered in this thesis.

torque which is used for the Whipple model, is that the steer dynamics can be neglected, which simplifies the model [36].

If we assume constant velocity, the roll dynamics of the Getz and Marsden bicycle model are given by

$$mh\ddot{\varphi}(t) = gm \sin(\varphi(t)) - ((1 - h\sigma \sin(\varphi(t))) \cos(\varphi(t))\sigma(t)v^2 + a \cos(\varphi(t))v\dot{\sigma}(t)) m \quad (2.3)$$

where $\sigma(t)$ denotes the curvature of the trajectory and is defined as

$$\sigma(t) = \frac{1}{R(t)} = \frac{\tan(\delta(t))}{b}. \quad (2.4)$$

Moreover, an extension of the Getz model was proposed by Yi et al. where the steering axis is no longer assumed to be vertical, i.e. $\nu \neq \frac{\pi}{2}$, which also provides a positive trail ($c > 0$) to the bicycle [37, 38]. Here, ν denotes the head angle, i.e., the angle between the steering axis and the ground (the corresponding caster angle is $\frac{\pi}{2} - \nu$). This extension offers a more realistic model of a bicycle, as most bicycles possess a positive trail. However, it has been

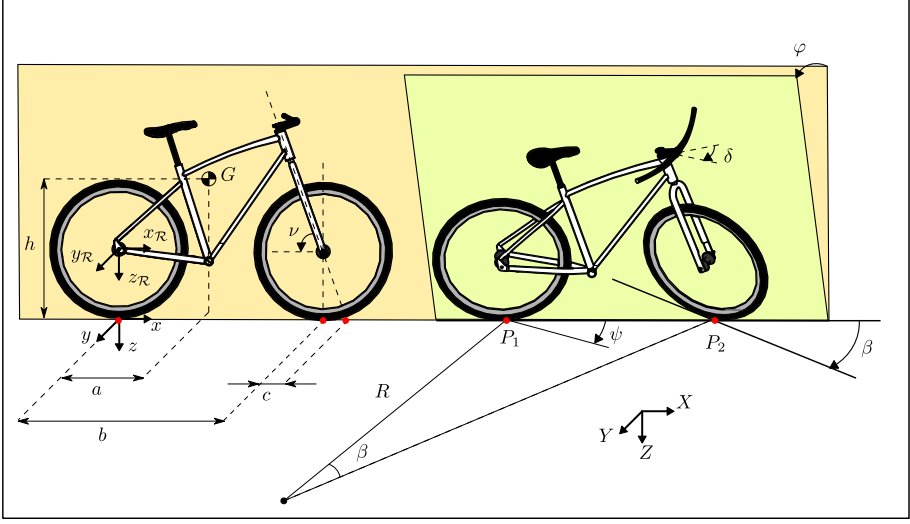


Figure 2.2: A bicycle riding on a flat horizontal plane.

shown that it is not a feature that is necessary for the self-stabilization of a bicycle [39]. The caster angle will also have an impact on the yaw angle ($\psi(t)$) of the bicycle and, in extension, the position ($x(t)$, $y(t)$) of the bicycle in the world frame (X , Y , Z). The pose [$x(t)$, $y(t)$, $\psi(t)$] of the bicycle can be computed for the horizontal plane using the following kinematic model:

$$\dot{x} = v(t) \cos(\psi(t)) \quad (2.5a)$$

$$\dot{y} = v(t) \sin(\psi(t)) \quad (2.5b)$$

$$\dot{\psi} = \frac{v(t) \tan(\beta(t))}{b} \quad (2.5c)$$

where $v(t)$ is the forward velocity of the rear wheel, and $\beta(t)$ is the effective steering angle at the ground and is given by

$$\beta(t) = \arctan\left(\frac{\tan(\delta)(t) \sin(v)}{\cos(\varphi)(t)}\right), \quad (2.6)$$

which is defined for all practical lean angles, i.e. $-\frac{\pi}{2} < \varphi < \frac{\pi}{2}$. With a non-zero caster angle, the roll dynamics of the model proposed by Yi et al. becomes

$$\begin{aligned} h^2 m \ddot{\varphi}(t) = & gm \left(h \sin \varphi(t) + \frac{ca \sin v}{b} \sigma(t) \cos \varphi(t) \right) \\ & - \left(1 - \frac{h \sigma(t)}{b} \sin \varphi(t) \right) \frac{h \sigma(t)}{b} \cos \varphi(t) v(t)^2 m \\ & - \frac{ahm}{b} \cos \varphi(t) \left(\sigma(t) \dot{v}(t) + v(t) \omega_{\sigma}(t) \right), \end{aligned} \quad (2.7)$$

where $\omega_e(t)$ is the effective steering angular velocity.

The more advanced model initially proposed by Yi, and later refined by Zhang and Yi, is slightly more complex compared to the Getz model, but has demonstrated success in various applications, including the development of balance controllers [5, 40], path tracking [37], and localization of a bicycle [41]. In contrast, the Whipple model has primarily been used to analyze rider and bicycle stability and human control [42, 17], though it has also contributed to the design of controllers for autonomous bicycles [3].

Additionally, numerous nonlinear models have been proposed in the literature, ranging from analytical models [43] to multibody dynamics software models [3, 10] and models designed based on system identification [18]. Given that research on bicycle dynamics modeling spans more than a decade, a comprehensive review is beyond the scope of this thesis. Instead, readers seeking a more detailed discussion of dynamic models for two-wheeled vehicles are referred to [2, 44, 45].

2.1 Data-Driven Control

While mathematical modeling can simplify the design of control schemes, it is also possible to develop controllers without relying on an explicit model, for instance, through Reinforcement Learning (RL) [46]. These controllers learn a control policy by iteratively interacting with the environment, where the algorithm receives feedback, either as penalties or rewards, based on their actions and guides the search. This allows for shaping different behaviors, such as improved robustness [47] or empirical stability margins [48]. Recent advancements in Deep Reinforcement Learning (DRL) have further enabled efficient policy learning for high-dimensional control tasks, such as balancing and trajectory tracking of bicycles in simulation [49]. However, RL methods are typically sample-inefficient, often requiring large numbers of simulations or experiments, and may be difficult to generalize to new scenarios. Moreover, an agent trained in simulations usually struggles when it is transferred to an experimental setup due to unmodelled dynamics, disturbances, noise, etc. [50]. An alternative, model-free, approach has emerged in recent years through *direct data-driven control*.

Data-driven control design can be divided into *indirect* and *direct* data-driven control design. As illustrated in Figure 2.3, indirect design refers to the process where a model is identified using system identification methods relying on input-output data from the system. In contrast, the direct design methods bypass the explicit modelling step and design a controller directly from the

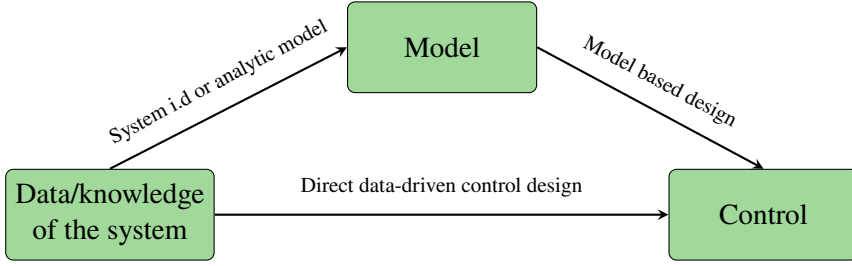


Figure 2.3: Direct data-driven control design bypasses the time-consuming step of modeling the system and designs controllers directly from measurements collected from the system.

input-output data [51]. In general, data-driven control is appealing for complex, high-dimensional, or uncertain systems, where accurate first-principles models are difficult and time-consuming to obtain. In this thesis, we focus on direct data-driven methods rooted in linear systems theory, where the underlying plant is represented as an LTI system. Moreover, data-driven control methods take advantage of technological advancements that allow for more data and computational power. By leveraging historical or real-time experimental data, direct data-driven control allows for the synthesis of controllers without the need for intermediate model identification steps [19].

A significant breakthrough in direct data-driven control is Willems’ Fundamental Lemma [20], which provides a rigorous mathematical foundation for using past input-output data to predict and control future system behavior. The lemma states that for an LTI system, a sufficiently rich, i.e., a persistently exciting set of input-output trajectories, contains all necessary information to describe the system’s dynamics. Let us consider the general discrete time LTI system

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \quad k \in \mathbb{N}_0, \quad (2.8)$$

where $\mathbf{x}_k \in \mathbb{R}^n$ and $\mathbf{u}_k \in \mathbb{R}^m$ denote the state and input vectors of real numbers of sample k , respectively. The matrices $\mathbf{A}^{n \times n}$ and $\mathbf{B}^{n \times m}$ represent the state and input matrices.

Now, assume that we have collected T samples of input–state data $\{\mathbf{u}_k, \mathbf{x}_k\}_{k=0}^{T-1}$. The input sequence $\mathbf{u}_{0:T-1} := \{\mathbf{u}_0, \dots, \mathbf{u}_{T-1}\} \in \mathbb{R}^{mT}$ is said to be *persistently exciting of order l* if the Hankel matrix

$$\mathcal{U}_0 = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_{T-l} \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_{T-l+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_{l-1} & \mathbf{u}_l & \cdots & \mathbf{u}_{T-1} \end{bmatrix},$$

has full rank ml . This implies that signal $\mathbf{u}$ must be sufficiently long, namely

$$T \geq (m + 1)l - 1. \quad (2.9)$$

Similarly, the state data can be structured in another Hankel matrix as

$$\mathcal{X}_0 = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{T-l} \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{T-l+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{l-1} & \mathbf{x}_l & \cdots & \mathbf{x}_{T-1} \end{bmatrix}.$$

The Fundamental Lemma holds under a set of assumptions; in addition to linearity and time invariance, the system also needs to be controllable and observable [51]. In its original form, the lemma also assumes noise-free measurements. Under these assumptions and if the input is persistently exciting of sufficiently high order, any l length trajectory $\{\mathbf{u}_k, \mathbf{x}_k\}_{k=0}^{l-1}$ of the system can be represented as a linear combination of the columns of the data Hankel matrices, i.e., there exists a vector $\mathbf{g}$ such that

$$\begin{bmatrix} \mathcal{U}_0 \\ \mathcal{X}_0 \end{bmatrix} \mathbf{g} = \begin{bmatrix} \mathbf{u}_{0:l-1} \\ \mathbf{x}_{0:l-1} \end{bmatrix}. \quad (2.10)$$

Based on these Hankel matrices, the goal is often to design a state-feedback control law, typically in the form

$$\mathbf{u}_k = K \mathbf{x}_k, \quad (2.11)$$

where K is the gain matrix to be determined. In the traditional setting, with a model, the feedback gain is typically obtained by solving a Riccati equation. In the direct data-driven framework, where no explicit model is available, one instead parameterizes all admissible closed-loop trajectories contained in the data and solves the LQR problem directly in terms of this parameterization [21, 52]. In the noise-free case, the resulting controllers are equivalent to their model-based counterparts. In practice, experimental data are corrupted by process and measurement noise, which can lead to reduced closed-loop performance if not properly accounted for. This has motivated a range of robust and regularized formulations that explicitly account for noise and modelling uncertainty, for instance by adding regularization terms in the design cost or by enforcing robustness constraints on the closed-loop behavior [53, 52, 54]. Other direct data-driven methods include predictive control, where the input sequence is optimized online [22], and robust state-feedback design [53], among others. By exploiting Willems' Fundamental Lemma, such direct methods can avoid the

time-consuming and complex task of system identification and instead formulate control problems purely in terms of data, which is particularly attractive for systems with uncertain or time-varying dynamics where traditional model-based controllers may struggle to maintain accuracy.

Finally, recent developments have extended the use of Willems' Fundamental Lemma to nonlinear and adaptive control settings, using tools such as Koopman operators, optimization, and machine learning methods to enhance the performance of data-driven control methods [21, 24, 55, 56]. These extensions have further strengthened the role of direct data-driven control as a practical and robust alternative to conventional control design, bridging the gap between data-driven machine learning approaches and classical model-based control theories.

2.2 Human Control

A human rider maintains balance on a bicycle through a combination of forward velocity, upper body lean, and torque applied to the handlebars. However, several experimental studies show that upper-body lean appears to play a minor role in balance. In treadmill experiments, Kooijman et al. observed that very little upper-body motion occurs and that stabilization is achieved almost entirely by steering control actions [42]. Moore et al. later confirmed this using a detailed motion capture analysis, showing that upper-body motions are mainly present at the pedalling frequency, suggesting that they are primarily a consequence of pedalling rather than a dedicated balance strategy [57]. Both studies also report that, at very low forward speeds, riders introduce another control input in the form of lateral knee movements, which are likely used to assist balance when steering alone is less effective [42, 57].

The observation that steering control is the main input was further supported by Zhang et al. [40], who studied a bicycle equipped with a flywheel. The flywheel introduced a lean torque disturbance without the rider's awareness. By analyzing the rider's response, it was shown that the rider primarily used actuation of the handlebars to compensate for the disturbance. These findings align with those of Kooijman and Moore, which support the conclusion that handlebar motion is the primary control input for balance and maneuvering. In a more recent research on rider-control identification, perturbation of the system in experiments was used along with system identification techniques to model the rider as a feedback controller acting primarily through steering torque, with additional contributions from upper-body motions at low speed or under challenging conditions [58].

Finally, to initiate a turn to the left, a rider typically first performs a subtle counter-steer to the right [59]. This brief counter-steer causes the bicycle–rider system to lean to the left, after which the rider turns the handlebars to the left to follow the desired path while maintaining balance. This counter-steering mechanism is executed unconsciously by most riders and is therefore not widely recognized outside the scientific literature. If counter-steering is prevented, riders quickly lose balance when attempting to turn, as shown in this video ². For a comprehensive survey on rider control and human–bicycle interaction, the reader is referred to [59].

²<https://www.youtube.com/watch?v=9cNmUNHSBac>

Chapter 3

Related Work

Several different methods have been proposed for the control of an autonomous bicycle. In this section, balance controllers, relying on a model-based approach, are covered first, followed by a section on trajectory tracking for autonomous bicycles. Finally, model-free control design is reviewed, including RL and direct data-driven control design.

3.1 Model-based control

Balancing an autonomous bicycle can be achieved using steering actuation and forward velocity [3, 5, 60], or by mounting a flywheel or a moving mass on the bicycle [8, 40, 61], which allows direct regulation of the lean angle. The advantage of direct lean angle regulation is that the system can be stabilized even when standing still; however, such approaches may struggle to maintain balance at higher velocities [5]. As a third option, the two former approaches can be combined, using direct lean-angle regulation at low speeds and switching to steering actuation once the bicycle has picked up speed [10, 62]. The bike used as the experimental platform in this thesis was originally developed in a project for verification of motorized vehicles' AEB systems [63], where it was important that the autonomous bicycle resembled an ordinary bicycle, which is difficult to realize with flywheels or a moving mass mounted on the frame. Therefore, this thesis focuses on balancing the bike using steering actuation in combination with forward velocity.

In 2005, Tanaka and Murakami balanced a bicycle on a roller using handlebar actuation at a given forward velocity [64]. With a reference lean angle of zero, the error between the measured lean angle and the reference was fed to a PD controller that actuated the handlebars. The PD controller was designed

based on a point-mass model of their bicycle. The controller was first evaluated in simulations and then implemented on a bike riding on a bicycle roller. Handlebar actuation for balancing was also used in the work of He et al. [5]. Using the point-mass model of Yi et al. [38], linearized around an equilibrium and for constant velocity, a combined feedforward and feedback controller was designed. In contrast to [64], the experiments were conducted outdoors rather than on a roller. During experiments, different forward velocities were used, and a lookup table was employed to switch the controller gains depending on the velocity. Balance using different forward speeds was also considered in the work of Yeh et al., where a Linear-Parameter-Varying (LPV) controller was used to balance a bicycle around campus at varying speeds [60]. The dynamics were decomposed into four linear subsystems, and the control gains were computed using Linear Matrix Inequalities (LMIs) to ensure robust stability under speed variations. The point-mass model proposed by Åström et al. [16] was used for control design. The controller was first evaluated in simulations and then on an instrumented bicycle, yielding impressive experimental results with good tracking performance.

The bicycle is widely recognized as one of the most energy-efficient modes of transportation [65], yet energy consumption in control algorithms for autonomous bicycles is rarely considered. To address this, Rodriguez-Rosa [11] proposed an adaptive control scheme aimed at reducing energy usage. A PI controller was designed using the point-mass model of Limebeer and Sharp [35]. The proportional and integral gains were dynamically adjusted as functions of the forward velocity to balance energy consumption, steady-state error, settling time, and disturbance rejection. Energy usage was quantified as the sum of squared control inputs, providing a clear metric for optimization. Simulation results demonstrated that the adaptive controller not only stabilized the bicycle but also significantly reduced energy consumption; at higher velocities, reductions of up to 98% were reported compared to a conventional PI controller with fixed gains across all considered speeds.

Since the bicycle is a nonlinear system, Defoort and Murakami employed nonlinear control techniques to stabilize it, specifically a second-order Sliding Mode Controller (SMC) combined with a Disturbance Observer (DOB) [4]. The control design was based on a nonlinear point-mass model. A comparative study of three strategies was carried out: SMC with a DOB, SMC without a DOB, and a PD controller with a DOB. Simulations on a nonlinear bicycle model showed that the SMC with a disturbance observer achieved significantly faster convergence to the desired lean angle than the other two controllers. For experimental validation, the same bicycle and a similar setup as in [64] were used. The results confirmed that the proposed strategy successfully maintained

balance for velocities between 0.4 and 3 m/s, and that the SMC with DOB provided the best overall performance. A known drawback of SMC, however, is that the switching nature of the control law may induce chattering in the control signal, potentially leading to high actuation effort and increased wear on the motors [66].

When balancing an autonomous bicycle, it is also important to consider both external and internal disturbances, such as wind, uneven terrain, friction in the joints, and other unmodeled dynamics. One way to address these issues is to design a controller that explicitly focuses on disturbance rejection, such as the Active Disturbance Rejection Controller (ADRC) proposed by Baquero-Suárez et al. [3]. Their design relies heavily on the Whipple model. First, a velocity analysis based on the eigenvalues of the characteristic polynomial in (2.2) was conducted to motivate the velocity range for experiments. The lower bound for the velocity was also established by analyzing the system zeros from steering torque to lean angle. This analysis revealed that the system becomes non-minimum phase, and thus more difficult to control, at velocities below $v_l = 0.78$ m/s, so the controller was designed for speeds above this limit. However, their analysis did not include the steering motor dynamics, which would alter both the non-minimum-phase range and the self-stabilizing range of the system. The Whipple model was then used to design the ADRC scheme. To evaluate the controller, a detailed CAD model of the bicycle was imported into Adams and controlled in co-simulation with Matlab. Subsequently, experiments were carried out on an instrumented bicycle for both constant and varying velocities in an outdoor environment. The results show that the bike could balance in both simulation and on straight asphalt tracks under varying forward velocities. However, noticeable oscillations remained in both the lean and steering angles. Disturbance rejection was also the focus in the work of He et al., where an ADRC scheme for balance was combined with a Deep Deterministic Policy Gradient (DDPG) controller for trajectory tracking [67]. The scheme was evaluated in co-simulation using V-REP and Matlab, and the results indicated that the proposed controller could balance the bicycle at varying velocities while tracking a reference trajectory. In the study, the controller was compared to the SMC–DOB proposed by Defoort and Murakami [4], and the ADRC was shown to stabilize the system faster. However, the control scheme has not yet been validated experimentally.

3.2 Trajectory tracking

Trajectory tracking for an autonomous bicycle is inherently more challenging than for many other wheeled robots due to the need for active control to main-

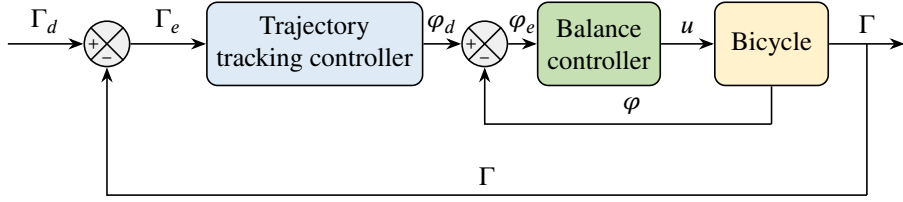


Figure 3.1: Cascaded control scheme for trajectory tracking and balance. The outer loop compares the desired pose Γ_d with the measured pose Γ to form the pose error Γ_e and computes a desired lean angle φ_d for the inner loop. The balance controller tracks φ_d using the lean angle error φ_e and generates the steering input u to the bicycle.

tain stability. When steering is used for balance, it is not straightforward to also use it for path tracking. A conflict may arise if the balance controller and trajectory tracking controller compute contrary control signals, i.e., one control signal results in a left steering action and one in a right steering action. A more effective approach is to use a cascaded control scheme, see Figure 3.1, where the trajectory tracking controller computes a desired lean angle for the balance controller to track.

Dao and Chen proposed a cascade control strategy to achieve both balance and trajectory tracking [68]. Their method employs an SMC in the inner loop to stabilize the bicycle by tracking a specified lean angle, while the outer loop uses a fuzzy logic controller with gain scheduling and integral action. In the simulation, the Whipple model and the parameters from the benchmark bicycle in [2] were used. From the results, it's clear that the integral action plays an important role in their trajectory tracking controller, especially when the disturbance is induced in the system. Using Getz's point-mass model, Yuan et al. [69] tackled trajectory generation and path following in simulation. They constructed the bicycle's planar trajectory using polynomial curves and employed Particle Swarm Optimization (PSO) to ensure compliance with constraints on the lean angle, while finding an optimal path. This formulation treats path tracking as an optimization problem subject to the bicycle's dynamic constraints, enabling accurate path following while maintaining stability. The proposed control strategy ensures accurate trajectory tracking while preserving stability, even for different speed scenarios. A nonlinear control approach for trajectory tracking in motorcycles was introduced by Yi et al. [37]. Since their work focused on motorcycles rather than bicycles, it considered significantly higher velocities, ranging from 5 m/s to 14 m/s. The controller was tested in numerical simulations with noise in the measured variables. Due to the high speeds and wide turns, the reported steering angles remained small, around 2° . Conse-

quently, its applicability to autonomous bicycles remains uncertain. However, all these approaches are evaluated on rather simple analytical models and may be difficult to realize on a real bicycle.

In the work of Baquero-Suárez et al., the ADRC was extended to trajectory tracking. By establishing a relationship between the bicycle's lean and yaw angles, they manipulated the desired lean angle to track the desired yaw angle. Their method was evaluated in simulations, where a realistic CAD model of the bicycle was controlled in Adams. Their method was evaluated at low velocities, 1.5 m/s, using wide circular paths and straight segments. Experimental results were also reported in the work of Zhao et al. [70], where path planning, path following, and obstacle avoidance for an autonomous bicycle were investigated. The path planning algorithm initially defines a linear path between two points and dynamically replans the trajectory in four phases upon detecting an obstacle. By sensing the yaw angle and position of the bicycle, a reference lean angle is computed, guiding the bike along the desired path. The results from experiments demonstrated that their bicycle successfully avoided static obstacles using a Lidar sensor for detection. Experimental results were also reported in the work of Yuh et al., who used a cascade LPV control for the first balance, and then the outer LPV loop for trajectory tracking [60]. The bicycle follows a virtual reference trajectory generated by the motion model proposed by Åström et al. [16]. The system is experimentally validated on a full-scale self-driving bicycle, achieving a trajectory tracking error of less than 0.18 m on campus roads.

3.3 Model-free control

Humans typically learn to balance a bicycle through repeated trial and error, gradually learning the control policy. A similar principle can be applied to balancing an autonomous bicycle using RL [71, 72, 73]. In the work by Choi et al. [71], a DDPG [74] is used to learn a policy for controlling the bicycle's speed, lean angle, and steering torque. Three different reward functions were compared: one where the focus was solely on the lean angle error, and two reward functions that had different weight terms for lean angle, lean rate, and acceleration. When the different reward functions were evaluated on a nonlinear bicycle model, the weighted reward function was superior to the reward function that only considered the lean angle error. However, the controller is only evaluated in simulation, so its behavior in a real-world setting remains unclear. Experimental evaluation of DDPG was considered in the work of Tuyen and Chung [72], using a LEGO bicycle, but the hardware's limited computational power was found to be insufficient for running the control algorithm effectively.

Another DRL algorithm is proposed for stabilizing a Whipple model of a bicycle and tracking a predefined path [73]. While the bicycle is balanced at varying velocities, it requires time-consuming training and lacks experimental validation.

Beyond purely learning-based schemes, several recent works combine RL with more traditional control methods. Guo et al. [75] propose a hybrid controller for a self-balancing bicycle robot without trail, where an FL law is tuned online using a synchronous actor–critic RL algorithm. Steering angle and driving velocity are treated as inputs, and the Lyapunov function is used to guarantee closed-loop stability. Simulations and experiments show improved balance performance compared to an FL controller. Sha et al. [76] introduce the Adaptive Residual Decision-Control Synthesis (ARDCS) framework, which pairs an inner LQR balance controller with an outer Dynamic Proximal Policy Optimization (DPPO) agent that generates desired lean angles and adds steering corrections. In extensive PyBullet simulations involving disturbance rejection, this cascaded control design outperforms standalone LQR and pure DRL, highlighting the potential of RL-classical control combinations for autonomous bicycle control.

Recently, direct data-driven control has emerged as an alternative approach for model-free control. It does not require extensive repetitions as in the case of RL. Additionally, it bypasses explicit modeling and instead derives control policies directly from system data. For unknown LTI systems in the noise-free case, it has been shown that these controllers coincide with the LQR and MPC obtained using traditional control design with a known model [22, 77, 21].

Within the direct data-driven framework, DeePO has been proposed as an adaptive, direct data-driven method that iteratively updates a feedback gain using online input–output data [23, 78]. DeePO relies on a covariance-based parameterization of the closed-loop system and computes policy gradients directly from data, without explicit model identification. The algorithm uses rank-one updates to efficiently compute the control policy. This enables real-time policy adaptation and makes DeePO particularly appealing for systems with slowly varying or uncertain dynamics. DeePO has been successfully applied to grid-connected power converters subject to changing grid conditions [24].

Another branch of direct data-driven control focuses on predictive control algorithms. In the model-based case, MPC has become a standard tool in robotics, process industry, and autonomous vehicles [79]. DeePC transfers the MPC idea to the model-free setting by using Hankel matrices constructed from past input–output data to predict future trajectories and optimize a control sequence over a receding horizon [22]. Regularized and robust variants have

been proposed to cope with noise and uncertainty [52, 80], and DeePC has been applied to a range of complex systems, such as power networks, quadcopters, and walking robots [51, 26, 27]. Recent work has further extended DeePC to LPV systems [81], nonlinear systems via Koopman operators [82, 83] or neural parameterization [84], and nonlinear output-feedback problems using online data selection (Select-DeePC) [85]. From an algorithmic standpoint, dimension reduction through singular value decomposition has been used to reduce the computational burden of DeePC [86]. Another option is to train a deep neural network to approximate the decision vector based on the Hankel matrices and input/output data of the system, thus removing the need to solve an optimization problem online [87]. The interested reader is referred to the recent review of Berberich and Allgöwer for a more detailed overview of the current state of research on DeePC [88].

To summarize, existing work on autonomous bicycles either focuses on model-based controllers, often with limited experimental validation, or on reinforcement learning and hybrid RL-classical schemes that remain largely confined to simulation studies or small-scale platforms. At the same time, direct data-driven control methods such as data-driven LQR, DeePO, and DeePC have mostly been developed and validated for linear, fully actuated systems in other domains, with only a few results on underactuated, nonlinear vehicles. In this thesis, we investigate both traditional and direct data-driven control approaches within a unified framework on an underactuated autonomous bicycle.

Chapter 4

Research Overview

In this section, the problem is formulated. Based on the problem formulation, four research questions are defined. Next, the research method and methodology are presented. Finally, the section concludes with a brief presentation of the autonomous bicycle and the hardware mounted on it.

4.1 Problem Formulation

Stabilizing a riderless bicycle, solely relying on the actuation of the steering rate, is a challenging task, especially at lower velocities, beneath the so-called *weave speed*. Based on the analysis of the Eigenvalues in Figure 2.1, the weave speed of the instrumented bicycle used in this thesis is around 3.8m/s; however, the evaluation of balance controllers is conducted at lower velocities. Therefore, the system we aim to control is nonlinear, open-loop unstable, non-holonomic, underactuated, and time-varying.

From the literature, it is evident that traditional control methods, such as PID and LQR, can successfully stabilize autonomous bicycles; however, their design relies on approximate models that are often linearized around a single equilibrium point. When the system deviates from this equilibrium, it will affect the control performance. Moreover, models are affected by parametric uncertainty and external and internal disturbances such as wind, uneven terrain, friction, and mechanical wear and tear, which can reduce the performance of a model-based control design. On the other hand, direct data-driven control methods often rely on persistently excited data. Directly applying a persistently excited input to the steering actuation may jeopardize the stabilization of the system, as if the bicycle leans to the right and is steered to the left, it will lose balance and fall over. Furthermore, direct data-driven control design often as-

sumes LTI systems; thus, the inherent nonlinear system poses challenges for direct data-driven control approaches.

Moreover, an autonomous bicycle should not only be able to balance but also track a reference trajectory. Trajectory tracking introduces additional challenges, as the system must simultaneously maintain stability and follow a desired trajectory while compensating for disturbances. Given the non-holonomic constraint, time-varying dynamics, and underactuated nature of a bicycle, achieving precise tracking requires careful coordination of steering and velocity inputs to regulate both lean angle and heading. As the velocity varies, the remaining dynamics of the system will vary; thus, it is important that a stabilizing controller can adapt to system changes. To evaluate the different control methods, a previously designed instrumented bicycle is utilized [89, 63]. However, before experiments are conducted, it is important to evaluate the approaches in realistic simulations. Finally, to simplify the integration of direct data-driven control methods and simplify the design, several hardware components need to be updated on the experimental platform.

Based on the problem formulation, the following research questions (RQ) are defined:

RQ 1. *How do traditional and direct data-driven control methods compare in terms of balancing an autonomous bicycle and tracking a reference lean angle at constant forward velocity?*

The first question aims to evaluate the performance of both traditional control methods and direct data-driven control methods in stabilizing an autonomous bicycle. Understanding their respective strengths and limitations, particularly in terms of balance control and reference tracking, is crucial for understanding how they can be combined.

RQ 2. *How can persistent excitation for direct data-driven controllers be achieved in open-loop unstable systems without compromising closed-loop stability and performance?*

The second research question investigates the problem of applying persistently exciting input signals in direct data-driven control design and the requirement to maintain stable closed-loop behavior. Many direct data-driven methods rely on probing signals to guarantee sufficient excitation of the system, but in open-loop unstable systems, such inputs can drive the state far from the desired operating point. For an autonomous bicycle, for instance, a badly timed steering

excitation may amplify a lean in the wrong direction and cause the bike to fall.

RQ 3. *How can model-based control design and direct data-driven control design be unified to improve stabilization of an autonomous bicycle at low speeds?*

Since traditional control methods rely on simplified models that may not fully capture system dynamics, and data-driven methods often require persistently excited data and are often designed for LTI systems, the third research question aims to investigate how direct data-driven control methods can be combined with model-based control design for enhancing the performance of the controller for nonlinear unstable systems.

RQ 4. *How do direct data-driven trajectory tracking controllers compare to model-based trajectory tracking controllers in terms of tracking a reference trajectory in simulation?* The fourth and final research question addresses the challenge of trajectory tracking for an autonomous bicycle. More specifically, it aims to investigate how model-based trajectory tracking control design can be replaced with a direct data-driven approach for realizing trajectory tracking.

4.2 Research Method

In this thesis, an iterative research approach is used. The methodology is illustrated in Figure 4.1. First, a research problem is identified, and relevant literature is reviewed. Depending on the findings of the literature review, the research problem may be reformulated. Based on the knowledge gained from the literature review, a solution to the problem is proposed. Next, the solution is implemented in Matlab for evaluation in simulation, where a detailed CAD model, designed in SolidWorks ¹, is imported to either Adams ² (**Paper A** and **Paper B**) or Simscape ³ (**Paper C, D, F**), and controlled through Simulink ⁴. If the results obtained from simulations are successful, the proposed controller may be evaluated in experiments, where it is implemented in LabVIEW ⁵ (**Paper A**) or

¹<https://www.solidworks.com/>

²<https://hexagon.com/products/product-groups/computer-aided-engineering-software/adams>

³<https://se.mathworks.com/products/simscape.html>

⁴<https://se.mathworks.com/products/simulink.html>

⁵<https://www.ni.com/sv-se/shop/labview.html>

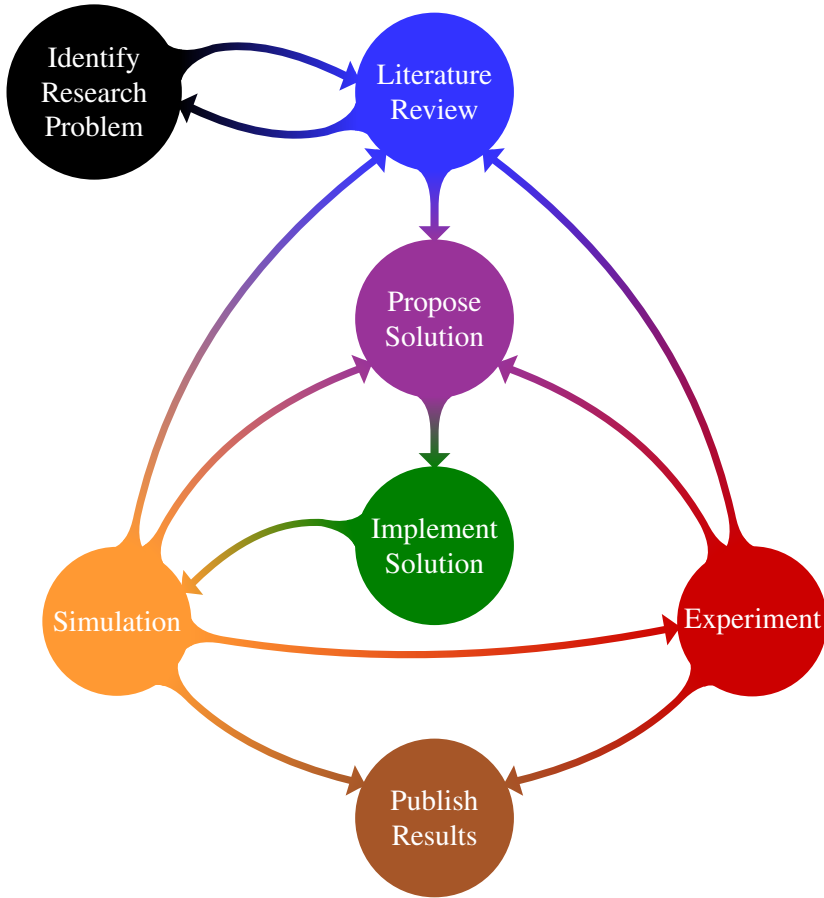


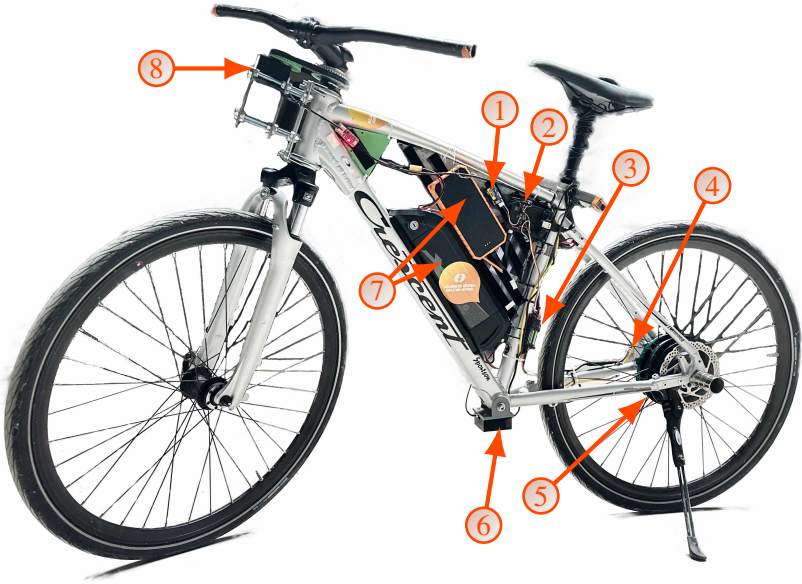
Figure 4.1: Research methodology used in the thesis.

ROS2⁶ (**Paper D**). The balance control framework in **Paper C** has been partly evaluated in experiments using LabVIEW [90]. Moreover, the trajectory tracking controllers implemented in **Paper B** and **Paper F** have only been evaluated in simulations due to the time-consuming and complex task of assessing them in experiments, and also the problem of how to establish ground truth data of the current pose of the bicycle. In both experiments and simulations, a quantitative analysis is performed on the collected data. In **Paper E**, the proposed approach is evaluated on a simple open-loop stable and controllable LTI system, as the main focus of this paper is on the improvements of DeePO, which are yet to be evaluated on the use case of the autonomous bicycle.

⁶<https://docs.ros.org/en/humble/index.htmls>

4.3 Experimental platform

In experiments, a regular-sized male model of an electric bicycle is utilized. The original battery and rear wheel Bafang motor, controlled through a Phoenix Edge HV 60 AMP Electronic Speed Controller (ESC), are used to drive the bicycle forward. A Hall sensor is used to measure the time intervals between 12 evenly distributed magnets on the rear wheel to estimate the speed of the bicycle. To measure the lean angle and lean rate of the bicycle, a VN-100 is utilized in **Paper A**, and later, replaced with an MTi-7 in **Paper D**. The main difference between the two sensors is that MTi-7 also offers GNSS data, which can be used in future work for localization of the bicycle, which is a prerequisite for experimental evaluation of the trajectory tracking controllers in **Paper B** and **Paper F**. In **Paper A**, the steering axis is controlled through a DCX32L Maxon motor together with a gear hub and encoder for sensing the position of the handlebar with respect to the mainframe, and the motor is controlled through a Junus motor controller. The Maxon motor and the Junus motor controller are replaced with a Dynamixel XH540-W270-T servo in paper **Paper D** due to the limitation of access to the Maxon motor, and the lightweight design of the Dynamixel. No additional motor controller is needed when using the Dynamixel servo. Finally, in **Paper A**, a National Instruments roboRIO is used as the main processing unit, and it consists of a Field Programmable Gate Array (FPGA) target and a real-time target. The code on both the real-time target and the FPGA is written in LabVIEW. The roboRIO is replaced with a Raspberry Pi 4b in **Paper D**, which simplifies the implementation of the control algorithms and enables the use of the ROS 2 framework on the bicycle. Figure 4.2 shows the instrumented bicycle with the hardware used in **Paper D**.



Hardware	
①	RC receiver
②	Raspberry Pi 4b
③	ESC
④	Hall sensor
⑤	Bafang RM G040.250.DC
⑥	Xsens MTi-7
⑦	Batteries
⑧	Dynamixel XH540-W270-T

Figure 4.2: Instrumented bicycle used in the experiments in **Paper D**. In **Paper A**, the same bicycle was used, but with some differences in the hardware setup.
Original photo: Henrik Falk

Chapter 5

Thesis Contributions

In this thesis, four main contributions are identified. The first contribution (**C1**) is a systematic quantitative comparison of traditional model-based controllers (PID, LQR, fuzzy, FL) and direct data-driven controllers, including the nonlinear design of De Persis et al. [21] and the DeePO algorithm [23], for balancing an autonomous bicycle at constant forward velocity. The second contribution (**C2**) develops and experimentally validates a unified control framework in which a model-based FL controller is combined in series with direct data-driven controllers, enabling safe data generation and improved balance performance on a nonlinear, open-loop unstable system. The third contribution (**C3**) is an in-depth experimental and algorithmic study of DeePO, from its first real-world implementation on the instrumented bicycle to several refinements that address probing, forgetting, and adaptation mechanisms. The fourth and final contribution (**C4**) concerns trajectory tracking, where a model-based PID–MPC cascade controller is compared with a data-driven FL–DeePO–DeePC architecture for tracking trajectories with the autonomous bicycle. A mapping between the research questions (**RQ1–RQ4**), the contributions (**C1–C4**), and **Papers A–F** is given in Table 5.1.

5.1 Research Contribution

C1 – The first contribution is a comparative quantitative analysis of several control methods for balancing an autonomous bicycle using the simple principle of steering in the direction of the fall. In **Paper A**, two PID controllers, using different tuning procedures, an LQR controller, and a fuzzy controller are designed and evaluated. The PID and LQR controllers are based on a linearization of the point-mass model proposed by Getz [33] around the upright equilibrium,

Table 5.1: Mapping between research questions, contributions, and included papers.

	RQ1	RQ2	RQ3	RQ4
C1	  			
C2	  	 	  	
C3		  	 	
C4				 

while the fuzzy controller is inspired by Abdolmalaki [91]. The controllers are first assessed in simulation, using both the linearized model and a nonlinear multibody model designed in SolidWorks and imported into Adams, see Figure 5.1(a). The Integrated Squared Error (ISE) of the lean angle,

$$\text{ISE}_\varphi = \sum_{i=0}^t (\varphi_i - \varphi_{d,i})^2, \quad (5.1)$$

is used as a performance metric, together with disturbance rejection and, in experiments, execution time on the embedded platform. Subsequently, the controllers are implemented on the embedded hardware, and experiments are performed with the autonomous bicycle riding on a narrow bicycle roller. The comparative study shows that all controllers are capable of balancing the bicycle, but that the PID and LQR controllers provide the best trade-off between performance, robustness, and computational cost. Experimental results from these roller experiments are shown in Figure 5.2(a).

Building on these results, **Papers C** and **D** extend the comparison to include nonlinear model-based and direct data-driven control methods. In both papers, a Single-Input Multiple-Output (SIMO) FL controller is proposed, based on a simplified nonlinear bicycle model inspired by Yi et al. [38]. The FL controller regulates the steering rate to balance the bicycle and track a desired lean angle and lean rate. To evaluate these controllers, the detailed SolidWorks model of the autonomous bicycle is imported into Simscape, which provides a high-fidelity multibody simulation environment with performance improvements in terms of simulation time compared to the Adams co-simulation used in **Paper A**. Figure 5.1 illustrates the multibody models used in the comparative studies.

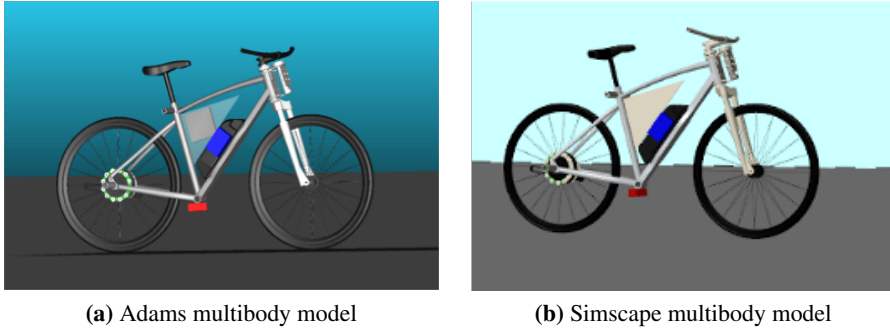
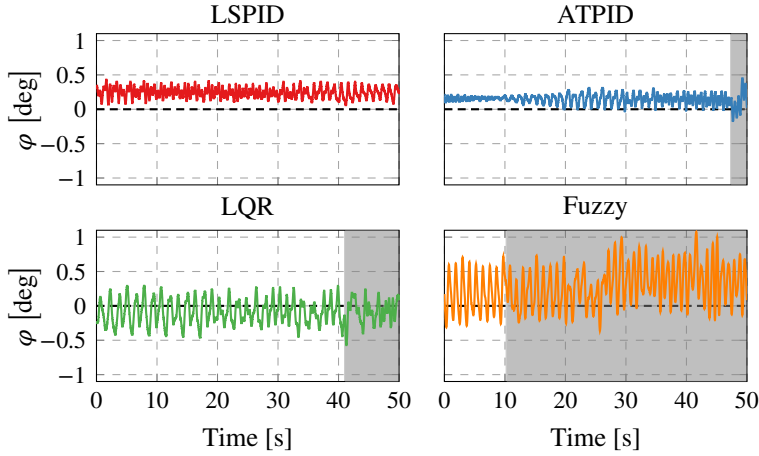


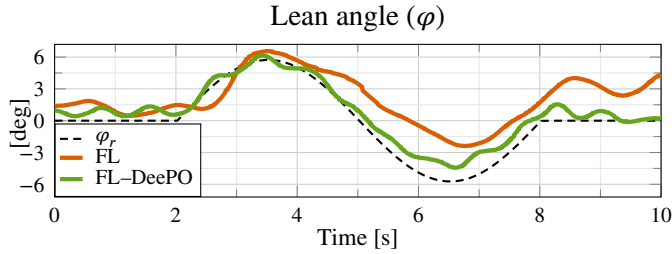
Figure 5.1: The Adams and Simscape multibody bicycle models used in simulation.

In **Paper C**, a direct data-driven state-feedback controller for nonlinear systems, based on the method by De Persis and Tesi [21], is first applied directly to the autonomous bicycle using data collected from the unstable system. This purely data-driven approach is shown to be highly sensitive to the amount of data used for the control design and to several design parameters, making its behavior difficult to predict and its tuning challenging. In particular, obtaining persistently exciting input signals while simultaneously maintaining balance proves non-trivial. These challenges are also highlighted in the experimental evaluation of the direct data-driven controller in the master thesis of Rönmark and Landré [90]. Motivated by these observations, **Paper C** proposes a unified control structure in which the FL controller is combined in series with the data-driven controller; this architecture is further developed in **C2**. **Paper D** further extends this comparative analysis by replacing the static outer-loop controller in **Paper C** with the DeePO algorithm [23], resulting in an adaptive unified FL–DeePO framework. The FL and FL–DeePO controllers are compared in both Simscape simulations and hardware experiments on the autonomous bicycle. Using the ISE of the lean angle and lean rate as performance metrics, the results show that the unified FL–DeePO controller consistently improves tracking performance compared to FL alone, while maintaining actuation constraints and coping with measurement noise and unmodeled dynamics. An illustrative comparison between FL and FL–DeePO in the experiment is shown in Figure 5.2(b) tracking a time-varying reference lean angle.

Taken together, **Papers A, C, and D** constitute **C1**, a systematic quantitative comparison of classical linear controllers (PID, LQR, fuzzy), nonlinear model-based FL, and direct data-driven controllers, both static and adaptive, for balancing an autonomous bicycle through steering actuation, evaluated across linearized models, multibody simulations, and real-world experiments.



(a) Classical model-based controllers on the roller (Paper A), the gray area indicates that the bicycle had gotten out of bounds on the narrow roller, and the experiment was aborted.



(b) FL vs FL-DeePO on flat ground tracking a time-varying reference (Paper D).

Figure 5.2: Experimental results underlying **C1**, **C2**, and **C3**. Note that the two sub-figures correspond to different experimental setups and are not meant as a direct quantitative comparison.

C2 – The second contribution addresses the gap between traditional model-based controllers and direct data-driven control methods for balancing an open-loop unstable, nonlinear system. Based on the insights from **C1** and the limitations of using a purely direct data-driven controller, a unified control framework is proposed in which a model-based FL controller is combined in series with a direct data-driven controller. The key idea is to use FL as an inner loop that stabilizes and partially linearizes the bicycle dynamics, while an outer-loop data-driven controller operates on the FL-stabilized system.

In **Paper C**, this framework is utilized with the static nonlinear state-feedback design of De Persis and Tesi [21]. Data are collected from the autonomous bicycle while the FL controller stabilizes it, and the resulting



Figure 5.3: Experimental setup in **Paper A** where an autonomous bicycle is riding on a bicycle roller.

Photo: Niklas Persson

FL–bicycle combination is treated as a new black-box system for the direct data-driven design. This approach resolves several practical challenges identified in **C1**. It enables the use of persistently exciting steering inputs while balance is maintained, it reduces the impact of model inaccuracies by delegating the fine-tuning of performance to the data-driven layer, and makes the closed-loop behavior significantly more predictable than for the purely data-driven controller. Simulations in Simscape show that the unified FL and data-driven controller yields lower ISE_{φ} and improved robustness to parameter variations compared to both FL alone and the direct data-driven controller applied directly to the physical system.

Paper D extends this unified framework by replacing the static outer-loop data-driven controller with the adaptive DeePO algorithm [23]. In this FL–DeePO architecture, the FL inner loop again stabilizes the system, while DeePO iteratively updates the feedback gains based solely on measured input and output data from the system. The combined controller is evaluated both in high-fidelity Simscape simulations and in experiments on the instrumented bicycle. The results demonstrate that the FL–DeePO framework can improve tracking of lean angle and lean rate compared to FL alone, while respecting actuator limits and coping with measurement noise, unmodeled dynamics, and time-varying parameters. This provides one of the first experimental demonstrations

of a direct data-driven controller for a nonlinear, non-holonomic, underactuated system, enhancing the performance of a model-based controller.

Finally, **Paper F** generalizes the SIMO unified framework to a MIMO setting, in which both the steering rate and the forward velocity are controlled. The MIMO FL controller is designed as the inner loop, and the unified FL-DeePO structure is extended to this higher-dimensional case. The study shows that the proposed architecture scales beyond single-input balance control and can handle simultaneous regulation of balance and speed, while preserving the safety and performance benefits observed in the SIMO case.

Taken together, **Papers C, D, and F** constitute **C2**, the development and experimental validation of a unified FL and data-driven control framework that enables safe generation of persistently exciting data in an unstable system and leverages model-based design for stabilization and data-driven methods for performance enhancement.

C3 – The third contribution is an investigation of the DeePO algorithm from both an experimental and an algorithmic perspective. In **Paper D**, we report, to the best of our knowledge, the first real-world implementation of DeePO, embedded on a Raspberry Pi and deployed on the autonomous bicycle within the unified FL-DeePO framework. The controller balances the bicycle on flat ground while tracking desired lean angle and lean rate signals, and the experiments quantify the performance gain over FL alone in terms of the ISE of the lean states and actuator effort. To cope with nonlinear and time-varying dynamics, a forgetting factor is incorporated in the DeePO update, and its influence on convergence and robustness is analysed. Moreover, the update period of the DeePO policy is systematically varied, showing that infrequent gain updates can preserve or even improve performance compared to updating at every sampling instant, which is important for computational load and implementation robustness. The experiments are conducted in an indoor warehouse with a flat concrete floor, see Figure 5.4.

In **Paper E**, we move to a controlled LTI setting to analyse fundamental limitations of DeePO related to probing noise and persistency of excitation. It is shown that the probing noise conventionally added to ensure rich data can induce undesirable state perturbations. To address this, a Perturbation-Free DeePO (PFDeePO) variant is proposed, which combines an update-freezing mechanism when the state is close to equilibrium with a small input-scaling excitation applied to the control signal. Simulations demonstrate that PFDeePO preserves the rank condition on the data while eliminating steady-state oscillations and maintaining or improving closed-loop performance. This study clarifies how probing strategies and adaptation rules affect the behaviour of DeePO



Figure 5.4: The instrumented bicycle controlled through a unified control framework consisting of FL and DeePO.

Photo: Henrik Falk

and informs the design choices used in the bicycle experiments.

Finally, in **Paper F**, DeePO is embedded as the adaptive inner-loop component of a cascaded FL–DeePO–DeePC architecture for simultaneous balance and trajectory tracking of the multibody bicycle model in Simscape. From the perspective of **C3**, the focus is on how DeePO behaves in this higher-dimensional MIMO setting, where both steering rate and forward acceleration are controlled, and how its adaptation interacts with the outer-loop data-enabled predictive controller. Practical modifications such as normalization of the DeePO data and a slightly revised formulation of the forgetting factor compared to **Paper D** are introduced and evaluated, and their impact on closed-loop performance is analysed. These results show that DeePO can be deployed as an adaptive enhancement of a model-based FL controller even when embedded in a more complex cascaded architecture.

In conclusion, **Papers D, E, and F** constitute **C3**, a comprehensive experimental and simulation-based assessment of DeePO, from its first real-world implementation and algorithmic refinements to its role as an adaptive component in unified model-based and direct data-driven control architectures for autonomous bicycle control.

C4 – The fourth contribution concerns trajectory tracking for an autonomous bicycle. In **Paper B**, a model-based cascaded control framework is proposed in which a robust PID controller in the inner loop balances the bicycle by regulating the steering rate, while an MPC in the outer loop

generates a desired lean angle and tracks a given trajectory. The Yi et al. [38] model is used for control design, and the full scheme is evaluated in high-fidelity co-simulations between Adams and Simulink, including measurement noise on the lean angle and disturbances on the steering input. The autonomous bicycle is shown to successfully follow two challenging reference trajectories, including a go-kart track, over a range of velocities. The path tracking performance is quantified using the mean squared error and the Hausdorff distance [92] between the bicycle trajectory and the reference, demonstrating that tight curves can be negotiated at comparatively high speeds when a carefully tuned PID–MPC cascade is employed.

Building on this model-based benchmark, **Paper F** investigates trajectory tracking in a data-driven setting. A cascaded FL–DeePO–DeePC architecture is proposed in which a MIMO FL controller stabilizes the lean angle and forward velocity, DeePO adaptively refines the FL feedback gains using closed-loop data from the bicycle, and DeePC in the outer loop computes reference trajectories for the FL–DeePO inner loop based solely on noisy input–output data. To enhance adaptability to nonlinear and time-varying dynamics, a forgetting-factor scheme is incorporated into DeePO, and regularization is introduced in DeePC together with a terminal constraint to improve numerical conditioning and closed-loop stability. The FL–DeePO–DeePC cascade control approach is validated in high-fidelity Simscape simulations, where its trajectory tracking performance is compared to the PID–MPC framework from **Paper B**. While the model-based controller achieves a smaller Hausdorff distance to the reference path, as seen in Figure 5.5, the data-driven scheme remains stable and feasible across a range of horizon and tuning parameters. It requires no explicit dynamic model of the bicycle.

Taken together, **Papers B** and **F** constitute **C4**: a comprehensive study of trajectory tracking for an autonomous bicycle, contrasting a model-based PID–MPC architecture with a fully data-driven FL–DeePO–DeePC cascade. This contribution quantifies the trade-offs between model-based and direct data-driven approaches in terms of tracking accuracy, robustness, and modelling effort, and demonstrates that full trajectory tracking can be achieved using only measured input–output data when an appropriate cascaded structure is adopted.

5.2 Overview of included papers

In this section, the six papers (**A–F**) are summarized, and how they relate to the contributions are given. Moreover, my personal contribution is highlighted for each paper.

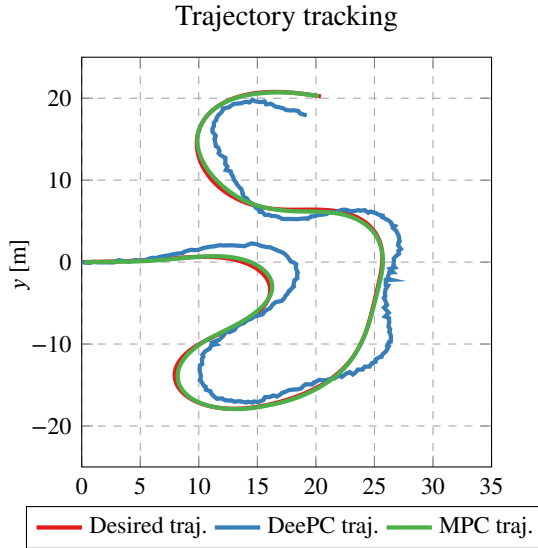


Figure 5.5: Results of FL-DeePC-DeePO (blue) and PID-MPC (green) tracking a reference trajectory (red).

5.2.1 Paper A

Title: A Comparative Analysis and Design of Controllers for Autonomous Bicycles

Authors: Niklas Persson, Tom Andersson, Anas Fattouh, Martin C. Ekström, Alessandro V. Papadopoulos.

Status: Published in European Control Conference (ECC), 2021

Abstract: In this paper, we develop and compare the performance of different controllers for balancing an autonomous bicycle. The evaluation is carried out both in simulation, using two different models, and experimentally, on a bicycle instrumented with only lightweight components, and leaving the bicycle structure practically unchanged. Two PID controllers, a Linear Quadratic Regulator (LQR), and a fuzzy controller are developed and evaluated in simulations where both noise and disturbances are induced in the models. The simulation shows that the LQR controller has the best performance in the simulation scenarios. Experimental results, on the other hand, show that the PID controllers provide better performance when balancing the instrumented bicycle.

Paper contributions This paper relates to contribution C1.

Personal Contributions I was the main author of the paper, contributing

to the simulations and writing most of the paper. The experimental results and the idea were joint work between Tom Andersson and me. The other co-authors contributed with feedback and improved the manuscript in collaboration with me.

5.2.2 Paper B

Title: Trajectory tracking and stabilisation of a riderless bicycle

Authors: Niklas Persson, Martin C. Ekström, Mikael Ekström, Alessandro V. Papadopoulos.

Status: Published in International Conference on Intelligent Transportation Systems (ITSC), 2021

Abstract: Trajectory tracking for an autonomous bicycle is considered in this paper. The trajectory tracking controller is designed using a Model Predictive Controller with constraints on the lean, steer, and heading angle as well as the position coordinates of the bicycle. The output from the trajectory tracking controller is the desired lean angle and forward velocity. Furthermore, a PID controller is designed to follow the desired lean angle, while maintaining balance, by actuation of the handlebar. The proposed control strategy is evaluated in numerous simulations where a realistic nonlinear model of the bicycle is traversing a go-kart track and a short track with narrow curves. The Hausdorff distance and Mean Squared Error are considered as measurements of the performance. The results show that the bicycle successfully can track desired trajectories at varying velocities.

Paper contributions This paper relates to contribution C4.

Personal Contributions I was the main author of the paper, contributing to the idea, the implementation, and the simulation setup. I was the main contributor to the manuscript, with valuable feedback and technical discussion with the other authors.

5.2.3 Paper C

Title: Direct Data-Driven Control Design for Autonomous Bicycles

Authors: Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.

Status: Published in International Conference on Automation Science and Engineering (CASE), 2024

Abstract: In this work, we address the problem of balancing an autonomous

bicycle using direct data-driven control. Firstly, we demonstrate that a direct implementation of data-driven approaches may not guarantee reliable performance, and is highly dependent on how the parameters are selected. To address this issue, we make the reasonable assumption that we have access to some inaccurate information about the system. We use this inaccurate information to design a feedback linearization, based on a simplified point mass model of the bicycle, which does not accurately represent the dynamics of the system. Next, we suggest an inner and outer-loop control strategy. In the inner loop, we implement the aforementioned feedback linearization controller. Subsequently, in the outer loop, we consider the combination of the autonomous bicycle and the feedback controller as a black box, and we design a direct data-driven controller from acquired data. We use a SolidWorks model of a real autonomous bicycle to evaluate the performance of our proposed control approach and to compare it with the direct data-driven controller design derived from acquired data of the bicycle without feedback linearization. The results show that our proposed strategy significantly improves the performance and makes the data-driven control approach more reliable across a broader range of parameter choices compared to a data-driven controller designed based on data from the system without feedback linearization. Finally, we show that introducing an additional integral-like state further enhances the system's performance.

Paper contributions This paper relates to contribution **C1** and **C2**.

Personal Contributions I was the main drive of the implementation, simulation environment and contributed to the idea together with the other authors. The manuscript was written in joint work with Mojtaba, with valuable feedback and technical discussions with Alessandro.

5.2.4 Paper D

Title: An Adaptive Data-Enabled Policy Optimization Approach for Autonomous Bicycle Control

Authors: Niklas Persson, Feiran Zhao, Mojtaba Kaheni, Florian Dörfler, Alessandro V. Papadopoulos.

Status: Submitted to Transaction on Control System Technology, Feb. 2025

Abstract: This paper presents a unified control framework that integrates a Feedback Linearization (FL) controller in the inner loop with an adaptive Data-Enabled Policy Optimization (DeePO) controller in the outer loop to balance an autonomous bicycle. While the FL controller stabilizes and partially linearizes the inherently unstable and nonlinear system, its

performance is compromised by unmodeled dynamics and time-varying characteristics. To overcome these limitations, the DeePO controller is introduced to enhance adaptability and robustness. The initial control policy of DeePO is obtained from a finite set of offline, persistently exciting input and state data. To improve stability and compensate for system nonlinearities and disturbances, a robustness-promoting regularizer refines the initial policy, while the adaptive section of the DeePO framework is enhanced with a forgetting factor to improve adaptation to time-varying dynamics. The proposed DeePO+FL approach is evaluated through simulations and real-world experiments on an instrumented autonomous bicycle. Results demonstrate its superiority over the FL-only approach, achieving more precise tracking of the reference lean angle and lean rate.

Paper contributions This paper relates to contribution **C1**, **C2** and **C3**.

Personal Contributions I implemented the algorithm in simulations and on the embedded system on the bicycle, and conducted all experiments and simulations. Feiran provided the idea of DeePO and provided technical details of the algorithm. The manuscript was a joint work between the authors, where Mojtaba and I were the main drivers.

5.2.5 Paper E

Title: A Modified Adaptive Data-Enabled Policy Optimization Control to Resolve State Perturbations

Authors: Mojtaba Kaheni, Niklas Persson, Vittorio De Iuliis, Costanzo Manes, Alessandro V. Papadopoulos.

Status: Accepted to Conference on Decision and Control (CDC), 2025

Abstract: This paper proposes modifications to the data-enabled policy optimization (DeePO) algorithm to mitigate state perturbations. DeePO is an adaptive, data-driven approach designed to iteratively compute a feedback gain equivalent to the certainty-equivalence LQR gain. Like other data-driven approaches based on Willems' fundamental lemma, DeePO requires persistently exciting input signals. However, linear state-feedback gains from LQR designs cannot inherently produce such inputs. To address this, probing noise is conventionally added to the control signal to ensure persistent excitation. However, the added noise may induce undesirable state perturbations. We first identify two key issues that jeopardize the desired performance of DeePO when probing noise is not added: the convergence of states to the equilibrium point, and the convergence of the controller to its optimal value. To address these challenges without relying on probing noise,

we propose Perturbation-Free DeePO (PFDeePO) built on two fundamental principles. First, the algorithm pauses the control gain updating in DeePO process when system states are near the equilibrium point. Second, it applies a multiplicative noise, scaled by a mean value of 1 as a gain for the control signal, when the controller converges. This approach minimizes the impact of noise as the system approaches equilibrium while preserving stability. We demonstrate the effectiveness of PFDeePO through simulations, showcasing its ability to eliminate state perturbations while maintaining system performance and stability.

Paper contributions This paper relates to contribution C3.

Personal Contributions I contributed with the simulation and assisted in writing the manuscript.

5.2.6 Paper F

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

Authors: Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.

Status: Submitted to European Control Conference (ECC), 2026

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.

Paper contributions This paper relates to contribution **C2**, **C3**, and **C4**.

Personal Contributions I was the main driver of the implementation, idea, simulation, and the manuscript, with valuable feedback from the other authors.

Chapter 6

Conclusions and Future Work

In this thesis, several traditional model-based controllers and direct data-driven control methods are used and combined to balance an autonomous bicycle using mainly steering actuation. The bicycle is a nonlinear, non-holonomic, underactuated, and in general open-loop unstable system. Moreover, direct data-driven methods assume linear time-invariant dynamics and rely on persistently exciting input signals. Within this setting, four research questions were formulated, concerning the comparison between traditional and direct data-driven controllers (RQ1), the generation of persistently exciting data in unstable systems (RQ2), a unified model-based and direct data-driven control design (RQ3), and the comparison between model-based and data-driven trajectory tracking (RQ4).

The first research question (RQ1) is addressed by **C1** through **Papers A, C, and D**. In **Paper A**, PID, LQR, and fuzzy controllers designed from a linearized point-mass model were evaluated in both high-fidelity simulations and roller experiments. The results showed that all considered controllers can stabilize the bicycle, but that PID and LQR offer the best trade-off between balance time, disturbance rejection, and implementation complexity. **Paper C** and **Paper D** extend this comparison to FL and direct data-driven controllers. A purely data-driven state-feedback design based on the method by De Persis and Tesi exhibited strong sensitivity to data and tuning, whereas when unified with FL or replaced by DeePO in an FL–DeePO architecture, the data-driven components systematically improved performance over FL alone in both simulations and experiments. In this way, **C1** shows that while classical model-based controllers remain strong baselines for balance, data-driven controllers can enhance performance when deployed in a suitable structure.

The second and third research questions (RQ2 and RQ3) focused on how to achieve persistent excitation for direct data-driven controllers in an open-loop

unstable system without compromising stability and how model-based and direct data-driven design can be unified to improve stabilization at low speeds. These questions are answered jointly by **C2** and **C3**, mainly through **Papers C, D, E, and F**. The key idea in **C2** is to use FL as an inner loop that stabilizes and partially linearizes the bicycle, and then treat the combination of bicycle and FL as a new black-box system for the outer-loop data-driven controller. This unified framework allows persistently exciting steering inputs to be applied while balance is maintained and transfers the burden of detailed performance tuning from the model-based design to the data-driven layer. **Paper C** demonstrated in simulation that this FL–data-driven scheme yields lower lean-angle ISE and more predictable behaviour than both FL alone and a purely data-driven controller. **Paper D** showed experimentally that an FL–DeePO controller further improves tracking of lean angle and lean rate on flat ground compared to FL alone, and **Paper F** generalized the unified framework to a MIMO setting, simultaneously regulating steering rate and forward velocity. From an algorithmic perspective, **Paper E** complemented these results by introducing PFDeePO in an LTI setting, which clarifies how probing noise, forgetting factors, and update-freezing mechanisms can be designed to maintain persistent excitation while avoiding excessive state perturbations.

The fourth research question (RQ4) addressed trajectory tracking and asked how direct data-driven trajectory tracking controllers compare to model-based controllers in terms of tracking a reference trajectory in simulation. This is the focus of **C4**, based on **Papers B and F**. In **Paper B**, a cascaded PID–MPC architecture was proposed, and **Paper F** introduced a fully data-driven alternative in which a MIMO FL controller and DeePO stabilize and regulate the bicycle, while DeePC in the outer loop computes control actions directly from measured input–output data. Although the PID–MPC controller achieved a shorter Hausdorff distance, the data-driven cascade remained stable and feasible over a range of tuning parameters and avoided the modelling effort required for MPC. Hence, **C4** shows that full trajectory tracking of an autonomous bicycle can be realized using only measured data, at the cost of a reduction in tracking accuracy compared to a carefully tuned model-based controller.

6.1 Future Work

The PID–MPC and FL–DeePO–DeePC frameworks have been evaluated in high-fidelity simulations; however, they are yet to be evaluated on the instrumented bicycle. In future work, the GNSS of the MTi-7 sensor, together with wheel odometry, can be used to estimate the pose and enable experimental trajectory tracking. A second direction concerns the further integration of

the PFDeePO ideas into the bicycle experiments. The modifications proposed in **Paper E** have so far been studied only in LTI simulations. Implementing PFDeePO on the bicycle and combining it with the unified FL–DeePO framework would be interesting to investigate in the future.

Bibliography

- [1] W. J. M. Rankine, "On the dynamical principles of the motion of velocipedes," *The Engineer*, vol. 28, pp. 79, 129, 153, 175, 1869.
- [2] 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.
- [3] 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.
- [4] 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.
- [5] J. He, M. Zhao, and S. Stasinopoulos, "Constant-velocity steering control design for unmanned bicycles," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 428–433, 2015.
- [6] B. T. Thanh and M. Parnichkun, "Balancing control of bicyrobo by particle swarm optimization-based structure-specified mixed H_2/H_∞ control," *International Journal of Advanced Robotic Systems*, vol. 5, no. 4, p. 39, 2008.
- [7] H. Yetkin, S. Kalouche, M. Vernier, G. Colvin, K. Redmill, and U. Ozguner, "Gyroscopic stabilization of an unmanned bicycle," in *American Control Conference*, pp. 4549–4554, 2014.
- [8] L. Keo and Y. Masaki, "Trajectory control for an autonomous bicycle with balancer," in *2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 676–681, IEEE, 2008.
- [9] N. C. Sanchez, L. A. Pastor, and K. Larson, "Autonomous bicycles: A new approach to bicycle-sharing systems," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, 2020.
- [10] M. R. García, D. A. Mántaras, J. C. Álvarez, and D. Blanco F, "Stabilizing an urban semi-autonomous bicycle," *IEEE Access*, vol. 6, pp. 5236–5246, 2018.

- [11] D. Rodriguez-Rosa, I. Payo-Gutierrez, F. J. Castillo-Garcia, A. Gonzalez-Rodriguez, and S. Perez-Juarez, "Improving energy efficiency of an autonomous bicycle with adaptive controller design," *Sustainability*, vol. 9, no. 5, p. 866, 2017.
- [12] B. Kutela, S. Das, and B. Dadashova, "Mining patterns of autonomous vehicle crashes involving vulnerable road users to understand the associated factors," *Accident Analysis & Prevention*, p. 106473, 2021.
- [13] L. Cui, S. Wang, Z. Zhang, and Z.-P. Jiang, "Asymptotic trajectory tracking of autonomous bicycles via backstepping and optimal control," *IEEE Control Systems Letters*, vol. 6, pp. 1292–1297, 2022.
- [14] D. Kondor, X. Zhang, M. Meghjani, P. Santi, J. Zhao, and C. Ratti, "Estimating the potential for shared autonomous scooters," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, pp. 4651–4662, 2022.
- [15] 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.
- [16] K. J. Åström, R. E. Klein, and A. Lennartsson, "Bicycle dynamics and control: adapted bicycles for education and research," *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, 2005.
- [17] J. D. Kooijman, A. L. Schwab, and J. P. Meijaard, "Experimental validation of a model of an uncontrolled bicycle," *Multibody System Dynamics*, vol. 19, pp. 115–132, 2008.
- [18] T.-K. Dao and C.-K. Chen, "A study of bicycle dynamics via system identification approaches," *Journal of the Chinese Institute of Engineers*, vol. 35, no. 7, pp. 853–868, 2012.
- [19] V. Breschi, A. Chiuso, and S. Formentin, "Data-driven predictive control in a stochastic setting: a unified framework," *Automatica*, vol. 152, p. 110961, 2023.
- [20] 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.
- [21] C. De Persis and P. Tesi, "Formulas for data-driven control: Stabilization, optimality, and robustness," *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 909–924, 2020.

- [22] J. Coulson, J. Lygeros, and F. Dörfler, “Data-enabled predictive control: In the shallows of the DeePC,” in *18th European Control Conference (ECC)*, pp. 307–312, 2019.
- [23] 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)*, pp. 6160–6165, 2023.
- [24] F. Zhao, R. Leng, L. Huang, H. Xin, K. You, and F. Dörfler, “Direct adaptive control of grid-connected power converters via output-feedback data-enabled policy optimization,” *arXiv preprint arXiv:2411.03909*, 2024.
- [25] L. Huang, J. Zhen, J. Lygeros, and F. Dörfler, “Quadratic regularization of data-enabled predictive control: Theory and application to power converter experiments,” *IFAC-PapersOnLine*, vol. 54, no. 7, pp. 192–197, 2021.
- [26] 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.
- [27] 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.
- [28] H. Wang, K. Zhang, K. Lee, Y. Mei, K. Zhu, V. Srivastava, J. Sheng, and Z. Li, “Mechanical design and data-enabled predictive control of a planar soft robot,” *IEEE Robotics and Automation Letters*, vol. 9, no. 9, pp. 7923–7930, 2024.
- [29] F. Whipple, “Stability of the motion of a bicycle,” *Quarterly Journal of Pure and Applied Mathematics*, vol. 30, pp. 312–348, 1899.
- [30] E. Carvallo, “Theorie du mouvement du monocycle et de la bicyclette,” *Journal de L'Ecole Polytechnique*, vol. 5, pp. 119–188, 1900.
- [31] M. Plöchl, J. Edelmann, B. Angrosch, and C. Ott, “On the wobble mode of a bicycle,” *Vehicle system dynamics*, vol. 50, no. 3, pp. 415–429, 2012.
- [32] J. K. Moore, M. Hubbard, A. L. Schwab, and J. D. Kooijman, “Accurate measurement of bicycle parameters,” in *Bicycle and Motorcycle Dynamics 2010 Symposium on the Dynamics and Control of Single Track Vehicles*, 2010.

- [33] 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, pp. 148–151 vol.1, 1994.
- [34] N. H. Getz and J. E. Marsden, "Control for an autonomous bicycle," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1397–1402, 1995.
- [35] D. J. Limebeer and R. S. Sharp, "Bicycles, motorcycles, and models," *IEEE Control Systems Magazine*, vol. 26, no. 5, pp. 34–61, 2006.
- [36] D. E. Meehan and A. L. Ruina, "Linear and nonlinear controllers of an autonomous bicycle have almost identical basins of attraction in a non-linear simulation," in *Bicycle and Motorcycle Dynamics 2019*, Symposium on the Dynamics and Control of Single Track Vehicles, 2020.
- [37] J. Yi, D. Song, A. Levandowski, and S. Jayasuriya, "Trajectory tracking and balance stabilization control of autonomous motorcycles," in *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2583–2589, IEEE, 2006.
- [38] J. Yi, Y. Zhang, and D. Song, "Autonomous motorcycles for agile maneuvers, part i: Dynamic modeling," in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, pp. 4613–4618, 2009.
- [39] J. D. Kooijman, J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, "A bicycle can be self-stable without gyroscopic or caster effects," *Science*, vol. 332, no. 6027, pp. 339–342, 2011.
- [40] Y. Zhang, P. Wang, J. Yi, D. Song, and T. Liu, "Stationary balance control of a bikebot," in *International Conference on Robotics and Automation (ICRA)*, pp. 6706–6711, 2014.
- [41] S. Miah, E. Milonidis, I. Kaparias, and N. Karcianas, "An innovative multi-sensor fusion algorithm to enhance positioning accuracy of an instrumented bicycle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1145–1153, 2020.
- [42] J. D. Kooijman, A. L. Schwab, and J. K. Moore, "Some observations on human control of a bicycle," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 4, pp. 2021–2028, 2009.

- [43] E. X. Wang, J. Zou, G. Xue, Y. Liu, Y. Li, and Q. Fan, “Development of efficient nonlinear benchmark bicycle dynamics for control applications,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 2236–2246, 2015.
- [44] A. L. Schwab and J. P. Meijaard, “A review on bicycle dynamics and rider control,” *Vehicle System Dynamics*, vol. 51, no. 7, pp. 1059–1090, 2013.
- [45] C. A. Manrique-Escobar, C. M. Pappalardo, and D. Guida, “On the analytical and computational methodologies for modelling two-wheeled vehicles within the multibody dynamics framework: A systematic literature review,” *Journal of Applied and Computational Mechanics*, vol. 8, no. 1, pp. 153–181, 2021.
- [46] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, “Reinforcement learning for control: Performance, stability, and deep approximators,” *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.
- [47] X. He and C. Lv, “Robotic control in adversarial and sparse reward environments: A robust goal-conditioned reinforcement learning approach,” *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 1, pp. 244–253, 2024.
- [48] M. Han, L. Zhang, J. Wang, and W. Pan, “Actor-critic reinforcement learning for control with stability guarantee,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6217–6224, 2020.
- [49] X. Zhu, X. Zheng, Y. Deng, Z. Chen, B. Liang, and Y. Liu, “Deep reinforcement learning-based control of bicycle robots on rough terrain,” in *9th International Conference on Control, Automation and Robotics (IC-CAR)*, pp. 103–108, 2023.
- [50] C. Tang, B. Abbatematteo, J. Hu, R. Chandra, R. Martín-Martín, and P. Stone, “Deep reinforcement learning for robotics: A survey of real-world successes,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, pp. 28694–28698, Apr. 2025.
- [51] 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.
- [52] F. Dörfler, P. Tesi, and C. De Persis, “On the role of regularization in direct data-driven LQR control,” in *IEEE 61st Conference on Decision and Control (CDC)*, pp. 1091–1098, 2022.

- [53] J. Berberich, A. Koch, C. W. Scherer, and F. Allgöwer, “Robust data-driven state-feedback design,” in *American Control Conference (ACC)*, pp. 1532–1538, IEEE, 2020.
- [54] P. Mattsson and T. B. Schön, “On the regularization in deepc,” *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 625–631, 2023.
- [55] M. Mejari and V. Breschi, “Direct data-driven design of LPV controllers and polytopic invariant sets with cross-covariance noise bounds,” *IEEE Control Systems Letters*, 2024.
- [56] R. Strässer, J. Berberich, M. Schaller, K. Worthmann, and F. Allgöwer, “Koopman-based control of nonlinear systems with closed-loop guarantees,” *arXiv preprint arXiv:2411.10359*, 2024.
- [57] J. K. Moore, J. D. Kooijman, A. L. Schwab, and M. Hubbard, “Rider motion identification during normal bicycling by means of principal component analysis,” *Multibody System Dynamics*, vol. 25, no. 2, pp. 225–244, 2011.
- [58] G. Dialynas, C. Christoforidis, R. Happee, and A. Schwab, “Rider control identification in cycling taking into account steering torque feedback and sensory delays,” *Vehicle System Dynamics*, vol. 61, no. 1, pp. 200–224, 2023.
- [59] J. D. Kooijman and A. L. Schwab, “A review on bicycle and motorcycle rider control with a perspective on handling qualities,” *Vehicle System Dynamics*, vol. 51, no. 11, pp. 1722–1764, 2013.
- [60] T.-J. Yeh, T.-C. Lin, and A. C.-B. Chen, “Robust balancing and trajectory control of a self-driving bicycle,” *IEEE Transactions on Control Systems Technology*, vol. 32, no. 6, pp. 2410–2417, 2024.
- [61] M.-H. Hsieh, Y.-T. Chen, C.-H. Chi, and J.-J. Chou, “Fuzzy sliding mode control of a riderless bicycle with a gyroscopic balancer,” in *2014 IEEE International Symposium on Robotic and Sensors Environments (ROSE) Proceedings*, pp. 13–18, IEEE, 2014.
- [62] P. Wang, Y. Gong, J. Yi, and T. Liu, “An integrated stationary/moving balance control of an autonomous bikebot,” in *2019 American Control Conference (ACC)*, pp. 3273–3278, IEEE, 2019.
- [63] M. Ekström and T. Eriksson, “Student report - autobike 2018.” <http://www.es.mdh.se/publications/5454->, February 2019. [Online; accessed 25-Jan-2023].

- [64] Y. Tanaka and T. Murakami, "Self sustaining bicycle robot with steering controller," in *The 8th IEEE International Workshop on Advanced Motion Control (AMC)*, pp. 193–197, IEEE, 2004.
- [65] D. J. MacKay, *Sustainable Energy-without the hot air*. UIT cambridge, 2008.
- [66] J. Slotine and W. Li, *Applied Nonlinear Control*. Prentice-Hall, 1991.
- [67] K. He, C. Dong, A. Yan, Q. Zheng, B. Liang, and Q. Wang, "Composite deep learning control for autonomous bicycles by using deep deterministic policy gradient," in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, pp. 2766–2773, IEEE, 2020.
- [68] T.-K. Dao and C.-K. Chen, "Path-tracking control of a riderless bicycle via road preview and speed adaptation," *Asian Journal of Control*, vol. 15, no. 4, pp. 1036–1050, 2013.
- [69] J. Yuan, F. Sun, and Y. Huang, "Optimal design of trajectory parameters and position tracking with balance for riderless bicycle," *Optimal Control Applications and Methods*, vol. 37, no. 1, pp. 72–89, 2016.
- [70] M. Zhao, S. Stasinopoulos, and Y. Yu, "Obstacle detection and avoidance for autonomous bicycles," *IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1310–1315, 2017.
- [71] 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.
- [72] L. P. Tuyen and T. Chung, "Controlling bicycle using deep deterministic policy gradient algorithm," in *International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pp. 413–417, 2017.
- [73] S. Weyrer, P. Manzl, A. Schwab, and J. Gerstmayr, "Path following and stabilisation of a bicycle model using a reinforcement learning approach," *arXiv preprint arXiv:2407.17156*, 2024.
- [74] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.
- [75] L. Guo, H. Lin, J. Jiang, Y. Song, and D. Gan, "Combined control algorithm based on synchronous reinforcement learning for a self-balancing bicycle robot," *ISA Transactions*, vol. 145, pp. 479–492, 2024.

- [76] S. Sha, Y. Liu, B. Huo, Z. Chen, and X. Zhao, “Hierarchical adaptive control framework for autonomous bicycles: Integrating residual decisions and dynamic optimization,” *IEEE Transactions on Automation Science and Engineering*, 2025.
- [77] F. Celi, G. Baggio, and F. Pasqualetti, “Closed-form and robust expressions for data-driven lq control,” *Annual Reviews in Control*, vol. 56, p. 100916, 2023.
- [78] 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.
- [79] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, “Review on model predictive control: An engineering perspective,” *The International Journal of Advanced Manufacturing Technology*, vol. 117, no. 5, pp. 1327–1349, 2021.
- [80] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, “Data-driven model predictive control with stability and robustness guarantees,” *IEEE Transactions on Automatic Control*, vol. 66, no. 4, pp. 1702–1717, 2020.
- [81] T. B. Hamdan, P. Coirault, G. Mercère, and T. Dairay, “Data enabled predictive control of lpv systems,” *Control Engineering Practice*, vol. 149, p. 105969, 2024.
- [82] C. Ren, H. Jiang, C. Li, W. Sun, and S. Ma, “Koopman-operator-based robust data-driven control for wheeled mobile robots,” *IEEE/ASME Transactions on Mechatronics*, vol. 28, no. 1, pp. 461–472, 2023.
- [83] R. Strässer, J. Berberich, and F. Allgöwer, “Robust data-driven control for nonlinear systems using the koopman operator,” in *IFAC World Congress*, (Yokohama, Japan), July 2023.
- [84] M. Lazar, “Neural data-enabled predictive control,” *IFAC-PapersOnLine*, vol. 58, no. 15, pp. 91–96, 2024.
- [85] J. Näf, K. Moffat, J. Eising, and F. Dörfler, “Choose wisely: Data-driven predictive control for nonlinear systems using online data selection,” *arXiv preprint arXiv:2503.18845*, 2025.
- [86] K. Zhang, Y. Zheng, C. Shang, and Z. Li, “Dimension reduction for efficient data-enabled predictive control,” *IEEE Control Systems Letters*, vol. 7, pp. 3277–3282, 2023.

- [87] X. Zhang, K. Zhang, Z. Li, and X. Yin, “Deep deepc: Data-enabled predictive control with low or no online optimization using deep learning,” *AIChE Journal*, vol. 71, no. 3, p. e18644, 2025.
- [88] J. Berberich and F. Allgöwer, “An overview of systems-theoretic guarantees in data-driven model predictive control,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 8, no. 1, pp. 77–100, 2025.
- [89] T. Andersson, N. Persson, A. Fattouh, and M. C. Ekström, “A loop shaping method for stabilising a riderless bicycle,” in *2019 European Conference on Mobile Robots (ECMR)*, pp. 1–6, 2019.
- [90] A. Rönmark and D. Landré, “Implementation of direct data-driven control on an instrumented bicycle,” 2024.
- [91] R. Y. Abdolmalaki, “Implementation of fuzzy inference engine for equilibrium and roll-angle tracking of riderless bicycle,” *arXiv e-prints*, 2017.
- [92] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*, vol. 317. Springer Science & Business Media, 2009.

II

Included Papers

Chapter 7

Paper A

A Comparative Analysis and Design of Controllers for Autonomous Bicycles

Niklas Persson, Tom Andersson, Anas Fattouh, Martin C. Ekström, Alessandro V. Papadopoulos.

In European Control Conference (ECC), 2021

Abstract

In this paper, we develop and compare the performance of different controllers for balancing an autonomous bicycle. The evaluation is carried out both in simulation, using two different models, and experimentally, on a bicycle instrumented with only lightweight components, and leaving the bicycle structure practically unchanged. Two PID controllers, a Linear Quadratic Regulator (LQR), and a fuzzy controller are developed and evaluated in simulations where both noise and disturbances are induced in the models. The simulation shows that the LQR controller has the best performance in the simulation scenarios. Experimental results, on the other hand, show that the PID controllers provide better performance when balancing the instrumented bicycle.

7.1 Introduction

Modern vehicles, equipped with sensors for mapping the surrounding environment and to detect and classify other road users struggle when it comes to detection of bicycles [1]. When the vehicle's autonomous emergency braking system is tested by the car safety performance assessment program, EuroNCAP, a bicycle with a dummy on top is placed on a moving platform¹. The platform then moves in a straight line in front or beside the vehicle being tested. A riderless bicycle which is design to have a minimal impact on its resemblance and can manoeuvre realistically would improve the testing environment for autonomous vehicles.

The control of a bicycle motion is an interesting research problem, that has been investigated for decades [2], with several different variants [3]. Its configuration with two inline wheels makes the bicycle a statically unstable system, making the control of the bicycle dynamics a very interesting control problem at low speeds [3]. A cyclist uses a combination of regulation on the forward speed, the steering angle, and the lean angle to balance the bicycle. A similar approach can also be used to control a driverless bicycle [3, 4]. However, direct control of the lean angle requires a flywheel, an inverted pendulum or something similar to be mounted on a bicycle and will, therefore, alter the appearance of the bicycle to a large degree, as well its usability.

Different control approaches have been proposed in the literature to design an autonomous bicycle, ranging from model-free to model-based [5, 2], from control-theoretic to machine learning [6, 7]. Several such approaches are evaluated in simulation, and the results strictly depend on the level of accuracy of the bicycle model. Furthermore, aspects like the computational complexity, the execution time, and implementation issues are hardly discussed.

As a first step to develop a riderless bicycle which can be used for testing of autonomous vehicles safety systems, several control strategies which are commonly used in literature for controlling bicycles are investigated and compared in this paper. The controllers are evaluated both in simulation, and on an instrumented bicycle running on a bicycle roller, to understand what is the more effective approach. Their performance is assessed both in terms of the ability of balancing the bicycle, and in terms of their execution time on the embedded control platform. In particular, three main control strategies are designed and compared: (i) a Proportional Integral Derivative (PID) controller, (ii) a Linear Quadratic Regulator (LQR), and (iii) a fuzzy controller.

¹<https://www.euroncap.com/en/vehicle-safety/the-ratings-explained/vulnerable-road-user-vru-protection/aeb-cyclist/>

7.2 Related Work

In the past, several researchers focused on the design of suitable bicycle dynamic models. The *Whipple model* [8] is often utilised such as in the work by Baquero-Suárez et al. [6] and Mejiard et al. [9]. The main drawback of the Whipple model is that it uses steering and lean torque as control variables which may be difficult to realise on a real autonomous bicycle. Moreover, the instrumented bicycle used in this paper has no direct regulation of the lean angle. Another commonly used bicycle model is the *point-mass model*, also known as the *inverted pendulum model*. For example, Sharma et al. [10] utilised the point-mass model to produce a fuzzy controller for stabilising a bicycle. Hauser et al. [11] used the point-mass model to investigate trajectory tracking for a motorcycle. The point-mass model only has angular inputs and outputs, thus eliminating the problems of dealing with torques. Both the point-mass model and the Whipple model can be used for the control design, thanks to their simplicity. However, more accurate models are needed to validate the control design in simulation before going to the implementation on the real bicycle. In this paper, we model the bicycle using Adams², a multibody dynamics software, and a linear model based on the point mass model.

Numerous control structures have been proposed to balance an autonomous bicycle. Tan et al. [5] developed a reinforcement learning approach to teach a humanoid to balance a bicycle in simulation. Shafiekhani et al. [7] developed adaptive critic-based neuro-fuzzy controller to solve the same problem. However, Meehan and Ruina [12] highlights that the complexity of designing complex nonlinear controllers for balancing a bicycle is often not worth the small performance benefits with respect to simpler control strategies. In [12], an LQR is compared with an dynamic programming optimal controller and the results shows that the two controllers have almost identical basin of attraction under reasonable constrained steer angles and rates.

In the work of García et al. [13], an autonomous bicycle is modelled and evaluated in several different scenarios, including starting from a stationary conditions. Thanks to an innovative design for a flywheel mounted on the bicycle together with steering control, the bicycle managed to balance in a forward speed range between 0–6m/s. To control the flywheel, an LQR controller is designed and to control the steering torque an intuitive controller [14] is used. Though, the 7.5kg flywheel peaks at around 500rpm which would consume a lot of energy. Furthermore, according to He et al. [15], methods involving direct regulation of the lean angle usually struggle when it comes to the balance of regular size bicycles, where the weight and velocity are often increased

²<https://www.mscsoftware.com/it/product/adams>

compared to a small bicycle.

Alternatively, regulation of the steering angle can be used to stabilise the bicycle. Such an approach was used in the work of Tanaka and Murakami [16], where the lean angle and lean rate were used in a PD controller to compute the desired steering acceleration. Vatanshevanopakorn and Parnichkun [17] proposed an LQR optimised for a bicycle model coupled with the dynamics of an electrical steering motor. The simulation results presented show that the proposed LQR controller structure was capable of stabilising the bicycle with an initial lean angle and steering angle other than zero. An LQR was also utilised in the work of Anjumol and Jisha to control a second-degree bicycle model [18]. The results obtained was compared with a posture controlled proposed by Tanaka et al. [19], and concluded that the LQR performed better than the posture controller in simulation.

In [15], three proportional gains are used in a feedforward and feedback loop scheme to stabilise a bicycle in both simulation and experiments. The control scheme utilises measurements of both the lean angle and the lean angle rate to compute a desired steering angle of the handlebar. The bicycle used in experiments is equipped with two motors, a few sensors, a battery, and a compactRIO which serves as the main processing unit. The results of the experiments are impressive, however, the bicycle is quite massive with a weight of 52.5kg.

In this work, we conduct a comparative performance evaluation of the main control approaches, designed for the instrumented bicycle. The way the handlebar is controlled depends on the motor and motor controller used, it can either be by steering position [15], steering torque [6], or steering velocity [12]. We control the steering position when using the PID and the fuzzy controller and steering velocity for the LQR to understand what can be more beneficial for future autonomous bicycles. The bicycle is designed with lightweight components and without altering the main structure of a regular bicycle.

7.3 Modeling of instrumented bicycle

7.3.1 Experimental platform

The instrumented bicycle, illustrated in Fig. 7.1 is based on a regular-sized electrical bicycle of a male model with the propulsion motor in the rear wheel (#1). The 11.6Ah and 36V battery is mounted on the frame of the bicycle (#5). In the design process of the instrumented bicycle, care has been taken into both the size and the weight of the components to fit all extra components, such as IMU and main processing unit, in a bicycle basket in the upcoming iteration of

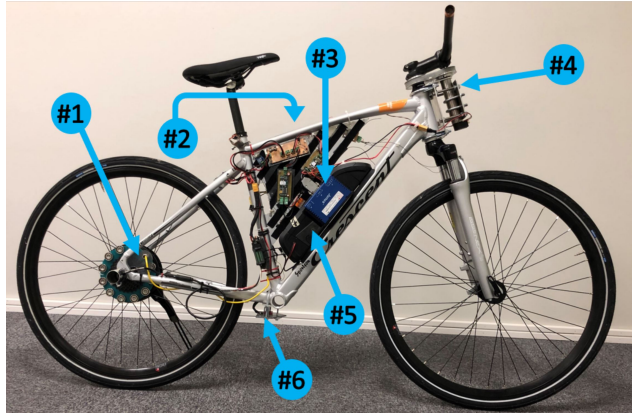


Figure 7.1: Instrumented bicycle. (#1) Propulsion motor; (#2) NI roboRIO; (#3) Motor controller; (#4) Steering motor; (#5) Battery; (#6) IMU.

the instrumented bicycle. A DCX32L Maxon motor together with a gear hub and encoder are utilised to control the handlebar through two cogwheels with a rubber band in between (#4). To control the steering motor a Junus motor controller is mounted on the side of the battery (#3) where the steering velocity ($\dot{\delta}$), is the input signal. Power distribution boards and the main processing unit, a National Instruments roboRIO (#2), are attached to the centre of the bicycle however on the opposite side of the one visualised in Fig 7.1. To measure the speed of the bicycle, a Hall sensor is used which measures the time between pulses of 12 evenly distributed magnets around the rear wheel (#1). To sense the lean angle a VectorNav VN-100 IMU is used and configured to output the lean angle and the lean rate of the bicycle (#6).

The roboRIO is equipped with both a dual-core ARM Cortex-A9CPU and

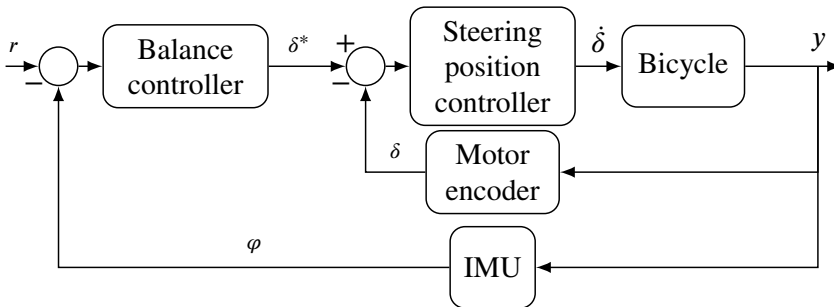


Figure 7.2: Control structure of the instrumented bicycle where a PD controller is used in an inner loop to control the steering position.

an Artix-7 FPGA and the code is written using LabVIEW³. The FPGA is used for acquiring sensor data and actuating the motors. To actuate the steering motor using a steering position, a PD controller is implemented on the FPGA which takes the error between the current steering position δ and the desired one δ^* and computes a steering velocity fed to the bicycle $\dot{\delta}$ as presented in Fig. 7.2. The parallel structured PD controller is tuned experimentally and is executing at 600Hz with $K_P = 0.1$, $K_D = 0.04$, and a filter time constant $T_f = 0.8$.

The different balancing controllers are realised on the CPU and execute at a frequency of 100Hz. The main geometrical features of the instrumented bicycle are illustrated in Fig. 7.3 and given in Table 7.1 along with constraints on the lean angle, lean rate, steering angle, and steering rate.

Table 7.1: Parameters of the instrumented bicycle.

Design parameters			
Parameter	Symbol	Unit	Value
CoG with respect to O (x)	a	[m]	0.473
CoG with respect to O (z)	h	[m]	0.515
Gravity	g	[m/s ²]	9.820
Wheelbase	b	[m]	1.080
Mass	m	[kg]	23.720
Wheel radius	r	[m]	0.349
Trail	c	[m]	0.087
Head angle	λ	[deg]	72.950
Constraints			
Lean angle	φ	[deg]	± 2
Lean rate	$\dot{\varphi}$	[deg/s]	50
Steer angle	δ	[deg]	± 15
Steer rate	$\dot{\delta}$	[deg/s]	70

7.3.2 Linear model

The point-mass model [3] describe the dynamics of the lean angle φ based on the steering angle δ and velocity $\dot{\delta}$ and is linearised about the equilibrium of zero lean and steer angle for Getz bicycle model [2]:

$$\ddot{\varphi} - mgh\varphi = \frac{v}{b}\dot{\delta} + \frac{mv^2h}{b}\delta, \quad (7.1)$$

³<https://www.ni.com/sv-se/shop/labview.html>

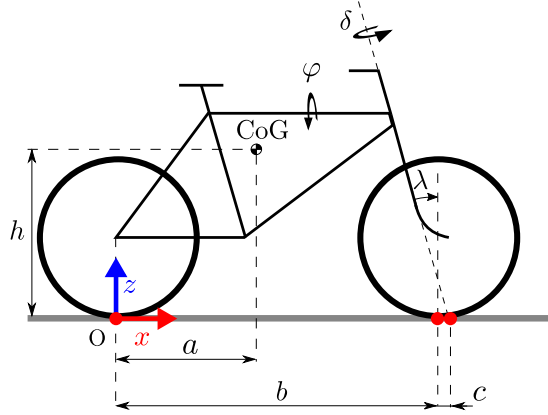


Figure 7.3: Geometrical features of a bicycle, where a and h corresponds to the measures horizontal and vertical measure of the CoG. The wheelbase is denoted by b .

where v is the forward velocity of the bicycle, and the physical parameters are reported in Table 7.1. The model assumes zero head angle ($\lambda = 0$), constant velocity (v), massless wheels and front fork. Instead, the total mass of the bicycle is lumped together forming a point mass m . The transfer function from δ to φ is

$$G(s) = \frac{\Phi(s)}{\Delta(s)} = \frac{av}{bh} \frac{s + \frac{v}{a}}{s^2 - \frac{g}{h}}. \quad (7.2)$$

However, as $G(s)$ does not model friction or dynamics of the steering setup including the steering position controller, $G(s)$ is put in series with another transfer function, $H(s)$. To obtain $H(s)$, a step response is recorded from the instrumented bicycle, see Fig. 7.4.A, and matched with a Second-Order Plus Dead Time (SOPDT) transfer function

$$H(s) = \frac{\Delta(s)}{\Delta^*(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} e^{-ds}, \quad (7.3)$$

where $\zeta = 0.6$, $\omega_n = 33.9$, and the time delay, $d = 0.015$. During the step response, the bicycle is held in an upright position with both wheels on the ground and the handlebar pointing forward. The input to $H(s)$ is the desired steering angle, δ^* and the output of $H(s)$ is the actual steering angle δ . The complete model, from δ^* to φ , is obtained as $P(s) = H(s)G(s)$ and is discretize using the zero-order hold method and a sample time $T_s = 0.01$ seconds.

For control strategies, such as LQR, the point-mass model given in (7.1) is

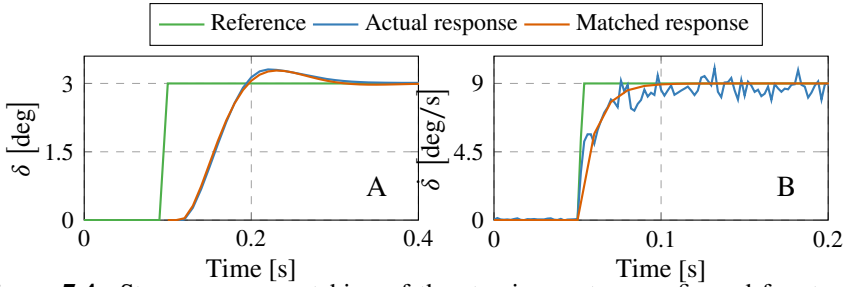


Figure 7.4: Step response matching of the steering system configured for steering position (A), and steering velocity (B). The matched responses are used in simulation to model the steering system including the motor and friction.

utilised in its state-space representation, with:

$$\mathbf{A}_1(v) = \begin{bmatrix} 0 & 1 & 0 \\ \frac{g}{h} & 0 & -\frac{v^2}{bh} \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_1(v) = \begin{bmatrix} 0 \\ -\frac{av}{bh} \\ 1 \end{bmatrix}, \quad (7.4)$$

$$\mathbf{C}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{D}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

where, the state vector consists of $\mathbf{x}_1 = [\varphi, \dot{\varphi}, \delta]^\top$. Note that the input, $u_1 = \dot{\delta}$ represents the steering velocity, instead of the desired steering position which is the input signal to the model given by the transfer function $P(s)$. When the LQR is implemented on the instrumented bicycle, the PD controller for steering position is bypassed, and the steering velocity is fed directly to the motor controller.

To include the dynamics of the steering system, comparable to the approach of the transfer function $P(s)$, another step response is matched. However, the reference is a steering velocity instead of a steering position since the model in (7.4) is using the steering velocity as its input. As a reference angular velocity, 9deg/s is commanded to the DC motor mounted on the instrumented bicycle. The Junus motor controller, used for controlling the steering system, logs the angular velocity data for post-processing. The result is presented in Fig. 7.4.B.

The matched step response in Fig. 7.4.B is converted to its state-space, and its matrices are:

$$\mathbf{A}_2 = [-100], \quad \mathbf{B}_2 = [1], \quad \mathbf{C}_2 = [100], \quad \mathbf{D}_2 = [0], \quad (7.5)$$

with the state vector $\mathbf{x}_2 = \dot{\delta}$, and the control input $u_2 = \dot{\delta}^*$ representing the desired steering velocity. The two state spaces in (7.4) and (7.5) are combined

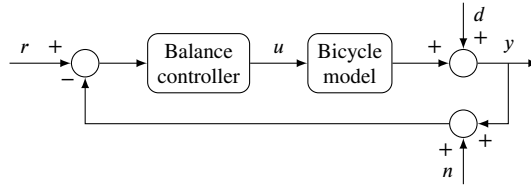


Figure 7.5: Reference feedback control scheme for balancing a bicycle.

in series as:

$$\begin{aligned} \mathbf{A}(v) &= \begin{bmatrix} \mathbf{A}_2 & \mathbf{0} \\ \mathbf{B}_1 \mathbf{C}_2 & \mathbf{A}_1 \end{bmatrix}, \quad \mathbf{B}(v) = \begin{bmatrix} \mathbf{B}_2 \\ \mathbf{B}_1 \mathbf{D}_2 \end{bmatrix}, \\ \mathbf{C} &= [\mathbf{D}_1 \mathbf{C}_2 \quad \mathbf{C}_1], \quad \mathbf{D} = [\mathbf{D}_1 \mathbf{D}_2]. \end{aligned} \quad (7.6)$$

The input signal is the desired steering velocity $u = u_2 = \dot{\delta}^*$.

7.3.3 Nonlinear model

The instrumented bicycle was disassembled and each part was measured, weighed, and designed in SolidWorks⁴. Both the rear wheel motor and the steering motor are defined as rotary motors in SolidWorks. The complete model is then exported to Adams which is a multibody dynamics simulation software that uses the Newton-Euler method to obtain the equations of motion.

In Adams, the steering motor of the bicycle is defined as either steering position or steering velocity depending on the control structure used. Gravity and a contact surface between the wheels and the ground are included as well. To model the Coulumb friction force between the ground and the wheels, the static and dynamic friction coefficients $\mu_s = 0.7$ and $\mu_d = 0.7$ are used as well as a stiction transition velocity of 0.2m/s and a friction transition velocity of 1m/s.

7.4 Control strategies

The closed-loop system in Fig. 7.5 is used to design control strategies for balancing the bicycle. The control signal u depends on which model is used as explained in the previous section. The reference signal is denoted with r , and the controlled variable with y . A load disturbance d , which intend to emulate a small physical side push of the bicycle is included in simulations. Additionally, the measurement noise n in the lean angle measurements is also considered.

⁴<https://www.solidworks.com/>

7.4.1 PID controllers

Two PID controllers, tuned using different methods, are used in this paper. Both PID controllers are designed using the ideal form of a PID controller in discrete time

$$U(z) = K_P \left(1 + K_I \cdot T_s \frac{1}{z-1} + K_D \cdot \frac{1}{T_s} \frac{z-1}{z} \right), \quad (7.7)$$

with $T_s = 0.01$ seconds. The first PID controller – in the following referred as LSPID – is tuned using a loop shaping method explained in detail in the work by Andersson et al. [20]. The general objectives of the loop shaping method is to ensure good stability, tracking performance and robustness to disturbances and uncertainties [21]. The control loop objectives are formulated as the following constraints on the loop transfer function frequency response:

- The target bandwidth is selected so that the open loop system should cross the 0 dB mark once with a phase margin of at least 30°
- For disturbance rejection, the gain for the frequency response of the open loop system below the target bandwidth should be high
- To assure robustness of plant uncertainties, the gain for the frequency response of the open loop system above the target bandwidth should be low

The cost function J , is defined as:

$$\begin{aligned} J = & w_1 (\omega_b - \omega_t)^2 \\ & + w_2 \int_{\omega_b}^{\pi/T_s} 20 \log |K(j\omega)P(j\omega)| d\omega \\ & - w_3 \int_0^{\omega_b} 20 \log |K(j\omega)P(j\omega)| d\omega, \end{aligned} \quad (7.8)$$

where $\omega_t = 15\text{rad/s}$ is the target bandwidth and ω_b is the bandwidth of the loop transfer function defined as:

$$\omega_b = \inf_{\omega} |K(j\omega)P(j\omega)| \leq 1, \quad (7.9)$$

and weights are chosen as $w_1 = 0.05$, $w_2 = w_3 = 5 \times 10^{-4}$.

An alternative tuning of the PID controller, referred to as ATPID, is performed using the Simulink PID tuner applied on the linear model $P(s)$. A cross-over frequency $\omega_c = 15\text{rad/s}$ and phase margin $\phi_m = 60^\circ$ were chosen in the design process which results in a stable controller with a rise time of 0.05s and settling time of 1.44s. The gains for the respective PID controller are reported in Table 7.2.

Table 7.2: The gains for the two PID controllers

PID gains		
Gain	LSPID	ATPID
K_P	2.514	3.167
K_I	1.544	1.326
K_D	0.074	0.069

7.4.2 Linear quadratic regulator

The LQR [22] in discrete time, is set to minimise the cost function:

$$J(\mathbf{u}) = \sum_{k=1}^{\infty} (\mathbf{x}(k)^{\top} \mathbf{Q} \mathbf{x}(k) + \mathbf{u}(k)^{\top} \mathbf{R} \mathbf{u}(k)) \quad (7.10)$$

where the $\mathbf{Q}$ and $\mathbf{R}$ are semi-definite positive matrices. The state-feedback control law is given by:

$$\mathbf{u}(k) = -\mathbf{K} \mathbf{x}(k-1) \quad (7.11)$$

where $\mathbf{K}$ is computed as

$$\mathbf{K} = (\mathbf{R} + \mathbf{B}^{\top} \mathbf{P} \mathbf{B})^{-1} \mathbf{B}^{\top} \mathbf{P} \mathbf{A} \quad (7.12)$$

and $\mathbf{P}$ is obtained by solving the discrete-time algebraic Riccati equation:

$$\begin{aligned} \mathbf{P}(k-1) = & -\mathbf{A}^{\top} \mathbf{P}(k) \mathbf{B} (\mathbf{B}^{\top} \mathbf{P}(k) \mathbf{B} + \mathbf{R})^{-1} \mathbf{B}^{\top} \mathbf{P}(k) \mathbf{A} \\ & + \mathbf{Q} + \mathbf{A}^{\top} \mathbf{P}(k) \mathbf{A}. \end{aligned} \quad (7.13)$$

The matrices $\mathbf{A}$ and $\mathbf{B}$, the state vector $\mathbf{x}$, and the input $\mathbf{u}$ are given by the state-space model of the system (7.6), converted to discrete time with a sampling time $T_s = 0.01\text{s}$. The $\mathbf{Q} = \text{diag}(Q_{ii})$ and $\mathbf{R} = \text{diag}(R_{jj})$ matrices were chosen according to Bryson's Rule [23], as

$$Q_{ii} = \frac{1}{\text{Max value of } x_i^2} \quad R_{jj} = \frac{1}{\text{Max value of } u_j^2} \quad (7.14)$$

with the constraints found in table 7.1. Now, by utilising equation (7.12) the following state-feedback gain, $\mathbf{K}$, is obtained:

$$\mathbf{K} = [22.46 \quad -37.35 \quad -4.91 \quad 8.77]^{\top}. \quad (7.15)$$

On the instrumented bicycle it is possible to access all states except the steering velocity, which is estimated from the steering position as:

$$\dot{\delta}(t) \approx \frac{\delta(t) - \delta(t - T_s)}{T_s}, \quad T_s = 0.01\text{s}. \quad (7.16)$$

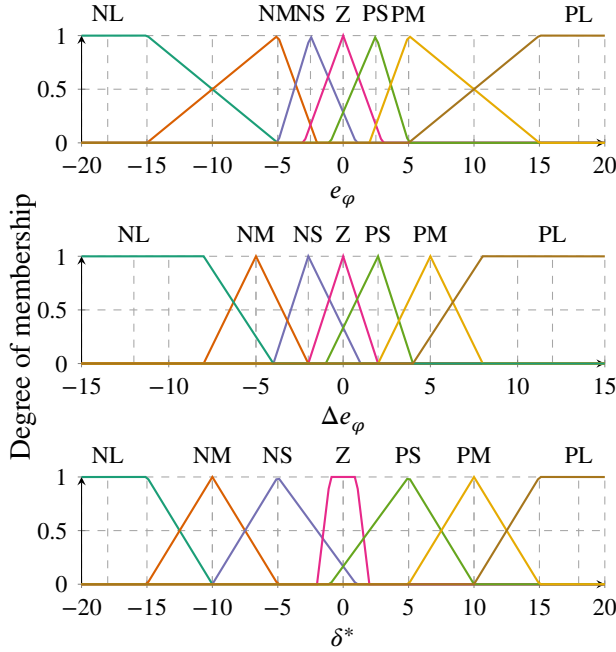


Figure 7.6: The fuzzy controllers membership functions, the first membership function is the lean angle error e_φ and the second membership function is the lean angle error difference Δe_φ . The output membership function presented in the bottom graph correlates to the desired steering angle δ^* .

7.4.3 Fuzzy controller

The fuzzy controller is inspired by the work of Abdolmalaki [24] and modified using a trial and error approach. The controller uses two inputs, the lean angle and the lean angle difference, and computes the desired steering angle, hence the linear model $P(s)$ is utilised. The input and output membership functions are shown in Fig. 7.6 where NL, NM, NS, Z, PS, PM, PL are short for Negative Large, Negative Medium, Negative Small, Zero, Positive Small, Positive Medium, and Positive Large respectively. The fuzzy rule set is defined in Table 7.3, using the same abbreviations as in Fig. 7.6. The columns represent the lean error e_φ and the rows represent the lean error difference Δe_φ . To represent implication and the “And” operation, the product function is utilised. The methods of aggregation and defuzzification are represented by the sum and the centroid functions respectively.

Table 7.3: The fuzzy rule set with “And” linguistic interception term.

$\Delta e_\varphi \backslash e_\varphi$	NL	NM	NS	Z	PS	PM	PL
NL	NL	NL	NM	NM	NS	NS	Z
NM	NL	NM	NM	NS	NS	Z	PS
NS	NM	NM	NS	NS	Z	PS	PS
Z	NM	NS	NS	Z	PS	PS	PM
PS	NS	NS	Z	PS	PS	PM	PM
PM	NS	Z	PS	PS	PM	PM	PL
PL	Z	PS	PS	PM	PM	PL	PL

7.5 Results

To evaluate the performance of the different controllers in simulation the Integrated Squared Error (ISE) of the lean angle and rejection of lean angle disturbances is used. For the experimental results, the execution time of the controllers on the platform is considered as well as their maximum balancing time of the bicycle on a bicycle roller.

7.5.1 Simulation setup

To evaluate the behaviour of the different controllers and models in simulation, Mathworks Simulink⁵ is utilised. The nonlinear model is exported from ADAMS and used in co-simulation through Simulink where it is put in series with the transfer functions of the step response matching to capture the dynamics of the steering setup. Both the linear and nonlinear models are set up in continuous time and the controllers are using a sampling time $T_s = 0.01s$, to transfer the data between the time domains rate transition blocks are utilised. The forward speed during simulations on both the linear and nonlinear model is set to 14km/h and reference lean r of 0 degrees. In simulations, Gaussian noise is added to the lean angle measurements with a standard deviation of 0.01° . The noise is measured by placing the VN-100 on a flat surface and collect the roll angle data for 30 minutes, this is repeated 3 times. To simulate a gentle push on the bicycle, a disturbance is induced in the lean angle measurements with an amplitude of 1 degree and lasts for 0.25s. The disturbance is activated after 5s. The constraints of the steering angle and steering velocity presented in Table 7.1 are implemented in simulation as saturation of the control signals.

⁵<https://se.mathworks.com/products/simulink.html>

7.5.2 Simulation results

The lean and steering angle of both the nonlinear and the linear models are shown in Fig. 7.7. The grey area in the subplots indicates where the disturbance in the lean angle measurements is injected. In the nonlinear case, the bicycle starts from zero and instantly accelerates up to 14km/h which makes the lean angle, and in extension the steering angle, to deviate a bit from zero in the beginning. The Integrated Square Error (ISE) are computed for the lean angle of the bicycle and presented in Table 7.4.

The most consistent control structure in simulation is the LQR which shows promising results for both models, for the linear case, this is not surprising as the LQR is optimised for that model. As the LQR behaves similarly on both models this suggests that the linear model is a good approximation of the more complex nonlinear model. However, it is obvious from the results of the PID controllers that the nonlinear model is more challenging to control for this type of controller. Especially the LSPID which has smaller gains struggles when it comes to disturbance rejection of the nonlinear model, this is confirmed by the ISE values for the LSPID. The ATPID, with a set of higher gains, are performing much better on both models. The ISE also verifies the consistency of the LQR and shows that the Fuzzy controller struggles on both models.

Table 7.4: The ISE values from the lean angle error in simulation

Integrated Squared Error (ISE) on lean angle		
	Nonlinear model	Linear model
LSPID	83.79	30.25
ATPID	31.25	24.16
LQR	27.24	23.39
Fuzzy	71.47	88.55

7.5.3 Experimental setup

In experiments, the instrumented bicycle is placed on a bicycle roller and reference lean of 0 degrees is used for all controllers⁶. The reason for choosing a bicycle roller is due to space limitations and weather conditions. At startup, the handlebar is pointing approximately forward and the bicycle is held by an operator in an upright position. After the bicycle has accelerated up to 14km/h, the operator releases the bicycle and the logging of data begins. In case the operator touches the bicycle, the remaining data does not qualify as self stabilising

⁶A footage of one experiment is available at this link: <https://youtu.be/9owSiGU-Z0o>

of the bicycle, which is indicated by the grey areas in Fig. 7.8. There were no disturbance injected in the lean angle measurements in the experiments, mainly due to the experimental setup using a bicycle roller.

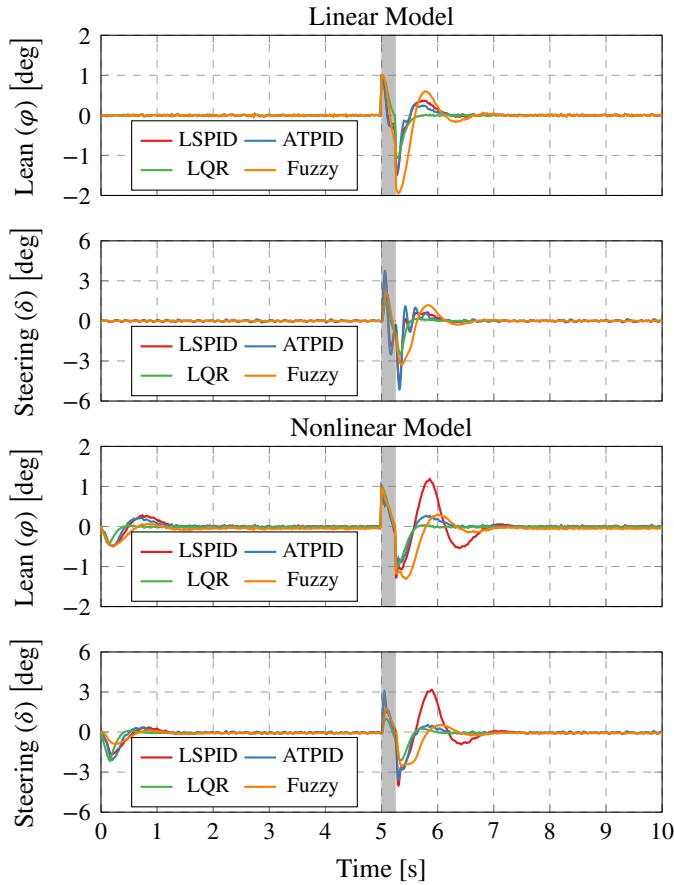


Figure 7.7: Results of the different control strategies when controlling the linear model and the nonlinear model in simulation.

7.5.4 Experimental results

The outcome of the experiments conducted on the roller is presented in Fig. 7.8. All four controllers manage to balance the bicycle. However, due to the narrow bicycle roller, the instrumented bicycle tends to go out of bounds and an operator needs to assist the bicycle to keep it on the roller. The Fuzzy controller manages to balance the bicycle, but with large oscillations in the lean angle

which makes it drift off the roller after 10 seconds. Perhaps with a better tuned fuzzy system, the bicycle balancing could be improved. However, the long execution time of the fuzzy system makes it a weak candidate compared to LQR and PID on a real-time embedded system. The two PIDs and the LQR manages to stabilise and keep the bicycle on the roller for over 40 seconds, with the LQR performing slightly worse than the PID's. A reason for this might be caused by the estimation of the steering velocity, which could be improved by a Kalman filter. The results also suggest that in experiments, it is more beneficial to use the steering position as the control signal compared to the steering velocity.

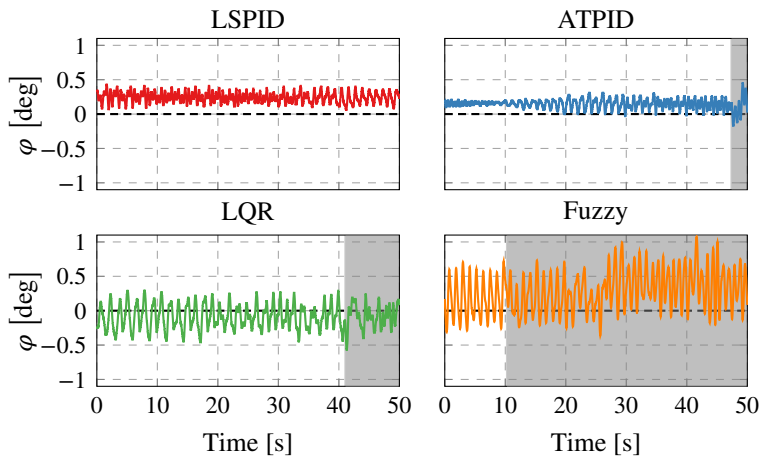


Figure 7.8: Results of the experiments conducted on the bicycle roller. The top row presents the lean angle of the bicycle while steering angle is reported in the second row for LSPID and ATPID. The third and the fourth row presents the lean angle and steering angle of the LQR and Fuzzy controller respectively. The gray area indicates that the bicycle has been aided by an operator, thus the experiment is considered to be aborted at that point.

The LSPID, which did not show the most promising results in simulation, produced the best results in experiments, indicating that the nonlinear validation model could be improved in future work. The most consistent controllers, when considering both experiment and simulation, are the ATPID and LQR.

To evaluate the execution time of the different control structures they are implemented on the roboRIO platform. Random numbers in the range $[-15, 15]$ deg are used as inputs to the controllers and the mean execution time and standard deviation of 10 000 loops are calculated for each controller. Table 7.5 shows the obtained averages for the three classes of controllers. Both the PID and the LQR show an execution time of few micro-seconds,

while the Fuzzy controller has an execution time in the order of hundreds of micro-seconds. Even though, all the controllers manage to complete the respective control laws within the sampling period ($T_s = 0.01\text{s}$), it is preferable to use the PID or the LQR controllers since it can allow additional functionalities to be implemented on the same computing platform, such as the motion planning, localization algorithms, and so on.

7.6 Conclusion and Future Work

To improve the evaluation process of vehicles ability to detect and classify bicycles, a driverless instrumented bicycle is designed with components which could fit into the bicycle frame or be hidden away in a bicycle basket. To evaluate and compare different control algorithms for stabilising the system, two different models are developed. The instrumented bicycle senses the lean angle and adjusts the steering angle to maintain balance. For the purpose of controlling the handlebar, two different tuned PID controllers, an LQR and a Fuzzy controller are evaluated and implemented in both simulations and on the instrumented bicycle. In the simulation, the controllers are compared both on a linear and a nonlinear model and a disturbance is induced in the lean angle measurements to evaluate the robustness of the four controllers. The outcome from the conducted experiments shows promising results when using the PID controllers or LQR, which all manages to keep the bicycle balanced for over 40 seconds on a narrow roller.

To maintain the silhouette of a bicycle, the next iteration of the instrumented bicycle should focus on embedding the components into the bicycle frame. Since the LQR and PID controller with the set of higher gains shows consistent results in simulation and experiments these controllers should be considered for balancing the bicycle in future work where path following will be addressed.

Table 7.5: Execution time of the different control strategies

Execution time		
Controller	Mean [μs]	Standard deviation [μs]
PID	8.24	4.78
LQR	9.97	3.96
Fuzzy	582.37	54.50

7.7 Acknowledgements

This work is part of a research project between Mälardalen University, Chalmers University, Volvo Cars, AstaZero, and Cycleurope and it is funded by Vinnova. The work is partly funded by Eskilstuna kommun och Eskilstuna Fabrikförening.

Bibliography

- [1] P. Fairley, “Self-driving cars have a bicycle problem [news],” *IEEE Spectrum*, vol. 54, no. 3, pp. 12–13, 2017.
- [2] N. H. Getz and J. E. Marsden, “Control for an autonomous bicycle,” in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1397–1402, 1995.
- [3] K. J. Åström, R. E. Klein, and A. Lennartsson, “Bicycle dynamics and control: adapted bicycles for education and research,” *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, 2005.
- [4] A. L. Schwab and J. P. Meijaard, “A review on bicycle dynamics and rider control,” *Vehicle System Dynamics*, vol. 51, no. 7, pp. 1059–1090, 2013.
- [5] J. Tan, Y. Gu, C. K. Liu, and G. Turk, “Learning bicycle stunts,” *ACM Trans. Graph.*, vol. 33, no. 4, pp. 1–12, 2014.
- [6] 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.
- [7] A. Shafiekhani, M. J. Mahjoob, and M. Akraminia, “Design and implementation of an adaptive critic-based neuro-fuzzy controller on an unmanned bicycle,” *Mechatronics*, vol. 28, pp. 115–123, 2015.
- [8] F. Whipple, “Stability of the motion of a bicycle,” *Quarterly Journal of Pure and Applied Mathematics*, vol. 30, pp. 312–348, 1899.
- [9] 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.

- [10] H. D. Sharma and N. Umashankar, "A Fuzzy Controller Design for an Autonomous Bicycle System," *2006 IEEE International Conference on Engineering of Intelligent Systems*, pp. 1–6, 2006.
- [11] J. Hauser, A. Saccon, and R. Frezza, "Aggressive motorcycle trajectories," *IFAC Proceedings Volumes*, vol. 37, no. 13, pp. 949–954, 2004.
- [12] D. E. Meehan and A. L. Ruina, "Linear and nonlinear controllers of an autonomous bicycle have almost identical basins of attraction in a non-linear simulation," in *Bicycle and Motorcycle Dynamics 2019*, Symposium on the Dynamics and Control of Single Track Vehicles, 2020.
- [13] M. R. García, D. A. Mántaras, J. C. Álvarez, and D. Blanco F, "Stabilizing an urban semi-autonomous bicycle," *IEEE Access*, vol. 6, pp. 5236–5246, 2018.
- [14] A. L. Schwab, J. D. Kooijman, and J. P. Meijaard, "Some recent developments in bicycle dynamics and control," in *European Conf. on Structural Control (ECSC)*, 2007.
- [15] J. He, M. Zhao, and S. Stasinopoulos, "Constant-velocity steering control design for unmanned bicycles," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 428–433, 2015.
- [16] Y. Tanaka and T. Murakami, "Self sustaining bicycle robot with steering controller," in *The 8th IEEE International Workshop on Advanced Motion Control (AMC)*, pp. 193–197, IEEE, 2004.
- [17] S. Vatanashevanopakorn and M. Parnichkun, "Steering control based balancing of a bicycle robot," in *2011 IEEE International Conference on Robotics and Biomimetics*, pp. 2169–2174, 2011.
- [18] M. A. Anjumol and V. R. Jisha, "Optimal stabilization and straight line tracking of an electric bicycle," in *2014 International Conference on Power Signals Control and Computations (EPSCICON)*, pp. 1–6, 2014.
- [19] Y. Tanaka and T. Murakami, "A study on straight-line tracking and posture control in electric bicycle," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 1, pp. 159–168, 2009.
- [20] T. Andersson, N. Persson, A. Fattouh, and M. C. Ekström, "A loop shaping method for stabilising a riderless bicycle," in *2019 European Conference on Mobile Robots (ECMR)*, pp. 1–6, 2019.

- [21] D. Bruijnen, R. van de Molengraft, and M. Steinbuch, "Optimization aided loop shaping for motion systems," in *IEEE Int. Conf. on Contr. Appl.*, pp. 255–260, IEEE, 2006.
- [22] V. Kumar E, J. Jerome, and K. Srikanth, "Algebraic approach for selecting the weighting matrices of linear quadratic regulator," in *2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, pp. 1–6, 2014.
- [23] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*. Pearson, 8th ed., 2019.
- [24] R. Y. Abdolmalaki, "Implementation of fuzzy inference engine for equilibrium and roll-angle tracking of riderless bicycle," *arXiv e-prints*, 2017.

Chapter 8

Paper B

Trajectory tracking and stabilisation of a riderless bicycle

Niklas Persson, Martin C. Ekström, Mikael Ekström, Alessandro V. Papadopoulos.

In International Conference on Intelligent Transportation Systems (ITSC),
2021

Abstract

Trajectory tracking for an autonomous bicycle is considered in this paper. The trajectory tracking controller is designed using a Model Predictive Controller with constraints on the lean, steer, and heading angle as well as the position coordinates of the bicycle. The output from the trajectory tracking controller is the desired lean angle and forward velocity. Furthermore, a PID controller is designed to follow the desired lean angle, while maintaining balance, by actuation of the handlebar. The proposed control strategy is evaluated in numerous simulations where a realistic nonlinear model of the bicycle is traversing a go-kart track and a short track with narrow curves. The Hausdorff distance and Mean Squared Error are considered as measurements of the performance. The results show that the bicycle successfully can track desired trajectories at varying velocities.

8.1 Introduction

The bicycle, with its slim structure of two inline wheels and a frame, is statically unstable but with proper actuation, it can be made stable [1]. A human can learn to balance and control a bicycle from an early age by using the principle of steering into the fall of the bicycle. This simple principle has also been replicated in autonomous self-balancing bicycles [2]. However, a cyclist does not only balance a bicycle but does in general also follow a path to a destination. A self-balancing bicycle, which also could navigate on its own could potentially be used in several applications, e.g., for message delivery service or bike-sharing [3]. Furthermore, autonomous road vehicles, such as cars and trucks, struggle to correctly detect and classify Vulnerable Road Users (VRU), such as cyclists, as discussed by Fairley [4]. Thus, a fully autonomous bicycle could be used on the test tracks to aid the evaluation process and improve the VRU detection and emergency braking system of other road vehicles. It is crucial that the evaluation of test tracks is conducted in realistic environments, with realistic behaviour of the VRU. An autonomous bicycle, which can navigate and balance will resemble a cyclist to a larger degree compared to the current bicycle targets which are mounted on a sledge and pulled in front of, or beside the car¹.

Motion planning and path tracking are important components for autonomous vehicles, and they have been extensively explored in the area of mobile robotics and autonomous cars [5]. However, the application of such techniques cannot be applied as an off-the-shelves solution to riderless bicycles, as some manoeuvres may make the bicycle fall. For example, path trackers for cars typically use the steering angle to manipulate the heading [6], while in the case of a bicycle that uses steer actuation to maintain balance, the desired steering angle could potentially interfere with its balance and cause a fall. Instead, a desired lean angle is commonly used to alter the heading of the bicycle, which in extension controls the steering angle by using the principle of steering into the fall [7, 8].

In this paper, we present a cascaded control architecture to (i) balance the autonomous bike, and (ii) track the desired trajectory. The inner controller is in charge of balancing the bicycle, and it is designed as a robust PID control loop. The outer controller, is in charge of the trajectory tracking task, controlling the desired lean-angle of the bicycle to adjust its heading, and it is designed as a Model Predictive Controller (MPC). The system is evaluated through co-

¹<https://www.euroncap.com/en/vehicle-safety/the-ratings-explained/vulnerable-road-user-vru-protection/aeb-cyclist/>

simulation between Adams² and Matlab, where noise in the lean angle measurements and disturbance on the steering system are included.

The rest of this paper is organized as follows. In Section 8.2 the related work is presented. The bicycle model is derived in Section 8.3 followed by Section 8.4 where the control design is discussed. Section 8.5 presents the results and the paper is concluded in Section 8.6 which also includes future research.

8.2 Related Work

The path tracking problem for bicycles has received some attention in previous research. However, most previous bicycle research focuses on the modelling and balance of unmanned bicycles. In this section, the modelling is discussed first, then previously proposed solutions to the path tracking problem are discussed.

8.2.1 Modelling

The dynamic stability of bicycles has been researched for over a century, with Whipple and his work on bicycle stability dated back to 1899 as one of the first contributions to the topic [9]. To use the Whipple model, a set of 25 parameters needs to be measured from the bicycle, which can be time-consuming and sometimes difficult to obtain accurate measures. A simplified dynamic model was presented in the work of Getz, where the mainframe of the bicycle was modelled as a point-mass [10, 11]. In comparison to the Whipple model, this model requires only 4 parameters to be measured. Extending Getz work, Yi et al. presented a similar bicycle model [12, 13]. However, an important difference between the models is that the model proposed by Getz assumes that the steering axis is vertical, while the model by Yi et al. allows for a tilted steering axis, providing a positive trail to the bicycle. This extension offers a more realistic model of a bicycle as most bicycles possess a positive trail, even though it is not necessary for the self-stabilisation of a bicycle [14]. Yi used the model to develop a nonlinear control for path tracking and balance of a motorcycle in simulations [12]. The model has also been used for developing balance controllers for bicycles in both the work of He et al. [15] and Zhang et al. [16], as well as for bicycle localisation in the work of Miah et al. [17].

²<https://www.mscsoftware.com/it/product/adams>

8.2.2 Path tracking

In order to solve the path tracking problem, Dao and Chen [8] used a multi-loop control. To control their bicycle model, a sliding mode controller was used for lean angle tracking and a fuzzy logic controller with gain scheduling and integral controller constituted the path tracking controller in the outer loop. The results are promising as small tracking errors are achieved, however, a discussion regarding discretization and sampling times of the system is lacking. This is an important topic if the transition from simulation to experiments is to be made, and requires a discrete-time algorithm. The path tracking problem was also addressed in the work of Baquero-Suárez et al. [7], where an Active Disturbance Rejection Controller (ADRC) was designed from the linearised equations of the Whipple model [18] to balance the bicycle. A control law was also established which relates the lean angle of the bicycle to its yaw angle, and the controller was extended with path tracking capabilities. A reference lean angle other than zero was fed to the outer loop of the controller which enables the bicycle to track the desired heading. Baquero-Suárez path tracker showed promising results in simulation where a forward velocity of 1.5m/s was considered when evaluated on both a straight path and a circular path with a radius of 15m. In this paper, higher forward velocities are considered as well as narrow curves.

A way to address the path tracking problem, which has proven successful for both mobile robots and autonomous vehicles is the MPC [5]. A desired property of the MPC is the possibility to incorporate constraints on the states and control signals of the system while tracking a reference by looking ahead. By recursively minimising a cost function over a finite time horizon an optimal control signal is computed. However, there is an important trade-off. Typically, the larger the time horizon, the higher the accuracy, but so is the execution time. An MCP were used by Frezza et al. [19] as a path tracking controller for a motorcycle. To evaluate the controller, a multi-body simulation environment was used and the obtained results showed good tracking accuracy. Clearly, the proposed controller produces satisfactory results when used on a high-speed motorcycle, but unfortunately, it was never evaluated for a lower speed range typical for a bicycle. Also, the sampling and execution times were not considered in their work. Path planning and path tracking as well as obstacle detection and avoidance for an autonomous bicycle were presented in the work by Zhao et al. [20]. The path planning algorithm considered a global linear path between two points and re-plans the path using four phases if an obstacle is detected by the onboard Lidar. To balance the bicycle, the controller presented in [15] was used. The simulation and experimental results are impressive, however, the

details of the path tracking algorithm are not disclosed as well as any details of sampling and execution time of the different systems.

In this paper, we propose an MPC controller to address the path tracking problem for a riderless bicycle. Instead of considering low velocities and wide curves as in [7], we are using operating velocities typical for a bicycle. Moreover, the reference path includes both wide and narrow curves as well as straights. Many of the previous path tracking controllers are only considered in simulation and the transition to an experimental setup, including sampling and execution times is not discussed [19, 8]. In this paper, different sampling times are considered for the inner and outer control loop when controlling the nonlinear bicycle model in simulation.

8.3 Bicycle model

To model the bicycle, the point-mass model presented in the work of Yi et al. [13] is used. The model assumes the bicycle to ride on a horizontal plane, where the interaction between the wheels and the ground is point contacts and the geometry and thickness of the wheels are neglected. Furthermore, the mass of the rear frame is considered as a point mass, and non-holonomic constraints are imposed resulting in $v_x = v$ and $v_y = 0$. There are three coordinate systems associated with the model, the navigation frame $\mathcal{N}(X, Y, Z)$ fixed on the ground plane, the wheelbase frame $\mathcal{W}(x, y, z)$ associated with the line between the point C_1 and C_2 and the last frame is attached to the rear frame of the bicycle $\mathcal{R}(x_R, y_R, z_R)$, as shown in Fig. 8.1. The lean angle, $\varphi(t)$, and steering angle, $\delta(t)$, are positive following the right-hand rule.

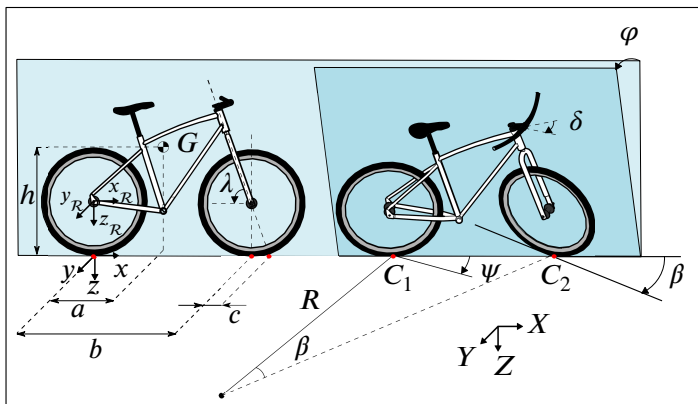


Figure 8.1: A bicycle riding on a flat horizontal plane.

The projected steering angle $\beta(t)$, i.e, the steering angle at the horizontal plane, can be obtained as:

$$\tan(\beta(t)) = \frac{\tan(\delta(t)) \sin(\lambda)}{\cos(\varphi(t))}, \quad (8.1)$$

where λ is the caster angle. Consider a bicycle riding on a path with radius R , then the curvature $\sigma(t)$ is:

$$\sigma(t) = \frac{b}{R(t)} = \tan(\beta(t)) \quad (8.2)$$

with b representing the wheelbase. The small angle approximation for the steer angle δ and lean angle φ yields:

$$\sigma(t) \approx \delta(t) \sin(\lambda), \quad (8.3)$$

and the respective curvature rate of change:

$$\omega_\sigma(t) = \dot{\sigma}(t) \approx \dot{\delta}(t) \sin(\lambda). \quad (8.4)$$

The lateral dynamics of the bicycle on the plane can be formulated as:

$$\begin{aligned} \dot{x} &= v(t) \cos(\psi(t)) \\ \dot{y} &= v(t) \sin(\psi(t)) \\ \dot{\psi}(t) &= \frac{\sigma(t)v(t)}{b} = \frac{\tan(\delta(t)) \sin(\lambda)}{b \cos(\varphi(t))} v(t). \end{aligned} \quad (8.5)$$

The roll dynamics of the bicycle can be expressed as:

$$\begin{aligned} h^2 \ddot{\varphi}(t)m &= gm \left(h \sin(\varphi(t)) + \frac{ca}{b} \sin(\lambda) \sigma(t) \cos(\varphi(t)) \right) - \\ &\left(1 - \frac{h}{b} \sigma(t) \sin(\varphi(t)) \right) \frac{h}{b} \sigma(t) \cos(\varphi(t)) v^2(t)m \\ &- \frac{ahm}{b} \cos(\varphi(t)) \left(\sigma(t) \dot{v}(t) - v(t) \omega_\sigma(t) \right). \end{aligned} \quad (8.6)$$

Here, the mass, m , cancels out and using the relationships in Eq. (8.1) and Eq. (8.2) the roll dynamics can be simplified as:

$$\begin{aligned} h^2 \ddot{\varphi}(t) &= g \left(h \sin(\varphi(t)) + \frac{cap^2}{b} \tan(\delta(t)) \right) - \\ &\left(1 - \frac{hp}{b} \tan(\delta(t)) \tan(\varphi(t)) \right) \frac{hp}{b} \tan(\delta(t)) v(t)^2 \\ &- \frac{ahp}{b} \tan(\delta(t)) \dot{v}(t) - \frac{ah}{b} \cos(\varphi(t)) v(t) \omega_\sigma(t). \end{aligned} \quad (8.7)$$

where $p = \sin(\lambda)$. To linearise the model, we apply the small angles approximation, and a constant velocity ($\dot{v} = 0$) obtaining:

$$h^2 \ddot{\varphi}(t) = g \left(h\varphi(t) + \frac{cap^2}{b} \delta(t) \right) - \left(1 - \frac{hp}{b} \delta(t)\varphi(t) \right) \frac{hp}{b} \delta(t)v^2 - \frac{ahp}{b} v \dot{\delta}(t) \quad (8.8)$$

Finally, we linearise at the equilibrium point $\bar{\varphi}(t) = 0$, $\bar{\delta}(t) = 0$ and $\bar{\dot{\delta}}(t) = 0$ using first order Taylor expansion and the linearised roll dynamics becomes

$$\ddot{\varphi}(t) = \frac{g}{h} \varphi(t) + \frac{gcap^2}{bh^2} \delta(t) - \frac{p}{bh} v^2 \delta(t) - \frac{ap}{bh} v \dot{\delta}(t). \quad (8.9)$$

Eq. (8.9) can be rewritten in state-space form, with the input $\mathbf{u}_\varphi = \dot{\delta}$, output $\mathbf{y}_\varphi = \varphi$, and the state $\mathbf{x}_\varphi = [\dot{\varphi}, \varphi, \delta]^\top$ as:

$$\begin{aligned} \mathbf{A}_\varphi &= \begin{bmatrix} 0 & \frac{g}{h} & \frac{p}{bh} \left(\frac{gcap}{h} - v^2 \right) \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{B}_\varphi &= \begin{bmatrix} -\frac{ap}{bh} v \\ 0 \\ 1 \end{bmatrix} \\ \mathbf{C}_\varphi &= [0 \quad 1 \quad 0], & \mathbf{D}_\varphi &= [0]. \end{aligned} \quad (8.10)$$

The relevant model parameters are obtained through measurements on the instrumented bicycle in [21] and presented in Table 8.1. To model the steering dynamics, including the steering motor and frictions, a step response matching is performed. The instrumented bicycle in [21] is held in an upright position with a steering angle of approximately 0 degrees, a reference angular velocity

Table 8.1: Parameters of the instrumented bicycle.

Design parameters			
Parameter	Symbol	Unit	Value
CoG with respect to O (x)	a	[m]	0.473
CoG with respect to O (z)	h	[m]	0.515
Gravity	g	[m/s ²]	9.820
Wheelbase	b	[m]	1.080
Mass	m	[kg]	23.720
Trail	c	[m]	0.087
Head angle	λ	[deg]	72.950

of 9deg/s is commanded to the motor and the results are sampled and stored on the motor controller. The step response is shown in Fig. 8.2 and the matched response is represented by a state-space model as:

$$\begin{aligned} \mathbf{A}_{\dot{\delta}} &= [-100], & \mathbf{B}_{\dot{\delta}} &= [100], \\ \mathbf{C}_{\dot{\delta}} &= [1], & \mathbf{D}_{\dot{\delta}} &= [0], \end{aligned} \quad (8.11)$$

with the desired steering velocity, $\mathbf{u}_{\dot{\delta}} = \dot{\delta}^*$, as input and the actual steering velocity, $\mathbf{y}_{\dot{\delta}} = \mathbf{x}_{\dot{\delta}} = \dot{\delta}$, serving as both the output and the state. The state-space model resulting from the series of the steer dynamics and the roll dynamics is:

$$\begin{aligned} \mathbf{A}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{A}_{\dot{\delta}} & \mathbf{0} \\ \mathbf{B}_{\varphi}\mathbf{C}_{\dot{\delta}} & \mathbf{A}_{\varphi} \end{bmatrix}, & \mathbf{B}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{B}_{\dot{\delta}} \\ \mathbf{B}_{\varphi}\mathbf{D}_{\dot{\delta}} \end{bmatrix}, \\ \mathbf{C}_{\dot{\delta}\varphi} &= [\mathbf{D}_{\varphi}\mathbf{C}_{\dot{\delta}} & \mathbf{C}_{\varphi}], & \mathbf{D}_{\dot{\delta}\varphi} &= [\mathbf{D}_{\varphi}\mathbf{D}_{\dot{\delta}}], \end{aligned} \quad (8.12)$$

with the state vector $\mathbf{x}_{\dot{\delta}\varphi} = [\mathbf{x}_{\dot{\delta}}, \mathbf{x}_{\varphi}^T]^T$, and the control input $\mathbf{u}_{\dot{\delta}\varphi} = \dot{\delta}^*$ representing the desired steering velocity.

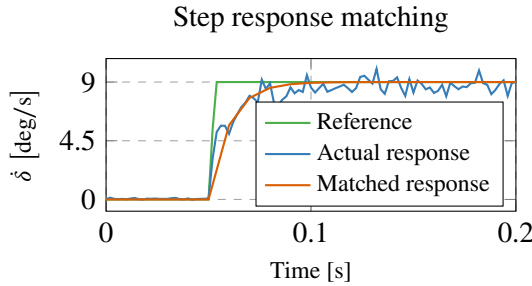


Figure 8.2: Step response matching of the steering system. The matched response is used in simulation to model the dynamics of the steering system.

8.4 Control design

In this section, the PID controller in charge of balancing the bicycle is derived first. The tuned PID controller is then put in series with the roll dynamics and appended to the lateral dynamics, and the complete model is used to design an MPC for the outer loop.

8.4.1 Balance controller

To maintain the balance of a bicycle, the front fork is steered in the same direction as the fall which will cause an acceleration in the opposite direction and

bring the bicycle to an upright position again. In the case of uneven terrain, side wind, or other disturbances the task of balance becomes more challenging. Therefore, disturbance rejection is an important property for the balancing controller and should be emphasised in the control design.

Consider the system in Fig. 8.3 where $R(s)$ is a real PID controller in parallel form

$$R(s) = K_p + \frac{K_i}{s} + \frac{K_d N}{1 + \frac{N}{s}}. \quad (8.13)$$

In the figure, $G(s)$ is the transfer function of the steering dynamics in Eq. (8.11), and $H(s)$ is the transfer function associated with Eq. (8.9), with input $\dot{\delta}$ and output φ , and it is equal to:

$$H(s) = \frac{\Phi(s)}{\dot{\Delta}(s)} = \frac{(gcap^2 - hpu^2) - ahpuv s}{bh^2 s^3 - bhgs}. \quad (8.14)$$

The roll dynamics, including the actuation, can be therefore expressed as the transfer function $P(s) = G(s)H(s)$. As illustrated in Fig. 8.3, a disturbance d , acting on the control signal $\dot{\delta}^*$, and noise n on the lean angle measurements φ are considered, and their effect should be minimized. The reference lean angle is denoted φ^* and the steering velocity which affects $H(s)$ is $\dot{\delta}$. The open-loop transfer function from the load disturbance to the system output is $Q(s) = \frac{P(s)}{1+R(s)P(s)}$.

The system with transfer function $H(s)$ has three poles in $s = 0$, and $s = \pm\sqrt{g/h}$, and therefore it is unstable with a pole in the right-half-plane. As a consequence, also the system with transfer function $P(s) = G(s)H(s)$ is unstable. As such, several autotuning techniques cannot be applied for the problem at hand [22]. Instead, the PID tuning problem, for a forward velocity

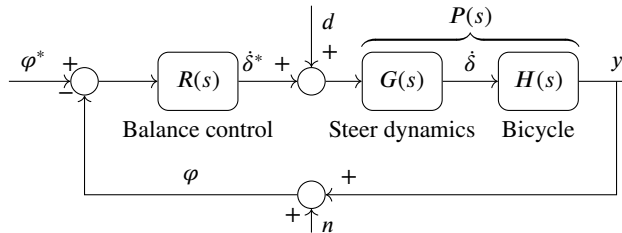


Figure 8.3: Balance controller $R(s)$ and the linear model $P(s)$.

$v = 14\text{km/h}$, is formulated as a nonlinear optimisation problem:

$$\begin{aligned}
 \min_{K_p, K_i, K_d, N} \quad & (\omega_c^{\text{des}} - \omega_c)^2 + w_1 \int_0^\infty t q(t)^2 dt + w_2 \int_0^\infty t \ell(t)^2 dt \\
 \text{s.t.} \quad & K_p, K_i, K_d \in [-200, 0], \\
 & N \in [10, 1000], \\
 & \omega_c^{\text{des}} = 60\text{rad/s}.
 \end{aligned} \tag{8.15}$$

with $w_1, w_2 \in [0, 1]$ and $w_1 + w_2 = 1$. Here, the desired crossover frequency is denoted ω_c^{des} and the crossover frequency of the open-loop response is ω_c . The function $q(t)$ is defined as the integrated mean squared value for the load disturbance. Similarly, $\ell(t)$ is the integrated squared error for the closed-loop step response. To leverage the influence of these performance measurements the weights, $w_1 = 0.5, w_2 = 0.5$ are utilised.

The nonlinear optimisation problem in Eq. (8.15) is solved by means of Particle Swarm Optimisation (PSO) [23] which is an iterative search algorithm. PSO does not require the optimisation problem to be differentiable and can search a large space of candidate solutions to the problem, however, it does not guarantee a globally optimal solution. A population size, or swarm size, is chosen initially as well as the size of the search space of possible solutions. Each particle in the population is given a random position in the search space and evaluate the cost function in Eq. (8.15) at their respective position. The result of the evaluation generates a velocity for the particle towards both its own best cost solution, but also towards the global best cost solution. The speed of the particle towards the local and global best solutions is also dependent on the stochastic local and global acceleration coefficients, $c_L \sim \mathcal{U}(0, \chi\varphi_1)$ and $c_G \sim \mathcal{U}(0, \chi\varphi_2)$, where $\varphi_{1,2}$ is chosen as 2.05 and with $\kappa = 1$, χ is computed as:

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{(\varphi^2 - 4\varphi)}|}. \tag{8.16}$$

The algorithm stops after a termination criterion is met or the maximum number of iterations, chosen as 1000, is reached. The resulting PID parameters are shown in Table 8.2.

The state-space matrices of the PID controller in (8.13) are defined as:

$$\begin{aligned}
 \mathbf{A}_R &= \begin{bmatrix} -N & 0 \\ 1 & 0 \end{bmatrix}, & \mathbf{B}_R &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \\
 \mathbf{C}_R &= [K_i - K_d N^2 \quad K_i N], & \mathbf{D}_R &= [K_p + K_d N],
 \end{aligned} \tag{8.17}$$

where the state vector is $\mathbf{x}_R = [e_1, e_2]^\top$, the output $\mathbf{y} = \dot{\delta}^*$, and the input $\mathbf{u} = e$ with $e = \varphi^* - \varphi$. Now, the open loop system in Fig 8.3 can be written as:

$$\begin{aligned} \mathbf{A}_O &= \begin{bmatrix} \mathbf{A}_R & \mathbf{0}^{(2 \times 4)} \\ \mathbf{B}_{\dot{\delta}\varphi} \mathbf{C}_R & \mathbf{A}_{\dot{\delta}\varphi} \end{bmatrix} & \mathbf{B}_O &= \begin{bmatrix} \mathbf{B}_R \\ \mathbf{B}_{\dot{\delta}\varphi} \mathbf{D}_R \end{bmatrix} \\ \mathbf{C}_O &= [\mathbf{D}_{\dot{\delta}\varphi} \mathbf{C}_R & \mathbf{C}_{\dot{\delta}\varphi}] & \mathbf{D}_O &= [\mathbf{D}_{\dot{\delta}\varphi} \mathbf{D}_R] \end{aligned} \quad (8.18)$$

and by closing the loop we obtain

$$\begin{aligned} \mathbf{A}_C &= \mathbf{A}_O - \mathbf{B}_O \mathbf{C}_O \\ \mathbf{B}_C &= \mathbf{B}_O \\ \mathbf{C}_C &= \mathbf{C}_O \\ \mathbf{D}_C &= \mathbf{D}_O. \end{aligned} \quad (8.19)$$

The input to the closed loop system is $\mathbf{u}_C = \varphi^*$, the output $\mathbf{y}_C = \varphi$, and the state vector $\mathbf{x}_C = [e_1, e_2, \dot{\delta}, \dot{\varphi}, \varphi, \delta]$.

Table 8.2: Computed optimal PID parameters.

PID parameters	
Parameter	Value
K_p	-82.6193
K_i	-69.4433
K_d	-22.4138
N	234.4655

8.4.2 Path tracking

A cyclist who does not preview the path ahead would struggle and ultimately could lose control and balance of the bicycle. Instead, a cyclist typically looks ahead and plans a path and by proper actuation of the handlebar, pedals, and body movements the path can be tracked. The MPC is an online optimal control algorithm that predicts the future behaviour of the plant for a given prediction horizon H_p , similar to how a cyclist tracks a path. Constraints on the states, output and input variables are considered as well. If the plant model and constraints are linear, the optimisation problem becomes convex and thus an optimal solution can be guaranteed. However, if the model does not successfully approximate the plant to a satisfying degree, the output of the MPC will be flawed. Thus, the results are highly dependent on the model and the design parameters of the MPC.

Consider the bicycle riding on a horizontal plane, the motion can be described by the lateral dynamics in Eq. (8.5), and given small angle approximation and a constant velocity, it can be written in state space form as:

$$\begin{aligned} \mathbf{A}_K &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ v_c & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_K = \begin{bmatrix} \frac{p}{b}v_c & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\ \mathbf{C}_K &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{D}_K = [0], \end{aligned} \quad (8.20)$$

with the state vector $\mathbf{x} = [\psi, x, y]^\top$, and the input $\mathbf{u} = [\delta, v]$. A model which includes both the motion of the bicycle, as well as its roll dynamics, can be obtained by augmenting the lateral dynamics in Eq. (8.20) to the closed loop dynamics in Eq. (8.19). When augmenting the systems, the steering angle δ , used for computing the heading ψ , can now be obtained from the state vector instead. The state matrices of the complete model are:

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} \mathbf{A}_K & \mathbf{0}^{(3 \times 5)} & \mathbf{B}_{K1} \\ \mathbf{0}^{(6 \times 3)} & \mathbf{A}_C & \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_{K2} & \mathbf{0}^{(3 \times 1)} \\ \mathbf{0}^{(6 \times 1)} & \mathbf{B}_C \end{bmatrix}, \\ \mathbf{C} &= [\mathbf{C}_K \quad \mathbf{C}_C], \quad \mathbf{D} = [\mathbf{0}^{(4 \times 2)}], \end{aligned} \quad (8.21)$$

where $\mathbf{B}_{K1}$ and $\mathbf{B}_{K2}$ corresponds to the first and second column of the $\mathbf{B}_K$ matrix respectively. The state vector is $\mathbf{x} = [\psi, x, y, e_1, e_2, \dot{\delta}, \dot{\phi}, \phi, \delta]^\top$, the input $\mathbf{u} = [v, \phi^*]$, and the output $\mathbf{y} = [\psi, x, y, \phi, \delta]^\top$.

Consider a reference trajectory $\Gamma(t) = [\psi_r(t), x_r(t), y_r(t)]^\top$. A new reference point on the path is extracted at each sampling interval $T_s = 0.1s$, and the $H_p - 1$ subsequent reference points are extracted which enables the bicycle to look ahead H_p points. However, as the linear model in Eq. (8.21) is linearised at its equilibrium state, i.e $\mathbf{x} = [\mathbf{0}]$, the model will more accurately describe the system close to the its equilibrium. Therefore, the measured output is set to zero for all states, and instead the difference in the bicycle frame, $\mathcal{W}$, is used as the reference output for the MPC with lean and steer angle set to zero. The difference are computed as:

$$\begin{aligned} \Delta\psi &= \psi_r - \psi \\ \Delta x &= \cos(\psi - \Delta\psi)(x_r - x) + \sin(\psi - \Delta\psi)(y_r - y) \\ \Delta y &= -\sin(\psi - \Delta\psi)(x_r - x) + \cos(\psi - \Delta\psi)(y_r - y), \end{aligned} \quad (8.22)$$

and the input reference to the MPC is $\mathbf{r} = [\Delta\psi, \Delta x, \Delta y, 0, 0]^\top$. The quadratic

cost function, optimised by the MPC, can now be formulated as:

$$\begin{aligned}
 \min_{\mathbf{u}_k} \quad & \sum_{k=0}^{H_p} \|\mathbf{y}_k - \mathbf{r}_k\|_{\mathbf{Q}}^2 + \sum_{k=0}^{H_u} \|\mathbf{u}_k - \mathbf{u}_{r_k}\|_{\mathbf{R}}^2 + \|\Delta \mathbf{u}_k\|_{\mathbf{S}}^2 \\
 \text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \\
 & \mathbf{y}_k = \mathbf{C}\mathbf{x}_k, \\
 & \mathbf{y}_{\min} \leq \mathbf{y}_k \leq \mathbf{y}_{\max} \quad k = 0, \dots, H_p, \\
 & \mathbf{u}_{\min} \leq \mathbf{u}_k \leq \mathbf{u}_{\max}, \\
 & \Delta \mathbf{u}_{\min} \leq \Delta \mathbf{u}_k \leq \Delta \mathbf{u}_{\max}
 \end{aligned} \tag{8.23}$$

where $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{S}$ are positive semi-definite weighting matrices penalizing the tracking error, control signals and control moves respectively. The control horizon is denoted H_u , $\mathbf{u}_r = [v, 0]^\top$ is the control reference signal, $\mathbf{u}$ is the control signal, $\mathbf{y}$ the output, and $\Delta \mathbf{u}$ represents the control move from one iteration to the next. The matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are given by the state-space model in Eq. (8.21) discretized using zero-order hold and a sampling time of $T_s = 0.1$ s. The design parameters for the MPC and their corresponding values are presented in Table 8.3. To estimate the states of the model, a Kalman filter is utilised and integral terms are used for estimating the output [24].

Table 8.3: MPC parameters and constraints.

Parameter	Value	Parameter	Value
$\mathbf{y}_{\min}$	$-\left[\pi, 50, 50, \frac{\pi}{6}, \frac{\pi}{3}\right]^\top$	$\mathbf{Q}$	$\text{diag}([5, 10, 5, 10, 0])$
$\mathbf{y}_{\max}$	$\left[\pi, 0.1, 50, \frac{\pi}{6}, \frac{\pi}{3}\right]^\top$	$\mathbf{R}$	$\text{diag}([0, 0])$
$\mathbf{u}_{\min}$	$\left[0.5v, \frac{-\pi}{6}\right]^\top$	$\mathbf{S}$	$\text{diag}([0.1, 0.1])$
$\mathbf{u}_{\max}$	$\left[1.5v, \frac{\pi}{6}\right]^\top$	H_u	4
$\Delta \mathbf{u}_{\min}$	$\left[-0.2, \frac{\pi}{3}\right]^\top$	H_p	10
$\Delta \mathbf{u}_{\max}$	$\left[0.2, \frac{\pi}{3}\right]^\top$		

8.5 Results

To evaluate the proposed control system, two different reference trajectories are considered. Instead of restricting the path to a circular or straight path as in [7], or a sinusoidal as in [8], a realistic path is considered in this paper. Using OpenStreetMap a go-kart track is exported to the Driving Scenario Designer in Matlab. This allows for a more realistic behaviour of the bicycle since cyclists

typically manoeuvre both straights and curves. The z-coordinates of the track is set to zero, *i.e.*, the bicycle is riding on a flat horizontal plane. The centerline of the track is extracted and used as the reference path for the bicycle. The second reference trajectory, which is much shorter than the go-kart track, consists of a few narrow curves and is a more challenging track. For both trajectories, six different nominal velocities: 10, 12, 14, 16, 18, and 20km/h are considered. On the short track, the simulations are repeated ten times for each nominal velocity. The model in Eq. (8.21) used for the MPC is updated for each nominal velocity, however, the PID parameters reported in Table 8.2 remains the same for all simulations.

To evaluate the performance of the proposed system, the Mean Squared Error (MSE) is utilised and computed as:

$$MSE = \frac{\sum_{k=0}^N (\|\mathbf{x}_k^{\text{bike}} - \mathbf{x}_{k-1}^{\text{ref}}\|^2) T_s}{t} \quad (8.24)$$

where $\mathbf{x}^{\text{bike}}$ and $\mathbf{x}^{\text{ref}}$ are the x and y coordinates of the bicycle and the reference trajectory respectively, $T_s = 0.1\text{s}$ is the sampling time, and t is the time it takes for the bicycle to go from its start position to its goal position. Since the reference trajectory $\mathbf{x}^{\text{ref}}$ should be ahead of the bicycle, the previous reference point is considered, *i.e.*, $\mathbf{x}_{k-1}^{\text{ref}}$. Moreover, to measure the maximum difference between the reference trajectory and the bicycle trajectory, the Hausdorff distance [25] is utilised. The Hausdorff distance can be used to measure the similarity of two parametrised curves, A and B , as:

$$d_H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b) \right\}, \quad (8.25)$$

where $d(a, b)$ is the Euclidian distance between the points a and b . On the short track, the standard deviation and the mean of the ten repetitions for each nominal velocity are computed.

8.5.1 Simulation setup

The ordinary-sized male bicycle in [21] was dismantled and each component was weighed, measured, and designed in SolidWorks³ and imported to Adams, a multibody dynamics simulation software. In Adams, the steering motor and the propulsion motor are defined as general point motions around the steering and rear-wheel axis respectively. Furthermore, the interaction between the

³<https://www.solidworks.com/>

wheels and the ground is modelled as a Coulomb friction force with the dynamic and static friction coefficients $\mu_d = \mu_s = 0.7$, a stiction transition velocity of 0.2m/s, and friction transition velocity of 1m/s. The bicycle, as visualised in Adams, is presented in Fig. 8.4.

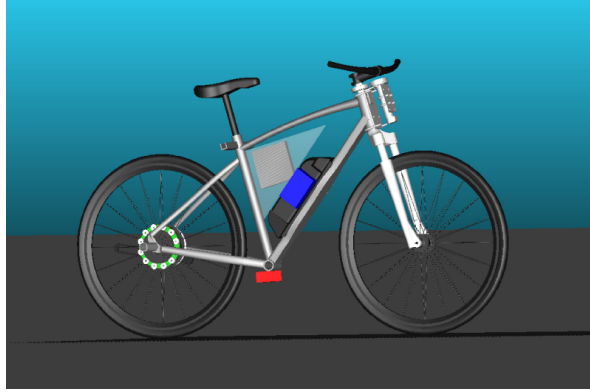


Figure 8.4: The instrumented bicycle designed in SolidWorks and imported to Adams is used as the plant in the co-simulation between Adams and Matlab.

The nonlinear bicycle model is then exported as a plant to Matlab Simulink, with the input $\mathbf{u} = [v, \delta]^\top$ and the output $\mathbf{y} = [\psi, x, y, v, \delta, \varphi]^\top$. The outer path tracking loop has a sampling time of $T_{s,outer} = 0.1$ s, while the inner stabilisation loop is set up with the sampling time $T_{s,inner} = 0.01$ s. To simulate the bicycle riding on uneven terrain, a disturbance $d \sim \mathcal{N}(0\text{rad/s}, 0.701^2\text{rad/s})$ is acting on the steering velocity input. Moreover, the lean angle measurement noise is an additive white Gaussian noise $n \sim \mathcal{N}(0\text{deg}, 10^{(-3/2)^2}\text{deg})$ and added to the lean angle measurements as shown in Fig. 8.5. The variance of the noise is estimated using the Inertial Measurement Unit (IMU) in [21] placed on a flat surface. The data from the IMU is collected over 30minutes and repeated three times.

8.5.2 Simulation results

The reference trajectory and the bicycle trajectory riding on the go-kart track for each of the nominal velocities are presented in Fig. 8.6. For the short track, the simulation is repeated ten times for each nominal velocity. The mean and standard deviation for the ten repetitions are presented in Fig. 8.7 and in Fig. 8.8 for the MSE and Hausdorff distance respectively. Note that at the nominal velocity of 18km/h only one repetition completed the trajectory (highlighted with a red asterisk in the figures). At 20km/h the bicycle fell over before reach-

ing the finish line in all ten repetitions. In Fig. 8.9 the short track trajectory is presented for the nominal velocity of 14km/h⁴. The variance, due to noise and disturbances, between the ten repetitions is represented in blue, and the reference path is highlighted in red.

8.5.3 Discussion

The results in Fig. 8.6 clearly shows that the proposed system is able to follow a trajectory in a real-life scenario at varying velocities. An MSE of 4.6cm is the highest value computed for the go-kart track and is obtained with the nominal velocity of 20km/h as seen in Fig. 8.7. The Hausdorff distances in Fig. 8.8 are all located at the narrow curves for all velocities, which indicates that the tracking performance is better for straights and wide curves. The Hausdorff distance and the MSE both increase slightly at the higher velocities of 18km/h and 20km/h. Since riding a bicycle at low speeds demands higher steering actuation compared to riding the same bicycle at higher speeds [26], the disturbance induced in the input steering signal will also have a greater impact on the performance at higher velocities

The short track in Fig. 8.9 is a more challenging track with narrow s-curves and short straights compared to the go-kart track in Fig. 8.6 or the tracks considered in [7, 12]. As a result, both the MSE and the Hausdorff distance in Fig. 8.7 and Fig. 8.8 are increased compared to MSE and Hausdorff distances at the go-kart track. For the short track, the velocity plays an important role in path tracking performance. The results up till 14km/h is consistent and with small variations, this is also highlighted in Fig. 8.9 for 14km/h. However, at higher velocities, the curves are too narrow and both the mean and standard deviation of the MSE and the Hausdorff distance at 16km/h is considerably higher

⁴<https://www.youtube.com/watch?v=Wq6Sv0X7erA>

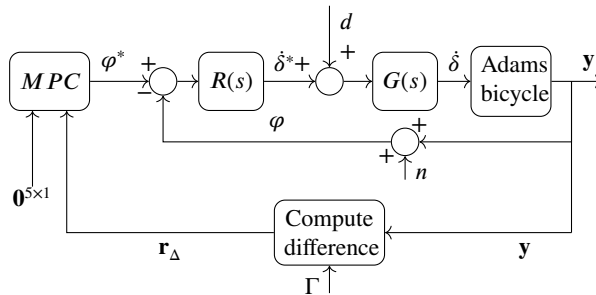


Figure 8.5: Simulation setup of the control system.

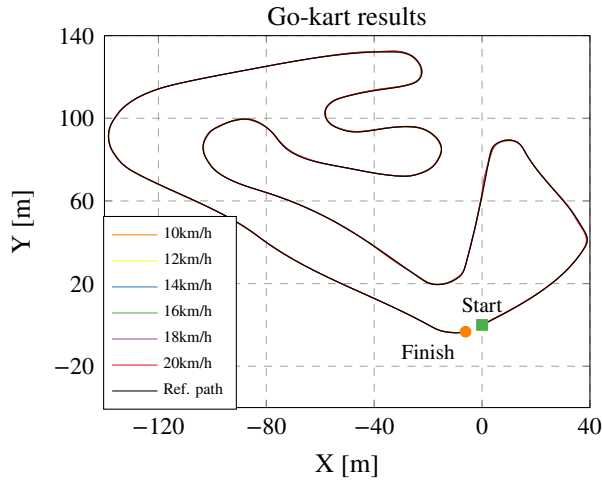


Figure 8.6: The Adams bicycle riding on a go-kart track.

compared to the low velocities. Above 16km/h, all except one simulation (at 18km/h) failed due to the bicycle falling over. The majority of falls takes place at the s-curve, either at the first curve or at the beginning of the second curve. When recovering the lean angle from the initial curve, the steering actuation is too aggressive in the opposite direction which causes the fall. One possible way to address this problem is with a longer prediction and control horizon, however, this would also increase the computational time. Introducing braking to the bicycle also needs to be investigated.

8.6 Conclusion

In this paper, an MPC is used to address the trajectory tracking problem for an autonomous bicycle. A point-mass model is used to model the bicycle, and the steering dynamics is obtained through a step response matching procedure. To balance the bicycle, a PID controller regulates the steering velocity. The PID parameters are computed by formulating the stabilisation of the bicycle as a nonlinear optimisation problem, solved by means of PSO. The bicycle model and the inner control loop are included in a prediction model, and an MPC is formulated for trajectory tracking. The balancing and path tracking capabilities of the autonomous bicycle are demonstrated in numerous co-simulations between Matlab and Adams where two different reference trajectories are considered. The Mean Squared Error and the Hausdorff distance is used to evaluate the path tracking performance. The results show that the riderless bicycle suc-

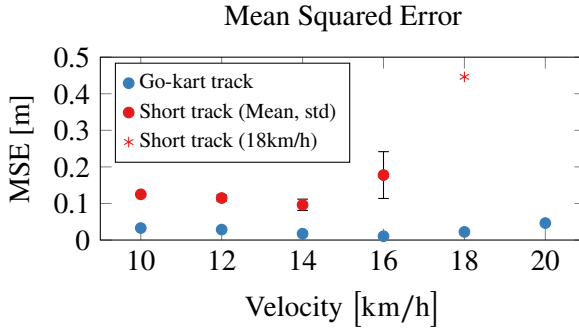


Figure 8.7: The Mean Squared Error between the reference path and the bicycle paths at each nominal velocity. The blue dots correspond to the MSE and computed for the go-kart track. The red dots and error bars correspond to the mean and standard deviation of the MSE for ten runs at each nominal velocity on the short track. The red asterisk marks the result for the nominal velocity of 18km/h, only one out of ten repetitions reached the final position. In the case of 20km/h the finish line was never reached in any of the ten repetitions.

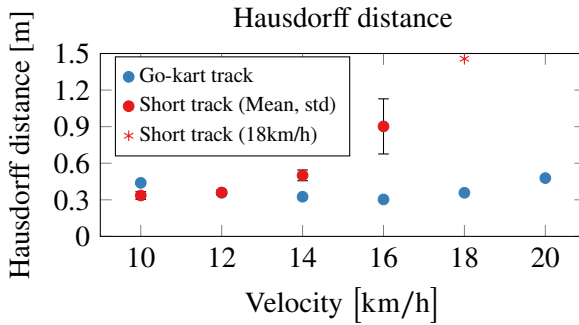


Figure 8.8: The Hausdorff distance at respective velocity for the long go-kart track is represented by the blue dots. Similarly, the red dots represent the mean Hausdorff distance, with error bars indicating the standard deviation, when riding on the short track with narrow curves. At the nominal velocity of 18km/h, only one repetition completed the whole track and the Hausdorff distance for this repetition is marked with the red asterisk. At 20km/h the finish position was never reached.

cessfully can balance and follow both reference trajectories in a range of velocities. For further evaluation of the system and to obtain experimental results, the MPC will be considered for implementation on an instrumented bicycle. However, a prerequisite for path tracking performance is reliable localisation, thus the localisation problem of a bicycle needs to be investigated first.

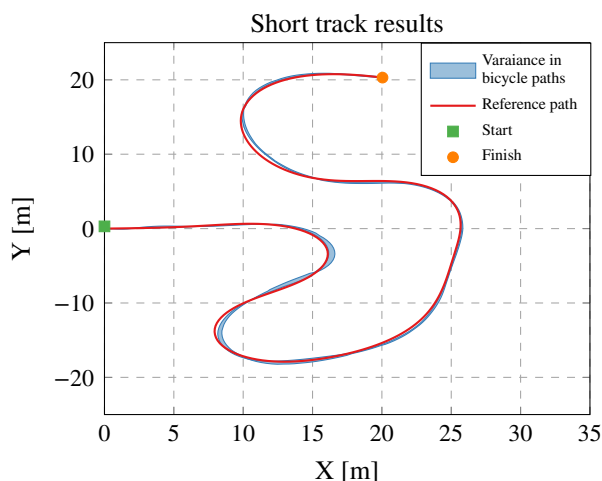


Figure 8.9: Simulations results for the co-simulation on the short track at the nominal velocity of 14km/h. The simulations are repeated ten times and the variance in bicycle path for these ten repetitions are highlighted in blue.

8.7 Acknowledgements

This work is part of a research project between Mälardalen University, Chalmers University, Volvo Cars, AstaZero, and Cycleurope and it is funded by Vinnova. The work is partly funded by Eskilstuna kommun och Eskilstuna Fabriksförening.

Bibliography

- [1] K. J. Åström, R. E. Klein, and A. Lennartsson, "Bicycle dynamics and control: adapted bicycles for education and research," *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, 2005.
- [2] A. L. Schwab and J. P. Meijaard, "A review on bicycle dynamics and rider control," *Vehicle System Dynamics*, vol. 51, no. 7, pp. 1059–1090, 2013.
- [3] N. C. Sanchez, L. A. Pastor, and K. Larson, "Autonomous bicycles: A new approach to bicycle-sharing systems," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, 2020.
- [4] P. Fairley, "Self-driving cars have a bicycle problem [news]," *IEEE Spectrum*, vol. 54, no. 3, pp. 12–13, 2017.

- [5] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.
- [6] N. H. Amer, H. Zamzuri, K. Hudha, and Z. A. Kadir, "Modelling and control strategies in path tracking control for autonomous ground vehicles: a review of state of the art and challenges," *Journal of intelligent & robotic systems*, vol. 86, no. 2, pp. 225–254, 2017.
- [7] 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.
- [8] T.-K. Dao and C.-K. Chen, "Path-tracking control of a riderless bicycle via road preview and speed adaptation," *Asian Journal of Control*, vol. 15, no. 4, pp. 1036–1050, 2013.
- [9] F. Whipple, "Stability of the motion of a bicycle," *Quarterly Journal of Pure and Applied Mathematics*, vol. 30, pp. 312–348, 1899.
- [10] 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, pp. 148–151 vol.1, 1994.
- [11] N. H. Getz and J. E. Marsden, "Control for an autonomous bicycle," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1397–1402, 1995.
- [12] J. Yi, D. Song, A. Levandowski, and S. Jayasuriya, "Trajectory tracking and balance stabilization control of autonomous motorcycles," in *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2583–2589, IEEE, 2006.
- [13] J. Yi, Y. Zhang, and D. Song, "Autonomous motorcycles for agile maneuvers, part i: Dynamic modeling," in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, pp. 4613–4618, 2009.
- [14] J. D. Kooijman, J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, "A bicycle can be self-stable without gyroscopic or caster effects," *Science*, vol. 332, no. 6027, pp. 339–342, 2011.
- [15] J. He, M. Zhao, and S. Stasinopoulos, "Constant-velocity steering control design for unmanned bicycles," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 428–433, 2015.

- [16] Y. Zhang, P. Wang, J. Yi, D. Song, and T. Liu, "Stationary balance control of a bikebot," in *International Conference on Robotics and Automation (ICRA)*, pp. 6706–6711, 2014.
- [17] S. Miah, E. Milonidis, I. Kaparias, and N. Karcianas, "An innovative multi-sensor fusion algorithm to enhance positioning accuracy of an instrumented bicycle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1145–1153, 2020.
- [18] 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.
- [19] R. Frezza, A. Beghi, and A. Saccon, "Model predictive for path following with motorcycles: application to the development of the pilot model for virtual prototyping," in *2004 43rd IEEE Conference on Decision and Control (CDC)*, vol. 1, pp. 767–772, IEEE, 2004.
- [20] M. Zhao, S. Stasinopoulos, and Y. Yu, "Obstacle detection and avoidance for autonomous bicycles," *IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1310–1315, 2017.
- [21] T. Andersson, N. Persson, A. Fattouh, and M. C. Ekström, "A loop shaping method for stabilising a riderless bicycle," in *2019 European Conference on Mobile Robots (ECMR)*, pp. 1–6, 2019.
- [22] A. O'Dwyer, *Handbook of PI and PID Controller Tuning Rules*. Imperial College Press, 3rd ed., 2009.
- [23] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [24] A. Zenere and M. Zorzi, "Model predictive control meets robust kalman filtering," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3774–3779, 2017.
- [25] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*, vol. 317. Springer Science & Business Media, 2009.
- [26] J. K. Moore, J. D. Kooijman, A. L. Schwab, and M. Hubbard, "Rider motion identification during normal bicycling by means of principal component analysis," *Multibody System Dynamics*, vol. 25, no. 2, pp. 225–244, 2011.

Chapter 9

Paper C

Direct Data-Driven Control Design for Autonomous Bicycles

Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.

In International Conference on Automation Science and Engineering (CASE),
2024

Abstract

In this work, we address the problem of balancing an autonomous bicycle using direct data-driven control. Firstly, we demonstrate that a direct implementation of data-driven approaches may not guarantee reliable performance, and is highly dependent on how the parameters are selected. To address this issue, we make the reasonable assumption that we have access to some inaccurate information about the system. We use this inaccurate information to design a feedback linearization, based on a simplified point mass model of the bicycle, which does not accurately represent the dynamics of the system. Next, we suggest an inner and outer-loop control strategy. In the inner loop, we implement the aforementioned feedback linearization controller. Subsequently, in the outer loop, we consider the combination of the autonomous bicycle and the feedback controller as a black box, and we design a direct data-driven controller from acquired data. We use a SolidWorks model of a real autonomous bicycle to evaluate the performance of our proposed control approach and to compare it with the direct data-driven controller design derived from acquired data of the bicycle without feedback linearization. The results show that our proposed strategy significantly improves the performance and makes the data-driven control approach more reliable across a broader range of parameter choices compared to a data-driven controller designed based on data from the system without feedback linearization. Finally, we show that introducing an additional integral-like state further enhances the system's performance.

9.1 Introduction

The seminal study by Willems et al. [1] revealed a paradigm-shifting concept that has immense potential for revolutionizing control system design. The study demonstrated that a finite set of system trajectories, generated through persistently exciting input, has the capability to represent the complete behavior of a controllable linear system. This finding implies that it is possible to describe controllable time-invariant linear systems solely based on finite historical data, eliminating the need for traditional state-space or transfer function representations. Such a breakthrough has sparked great interest within the control system community, as it offers a promising avenue to streamline control system design by potentially circumventing the costly and time-consuming step of system identification or complex first principle modeling. For instance, [2, 3, 4] propose predictive data-driven approaches, while [5, 6, 7] focus on optimal data-driven control methods and [8, 9, 10] present robust and nonlinear control design.

However, the Fundamental Lemma mentioned in [1], which serves as the basis for data-driven approaches, only applies to linear systems, while, in practical scenarios, most systems exhibit inherent nonlinearity. Nonlinear data-driven control approaches, introduced up to date, rely on techniques such as linearization [8, 9] or Koopman operators shifting [10, 11]. As a result, the performance of these approaches in practical implementations relies on the accuracy of the linear representation of the nonlinear system.

The case study in this article is an autonomous bicycle. Bicycle dynamics are nonlinear and they exhibit an unstable behavior at low velocities, making it a good illustration of many interesting issues in control design. Various model-based methods have been proposed for balancing an autonomous bicycle, including Sliding Mode Controller (SMC) [12], state feedback controller [13], Feedback Linearization (FL) controller [14], and by combining FL with LQR and LQI controllers [15], to name a few. In this paper, we investigate how to design a data-driven control approach for balancing a bicycle.

Figure 9.1 shows the results obtained by directly applying to the bicycle the data-driven control method for nonlinear systems proposed by De Persis and Tesi [8], referred to as DPT¹ throughout this paper. The controller aims to track a reference lean angle by actuating the steering velocity. The figure shows the Integrated Squared Error (ISE)² between the reference and the measured lean angle, as a function of the number of samples, T , used to design the DPT

¹The name comes from the initial of the authors of [8].

²Note that in the figure the ISE is clamped to 1 and marked with $\star$ when the resulting design is unstable.

controller. Moreover, the figure shows how the results change as a function of the design parameter γ required for the DPT controller. This design parameter defines the entity of the allowable disturbance, which includes the entity of the system's disturbance and of the mismatch between the nonlinear and linearized system—more details in [8]. The historical input and output data were sampled at a rate of 100Hz and were obtained by exciting the system input with $u(t) = \frac{36}{\pi} \cos((9 - 0.4t)t)$ deg/s while the bicycle was riding at a constant velocity of 8km/h.

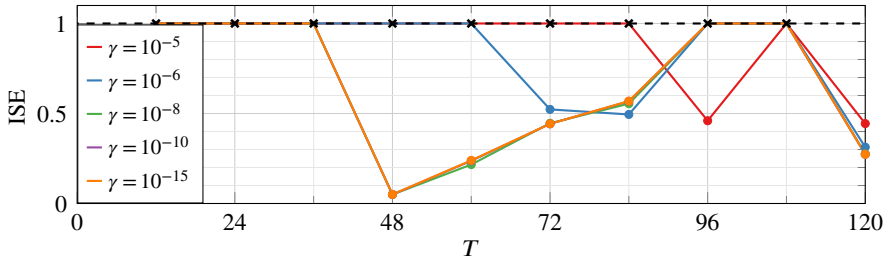


Figure 9.1: Integrated squared error (ISE) for an autonomous bicycle tracking a reference lean angle using the DPT controller in simulations with varying parameters. ISE values above 1 are considered unstable and are truncated.

As can be observed from the initial results depicted in Figure 9.1, the performance of the controller depends in a nontrivial way on the sampling size T , and the parameter γ , and it seems difficult to draw any conclusions on how these parameters should be chosen for reliable performance. However, in most practical systems, valuable information about some of the system's dynamics or parameters is available, which could be exploited in the control design. This information might not be sufficiently precise or reliable for designing a controller based solely on it, but it can still be precise enough to mitigate the nonlinear behaviors of a practical system and make the adoption of data-driven approaches smoother.

9.1.1 Statement of Contributions

The main contributions of this article are as follows:

- We apply the DPT control for nonlinear systems introduced in [8] to an accurately mechanically modeled autonomous bicycle designed in *Solid-Works*.
- We demonstrate that the minimum number of acquired samples required to design data-driven controllers, as determined in idealized studies, may

not be sufficient in practical implementations when dealing with nonlinear unstable systems.

- We enhance the performance of data-driven control for nonlinear systems by incorporating prior information about the system and adding an inaccurate FL controller to the control design.
- We show that introducing an additional integral-like state can further enhance the system's performance.

9.1.2 Organization

The rest of the paper is structured as follows, first, in Section 9.2 the control design and fundamental concepts of data-driven control are presented. The control setting and the simulation setup are discussed in Section 9.3 followed by the results in Section 9.4. Finally, concluding remarks and future work are given in Section 9.5.

9.1.3 Notation

Throughout this paper, $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{R}$ represent the sets of integers, positive integers, and real numbers, respectively. Unless clearly specified otherwise, scalars are denoted as x , while $\mathbf{x}$, $\mathbf{x}$, and $\mathbf{X}$ denote a (column) vector, a signal, and a matrix, respectively. Given a signal $\mathbf{s} : \mathbb{Z} \rightarrow \mathbb{R}^d$, of size d , we denote the trimmed $\mathbf{s}$ within the time interval $[k, k + T]$, where $T \in \mathbb{N}$ and $k \in \mathbb{Z}$, by $\mathbf{s}_{\{k:k+T\}}$, i.e.

$$\mathbf{s}_{\{k:k+T\}} \triangleq \{\mathbf{s}(k), \dots, \mathbf{s}(k + T)\}.$$

Furthermore, we denote the matrix that contains T successive samples of $\mathbf{s}$ starts from k by $\mathbf{S}_{[k,T]}$, i.e.

$$\mathbf{S}_{[k,T]} \triangleq [\mathbf{s}(k), \dots, \mathbf{s}(T + k - 1)].$$

Furthermore, $\text{diag}[x_1, \dots, x_n]$ denotes a diagonal matrix with the elements $x_1, \dots, x_n$ on its diagonal. Finally, $\text{tr}(\mathbf{X})$, $\text{rank}(\mathbf{X})$, and $\mathbf{X}'$ denote the trace, rank, and transpose of matrix $\mathbf{X}$, respectively.

9.2 Direct Data-Driven Control

This section introduces some preliminaries for designing controllers based solely on the historical input and output data of a system. Next, the DPT controller [8] is presented.

Consider a discrete linear time-invariant system

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \quad k \in \mathbb{N}, \quad (9.1)$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$ represent the state and input vectors, respectively. We assume that T historical samples of $\mathbf{u}$ are accessible, where $T \in \mathbb{N}$. In other words, the signal $\mathbf{u}_{\{0:T-1\}}$ is known. We recall the fundamental property introduced by Willems et al. [1] that establishes a necessary condition for the T -long historical data of $\mathbf{x}(k)$ and $\mathbf{u}(k)$ to effectively represent the dynamical system described by (9.1). Let us begin by defining the concept of persistently exciting inputs.

Definition 9.2.1 ([1]). *The signal $\mathbf{u}_{\{0:T-1\}} \in \mathbb{R}^m$ is said to be persistently exciting of order l if*

$$\mathcal{V}_0 = \begin{bmatrix} \mathbf{u}(0) & \mathbf{u}(1) & \cdots & \mathbf{u}(T-l) \\ \mathbf{u}(1) & \mathbf{u}(2) & \cdots & \mathbf{u}(T-l+1) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}(l-1) & \mathbf{u}(l) & \cdots & \mathbf{u}(T-1) \end{bmatrix},$$

has full rank ml .

■

Definition 9.2.1 implies that signal $\mathbf{u}$ must be sufficiently long, namely

$$T \geq (m+1)l - 1, \quad (9.2)$$

to be persistently exciting. The following lemmas play a key role in the data-driven approaches used in this paper.

Lemma 9.1 ([1]). *If the system (9.1) is controllable and $\mathbf{u}_{\{0:T-1\}}$ is persistently exciting of order $n+1$, then*

$$\text{rank}(\mathcal{W}_0) = n + m,$$

where

$$\mathcal{W}_0 := \begin{bmatrix} \mathbf{U}_{[0,T]} \\ \mathbf{X}_{[0,T]} \end{bmatrix}.$$

■

In [8], De Persis and Tesi transformed the state space representation in (9.1) into a form that exclusively relies on historical data.

Lemma 9.2 ([8]). *If $\mathbf{u}_{\{0:T-1\}}$ is persistently exciting, then the system (9.1) with a state feedback $\mathbf{u} = \mathbf{K}\mathbf{x}$ can be represented by:*

$$\mathbf{x}(k+1) = \mathbf{X}_{[1,T]} \mathcal{G}_{\mathbf{K}} \mathbf{x}(k), \quad (9.3)$$

where $\mathcal{G}_{\mathbf{K}}$ is a $T \times n$ matrix that satisfies

$$\begin{bmatrix} \mathbf{K} \\ \mathbf{I}_n \end{bmatrix} = \mathcal{W}_0 \mathcal{G}_{\mathbf{K}}, \quad (9.4)$$

and as a result

$$\mathbf{u}(k) = \mathbf{U}_{[0,T]} \mathcal{G}_{\mathbf{K}} \mathbf{x}(k). \quad (9.5)$$

■

From (9.3) we notice that in the closed-loop discrete system (9.1) under state-feedback control $\mathbf{u} = \mathbf{K}\mathbf{x}$ it holds,

$$\mathbf{A} + \mathbf{B}\mathbf{K} = \mathbf{X}_{[1,T]} \mathcal{G}_{\mathbf{K}}. \quad (9.6)$$

Thus, one can find an appropriate $\mathcal{G}_{\mathbf{K}}$ so that $\mathbf{X}_{[1,T]} \mathcal{G}_{\mathbf{K}}$ satisfies the classical Lyapunov stability condition. In [8], De Persis and Tesi proved that if $\text{rank}(\mathcal{W}_0) = n + m$, then $\mathbf{K}$ defined by

$$\mathbf{K} = \mathbf{U}_{[0,T]} \mathbf{Q} (\mathbf{X}_{[0,T]} \mathbf{Q})^{-1}, \quad (9.7)$$

asymptotically stabilizes (9.1) by any matrix $\mathbf{Q}$ that satisfies

$$\begin{bmatrix} \mathbf{X}_{[0,T]} \mathbf{Q} & \mathbf{X}_{[1,T]} \mathbf{Q} \\ \mathbf{Q}' \mathbf{X}'_{[1,T]} & \mathbf{X}_{[0,T]} \mathbf{Q} \end{bmatrix} > 0. \quad (9.8)$$

9.2.1 DPT Control for Nonlinear Systems

Based on the results of the data-driven state-feedback controller in (9.7), the authors in [8] investigated the conditions under which the linearization around the equilibrium point can be implemented to drive a data-driven approach for controlling smooth nonlinear systems. Consider a nonlinear system defined by

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)), \quad (9.9)$$

suppose that f is continuous and differentiable and $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$, which serves as the target for moving the system's states towards, is known. By linearization around this equilibrium point, one can rewrite (9.9) as

$$\Delta \mathbf{x}(k+1) = \mathbf{A} \Delta \mathbf{x}(k) + \mathbf{B} \Delta \mathbf{u}(k) + \mathbf{d}(k), \quad (9.10)$$

where $\Delta \mathbf{x}(k) := \mathbf{x}(k) - \bar{\mathbf{x}}$, $\Delta \mathbf{u}(k) := \mathbf{u}(k) - \bar{\mathbf{u}}$ and

$$\mathbf{A} := \left. \frac{\partial f}{\partial \mathbf{x}} \right|_{(\mathbf{x}, \mathbf{u})=(\bar{\mathbf{x}}, \bar{\mathbf{u}})}, \quad \mathbf{B} := \left. \frac{\partial f}{\partial \mathbf{u}} \right|_{(\mathbf{x}, \mathbf{u})=(\bar{\mathbf{x}}, \bar{\mathbf{u}})}. \quad (9.11)$$

In [8], it is proved that if the matrices

$$\begin{bmatrix} \Delta \mathbf{U}_{[0,T]} \\ \Delta \mathbf{X}_{[0,T]} \end{bmatrix}, \quad \Delta \mathbf{X}_{[1,T]},$$

have full rank and $\mathbf{d}(k)$ is upper bounded such that there exists $\gamma > 0$ that satisfies

$$\mathbf{D}_{[0,T]} \mathbf{D}'_{[0,T]} \leq \gamma \Delta \mathbf{X}_{[1,T]} \Delta \mathbf{X}'_{[1,T]}, \quad (9.12)$$

then the state-feedback $\mathbf{K} = \Delta \mathbf{U}_{[0,T]} \mathbf{Q} (\Delta \mathbf{X}_{[0,T]} \mathbf{Q})^{-1}$, which has similar shape to (9.7), locally stabilizes the system by any $(\mathbf{Q}, \mu)$ that satisfies

$$\begin{bmatrix} \Delta \mathbf{X}_{[0,T]} \mathbf{Q} - \mu \Delta \mathbf{X}_{[1,T]} \Delta \mathbf{X}'_{[1,T]} & \Delta \mathbf{X}_{[1,T]} \mathbf{Q} \\ \mathbf{Q}' \Delta \mathbf{X}'_{[1,T]} & \Delta \mathbf{X}_{[0,T]} \mathbf{Q} \end{bmatrix} > 0, \quad (9.13)$$

$$\begin{bmatrix} \mathbf{I}_T & \mathbf{Q} \\ \mathbf{Q}' & \Delta \mathbf{X}_{[0,T]} \mathbf{Q} \end{bmatrix} > 0, \quad \frac{\mu^2}{(4+2\mu)} > \gamma.$$

Remark 1. In DPT, it is assumed that the exact dynamical system model is unknown. Consequently, the precise value of $\mathbf{D}_{[0,T]}$ in (9.12) remains unknown. Therefore, determining the lower bound of γ that satisfies (9.13) through mathematical means is not feasible. As a result, the tuning of γ in DPT must be conducted experimentally.

9.3 Autonomous Bicycle Control Design

The results depicted in Figure 9.1 show that the DPT controller in (9.13) may not always deliver reliable performance. Under certain parameter combinations, the system becomes unstable. Furthermore, the inherent instability of a bicycle at low velocities complicates the task of sampling input and output data. For example, when the bicycle leans to the left and is steered to the right, it quickly falls over. In contrast, steering to the left eventually stabilizes it and brings it back to an upright position. These characteristics render some input signals infeasible for acquiring data.

In this section, we make a reasonable assumption that some imprecise and simplified information about the system is available. We address both of the aforementioned challenges by implementing output FL [16], using the available inaccurate information to reduce the nonlinear behavior of the system and push

it toward linearity. Subsequently, to improve the closed-loop performance of the system compared to relying solely on FL or data-driven control, we design a data-driven controller for the system composed of the autonomous bicycle and the FL. Given that the nonlinear data-driven control approaches introduced up to date rely on techniques such as linearization [8, 9] or Koopman operators shifting [10, 11], this modification can offer performance improvements by reducing the nonlinearities of the system. Additionally, this section provides details on the multibody simulation of the bicycle and an explanation of how the data was collected.

9.3.1 Control Design

Consider the following point mass model of a bicycle [17]:

$$\begin{aligned} h^2 \ddot{\varphi}(t)m = & gm \left(h \sin(\varphi(t)) + \frac{ca}{b} \sin(\lambda) \sigma(t) \cos(\varphi(t)) \right) - \\ & \left(1 - \frac{h}{b} \sigma(t) \sin(\varphi(t)) \right) \frac{h}{b} \sigma(t) \cos(\varphi(t)) v^2(t)m \\ & - \frac{ahm}{b} \cos(\varphi(t)) \left(\sigma(t) \dot{v}(t) - v(t) \omega_\sigma(t) \right), \end{aligned} \quad (9.14)$$

where h and a are the vertical and horizontal distances between the center of gravity and the contact point between the rear wheel and the ground, denoted P_1 in Figure 9.2. Gravity, wheelbase, mass, trail, and head angle are denoted g , b , m , c , and λ respectively. The curvature of the bicycle is represented by

$$\sigma(t) = \frac{\tan(\delta(t)) \sin(\lambda)}{\cos(\varphi(t))}, \quad (9.15)$$

where δ and φ represent the steering and lean angles respectively, and $\omega_\sigma(t) = \dot{\sigma}(t) \approx \dot{\delta}(t) \sin(\lambda)$. By assuming a constant forward velocity ($\dot{v} = 0$) and a vertical steering axis ($\lambda = 90^\circ, c = 0$), (9.14) can be simplified as:

$$\begin{aligned} \ddot{\varphi}(t) = & \frac{g}{h} \sin(\varphi(t)) + \frac{a}{bh} \cos(\varphi(t)) v \dot{\delta}(t) - \\ & \left(\frac{1}{bh} - \frac{1}{b^2} \tan(\delta(t)) \tan(\varphi(t)) \right) \tan(\delta(t)) v^2, \end{aligned} \quad (9.16)$$

with $\mathbf{x}(t) = [\varphi(t), \dot{\varphi}(t), \delta(t)]'$ and $u(t) = \dot{\delta}(t)$. Furthermore, the model assumes that the steering bar can be controlled without any delay, i.e., the steering dynamics are neglected. Although (9.16) is not full-state linearizable, it can be partially linearized by output FL.

Let $y(t) = \varphi(t)$, $\dot{y}(t) = \dot{\varphi}(t)$, and let the desired output and its time derivatives be denoted $y_d(t)$, $\dot{y}_d(t)$, and $\ddot{y}_d(t)$. If we choose $u = \frac{1}{p(x)}(w - f(x))$, where

$$\begin{aligned} f(x) &= -\left(\frac{1}{bh} - \frac{1}{b^2} \tan(x_3) \tan(x_1)\right) \tan(x_3) v^2, \\ &\quad + \frac{g}{h} \sin(x_1) \\ p(x) &= \frac{a}{bh} \cos(x_1) v, \\ w &= \ddot{y}_d(t) + k_1 (\dot{y}_d(t) - \dot{y}(t)) + k_2 (y_d(t) - y(t)). \end{aligned} \quad (9.17)$$

Then, by properly selecting k_1 and k_2 , the nonlinearities of the system are partially canceled.

Remark 2. *It is worth highlighting that the objective of FL in (9.17) is to push the nonlinear dynamics of the system toward linearity. The stability of the overall system will be achieved by the outer loop data-driven controller. Therefore, there is no necessity to investigate the stability of the internal dynamic or to select k_1 and k_2 in a way that stabilizes the observable dynamics. However, choosing $k_1 > 0$ and $k_2 > 0$ to satisfy the Routh–Hurwitz stability criterion of the linearized system makes the sampling phase smoother.*

Since (9.16) does not perfectly represent the actual system, applying (9.17) results in the presence of nonlinearities in the data sampled from the multibody model of the bicycle. Furthermore, perfect knowledge of the system and all the system's parameters is rarely available in practical applications. Therefore, we consider the mismatches, as listed in Table 9.1, between the parameters used in the multibody model and the control law. Throughout the paper, we choose the feedback gains in (9.17) to $k_1 = 10$ and $k_2 = 10$. Optimization of these parameters could improve the results, however, it is out of the scope of this paper.

As depicted in Figure 9.3, we propose the implementation of FL, introduced in (9.17), as the inner control loop responsible for mitigating the nonlinear behavior of the system through the control signal $\mathbf{u}_i(t)$. We consider our autonomous bicycle and the FL controller, depicted by the gray rectangle in the figure, as a new black box, for which we aim to design a data-driven controller. During the first T samples, we position the switch in (a) to enable the transmission of the persistently exciting signal $\mathbf{u}_{\{0:T-1\}}$ to the system as the control signal of the outer loop, $\mathbf{u}_o(t)$. Subsequently, we position the switch in (b) and apply the designed DPT state-feedback controller $\mathbf{u}_o(t) = \mathbf{u}_{DPT}(k) = \mathbf{K}\mathbf{x}(k)$

using (9.13). The feedback linearization control in the inner loop reduces the nonlinearities of the autonomous bicycle, bringing the overall system closer to linearity. Consequently, we anticipate that data-driven results should fit better, leading to improved outcomes compared to the results presented in Figure 9.1. It is important to notice that the control signal, $\mathbf{u}(t)$, is the summation of the inner and outer control signal, $\mathbf{u}(t) = \mathbf{u}_i(t) + \mathbf{u}_o(t)$.

9.3.2 Simulation setup

To evaluate our proposed control setting, we employ a multi-body simulation model of an instrumented bicycle [18]. The instrumented bicycle is initially designed using *SolidWorks* and subsequently imported into Mathworks *Sim-scape*. Two revolute joints connect the rear wheel and the front wheel to the bicycle's frame and fork, respectively, as depicted by 1 and 2 in Figure 9.2. Additionally, a third revolute joint connects the fork to the frame, highlighted as 3 in the figure, and is actuated using the proposed controller.

To model the steering dynamics, including the steering motor, in the multi-body simulation we employ a step response matching procedure [18]. The resulting transfer function, from the input, $\mathbf{u}(t)$, to the steering rate, $\dot{\delta}(t)$, is

$$H(s) = \frac{100 + s}{100}, \quad (9.18)$$

and is incorporated in series with the bicycle model, as illustrated in Figure 9.3. The gray rectangle in the figure illustrates the system that is treated as a black box when sampling data between the points $\mathbf{u}_o(k)$ and $\mathbf{x}(k)$. The switch in the figure represents the setup of the system when we sample data (a) and when the DPT control stabilizes the system in a feedback loop (b). The normal force between the tires and the ground is modeled with a stiffness of 10^6N/m , a damping coefficient of 10^3N/(m/s) , and a transition region width of 10^{-4}m . In addition, we utilize a smooth stick-slip method to model the friction between the tires and the ground. This method incorporates static and dynamic friction param-

Table 9.1: Actual and estimated parameters of the instrumented bicycle.

Geometric parameters				
Parameter	Symbol	Unit	Real value	Estimated value
CoG w.r.t. P_1 (x)	a	m	0.473	0.550
CoG w.r.t. P_1 (z)	h	m	0.515	0.700
Wheelbase	b	m	1.080	1.200



Figure 9.2: Instrumented bicycle with three revolute joints highlighted as 1, 2 and 3. P_1 represents the contact point between the rear wheel and the ground.

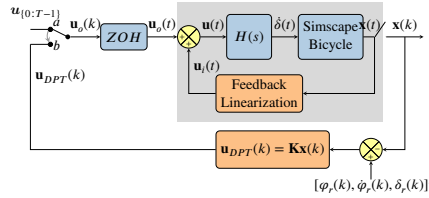


Figure 9.3: Block diagram of the control system that consists of an inner loop with a FL control producing $\mathbf{u}_i(t)$, and an outer loop DPT control that provides $\mathbf{u}_o(t)$ when the switch is in position (b). The DPT control is designed after T sampling steps, with the switch in position (a), by solving (9.13). The overall control input is $\mathbf{u}(t)$, where $\mathbf{u}(t) = \mathbf{u}_i(t) + \mathbf{u}_o(t)$.

eters of 0.9 and 0.75, respectively. The critical velocity, which determines the friction parameter to consider, is set to 10^{-3} m/s.

Let $\varphi(k)$ and $\delta(k)$ denote the bicycle lean and steer angles at time step k , respectively. The control signal, $\mathbf{u}_o(k)$ represents the control input to the system highlighted in gray in Fig. 9.3, and is utilized to stabilize the bicycle by steering in the direction of the lean angle. The simulation is initialized with zero steer and lean angle, i.e., $\delta(0) = 0$ and $\varphi(0) = 0$, along with a forward velocity of 8 km/h. It is worth noting that this velocity is below the so-called self-stabilization speed of the bicycle [19]. As a result, without proper actuation, the bicycle would inevitably fall over.

9.3.3 Data collection

We explore two scenarios, related to the states that we sampling data from in our study,

- *Scenario 1:* $\mathbf{x}(k) = [\varphi_e(k), \dot{\varphi}_e(k), \delta_e(k)]'$,
- *Scenario 2:* $\mathbf{x}(k) = [\varphi_e(k), \dot{\varphi}_e(k), \delta_e(k), \sum_{i=0}^k \varphi_e(i)]'$,

where $\varphi_e(k)$, $\dot{\varphi}_e(k)$ and $\delta_e(k)$ denote the difference between the lean angle, lean rate, and steering angle, with the desired values, respectively, i.e. $\varphi_e(k) = \varphi_r(k) - \varphi(k)$. Furthermore, $\sum_{i=0}^k \varphi_e(i)$ is the sum of the lean angle error. In the first scenario, as proposed for designing the model-based LQR controller in [18], we assume that the state vector consists of $[\varphi_e(k), \dot{\varphi}_e(k), \delta_e(k)]'$. It is important to note that we implicitly assume the order of the autonomous bicycle system to be 3 in this scenario. The exact order of the system is a crucial parameter, typically assumed to be known in published articles in the field of data-driven control, as authors often start their simulations based on a known mathematical model. However, practical systems generally have more actual states than their simplified mathematical models account for. These neglected states, along with disturbances in (9.9), which include both nonlinearities and actual system disturbances, can degrade the performance of the control system and lead to steady-state errors. To address this undesired behavior, we introduce an additional state, denoted as $\sum_{i=0}^k \varphi_e(i)$, in the second scenario. This approach draws inspiration from linear control theory, where proportional feedback of the integral of the error is commonly employed to compensate for steady-state errors [20].

Data were sampled at a rate of 100Hz from experiments conducted at a constant velocity of 8 km/h. Two datasets were collected, with the first dataset sampled from the unstable system without FL, i.e. $\mathbf{u}_i(t) = 0$. In this system, the input signal was a sinusoidal function, represented as $\mathbf{u}_{\{0:T-1\}} = \frac{36}{\pi} \cos((9 - 0.4t)t)$ deg/s. This choice was motivated by the highly unstable nature of a bicycle at low velocities, making it necessary to gather sufficient data to capture the primary dynamics of the bicycle.

In the second dataset, when the system is initially stabilized using FL, the restrictions on the input signal can be neglected. Instead, an input signal $\mathbf{u}_{\{0:T-1\}}$ sampled from a uniform distribution $\mathcal{U}(-1.5, 1.5)$ rad/s is utilized. For both datasets, we considered 120 samples, as shown in Figure 9.4, for control design. It is important to note that both of our proposed data collection settings ensure that the input signal remains persistently exciting, satisfying the condition in (9.2). The dataset from the unstable system was used to design

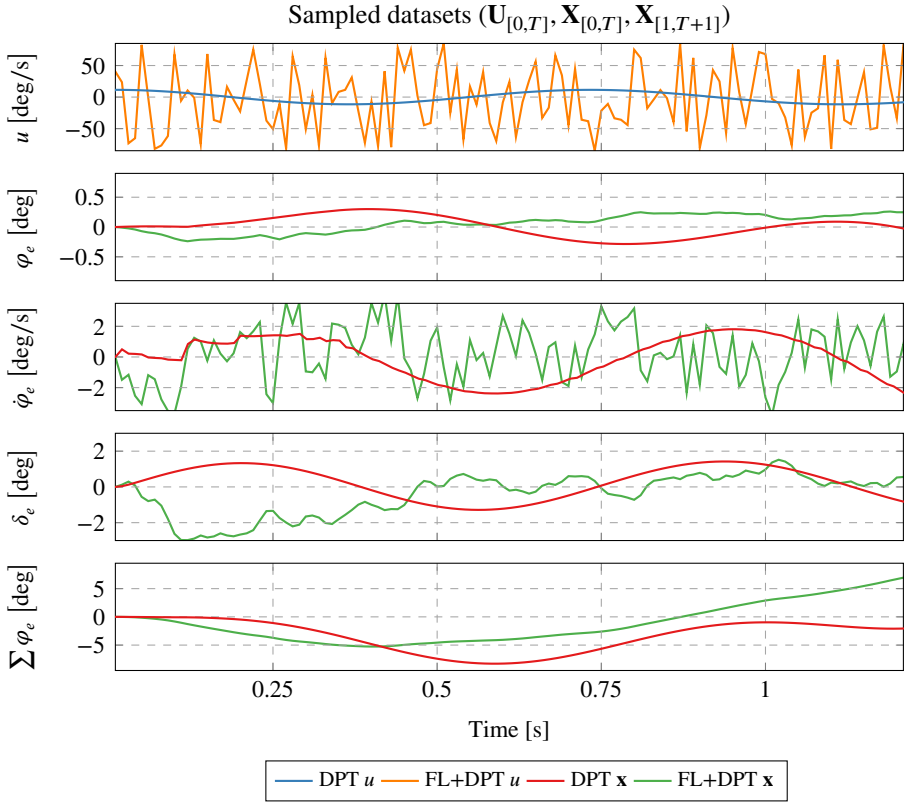


Figure 9.4: Collected data from the control signal, u , at the top followed by the four states φ_e , $\dot{\varphi}_e$, δ_e , $\sum \varphi_e$ respectively.

the DPT control, whilst the second dataset, from the stable system, was used to design the feedback linearization

DPT (FL+DPT) control.

9.4 Numerical Results

In the simulation, the bicycle maintains a constant forward velocity of 8 km/h and tracks a reference lean angle. The reference lean angle, represented by a black dashed line in Figure 9.5, consists of two sections. The first section features a sinusoidal pattern, $\varphi_r = \frac{36}{\pi} \sin(t)$ deg/s, followed by a section with a constant lean angle of zero degrees. Both $\dot{\varphi}$ and δ track a reference of zero.

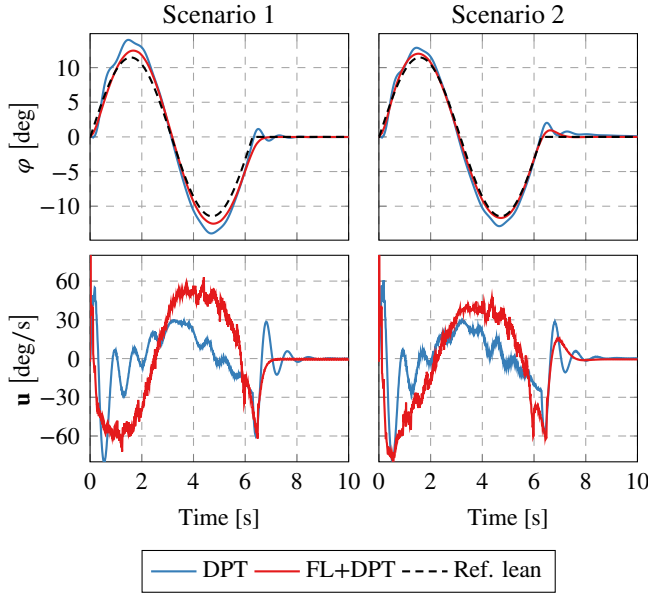


Figure 9.5: Tracking comparison of DPT and FL+DPT in Scenario 1 and Scenario 2. The roll angle is presented in the plots at the top of the figure, and the steering velocity input at the bottom of the figure.

To solve (9.13) for various combinations of γ and T , we employed CVX [21]. In Figure 9.5, a comparison is presented between the two different controllers and the two scenarios. The performance displayed in Figure 9.5 is based on 120 historical samples from Scenario 1 to design DPT and 120 historical samples from Scenario 2 to design FL+DPT³. For both controllers, we used $\gamma = 10^{-15}$. Table 9.2 reports the corresponding feedback gain matrices, defined by:

$$\begin{aligned} \text{Scenario 1:} \quad & \mathbf{K} = [k_{\varphi_e}, k_{\dot{\varphi}}, k_{\delta_e}], \\ \text{Scenario 2:} \quad & \mathbf{K} = [k_{\varphi_e}, k_{\dot{\varphi}}, k_{\delta_e}, k_{\Sigma \varphi_e}]. \end{aligned} \quad (9.19)$$

To investigate the impact of γ in (9.13) and T , an extensive simulation study was conducted for both Scenarios 1 and 2. The simulations encompassed five distinct values for γ in (9.13). For each scenario, and each value of γ , 10 different sample sets, $T_1, T_2, \dots, T_{10}$, was considered where each sample set increased by 12 samples, i.e., $T_1 = 12, T_2 = 24, \dots, T_{10} = 120$. The comparison of results, illustrated in Figure 9.6, focused on evaluating the Integrated

³https://youtu.be/z6_i4nLCoXk

Table 9.2: Resulting gains for the different controllers with $\gamma = 10^{-15}$ and $T = 120$.

	K			
	k_{φ_e}	$k_{\dot{\varphi}_e}$	k_{δ_e}	$k_{\Sigma \varphi_e}$
DPT				
Scenario 1	-43.8419	-1.4154	-3.7688	—
Scenario 2	-45.9845	-2.4332	-2.9955	-0.2471
FL+DPT				
Scenario 1	-140.0391	-10.2119	8.2020	—
Scenario 2	-165.1003	-11.6436	9.9205	-4.4944

Squared Error (ISE) between the lean angle and the reference lean angle, i.e.

$$\text{ISE}(k) = \sum_{i=0}^k (\varphi(i) - \varphi_r(i))^2. \quad (9.20)$$

ISE values larger than the set threshold of 1, are truncated to 1 and marked with an $\times$ in Figure 9.6. This indicates that the controller was not able to stabilize the bicycle or that the LMI in (9.13) were unfeasible, resulting in no feedback gains.

9.4.1 Discussion

The simulation results presented in Figure 9.5 demonstrate the effectiveness of both DPT and FL+DPT controllers. Remarkably, these controllers, relying solely on 120 samples of input and output data, not only balance an autonomous bicycle but also successfully track a nonzero reference lean angle with small deviations. The effectiveness of the proposed method of adding FL is clear when comparing the results in Figure 9.1 with Figure 9.6. By adding FL, based on a simplified model with errors in the parametric estimation, the performance in terms of the ISE is improved for almost all combinations of γ and T . Moreover, Figure 9.6, for $T \geq 48$ the performance variations between different values of γ are very small in both scenarios, thus, increasing the reliability compared to the results in Figure 9.1, where the DPT controller was design from data acquired from the unstable system. From Figure 9.6 it is also evident that the results from the initially stable system using FL, highlighted by the yellow line in the figure, are improved by adding DPT for $T \geq 48$. Furthermore, the results in Figures 9.5 and 9.6 clearly show that by adding a fourth state, $\sum_{i=0}^k \varphi_e(i)$, the performance further improves. In practical applications, finding the exact

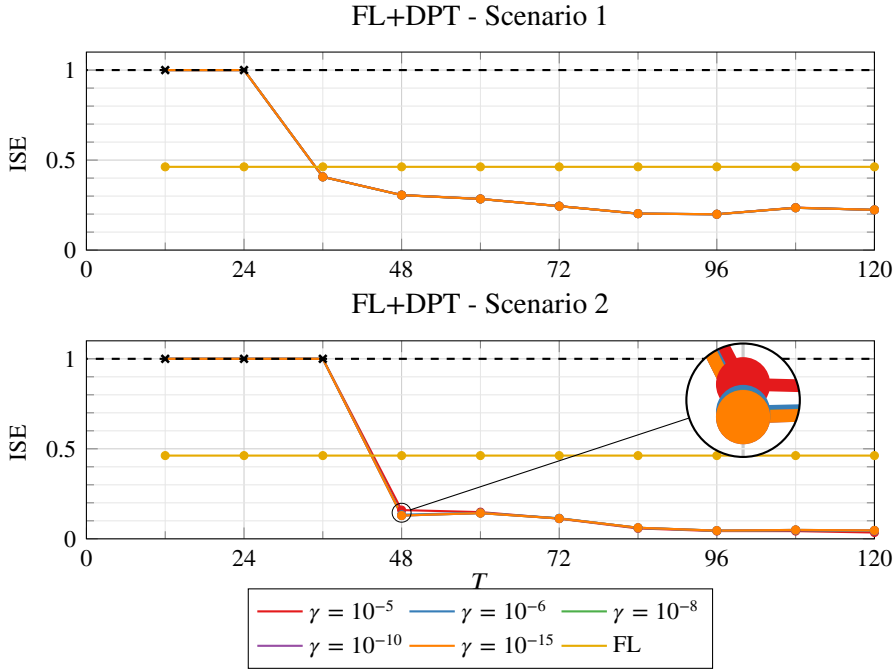


Figure 9.6: Performance, in terms of the ISE, by applying FL+DPT. The top plot shows the results from Scenario 1 when three states are considered in the sampling process and the feedback loop. Scenario 2, given in the bottom plot, shows the results when a fourth state, $\sum_{i=0}^k \varphi_e(i)$, is added to the control design.

order of a system can be challenging [22]. As a result, in practice, we typically measure or estimate a reduced-dimensional vector of states and implicitly assume that the system's order is limited to them. However, it is clear from the presented results that it can be sensible to modify the number of states.

Moreover, Lemma 9.1 along with (9.2) implies that only 7 samples in Scenario 1 and 9 samples in Scenario 2 might be sufficient to obtain a stabilizing controller. However, the outcomes illustrated in Figures 9.1 and 9.6 reveal that designing a data-driven feedback controller based on a limited set of historical samples can potentially compromise the stability of practical systems. As a result, it is advisable to refrain from relying solely on a small subset of historical samples that may not fully capture the nonlinear characteristics and complexities of a system.

9.5 Conclusion

In this paper, we utilize FL based on an imprecise and simplified point mass model of a bicycle. The primary objective is to mitigate nonlinearities while simultaneously achieving system stability, which simplifies the process of acquiring input/output data. Subsequently, we regard the bicycle, initially stabilized by FL, as a new system for which we have limited information. We treat it as a black box, and our goal is to develop a nonlinear direct data-driven controller for its regulation. To evaluate the performance of the approach, numerous simulations are conducted using a realistic Solidworks model of an autonomous bicycle. The results indicate that our proposed method enhances overall outcomes and makes the control approach more reliable across a broader spectrum of parameter choices.

Future research directions include conducting experimental evaluations of the proposed approaches on the instrumented bicycle. Additionally, the results presented in this paper can be extended by exploring the path-tracking capabilities of an autonomous bicycle using data-driven control. Moreover, recognizing the inevitability of the bicycle encountering various working conditions during its operation (e.g., different wind speeds, road conditions, etc.), and exploring dynamical data-driven designs that allow for online updating of the controller are beneficial.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Pietro Tesi for generously sharing the LMI solver codes used in his simulation in [8]. The availability of these codes facilitated our research and contributed to the successful implementation of our study.

Bibliography

- [1] 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.
- [2] D. Alpayo, F. Dörfler, and J. Lygeros, “An extended kalman filter for data-enabled predictive control,” *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 994–999, 2020.

- [3] J. Coulson, J. Lygeros, and F. Dörfler, “Distributionally robust chance constrained data-enabled predictive control,” *IEEE Transactions on Automatic Control*, vol. 67, no. 7, pp. 3289–3304, 2022.
- [4] L. Huang, J. Zhen, J. Lygeros, and F. Dörfler, “Robust data-enabled predictive control: Tractable formulations and performance guarantees,” *IEEE Transactions on Automatic Control*, vol. 68, no. 5, pp. 3163–3170, 2023.
- [5] Z. Yuan and J. Cortés, “Data-driven optimal control of bilinear systems,” *IEEE Control Systems Letters*, vol. 6, pp. 2479–2484, 2022.
- [6] C. D. Persis and P. Tesi, “On persistency of excitation and formulas for data-driven control,” in *Conference on Decision and Control (CDC)*, pp. 873–878, 2019.
- [7] G. Baggio, V. Katewa, and F. Pasqualetti, “Data-driven minimum-energy controls for linear systems,” *IEEE Control Systems Letters*, vol. 3, no. 3, pp. 589–594, 2019.
- [8] C. De Persis and P. Tesi, “Formulas for data-driven control: Stabilization, optimality, and robustness,” *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 909–924, 2020.
- [9] C. De Persis and P. Tesi, “Low-complexity learning of linear quadratic regulators from noisy data,” *Automatica*, vol. 128, p. 109548, 2021.
- [10] C. Ren, H. Jiang, C. Li, W. Sun, and S. Ma, “Koopman-operator-based robust data-driven control for wheeled mobile robots,” *IEEE/ASME Transactions on Mechatronics*, vol. 28, no. 1, pp. 461–472, 2023.
- [11] R. Strässer, J. Berberich, and F. Allgöwer, “Robust data-driven control for nonlinear systems using the koopman operator,” in *IFAC World Congress*, (Yokohama, Japan), July 2023.
- [12] 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.
- [13] C.-F. Huang, Y.-C. Tung, and T.-J. Yeh, “Balancing control of a robot bicycle with uncertain center of gravity,” in *International Conference on Robotics and Automation (ICRA)*, pp. 5858–5863, 2017.

- [14] 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, pp. 148–151 vol.1, 1994.
- [15] A. Owczarkowski, D. Horla, and J. Zietkiewicz, “Introduction of feedback linearization to robust lqr and lqi control – analysis of results from an unmanned bicycle robot with reaction wheel,” *Asian Journal of Control*, vol. 21, no. 2, pp. 1028–1040, 2019.
- [16] M. Kaheni, M. H. Zarif, A. A. Kalat, and L. Chisci, “Robust feedback linearization for input-constrained nonlinear systems with matched uncertainties,” in *2018 European Control Conference (ECC)*, pp. 2947–2952, 2018.
- [17] J. Yi, Y. Zhang, and D. Song, “Autonomous motorcycles for agile maneuvers, part i: Dynamic modeling,” in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, pp. 4613–4618, 2009.
- [18] 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)*, pp. 1570–1576, 2021.
- [19] J. D. Kooijman, A. L. Schwab, and J. P. Meijaard, “Experimental validation of a model of an uncontrolled bicycle,” *Multibody System Dynamics*, vol. 19, pp. 115–132, 2008.
- [20] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*. Boston, MA: Pearson, 7th ed., 2015.
- [21] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1.” <http://cvxr.com/cvx>, Mar. 2014.
- [22] D. Liu, Y. Bao, and H. Li, “Machine learning-based stochastic subspace identification method for structural modal parameters,” *Engineering Structures*, vol. 274, p. 115178, 2023.

Chapter 10

Paper D

An Adaptive Data-Enabled Policy Optimization Approach for Autonomous Bicycle Control

Niklas Persson, Feiran Zhao, Mojtaba Kaheni, Florian Dörfler, Alessandro V. Papadopoulos.

Submitted to Transaction on Control System Technology, Feb. 2025

Abstract

This paper presents a unified control framework that integrates a Feedback Linearization (FL) controller in the inner loop with an adaptive Data-Enabled Policy Optimization (DeePO) controller in the outer loop to balance an autonomous bicycle. While the FL controller stabilizes and partially linearizes the inherently unstable and nonlinear system, its performance is compromised by unmodeled dynamics and time-varying characteristics. To overcome these limitations, the DeePO controller is introduced to enhance adaptability and robustness. The initial control policy of DeePO is obtained from a finite set of offline, persistently exciting input and state data. To improve stability and compensate for system nonlinearities and disturbances, a robustness-promoting regularizer refines the initial policy, while the adaptive section of the DeePO framework is enhanced with a forgetting factor to improve adaptation to time-varying dynamics. The proposed DeePO+FL approach is evaluated through simulations and real-world experiments on an instrumented autonomous bicycle. Results demonstrate its superiority over the FL-only approach, achieving more precise tracking of the reference lean angle and lean rate.

10.1 Introduction

An autonomous bicycle is a bicycle, equipped with electric motors, sensors, algorithms, and control systems that allow the bicycle to navigate and operate without human intervention. Autonomous bicycles are an exciting area of research and development with numerous potential applications that can improve transportation, safety, and efficiency. In bicycle-sharing systems, autonomous bicycles can enhance the user experience by autonomously traveling to a person who has requested one, eliminating the need for individuals to walk toward the bicycle [1]. Additionally, autonomous bicycles can streamline fleet management by enabling bicycles to autonomously navigate to charging stations for recharging. This eliminates the need for operators to manually collect, load, and transport bicycles to charging stations, making the process more efficient. Similar applications of the self-balancing feature of autonomous bicycles include steering assistants for individuals with limited physical capabilities [2], among others.

Another notable application of autonomous bicycles is their ability to replace conventional bicycles in test tracks for evaluating the performance of various autonomous safety features in vehicles. Bicycles are often forced to share road segments with other motorized vehicles, which places cyclists at a higher risk of injury [3]. One way to reduce the risk is to use autonomous emergency braking (AEB) and autonomous emergency steering (AES) systems in motorized vehicles. The sensors in the vehicles detect and classify vulnerable road users (VRUs), including pedestrians and cyclists, and brakes or steers to avoid a collision. When organizations like EuroNCAP evaluate the AEB and AES systems on test tracks, a bicycle target placed on a moving platform is utilized¹. Since the target is mounted on the platform, its movements are also constrained by its linear motion. An autonomous bicycle, which can better represent a cyclist's maneuvers and sometimes unpredictable behavior, would enhance the testing area, thereby improving the reliability of safety tests in vehicles.

A bicycle is a nonlinear system that becomes self-stabilized at sufficient forward speed but remains unstable at lower velocities [4]. A human rider can employ three different control actions to maintain bicycle stability: adjusting the forward velocity, steering, and shifting their center of gravity to control the lean angle. At fixed speeds, similar to human control of a bicycle, an autonomous bicycle can be balanced either by regulating the steering angle [5, 6, 7] or by directly controlling the lean angle [8, 9]. However, the latter approach requires mounting a flywheel or a moving mass on the bicycle, significantly altering

¹<https://www.euroncap.com/en/vehicle-safety/the-ratings-explained/vulnerable-road-user-vru-protection/aeb-cyclist/>

its appearance, which may be undesirable for certain applications, i.e., at test tracks of safety features of vehicles, where it is important that the autonomous bicycle resembles an ordinary bicycle. In addition, as revealed by Kooijman et al. [10] and later confirmed by Moore et al. [11], controlling the lean angle has only a minor effect on balance at typical riding velocities. Therefore, this paper focuses on balancing the bicycle through steering control. Like human control at relatively high speeds, the bicycle should be steered in the direction of the fall, i.e., if it leans to the right, it must be steered to the right. Moreover, balancing the bicycle through steering control offers a more energy-efficient alternative than flywheels or a moving mass [12].

A wide range of control approaches have been proposed for balancing autonomous bicycles based on the simple principle of steering in the direction of the fall. For example, PID and LQR controllers have been designed based on a linearized model around the upright equilibrium of the bicycle and are evaluated around this equilibrium [6]. Both PID and LQR offer simple design methodologies; however, as a natural consequence of linearization, their performance may degrade as the system deviates from the equilibrium point used for linearization. One way to address this drawback is using a nonlinear controller, such as the sliding mode controller (SMC). The second order SMC proposed in the work of Defoort et al. [7] is designed using a nonlinear point-mass model of a bicycle and evaluated through both simulations and on an instrumented bicycle riding on a bicycle roller. However, SMC may induce chattering in the control signal due to the switching nature of the control law, potentially leading to high actuation efforts from the motors [13].

Active disturbance rejection control (ADRC) is proposed in [14] to handle unmodeled noise and disturbances. This control design is based on the so-called Whipple model [4], a fourth-order model linearized around small lean and steering angles. The model requires 25 physical parameters to be measured or estimated from the bicycle [15]. The ADRC is first evaluated in simulation using a detailed CAD model of their bicycle, imported into ADAMS, and controlled through co-simulation with MATLAB. Next, experiments were conducted. The results demonstrate that the bicycle can balance in both simulations and on straight asphalt tracks under varying forward velocities. However, noticeable oscillation was observed in both the lean and steering angles. Robustness against speed variations and disturbances is also considered in the recent work of Yeh et al. [5], in which a linear-parameter-varying (LPV) controller is designed based on a point mass model of a bicycle [16]. The controller was evaluated in both simulations and experiments conducted on an instrumented bicycle.

However, all these control methods rely heavily on an accurate system

model. While in practice, the mathematical models of autonomous bicycles are reasonably well understood [4, 17], the uncertainties, such as external disturbances (e.g., wind, street slope), and internal variations (e.g., changes in friction, mechanical and electrical couplings, and shifts in the center of gravity), can significantly degrade the performance of model-based approaches over time.

There are also direct approaches to bicycle control design that bypass the requirement of an explicit model. A notable instance is policy optimization (PO), an essential approach of modern reinforcement learning (RL) [18, 19, 20]. By computing the policy gradient from system trajectories, the PO updates the control policy with gradient descent methods. The work by Choi et al. [18] employs Deep Deterministic Policy Gradient (DDPG) to learn a policy for controlling the bicycle's speed, lean angle, and steering torque. This approach is evaluated on a nonlinear bicycle model; however, its real-world transferability is not tested in [18]. The work by Tuyen and Chung [19] evaluates DDPG on a miniature bicycle. However, the limited computational power of the hardware is insufficient for the control algorithm. Another deep RL algorithm is proposed for stabilizing a Whipple model of a bicycle and tracking a predefined path [20]. While the bicycle is balanced at varying velocities, they require time-consuming training and lack experimental validation.

Recently, there has been a growing trend of direct data-driven control methods motivated by behavioral system theory and subspace methods [21, 22, 23, 24, 25, 26]. For example, the seminal work [22] proposes a data-based parameterization for linear systems and reformulates the LQR problem as a convex program with a batch of persistently exciting data. Thus, the optimal LQR gain can be found without any explicit model or identification. To enhance adaptability, our previous work [27, 28] proposes a covariance parameterization for the LQR problem, based on which a data-enabled policy optimization (DeePO) method is developed to learn the LQR gain directly from on-line closed-loop data. In contrast to the previous policy optimization methods [18, 19, 20] that require multiple long trajectories to compute a single policy gradient, the DeePO method uses the covariance of online data to update the policy sample-by-sample. Besides its sample efficiency, the DeePO method is also computationally efficient, performing only a single step of gradient descent to update the policy per time step. Under persistently exciting input, the DeePO algorithm is shown to have non-asymptotic convergence guarantees to the optimal LQR gain. The covariance parameterization or the DeePO method has been extended to linear parameter-varying control [29], model-reference control [30], output-feedback control, and validated in a power converter system via simulations [31].

This paper proposes a unified DeePO framework for nonlinear bicycle control. To deal with the nonlinearities, we first design an output feedback linearization (FL) controller using well-established parametric bicycle models. This controller partially cancels the nonlinearities of the bicycle system while simultaneously stabilizing it. It is important to note that although the current state-of-the-art bicycle models are accurate, the model-based controllers may fail to achieve the desired performance due to unmodeled dynamics, parametric uncertainty, various disturbances, and time-varying dynamics (e.g., wind, varying friction in different environments, mechanical wear and tear, and coupling aging). To compensate for these disturbances and uncertainties, we integrate a DeePO controller on top of the FL layer. This allows us to adaptively fine-tune the feedback gains and effectively handle time-varying dynamics, disturbances, and unmodeled effects. Our approach is evaluated on both a realistic multi-body dynamic model of the bicycle and in hardware experiments conducted on an instrumented bicycle in an indoor environment.

To the best of our knowledge, this study is the first to report a real-world implementation of DeePO. While previous works have explored DeePO in simulations [28, 31], our study demonstrates its feasibility in a practical setting, validating its effectiveness in handling real-world nonlinearities, uncertainties, disturbances, and unmodeled dynamics. This contribution marks an important step toward bridging the gap between theory and real-world deployment of adaptive data-driven control methods.

10.1.1 Statement of Contributions

The contributions of this paper are summarized below.

- We introduce a unified control framework, combining FL with DeePO, for controlling a nonlinear autonomous bicycle.
- We integrate a forgetting factor into the conventional DeePO framework to effectively handle time-varying dynamics.
- This study is the first to apply DeePO in a real-world case study, specifically showcasing its effectiveness in controlling a bicycle through steering.
- Through both high-fidelity simulations and hardware experiments, we demonstrate that updating the feedback gains at every iteration of the DeePO algorithm's execution is not necessary. In fact, updating the feedback gain in DeePO with lower frequencies can still maintain or even improve control performance.

10.1.2 Organization

The rest of the paper is organized as follows: Section 10.2 formally states the control problem and provides an overview of our unified control framework. Section 10.3 outlines the details of the DeePO algorithm, including a regularizer for the initial policy and incorporating a forgetting factor into the framework. Section 10.4 presents the simulation and experimental results. Finally, Section 10.5 presents the concluding remarks.

10.1.3 Notation

We use I_n to denote the n -by- n identity matrix. We use $\rho(\cdot)$ to denote the spectral radius of a square matrix. $A^\top$, $\text{Tr}(A)$, and $A^\dagger$ represent the transpose, trace, and pseudoinverse of matrix A , respectively. We use $\text{diag}(a, b, \dots, c)$ to denote a diagonal matrix with diagonal elements being $a, b, \dots, c$. The 2-norm of matrix A is denoted $\|A\|$. We denote the continuous-time signal x with $x(t)$ and discrete-time with x_t .

10.2 Autonomous bicycle control

This paper addresses the problem of designing a unified control method for balancing an autonomous bicycle while tracking a reference lean angle. First, we use a simple point-mass model to design an inner-loop FL control. Next, DeePO enhances system performance by adapting feedback gains based on input-output data from sensors mounted on the bicycle.

10.2.1 Bicycle Dynamics and Mathematical Modeling

We consider a simple nonlinear model to represent the bicycle dynamics as in [32].

$$\ddot{\varphi}(t) = \frac{g}{h} \sin(\varphi(t)) + \frac{a}{bh} \cos(\varphi(t))v\dot{\delta}(t) - \left(\frac{1}{bh} - \frac{1}{b^2} \tan(\delta(t)) \tan(\varphi(t)) \right) \tan(\delta(t))v^2, \quad (10.1)$$

where $\varphi(t)$, $\dot{\varphi}(t)$, $\delta(t)$, and $\dot{\delta}(t)$ represent the lean angle, lean rate, steering angle, and the controlled steering rate, respectively. The contact point between the rear wheel and the ground is denoted by p_1 . Additionally, the vertical and horizontal distances between the bicycle's center of gravity and p_1 are denoted by a and h , respectively. The wheelbase is denoted by b , while g represents the gravitational constant, and v represents the forward velocity. This model

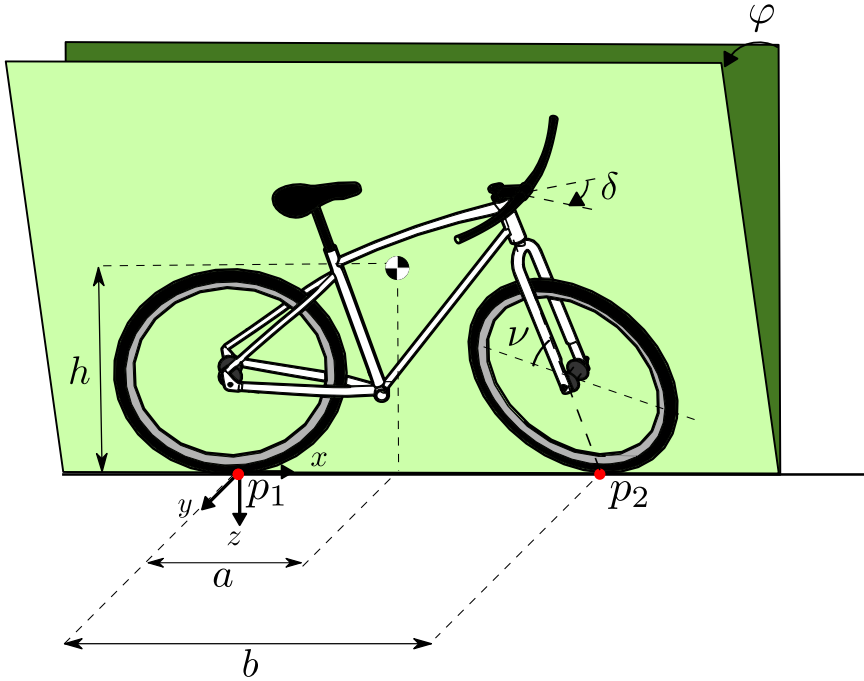


Figure 10.1: Illustration of the parameters used in the bicycle model in (10.1).

assumes a vertical steering axis, i.e., $\nu = \frac{\pi}{2}$, which results in zero trail. Furthermore, it is assumed that the steering axis can be controlled without delay and that the bicycle travels at a constant forward velocity. The visual representation of the parameters in (10.1) is shown in Fig. 10.1.

A bicycle is self-stabilized between the so-called *weave speed* and *capsize speed*. By analyzing the eigenvalues of the Whipple model and identifying the region where they are all negative, the self-stable region of a bicycle can be localized [15]. A similar eigenvalue analysis for the instrumented bicycle considered in this paper was previously conducted, where the 25 parameters required for the Whipple model were measured [33]. Based on this analysis, we focus on forward speeds of approximately 8 km/h (2.22 m/s), below the weave speed, as shown in Fig. 10.2. Thus, the system we aim to control is open-loop unstable, nonlinear, and non-holonomic, presenting a challenging control problem. Due to the system's inherent instability, applying a persistently exciting input without additional stabilization can lead to a loss of balance and cause the system to diverge. Specifically, an uncontrolled persistently exciting input could destabilize steering actions, such as turning left while leaning right, making it impractical to rely solely on such input for collecting persistently exciting

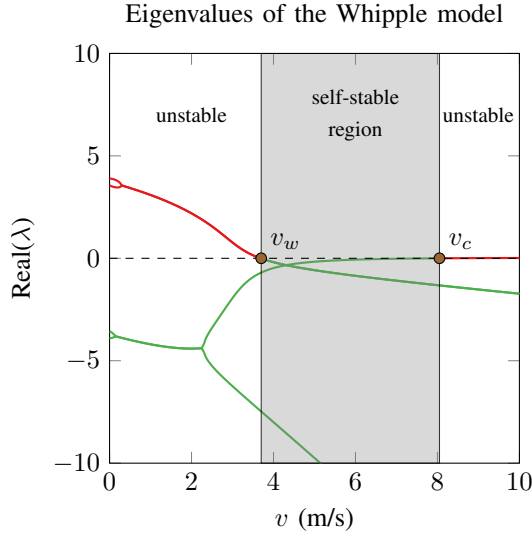


Figure 10.2: Stable and unstable regions of the instrumented bicycle, with the weave speed and capsize speed denoted by v_w and v_c , respectively.

data. In the following, we present an FL controller that balances the bicycle, simplifying the acquisition of persistently exciting data and mitigating some of the system's nonlinearities.

10.2.2 Control overview

A common approach for realizing a persistently exciting input is to utilize a random signal [34]. However, this approach is not directly applicable as it jeopardizes the bicycle's balance. Instead, we pre-stabilize the bicycle with an inner control loop using FL. Since the relative degree between the output and the number of states in the model given by (10.1) does not match, the system can only be partially linearized using output FL [13].

If we choose $x = [x_1, x_2, x_3] = [\varphi(t), \dot{\varphi}(t), \delta(t)]$, $y(t) = \varphi(t)$, $\dot{y}(t) = \dot{\varphi}(t)$, and represent the reference output as $y_r = [y_r(t), \dot{y}_r(t), \ddot{y}_r(t)]$, we can express the considered FL control law as:

$$u(t) = \dot{\delta}(t) = \frac{1}{p(x)}(w - f(x)), \quad (10.2)$$

where

$$\begin{aligned}
 f(x) &= -\left(\frac{1}{bh} - \frac{1}{b^2} \tan(x_3) \tan(x_1)\right) \tan(x_3) v^2 \\
 &\quad + \frac{g}{h} \sin(x_1) \\
 p(x) &= \frac{a}{bh} \cos(x_1) v, \\
 w &= \ddot{y}_r(t) + k_1 (\dot{y}_r(t) - \dot{y}(t)) + k_2 (y_r(t) - y(t)), \quad (10.3)
 \end{aligned}$$

with appropriate choices of $k_1 > 0$ and $k_2 > 0$ to partially compensate for the system's nonlinearities. However, the steering angle $\delta(t)$ remains an internal state that is not directly linearized, meaning some nonlinear dynamics persist. In particular, terms involving $\tan(\delta(t))$ introduce coupling effects that remain even after feedback linearization. Additionally, since the steering angle evolves according to $u = \dot{\delta}$, it can drift over time uncontrolled, requiring further regulation to prevent undesired effects on system stability. Furthermore, the proposed FL controller is designed based on continuous-time dynamics, with the model and control parameters provided in (10.1) and Table 10.1, respectively. However, in practice, we implement it using a sampled-data approach with a hold mechanism, which may introduce inaccuracies and lead to performance degradation due to the discrete nature of the implementation. This discrete approach may not fully capture the continuous dynamics of the system [35, 36]. Moreover, parameters in (10.2) and (10.3) are subject to parametric uncertainty, resulting in an inaccurate canceling of nonlinearities. Nevertheless, we demonstrate that the potential limitations of the FL controller can be mitigated by incorporating DeePO.

The proposed FL controller functions as an inner control loop to stabilize the bicycle, enabling the use of an additive random signal as either a persistently

Table 10.1: Model and control parameters for FL control

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 ²
k_1	-	1	-
k_2	-	6	-

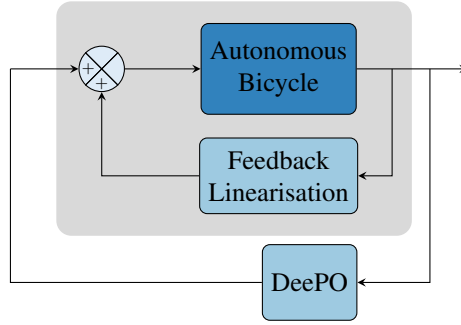


Figure 10.3: Control overview

exciting input or a performance enhancing adaptive control. In the remainder of the paper, we consider the autonomous bicycle with FL as our target system to control by DeePO, as highlighted by the gray box in Fig. 10.3. With this stable inner-loop system in place, we shift our focus to enhancing performance and compensating for the remaining nonlinearities using an adaptive, direct data-driven control approach in the outer loop.

10.3 Data-enabled policy optimization for autonomous bicycle control

This section first describes a brief overview of the linear quadratic regulator (LQR). Then, we propose a data-enabled policy optimization with a forgetting factor for adaptive learning of the LQR based on [27, 28].

10.3.1 The linear quadratic regulator

Consider a linear time-invariant system

$$\begin{cases} x_{t+1} = Ax_t + Bu_t + w_t \\ h_t = \begin{bmatrix} Q^{1/2} & 0 \\ 0 & R^{1/2} \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} \end{cases} \quad (10.4)$$

Here, x_t is the state, u_t is the control input, w_t represents the noise, and h_t is the performance signal of interest, (A, B) are controllable, and the weighting matrices (Q, R) are positive definite.

The LQR problem aims to find an optimal state-feedback gain $K \in \mathbb{R}^{m \times n}$ that minimizes the $\mathcal{H}_2$ -norm of the transfer function $\mathcal{T}(K) : w \rightarrow h$ of the

closed-loop system

$$\begin{bmatrix} x_{t+1} \\ h_t \end{bmatrix} = \begin{bmatrix} A + BK & I_n \\ \begin{bmatrix} Q^{1/2} \\ R^{1/2}K \end{bmatrix} & 0 \end{bmatrix} \begin{bmatrix} x_t \\ w_t \end{bmatrix}. \quad (10.5)$$

When $A + BK$ is stable, it holds that [37]

$$\|\mathcal{T}(K)\|_2^2 = \text{Tr}((Q + K^\top RK)\Sigma_K) =: C(K), \quad (10.6)$$

where Σ_K is the closed-loop state covariance matrix obtained as the positive definite solution to the Lyapunov equation

$$\Sigma_K = I_n + (A + BK)\Sigma_K(A + BK)^\top. \quad (10.7)$$

We refer to $C(K)$ as the LQR cost and to (10.6)-(10.7) as a *policy parameterization* of the LQR.

The optimal LQR gain K^* is unique and can be found by, e.g., solving an algebraic Riccati equation with (A, B) [37]. When (A, B) is unknown, data-driven methods can be used to learn the LQR gain from input-state data. In the sequel, we propose a covariance parameterization method for direct data-driven learning of the LQR.

10.3.2 Data-driven covariance parametrization of the LQR with exponential weighted data

Consider the t -long time series of states, inputs, noises, and successor states

$$\begin{aligned} X_{0,t} &:= \begin{bmatrix} x_0 & x_1 & \dots & x_{t-1} \end{bmatrix} \in \mathbb{R}^{n \times t}, \\ U_{0,t} &:= \begin{bmatrix} u_0 & u_1 & \dots & u_{t-1} \end{bmatrix} \in \mathbb{R}^{m \times t}, \\ W_{0,t} &:= \begin{bmatrix} w_0 & w_1 & \dots & w_{t-1} \end{bmatrix} \in \mathbb{R}^{n \times t}, \\ X_{1,t} &:= \begin{bmatrix} x_1 & x_2 & \dots & x_t \end{bmatrix} \in \mathbb{R}^{n \times t}, \end{aligned} \quad (10.8)$$

which satisfy the system dynamics

$$X_{1,t} = AX_{0,t} + BU_{0,t} + W_{0,t}. \quad (10.9)$$

Assume that the data is *persistently exciting (PE)* [21], i.e., the block matrix of input and state data

$$D_{0,t} := \begin{bmatrix} U_{0,t} \\ X_{0,t} \end{bmatrix} \quad (10.10)$$

has full row rank

$$\text{rank}(D_{0,t}) = m + n. \quad (10.11)$$

Define the covariance of exponentially weighted data as

$$\Phi_t := \frac{1}{t} D_{0,t} S_\lambda D_{0,t}^\top, \quad (10.12)$$

where $\lambda \in (0, 1)$ is a forgetting factor and $S_\lambda := \text{diag}\{\lambda^{t-1}, \lambda^{t-2}, \dots, 1\}$. Compared with [28], the forgetting factor here makes the weight of past data decay exponentially, such that the sample covariance can also reflect and adapt to the behavior of time-varying or nonlinear systems.

Since $D_{0,t}$ has full row rank and $S_\lambda > 0$, the covariance matrix is positive definite, i.e., $\Phi_t > 0$. Then, for any gain K , there exist a matrix V such that

$$\begin{bmatrix} K \\ I_n \end{bmatrix} = \Phi_t V. \quad (10.13)$$

We refer to (10.13) as the *covariance parameterization* with exponentially weighted data and to $V \in \mathbb{R}^{(n+m) \times n}$ as the *parameterized policy*.

With (10.13), the LQR problem (10.6)-(10.7) can be expressed by raw data matrices $(X_{0,t}, U_{0,t}, X_{1,t})$ and the optimization matrix V . For brevity, let $\bar{X}_{0,t} = X_{0,t} S_\lambda D_{0,t}^\top / t$ and $\bar{U}_{0,t} = U_{0,t} S_\lambda D_{0,t}^\top / t$ be a partition of Φ_t , and let $\bar{W}_{0,t} = W_{0,t} S_\lambda D_{0,t}^\top / t$ be the noise-state-input covariance, and finally define the covariance with respect to the successor state as $\bar{X}_{1,t} = X_{1,t} S_\lambda D_{0,t}^\top / t$. Then, the closed-loop matrix can be written as

$$A + BK = [B, A] \begin{bmatrix} K \\ I_n \end{bmatrix} \stackrel{(10.13)}{=} [B, A] \Phi_t V \stackrel{(10.9)}{=} (\bar{X}_{1,t} - \bar{W}_{0,t}) V. \quad (10.14)$$

Following the certainty-equivalence principle [38], we disregard the unmeasurable $\bar{W}_{0,t}$ for the design and use $\bar{X}_{1,t} V$ as the closed-loop matrix. After substituting $A + BK$ with $\bar{X}_{1,t} V$ in (10.6)-(10.7) and leveraging (10.13), the LQR problem becomes

$$\begin{aligned} & \underset{V}{\text{minimize}} \quad J_t(V) := \text{Tr} \left((Q + V^\top \bar{U}_{0,t} R \bar{U}_{0,t} V) \Sigma_t(V) \right), \\ & \text{subject to} \quad \bar{X}_{0,t} V = I_n, \end{aligned} \quad (10.15)$$

where $\Sigma_t(V) = I_n + \bar{X}_{1,t} V \Sigma_t(V) V^\top \bar{X}_{1,t}^\top$ is a covariance parameterization of (10.7), and the original gain matrix can be recovered as $K = \bar{U}_{0,t} V$. We refer to (10.15) as the covariance-parameterized LQR problem, which is direct data-driven and does not involve any explicit SysID.

10.3.3 Data-enabled policy optimization for adaptive LQR control with exponentially weighted data

In previous work [27, 28], a data-enabled policy optimization (DeePO) method for direct adaptive learning of the LQR was proposed, where the control policy is parameterized by sample covariance and updated recursively using gradient methods. In this subsection, we propose a DeePO algorithm based on our covariance parameterization with exponentially weighted data (10.13), detailed in Algorithm 10.1.

Algorithm 10.1 alternates between control (line 2) and policy update (lines 3-6). The DeePO algorithm uses online gradient descent of (10.15) to recursively update V . At time t , we apply the linear state feedback policy $u_t = K_t x_t + e_t$ for control and observe the new state x_{t+1} , where e_t is a probing noise used to ensure the PE rank condition (10.11). To update the policy, we first use $(X_{0,t+1}, U_{0,t+1}, X_{1,t+1})$ to formulate the covariance-parameterized LQR problem (10.15). Then, instead of solving this optimization problem optimality, we only take a single step of projected gradient descent towards its solution in (10.23). Here, the projection

$$\Pi_{\bar{X}_{0,t+1}} := I_{n+m} - \bar{X}_{0,t+1}^\dagger \bar{X}_{0,t+1} \quad (10.16)$$

onto the nullspace of $\bar{X}_{0,t+1}$ is to ensure the subspace constraint in (10.15). Define the feasible set of (10.15) (i.e., the set of stable closed-loop matrices) as $S_t := \{V \mid \bar{X}_{0,t} V = I_n, \rho(\bar{X}_{1,t} V) < 1\}$. Then, the gradient can be computed as follows.

Lemma 10.1 ([28]). *For $V \in S_t$, the gradient of $J_t(V)$ with respect to V is given by*

$$\nabla J_t(V) = 2 \left(\bar{U}_{0,t}^\top R \bar{U}_{0,t} + \bar{X}_{1,t}^\top P_t \bar{X}_{1,t} \right) V \Sigma_t(V), \quad (10.17)$$

where P_t satisfies the Lyapunov equation

$$P_t = Q + V^\top \bar{U}_{0,t}^\top R \bar{U}_{0,t} V + V^\top \bar{X}_{1,t}^\top P_t \bar{X}_{1,t} V. \quad (10.18)$$

Algorithm 10.1 is *direct and adaptive* in the sense that it directly uses online closed-loop data to update the policy. Thanks to the forgetting factor, it can rapidly adapt to changes in system behavior reflected in the data. As in [28], Algorithm 10.1 can also be implemented recursively. We write the sample covariance recursively as

$$\Phi_{t+1} = \frac{\lambda t}{t+1} \Phi_t + \frac{1}{t+1} \phi_t \phi_t^\top, \quad (10.19)$$

where $\phi_t = [u_t^\top, x_t^\top]^\top$. By the Sherman-Morrison formula [39], its inverse Φ_{t+1}^{-1} satisfies

$$\Phi_{t+1}^{-1} = \frac{t+1}{\lambda t} \left(\Phi_t^{-1} - \frac{\Phi_t^{-1} \phi_t \phi_t^\top \Phi_t^{-1}}{\lambda t + \phi_t^\top \Phi_t^{-1} \phi_t} \right). \quad (10.20)$$

Furthermore, the rank-one update of the parameterized policy is given by

$$\begin{aligned} V_{t+1} &= \frac{t+1}{t} \left(\Phi_t^{-1} - \frac{\Phi_t^{-1} \phi_t \phi_t^\top \Phi_t^{-1}}{t + \phi_t^\top \Phi_t^{-1} \phi_t} \right) \Phi_t V_t' \\ &= \frac{t+1}{\lambda t} \left(V_t' - \frac{\Phi_t^{-1} \phi_t \phi_t^\top V_t'}{\lambda t + \phi_t^\top \Phi_t^{-1} \phi_t} \right), \end{aligned} \quad (10.21)$$

where Φ_t^{-1} and V_t' are given from the last iteration.

Remark 3. Using the forgetting factor λ may asymptotically lead to failure of the rank-one update as time tends to infinity. To see this, we notice that the covariance update in (10.19) satisfies stable linear dynamics. This implies a loss of persistency of excitation and Φ_t will tend to zero, and hence Φ_t^{-1} will grow to infinity. A simple remedy is to reset the covariance Λ_t occasionally, i.e., set $\Lambda_t = I_{n+m}$, $\forall t \in \{T, 2T, \dots\}$. Since the autonomous bicycle operates only for a finite time, we do not reset the covariance in our subsequent experiments in Section 10.4. Another approach is to use sliding window data rather than exponentially weighted data for the covariance parameterization (10.13), where a key is to select an optimal window size to balance data informativity and adaptation efficiency. We leave this exploration to future work. ■

Remark 4. The stepsize η_t should be set according to the signal-to-noise ratio (SNR) of online data. For example, when the SNR is large, we are confident with the gradient direction, and the stepsize can be chosen more aggressively; on the contrary, when the SNR is small, the stepsize should be small to prevent the policy from moving out of the stability region. To this end, we set the stepsize as

$$\eta_t = \frac{\eta_0}{\left\| \bar{U}_{0,t} \Pi_{\bar{X}_{0,t}} \bar{U}_{0,t}^\top \right\|}, \quad t \geq t_0, \quad (10.22)$$

where η_0 is a constant, and the denominator is used to quantify the SNR. Another motivation of the denominator is from [40, Lemma 3], which reveals the equivalence between data-enabled and model-based policy gradients up to the data matrix $\bar{U}_{0,t} \Pi_{\bar{X}_{0,t}} \bar{U}_{0,t}^\top$. ■

Algorithm 10.1 DeePO for direct adaptive LQR control

Require: Offline data $(X_{0,t_0}, U_{0,t_0}, X_{1,t_0})$, an initial policy K_{t_0} , and a stepsize η .

- 1: **for** $t = t_0, t_0 + 1, \dots$ **do**
- 2: Apply $u_t = K_t x_t + e_t$ and observe x_{t+1} .
- 3: Update covariance matrices Φ_{t+1} and $\bar{X}_{1,t+1}$.
- 4: **Policy parameterization:** given K_t , solve V_{t+1} via

$$V_{t+1} = \Phi_{t+1}^{-1} \begin{bmatrix} K_t \\ I_n \end{bmatrix}.$$

- 5: **Update of the parameterized policy:** perform one-step projected gradient descent

$$V'_{t+1} = V_{t+1} - \eta_t \Pi_{\bar{X}_{0,t+1}} \nabla J_{t+1}(V_{t+1}), \quad (10.23)$$

where the gradient $\nabla J_{t+1}(V_{t+1})$ is given by Lemma 10.1.

- 6: **Gain update:** update the control gain by

$$K_{t+1} = \bar{U}_{0,t+1} V'_{t+1}.$$

- 7: **end for**

Algorithm 10.1 requires the initial policy to be stabilizing. A potential approach is to solve the covariance-parameterized LQR (10.15) with the offline data $(X_{0,t_0}, U_{0,t_0}, X_{1,t_0})$. However, due to the nonlinearity in the system dynamics of the autonomous bicycle, the solution of covariance-parameterized LQR may be destabilizing. Next, we propose a robustness-promoting regularizer for the covariance parameterization to obtain a stabilizing initial policy.

10.3.4 Learning an initial stabilizing policy using robustness promoting regularization

The feasibility of the covariance-parameterized LQR problem (10.15) depends on that of the Lyapunov equation

$$\Sigma = I_n + \bar{X}_1 V \Sigma V^\top \bar{X}_1^\top, \quad (10.24)$$

where $\bar{X}_1 V$ is regarded as the closed-loop matrix. However, having assumed certainty-equivalence by the covariance parameterization (10.13) and the relation $A + BK = (\bar{X}_1 - \bar{W}_0)V$, the Lyapunov equation that should be met

is

$$\Sigma = I_n + (\bar{X}_1 - \bar{W}_0)V\Sigma V^\top(\bar{X}_1 - \bar{W}_0)^\top. \quad (10.25)$$

The gap between the right-hand side of (10.24) and (10.25) is

$$\begin{aligned} & \bar{W}_0V\Sigma V^\top\bar{W}_0^\top - \bar{W}_0V\Sigma V^\top\bar{X}_1^\top - \bar{X}_1V\Sigma V^\top\bar{W}_0^\top \\ &= \frac{1}{t^2}W_0D_0^\top V\Sigma V^\top D_0W_0^\top \\ & - \frac{1}{t^2}(W_0D_0^\top V\Sigma V^\top D_0X_1^\top + X_1D_0^\top V\Sigma V^\top D_0W_0^\top). \end{aligned} \quad (10.26)$$

To reduce the gap, it suffices to make $\text{Tr}(D_0^\top V\Sigma V^\top D_0/t)$ small. To this end, we introduce the regularizer $\text{Tr}(V\Sigma V^\top \Phi)$ to the covariance-parameterized LQR problem (10.15), leading to

$$\begin{aligned} & \underset{V, \Sigma \geq 0}{\text{minimize}} \quad J_t(V) + \gamma \text{Tr}(V\Sigma V^\top \Phi), \\ & \text{subject to} \quad \Sigma = I_n + \bar{X}_1V\Sigma V^\top\bar{X}_1^\top, \bar{X}_0V = I_n \end{aligned} \quad (10.27)$$

with gain matrix $K = \bar{U}_0V$, where $\gamma > 0$ is the regularization coefficient. We refer to (10.27) as the regularized covariance parameterization of the LQR problem.

To obtain an initial stabilizing policy for Algorithm 10.1, we solve (10.27) with offline data $(X_{0,t_0}, U_{0,t_0}, X_{1,t_0})$.

10.3.5 Control gain update rate

Rapid changes in an adaptive control policy, K_t can potentially induce oscillations and, in the worst case, render the system unstable [41]. Moreover, the control policy at certain time intervals may be significantly influenced by measurement noise, meaning that updates could be driven more by noise than by the actual system dynamics.

To address these potential issues, we propose updating the DeePO control gain less frequently than the sampling frequency. To regulate the update frequency, we introduce the parameter ξ , which determines the intervals at which the controller in line 6 of Algorithm 10.1 is updated. For instance, if $\xi = 1$, the control gain is updated at every iteration, whereas if $\xi = 100$, the gain is updated every 100 iterations.

10.4 Simulations and experiments

In this section, we first provide details of the instrumented bicycle used in the experiments. Next, we describe the simulation setup, followed by the results

obtained from the simulations. Finally, we present the details of the experiments and the corresponding results.

10.4.1 Instrumented Bicycle

The bicycle we consider in experiments is a men's model of an electric bicycle, as shown in Fig. 10.4. The factory-installed rear wheel motor and the battery mounted on the main frame are used to drive the bicycle forward. The rear wheel is controlled through a Phoenix Edge HV 60 AMP Electronic Speed Controller (ESC), and the rear wheel velocity is estimated using 12 evenly distributed magnets on the rear wheel and a Hall sensor. A PI controller is manually tuned to maintain an approximately constant forward velocity. An Xsens MTi-7 GNSS/INS at the bottom bracket shell measures the lean angle and rate.

Furthermore, a Dynamixel XH540-W270-T servo with a cog belt is mounted on the bicycle's main frame to control the handlebar. The servo is controlled through a velocity command, $u_t = \dot{\delta}_t$ rad/s, using an integer between -167 and 167 with a resolution of approximately 0.024 rad/s per unit. The control signal is saturated at ± 4 rad/s. The control algorithms are implemented, and the data is processed using ROS2 Humble running on a Raspberry Pi 4b with Ubuntu 20.04. Finally, a radio controller (RC) lets the operator send wireless commands to the RC receiver mounted on the bicycle.

10.4.2 Simulation setup

A CAD model of the instrumented bicycle was designed using SolidWorks. The CAD model is imported into MathWorks Simscape and controlled through Simulink. The rear and front wheels are connected to the mainframe through revolute joints, where the rear joint is actuated and given a constant speed corresponding to a forward velocity of 8 km/h. A third revolute joint connects the steering axis to the bicycle's mainframe and is actuated through the control signal $u(t) = \dot{\delta}(t)$. The steering dynamics are modeled using an identified steering step response matching procedure [6], from the control signal $u(t)$ to the steering rate $\dot{\delta}(t)$ the resulting transfer function is:

$$H(s) = \frac{100 + s}{100}. \quad (10.28)$$

The transfer function is placed in series with the bicycle model, as shown in Fig. 10.5. The control signal is saturated at ± 4 rad/s. The figure also highlights that the input signal, u_t , is composed of two parts. First, an inner loop control signal u_t^i from the FL controller with the parameters given in Table 10.1, and second, an outer control loop signal u_t^o which originates from a persistently

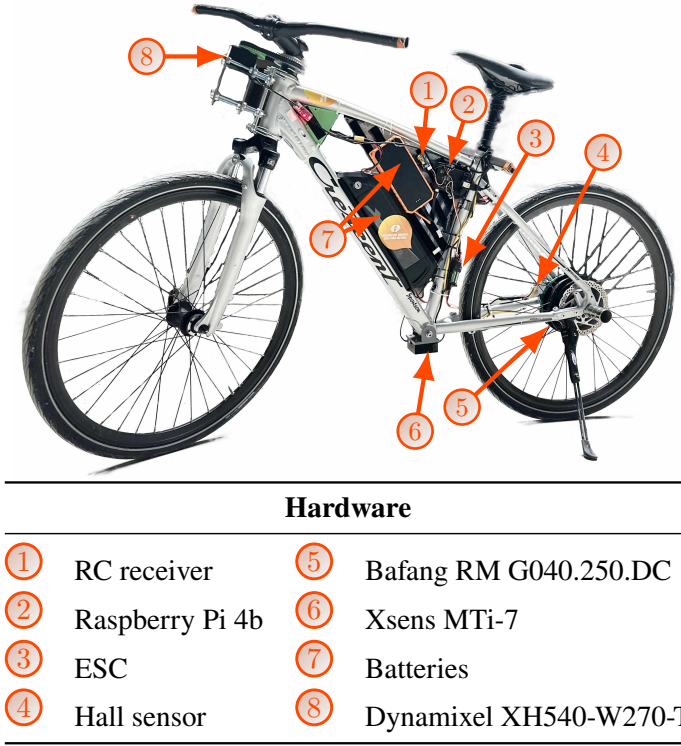


Figure 10.4: Instrumented bicycle used in the experiments.

excited input $u_t^{PE} = \mathcal{N}(0, \sigma_{PE})$ with $\sigma_{PE} = 0.2$ rad/s when the switch is in position α , and from DeePO when the switch is in position β . The lean angle, lean rate, and steering angle measurements are induced with zero-mean, normally distributed noise using $\sigma_\phi = \sigma_\delta = 0.5$ deg and $\sigma_{\dot{\phi}} = 0.5$ deg/s. The forward velocity is set to a constant value of 8 km/h.

The tracking error is used as the state vector, i.e., $x_t^e = [\varphi_t^r - \varphi_t, \dot{\varphi}_t^r - \dot{\varphi}_t]^\top$. The state variables and the persistently exciting input signal are sampled at 100 Hz. The same execution rate is used for DeePO and the FL controller, generating a new control signal for the steering motor every 0.01 s.

An initial simulation is performed with the switch in Fig. 10.5 in position α , and the bicycle tracks a lean angle reference $\varphi_t^r = 0$ and its derivative $\dot{\varphi}_t^r = 0$ for 10 seconds. We collect $T = 200$ samples of the acquired data to construct $U_{0,T}$, $X_{0,T}$, $X_{1,T}$ using (10.8). An initial policy is obtained by solving the regularized covariance-parameterized LQR problem (10.27) using CVX [42] with $Q = \text{diag}[1, 0.01]$ and $R = 10^{-4}$. Higher values of γ promote robustness against uncertainties in the system, while lower values prioritize performance.

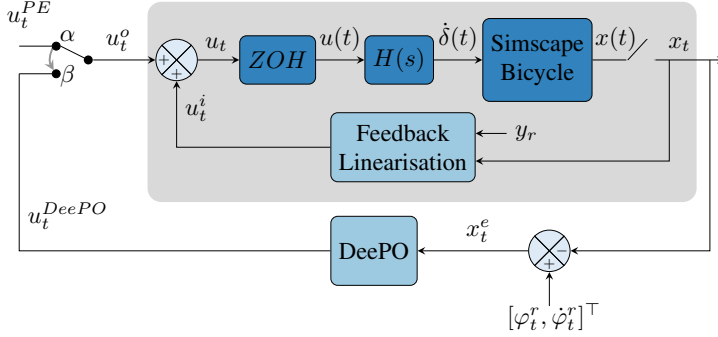


Figure 10.5: Control setup where the gray box represents the system controlled using DeePO.

In our simulations, we set it $\gamma = 1$, which balances robustness and performance in our initial control policy.

Next, the switch in Fig. 10.5 is set to position β , and Algorithm 10.1 is used to update the control policy at every time sample, i.e., $\xi = 1$ and using a forgetting factor $\lambda = 1 - 10^{-4}$, and learning rate $\eta = 10^{-3}$. To ensure a persistently exciting input, the probing noise e_t is a zero-mean normally distributed random number, which is added to the DeePO output and constructs the input to the system as:

$$u_t^{DeePO} = u_t^{DeePO} + e_t, \quad (10.29)$$

where $e_t = \mathcal{N}(0, 0.2u_t^{DeePO})$. Moreover, the bicycle tracks a time-varying reference, as shown in Fig. 10.6.

To evaluate the update rate for the gain update in Algorithm 10.1 (line 6), multiple simulations are conducted where ξ varies while the rest of the parameters are kept fixed. Four different update rates for the gain update are considered: $\xi = 1, 10, 50$, and 100 . Moreover, one simulation is conducted with only the FL controller as a baseline. The forgetting factor is evaluated in a similar fashion using $\lambda = 1 - 10^{-\zeta}$ with $\zeta = \infty, 2, 3, 4, 5, 6$, where $\zeta = \infty$ corresponds to a controller without a forgetting factor. In these simulations, the control update rate is set to $\xi = 1$, and the rest of the parameters are kept at their initial values. The performance of the controllers is evaluated using the integrated squared error of the lean angle and the lean rate with respect to their respective reference values:

$$\text{ISE}_\varphi = \sum_{i=0}^t (\varphi_i - \varphi_i^r)^2, \quad \text{ISE}_{\dot{\varphi}} = \sum_{i=0}^t (\dot{\varphi}_i - \dot{\varphi}_i^r)^2. \quad (10.30)$$

10.4.3 Simulation results

The results for tracking a reference lean angle and lean rate are given in the top and middle plots of Fig. 10.6. The bottom plot represents the contribution of the DeePO algorithm to the system's total control signal. The update rate of the control gain is $\xi = 1$, i.e., the control gain is updated with every sample, in Fig. 10.6. The results highlight the effectiveness of our unified control framework using FL and DeePO for tracking the reference lean angle, φ_t^r , and lean rate, $\dot{\varphi}_t^r$. The tracking is smoother, and the error is reduced compared to using only FL, as shown in the top and middle plots of Fig. 10.6. This improvement demonstrates DeePO's ability to adapt and refine the control policy iteratively, even under simulated nonlinear dynamics and sensor noise. The bottom plot of Fig. 10.6 shows the contribution of DeePO to the total control signal and highlights that DeePO complements the FL controller and enables fine-tuning of the control signal online.

In Fig. 10.7, the control policy for different update rates with respect to time is presented, and the corresponding ISE values are reported in Fig. 10.8, together with the ISE values when using only FL. These results clearly show that updating the control policy applied to the bicycle at every iteration of the algorithm is unnecessary. In fact, the performance improves for some of the lower update rates, i.e., higher values of ξ . However, it is not obvious how this parameter should be chosen, as for $\xi = 10$, the performance is slightly worse than $\xi = 1$. On the other hand, the performance is improved for $\xi = 50$ and $\xi = 100$. However, a low update rate may reduce DeePO's adaptation to fast changes in the system's behavior, reducing performance in systems with rapidly changing dynamics. Moreover, from Fig. 10.7, the control policy does not seem to converge, which may be due short simulation horizon.

Finally, Fig. 10.9 presents the ISE values when the forgetting factor varies while the remaining control parameters are kept fixed. The results show that introducing a forgetting factor in DeePO may further enhance control performance. An intermediate forgetting factor of $\lambda = 1 - 10^{-5}$ yields the best performance, while $\lambda = 1 - 10^{-2}$ results in the largest ISE value. Though varying ξ and λ may offer performance enhancements, finding the optimal and likely state-dependent values, for these parameters remains an area for future research. Moreover, the consistency of the simulation results highlights the efficiency of our unified control framework, which integrates FL and DeePO under varying circumstances.

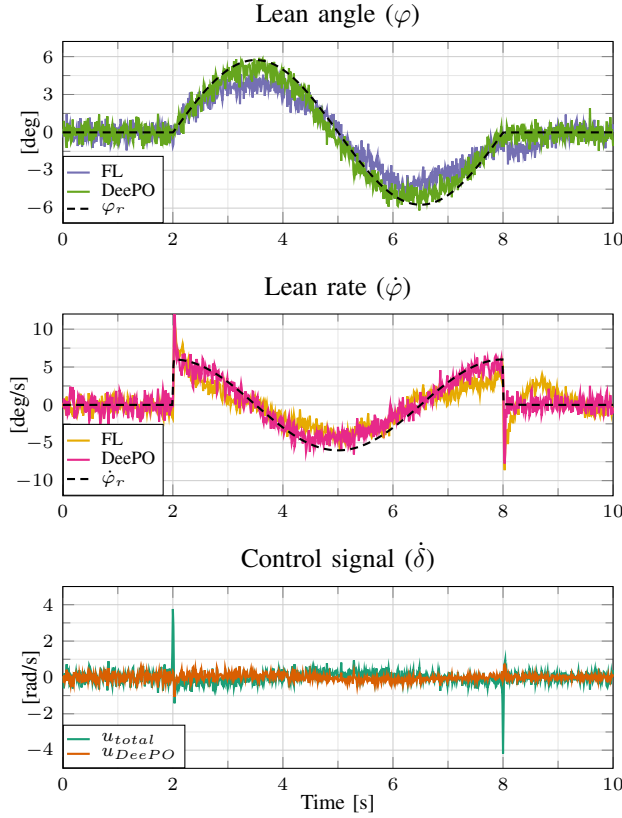


Figure 10.6: Tracking performance of the DeePO algorithm in simulation with $\xi = 1$. The top and middle plots illustrate the lean angle and rate tracking results for the DeePO+FL setup and the FL-only approach. The bottom plot presents the total control signal along with DeePO’s contribution.

10.4.4 Experimental setup

An initial experiment is conducted in which a dataset is collected from the sensors and the control signal. Similar to the initial simulation, the FL controller with the parameters from Table 10.1 together with a persistently exciting input as $u_{PE} = \mathcal{N}(0, \sigma_{PE})$ are used as the input to the system, with $\sigma_P E = 0.2$ rad/s. Furthermore, we consider tracking of a lean angle and lean rate reference of zero, and the tracking errors make up the states as in the simulation. The experiments are conducted in an indoor warehouse building with a flat concrete floor; see Fig. 10.10. The instrumented bicycle starts from a standstill, and the operator starts the bicycle using a switch on the RC controller and assists in balancing the bicycle until it reaches its constant goal velocity of 8 km/h. Using

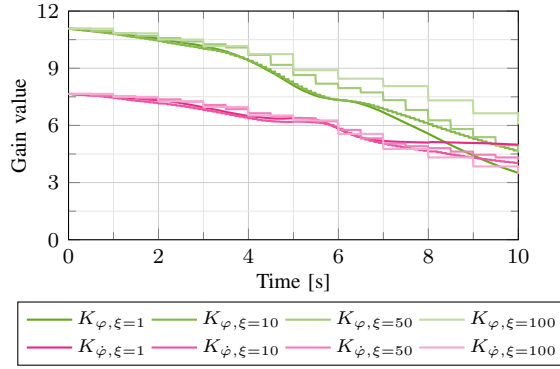


Figure 10.7: Evaluation of the control policy in simulation over time with different update rates of the control gain.

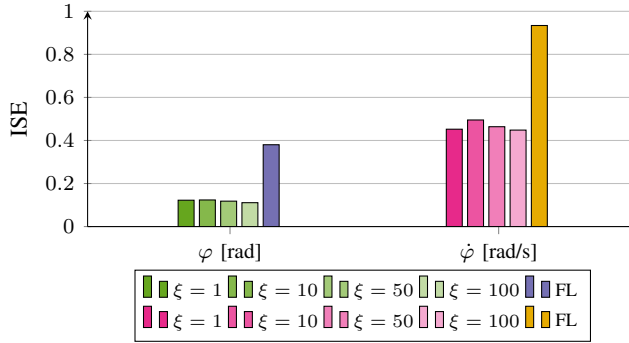


Figure 10.8: Integrated squared error of the lean angle and lean rate for different values of ξ in simulation. The performance of only using FL is also included as a baseline.

a second switch on the RC controller, the experiment is initiated, and the persistently exciting input, together with the FL controller, regulates the steering velocity of the handlebar and, by extension, the bicycle's balance. The Xsens MTi-7 and the Dynamixel data are collected at a sampling frequency of 100 Hz on the Raspberry Pi. Algorithm 10.1 and the FL controller, implemented on the Raspberry Pi, compute a new control signal every 0.01 s and transmit it to the Dynamixel servo.

The sampled input and state data are post-processed in Matlab, where they are used to construct $U_{0,T}$, $X_{0,T}$, $X_{1,T}$, with $T = 300$. Next, the initial policy is obtained by solving (10.27) in Matlab, with $Q = I_2$, $R = 0.01$, and $\gamma = 1$. In the subsequent experiments, we set the forgetting factor and learning rate as $\lambda = 1 - 10^{-4}$ and $\eta = 10^{-3}$, respectively. While γ , λ , and η are kept the same in the experiments as in the simulation, the Q , R , and T are changed in exper-

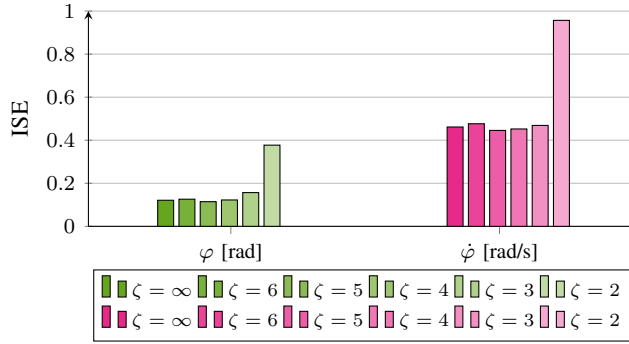


Figure 10.9: Integrated squared error of tracking the lean angle and lean rate references for different values of the forgetting factor in DeePO. The forgetting factor is defined as $\lambda = 1 - 10^{-\zeta}$.

iments. The increase in the number of data samples can be explained by the idealized dynamics in simulation compared to experiments where, for instance, mechanical imperfections, delays, and external disturbances are present. Thus, a larger dataset ensures a more reliable estimation of the system's behavior. Moreover, the increase in the weight of $\dot{\varphi}$ in Q is justified by the sensor noise and the need for robustness in experiments. Finally, in simulations, the resolution of the steering motor is neglected, and the complex dynamics of the actuating steering system are not fully captured. Thus, we can allow for more aggressive control actions using a lower value of R . On the other hand, in experiments, the resolution of the motor limits the possible steering commands. A larger R in experiments ensures a smoother, more physically feasible control input. The same time-varying reference lean angle and lean rate used in simulations are also utilized in the experiments. Furthermore, we conduct several experiments where ξ varies as $\xi = 1, 10, 50$, and 100 . Additionally, one experiment is conducted with only the FL controller as a baseline.

10.4.5 Experimental results

The lean angle and lean rate tracking performance of the DeePO algorithm is illustrated in the top and middle plots of Fig. 10.11 using $\xi = 1$. The results of using only FL are also included in the figure. The control signal of DeePO and the total control signal of the system are highlighted in the bottom plot of Fig. 10.11. The results demonstrate the effectiveness of DeePO and its robust tracking of the reference lean angle and lean rate, with noticeable improvements compared to the FL controller, as evident from the first two plots of Fig. 10.11. The results also show how DeePO adapts over time, even in the



Figure 10.10: The indoor environment where the experiments were conducted on a flat concrete floor.

presence of nonlinearities, sensor noise, and external disturbances (e.g., floor imperfections or variations in tire grip). However, the oscillations in lean angle and lean rate have a much higher amplitude in experiments compared to simulations, which the steering motor's resolution limitation could partly explain. In experiments, the resolution is limited to 0.024 rad/s per unit, a factor not considered in simulations. Moreover, simulations have unmodelled dynamics and environmental details compared to experiments, such as joint friction, uneven terrain, and approximations made in the model.

The evolution of the control gain values for the lean angle and lean rate are reported in Fig. 10.12 with varying values for gain update frequency ξ . As observed in simulations, the evolution of the control gains in experiments follows the same trend, even though the update frequencies vary. However, the difference between the evolution of the control gains in experiments and the control gains of the simulation, as presented in Fig. 10.7, is quite different, which indicates a gap between the simulations and experiments. It also highlights the usefulness of adaptive control methods, which can refine the feedback gain based on online experiment data. The gap between simulations and experiments is also evident when comparing the ISE values in Fig. 10.8 and

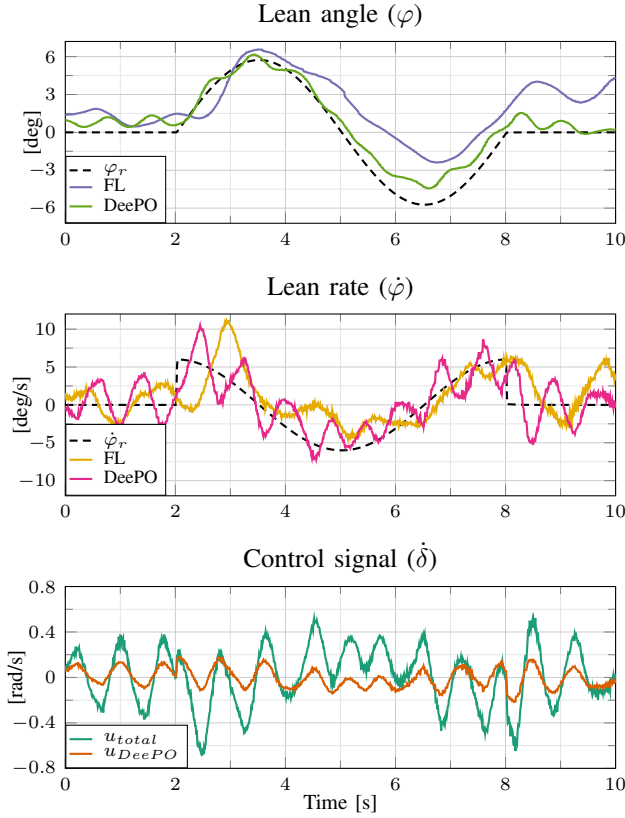


Figure 10.11: Experimental results of DeePO where the control policy is updated at every time step, i.e., $\xi = 1$.

Fig. 10.13. In experiments, update at every time step or intermediate rates of $\xi = 50$ produces significantly better results than update at lower rates or using only FL. The considerably higher ISE for $\xi = 100$ and using only FL further highlights the limitations of static or infrequent control gain updates. In simulations where we have control over the initial conditions, noise, and external disturbances, the results are more aligned, and the impact of ξ is not as evident as in experiments. One particular challenge in the experiments was finding a suitable pre-stabilizing control form initial data and control of the initial conditions. A video demonstration of the simulations and experiments is available online ².

²<https://youtu.be/5RKNr6tPiuwonline>

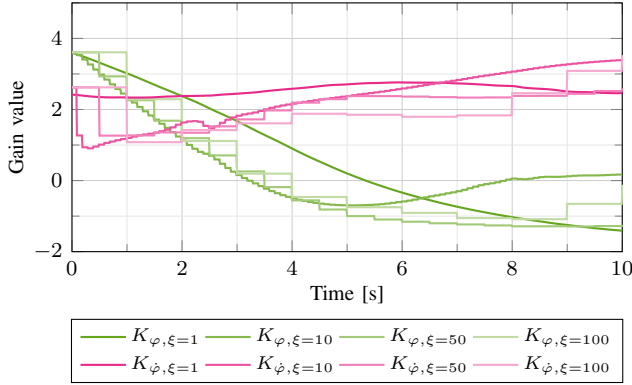


Figure 10.12: Evolution of control gains in experiments for different values of ξ .

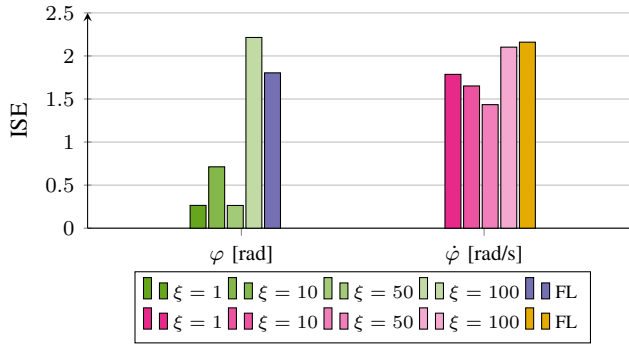


Figure 10.13: Integrated squared error for different values of ξ and when using the FL controller alone in experiments.

10.5 Conclusion

This paper introduced a unified framework that balances an autonomous bicycle by combining an FL controller in the inner loop and DeePO in the outer loop. The primary objective of the FL loop is to stabilize and partially linearize an otherwise unstable and nonlinear system. However, practical systems often contain unmodeled dynamics and time-varying characteristics that can degrade the performance of the FL controller when used in isolation. To address these challenges, we integrated a DeePO controller on top of the FL loop.

We derived an initial control policy using a finite set of offline, persistently exciting input and state data. To handle the nonlinearities and disturbances that may degrade the policy's performance, we introduced a robustness-promoting regularizer to refine the initial stabilizing policy and a forgetting factor in the

DeePO framework to adapt to the time-varying nature of our case study.

We demonstrated the effectiveness of the DeePO+FL approach through both simulations and real-world experiments on an autonomous bicycle. The results clearly showed that DeePO+FL outperforms the FL-only approach, particularly in terms of tracking the reference lean angle and lean rate more accurately. Additionally, we evaluated the impact of the control gain update frequency and found that performance improvements could be achieved with a lower update frequency. However, determining the optimal update rate remains an open question for future research.

The experimental and simulation results demonstrated that the proposed controller effectively adapts to the system dynamics despite the presence of nonlinearities, sensor noise, and hardware limitations. Our work illustrates the potential of direct data-driven methods to adapt and control nonlinear systems, such as an autonomous bicycle, relying solely on data. In the future, we plan to enhance the DeePO algorithm by incorporating the robustness regularizer in its online component. Additionally, exploring direct data-driven navigation for the bicycle is another exciting direction for further research.

Bibliography

- [1] N. C. Sanchez, L. A. Pastor, and K. Larson, "Autonomous bicycles: A new approach to bicycle-sharing systems," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, 2020.
- [2] 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.
- [3] B. Kutela, S. Das, and B. Dadashova, "Mining patterns of autonomous vehicle crashes involving vulnerable road users to understand the associated factors," *Accident Analysis & Prevention*, p. 106473, 2021.
- [4] 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.
- [5] T.-J. Yeh, T.-C. Lin, and A. C.-B. Chen, "Robust balancing and trajectory control of a self-driving bicycle," *IEEE Transactions on Control Systems Technology*, vol. 32, no. 6, pp. 2410–2417, 2024.

- [6] 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)*, pp. 1570–1576, 2021.
- [7] 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.
- [8] Y. Zhang, P. Wang, J. Yi, D. Song, and T. Liu, “Stationary balance control of a bikebot,” in *International Conference on Robotics and Automation (ICRA)*, pp. 6706–6711, 2014.
- [9] P. Wang, J. Yi, T. Liu, and Y. Zhang, “Trajectory tracking and balance control of an autonomous bikebot,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2414–2419, 2017.
- [10] J. D. Kooijman, A. L. Schwab, and J. K. Moore, “Some observations on human control of a bicycle,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 4, pp. 2021–2028, 2009.
- [11] J. K. Moore, J. D. Kooijman, A. L. Schwab, and M. Hubbard, “Rider motion identification during normal bicycling by means of principal component analysis,” *Multibody System Dynamics*, vol. 25, no. 2, pp. 225–244, 2011.
- [12] D. Rodriguez-Rosa, I. Payo-Gutierrez, F. J. Castillo-Garcia, A. Gonzalez-Rodriguez, and S. Perez-Juarez, “Improving energy efficiency of an autonomous bicycle with adaptive controller design,” *Sustainability*, vol. 9, no. 5, p. 866, 2017.
- [13] J. Slotine and W. Li, *Applied Nonlinear Control*. Prentice-Hall, 1991.
- [14] 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.
- [15] J. D. Kooijman, A. L. Schwab, and J. P. Meijaard, “Experimental validation of a model of an uncontrolled bicycle,” *Multibody System Dynamics*, vol. 19, pp. 115–132, 2008.

- [16] K. J. Åström, R. E. Klein, and A. Lennartsson, “Bicycle dynamics and control: adapted bicycles for education and research,” *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, 2005.
- [17] S. Bruni, J. Meijaard, G. Rill, and A. Schwab, “State-of-the-art and challenges of railway and road vehicle dynamics with multibody dynamics approaches,” *Multibody System Dynamics*, vol. 49, pp. 1–32, 2020.
- [18] 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.
- [19] L. P. Tuyen and T. Chung, “Controlling bicycle using deep deterministic policy gradient algorithm,” in *International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pp. 413–417, 2017.
- [20] S. Weyrer, P. Manzl, A. Schwab, and J. Gerstmayr, “Path following and stabilisation of a bicycle model using a reinforcement learning approach,” *arXiv preprint arXiv:2407.17156*, 2024.
- [21] 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.
- [22] C. De Persis and P. Tesi, “Formulas for data-driven control: Stabilization, optimality, and robustness,” *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 909–924, 2020.
- [23] J. Coulson, J. Lygeros, and F. Dörfler, “Data-enabled predictive control: In the shallows of the DeePC,” in *18th European Control Conference (ECC)*, pp. 307–312, 2019.
- [24] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, “Data-driven model predictive control with stability and robustness guarantees,” *IEEE Transactions on Automatic Control*, vol. 66, no. 4, pp. 1702–1717, 2020.
- [25] F. Dörfler, P. Tesi, and C. De Persis, “On the role of regularization in direct data-driven LQR control,” in *IEEE 61st Conference on Decision and Control (CDC)*, pp. 1091–1098, 2022.
- [26] H. J. Van Waarde, J. Eising, H. L. Trentelman, and M. K. Camlibel, “Data informativity: a new perspective on data-driven analysis and control,” *IEEE Transactions on Automatic Control*, vol. 65, no. 11, pp. 4753–4768, 2020.

- [27] 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)*, pp. 6160–6165, 2023.
- [28] 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.
- [29] M. Mejari and V. Breschi, “Direct data-driven design of LPV controllers and polytopic invariant sets with cross-covariance noise bounds,” *IEEE Control Systems Letters*, 2024.
- [30] M. Mejari, V. Breschi, S. Formentin, and D. Piga, “Bias correction and instrumental variables for direct data-driven model-reference control,” *arXiv preprint arXiv:2411.05740*, 2024.
- [31] F. Zhao, R. Leng, L. Huang, H. Xin, K. You, and F. Dörfler, “Direct adaptive control of grid-connected power converters via output-feedback data-enabled policy optimization,” *arXiv preprint arXiv:2411.03909*, 2024.
- [32] 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)*, pp. 114–120, 2024.
- [33] N. Persson, *Control and Navigation of an Autonomous Bicycle*. Mälardalen University (Sweden), 2023.
- [34] M. Alsalti, V. G. Lopez, and M. A. Müller, “On the design of persistently exciting inputs for data-driven control of linear and nonlinear systems,” *IEEE Control Systems Letters*, vol. 7, pp. 2629–2634, 2023.
- [35] R. Kimber and W. Gray, “On sampled-data implementations of feedback linearizing controllers,” in *30th IEEE Conference on Decision and Control (CDC)*, pp. 1873–1874, 1991.
- [36] J. Grizzle and P. Kokotovic, “Feedback linearization of sampled-data systems,” *IEEE Transactions on Automatic Control*, vol. 33, no. 9, pp. 857–859, 1988.
- [37] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

- [38] F. Dörfler, P. Tesi, and C. De Persis, “On the certainty-equivalence approach to direct data-driven LQR design,” *IEEE Transactions on Automatic Control*, vol. 68, no. 12, pp. 7989–7996, 2023.
- [39] 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.
- [40] S. Kang, F. Zhao, and K. You, “Linear convergence of data-enabled policy optimization for linear quadratic tracking,” *arXiv preprint arXiv:2410.05596*, 2024.
- [41] I. D. Landau, R. Lozano, M. M’Saad, and A. Karimi, *Adaptive control: algorithms, analysis and applications*. Springer Science & Business Media, 2011.
- [42] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1.” <http://cvxr.com/cvx>, Mar. 2014.

Chapter 11

Paper E

A Modified Adaptive Data-Enabled Policy Optimization Control to Resolve State Perturbations

Mojtaba Kaheni, Niklas Persson, Vittorio De Iuliis, Costanzo Manes, Alessandro V. Papadopoulos.

Accepted to Conference on Decision and Control (CDC), 2025

Abstract

This paper proposes modifications to the data-enabled policy optimization (DeePO) algorithm to mitigate state perturbations. DeePO is an adaptive, data-driven approach designed to iteratively compute a feedback gain equivalent to the certainty-equivalence LQR gain. Like other data-driven approaches based on Willems’ fundamental lemma, DeePO requires persistently exciting input signals. However, linear state-feedback gains from LQR designs cannot inherently produce such inputs. To address this, probing noise is conventionally added to the control signal to ensure persistent excitation. However, the added noise may induce undesirable state perturbations. We first identify two key issues that jeopardize the desired performance of DeePO when probing noise is not added: the convergence of states to the equilibrium point, and the convergence of the controller to its optimal value. To address these challenges without relying on probing noise, we propose Perturbation-Free DeePO (PFDeePO) built on two fundamental principles. First, the algorithm pauses the control gain updating in DeePO process when system states are near the equilibrium point. Second, it applies a multiplicative noise, scaled by a mean value of 1 as a gain for the control signal, when the controller converges. This approach minimizes the impact of noise as the system approaches equilibrium while preserving stability. We demonstrate the effectiveness of PFDeePO through simulations, showcasing its ability to eliminate state perturbations while maintaining system performance and stability.

11.1 Introduction

Traditionally, the design of controllers has relied on two primary approaches: first-principle modeling which is based on established physical laws and principles specific to the system's domain, or system identification that uses available data to construct a mathematical representation. These mathematical models, are then employed to analyze the system's behavior under various inputs and to derive a control law that satisfies specified performance criteria.

However, the seminal work by Willems *et al.* [1] introduced a groundbreaking concept with far-reaching implications for control system design. This study revealed that a finite set of system trajectories, generated using persistently exciting inputs, is sufficient to describe the complete behavior of a controllable linear time-invariant (LTI) system. It implies that controllable LTI systems can be fully characterized using only finite historical data, eliminating the need for traditional model-based representations. This result has sparked widespread interest within the control systems community, as it presents a promising alternative for simplifying controller design. By potentially circumventing the expensive and time-consuming processes of system identification or first-principle modeling, this approach offers a new framework for streamlining and accelerating the development of control systems.

Linear quadratic regulators (LQR) are a widely adopted control design method due to their ability to balance the trade-off between convergence rate and control effort, which can be adjusted based on the designer's preferences. In LTI systems, when the mathematical state-space representation of the system is available, solving a straightforward optimization problem provides the state-feedback gain that minimizes the corresponding cost function [2]. In cases where, instead of the state-space representation, sufficiently rich measurements are available, it is still possible to design an LQR controller by conventional approaches, but an initial identification step is required to first find the system matrices. This approach is typically referred to in the literature as *indirect data-driven* LQR design [3, 4, 5, 6].

In contrast, several methods have recently been proposed to design LQR controllers for LTI systems directly from data, without the need to identify the system matrices [7, 8, 9, 10, 11, 12, 13, 14]. These approaches are commonly known as *direct data-driven* LQR design. Most of the direct data-driven LQR design methods rely on pre-measured datasets to compute the optimal feedback gain. A natural consideration is to allow the control system to leverage the data it measures during operation to improve or fine-tune the LQR performance. Toward this goal, the Data-enabled Policy Optimization (DeePO) algorithm was recently proposed [15, 16]. DeePO incorporates an adaptation

feature on top of direct data-driven LQR design. At each time step, newly measured input and state data are added to the previously stored dataset, and the control feedback gain is updated iteratively using a learning rate, steering the design towards reducing the objective function's cost based on the updated data. Furthermore, it has been proven that DeePO computes a feedback gain equivalent to the certainty-equivalence LQR gain typically derived using indirect data-driven techniques [16]. DeePO has been evaluated in simulations on a power converter system [17], and was used for balancing an autonomous bicycle in real experiments [18]. Similar to other data-driven methods based on Willems' fundamental lemma, DeePO requires persistently exciting input signals. On the other hand, linear state-feedback gains from LQR designs cannot inherently generate such inputs. To overcome this limitation, probing noise is conventionally added to the control signal to ensure persistent excitation. This added noise often induces undesirable state perturbations. In this article, we introduce PFDeePO to resolve state perturbations in DeePO.

11.1.1 Notation

Throughout this paper, unless clearly stated otherwise, the symbols $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}$ denote the sets of, natural, integer, and real numbers, respectively. Scalars are represented by lowercase letters such as x , while $\mathbf{x}$ and $\mathbf{X}$ denote a (column) vector and a matrix, respectively. The notation $\mathbf{X} < 0$ ($\mathbf{X} \leq 0$) and $\mathbf{X} > 0$ ($\mathbf{X} \geq 0$) indicates that $\mathbf{X}$ is negative (semi-) definite and positive (semi-) definite, respectively. Additionally, $\mathbf{I}_i$ denotes the $i \times i$ identity matrix. The symbols $\mathbf{0}_{i \times j}$ and $\mathbf{1}_{i \times j}$ represent all-zeros and all-ones matrices of size $i \times j$, respectively. The 2-norm of matrix $\mathbf{X}$ is denoted by $\|\mathbf{X}\|$. $\mathbf{X}^\top$, $\text{tr}(\mathbf{X})$, and $\mathbf{X}^\dagger$ represent the transpose, trace, and pseudoinverse of matrix $\mathbf{X}$, respectively. The notation $\mathbf{x}^i$ and $\mathbf{X}^i$ refer to the i -th coordinate of $\mathbf{x}$ and the i -th row of $\mathbf{X}$, respectively. Meanwhile, $\mathbf{X}^{i,j}$ denotes the element located in the i -th row and j -th column of $\mathbf{X}$. $\mathcal{N}(\mathbf{x}, \mathbf{X})$ represents a multivariate Gaussian distribution with a mean vector $\mathbf{x}$ and a covariance matrix $\mathbf{X}$. $\Pi_{\mathbf{X}}$ stands for projection operator on $\mathbf{X}$. Finally, $\underline{\sigma}(\mathbf{X})$ denotes the smallest singular value of $\mathbf{X}$.

11.2 Data-Driven Policy Optimization for LQR Learning

11.2.1 Background

Consider an LTI discrete-time system, represented in state space form as:

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k + \boldsymbol{\omega}_k, \\ \mathbf{z}_k &= \begin{bmatrix} \mathbf{Q}^{1/2} & \mathbf{0}_{n \times m} \\ \mathbf{0}_{m \times n} & \mathbf{R}^{1/2} \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{u}_k \end{bmatrix}, \end{aligned} \quad (11.1)$$

where $k \in \mathbb{N}$ is the index for counting samples, $\mathbf{x} \in \mathbb{R}^n$ is the state, $\mathbf{u} \in \mathbb{R}^m$ represents the input, and $\boldsymbol{\omega}_k$ is noise. Furthermore, let $\mathbf{z}_k \in \mathbb{R}^{n+m}$ represent the performance signal. We assume that the pair $(\mathbf{A}, \mathbf{B})$ is controllable, and that $(\mathbf{Q}, \mathbf{R})$ are positive definite square matrices with compatible dimensions. The objective of the LQR design is to determine a state feedback controller, $\mathbf{K} \in \mathbb{R}^{m \times n}$, that minimizes the $\mathcal{H}_2$ -norm of the transfer function $\mathcal{T}(\mathbf{K}) : \boldsymbol{\omega} \mapsto \mathbf{z}$ of the following closed-loop system:

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{z}_k \end{bmatrix} = \begin{bmatrix} \mathbf{A} + \mathbf{BK} & \mathbf{I}_n \\ \begin{bmatrix} \mathbf{Q}^{1/2} \\ \mathbf{R}^{1/2}\mathbf{K} \end{bmatrix} & \mathbf{0}_{(m+n) \times n} \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \boldsymbol{\omega}_k \end{bmatrix}. \quad (11.2)$$

As discussed in [19], the $\mathcal{H}_2$ -norm of the transfer function $\mathcal{T}(\mathbf{K})$ obtained from (11.2) can be expressed as:

$$C(\mathbf{K}) \triangleq \|\mathcal{T}(\mathbf{K})\|^2 = \text{tr} \left((\mathbf{Q} + \mathbf{K}^\top \mathbf{R} \mathbf{K}) \boldsymbol{\Sigma}_{\mathbf{K}} \right). \quad (11.3)$$

where $C(\mathbf{K})$ represents the cost function, and $\boldsymbol{\Sigma}_{\mathbf{K}}$ is typically referred to as the closed-loop state covariance matrix, which is the solution to the following Lyapunov equation:

$$\boldsymbol{\Sigma}_{\mathbf{K}} = \mathbf{I}_n + (\mathbf{A} + \mathbf{BK}) \boldsymbol{\Sigma}_{\mathbf{K}} (\mathbf{A} + \mathbf{BK})^\top. \quad (11.4)$$

Therefore, the LQR design can be summarized as:

$$\begin{aligned} \min_{\boldsymbol{\Sigma}_{\mathbf{K}} \geq 0, \mathbf{K}} \quad & C(\mathbf{K}) = \text{Tr} \left((\mathbf{Q} + \mathbf{K}^\top \mathbf{R} \mathbf{K}) \boldsymbol{\Sigma}_{\mathbf{K}} \right) \\ \text{subject to} \quad & \boldsymbol{\Sigma}_{\mathbf{K}} = \mathbf{I}_n + (\mathbf{A} + \mathbf{BK}) \boldsymbol{\Sigma}_{\mathbf{K}} (\mathbf{A} + \mathbf{BK})^\top. \end{aligned} \quad (11.5)$$

To directly compute the optimal feedback gain matrix $\mathbf{K}$ from (11.5), the system matrices $(\mathbf{A}, \mathbf{B})$ must be known. However, if $(\mathbf{A}, \mathbf{B})$ are unknown, it may still be possible to incorporate an identification step to estimate them. Suppose

signals of length t of states, inputs, noises, and successor states, which do not necessarily need to be consecutive. These signals are defined as follows:

$$\begin{aligned}\mathbf{X}_0 &\triangleq [\mathbf{x}_0 \quad \mathbf{x}_1 \quad \cdots \quad \mathbf{x}_{t-1}], \\ \mathbf{X}_1 &\triangleq [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_t], \\ \mathbf{U}_0 &\triangleq [\mathbf{u}_0 \quad \mathbf{u}_1 \quad \cdots \quad \mathbf{u}_{t-1}], \\ \mathbf{W}_0 &\triangleq [\boldsymbol{\omega}_0 \quad \boldsymbol{\omega}_1 \quad \cdots \quad \boldsymbol{\omega}_{t-1}].\end{aligned}\tag{11.6}$$

The input signal $\mathbf{U}_0$ must be *sufficiently rich* to effectively represent the dynamical system described by (11.1). This property is commonly referred to as *persistently exciting*, and is formally defined as follows:

Definition 11.2.1 ([1]). *A signal $\mathbf{U}_0$ is said to be persistently exciting of order l when*

$$\mathcal{U}_0 = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_{t-l} \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_{t-l+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_{l-1} & \mathbf{u}_l & \cdots & \mathbf{u}_{t-1} \end{bmatrix},\tag{11.7}$$

has full rank ml . ■

The following lemma is also useful for determining the persistent excitation of a system.

Lemma 11.1 ([1]). *If the system (11.1) is controllable and $\mathbf{U}_0$ is persistently exciting of order $n + 1$, then $\text{rank}(\mathcal{D}) = n + m$, where*

$$\mathcal{D} \triangleq \begin{bmatrix} \mathbf{U}_0 \\ \mathbf{X}_0 \end{bmatrix}.\tag{11.8}$$
■

If $\mathbf{U}_0$ is persistently exciting, the estimates of the system matrices $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$ can be obtained by solving the following optimization problem:

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \min_{\mathbf{A}, \mathbf{B}} \left\| \mathbf{X}_1 - [\mathbf{B} \quad \mathbf{A}] [\mathbf{U}_0^\top \quad \mathbf{X}_0^\top]^\top \right\|.\tag{11.9}$$

The LQR controller can then be designed by substituting $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$, obtained from (11.9), in place of the true system matrices $(\mathbf{A}, \mathbf{B})$ in (11.5). This approach is commonly referred to as *certainty-equivalence* and is a typical strategy in *indirect data-driven* LQR design [4, 20, 21, 22].

Recently, several methods have been proposed in the literature to bypass the identification step in (11.9) by directly leveraging the data introduced in

(11.6). These methods are commonly referred to as *direct data-driven* LQR design [7, 8, 9, 10, 11, 12, 13]. From Lemma 11.1, we know that if $\mathbf{U}_0$ is persistently exciting of order $n + 1$, then $\text{rank}(\mathcal{D}) = n + m$. Consequently, by the Rouché–Capelli theorem [23], there exists a matrix $\mathbf{G} \in \mathbb{R}^{t \times n}$ such that:

$$\begin{bmatrix} \mathbf{K} \\ \mathbf{I}_n \end{bmatrix} = \mathcal{D}\mathbf{G}. \quad (11.10)$$

Substituting into the state-space representation, we have:

$$\mathbf{A} + \mathbf{B}\mathbf{K} = [\mathbf{B} \quad \mathbf{A}] \begin{bmatrix} \mathbf{K} \\ \mathbf{I}_n \end{bmatrix} = [\mathbf{B} \quad \mathbf{A}] \mathcal{D}\mathbf{G}. \quad (11.11)$$

On the other hand, the measured data in (11.6) must satisfy the system dynamics:

$$\mathbf{X}_1 = \mathbf{A}\mathbf{X}_0 + \mathbf{B}\mathbf{U}_0 + \mathbf{W}_0. \quad (11.12)$$

By substituting the definition of $\mathcal{D}$ from Lemma 11.1 into (11.11) and incorporating (11.12), we obtain:

$$\mathbf{A} + \mathbf{B}\mathbf{K} = (\mathbf{X}_1 - \mathbf{W}_0)\mathbf{G}. \quad (11.13)$$

Since $\mathbf{W}_0$ is unknown and cannot be directly accounted for, we approximate $\mathbf{A} + \mathbf{B}\mathbf{K}$ with $\mathbf{X}_1\mathbf{G}$ and $\mathbf{K}$ with $\mathbf{U}_0\mathbf{G}$ in the LQR optimization problem. This leads to the following direct data-driven LQR optimization formulation:

$$\begin{aligned} \min_{\Sigma_{\mathbf{K}} \geq 0, \mathbf{G}} \quad & C(\mathbf{G}) = \text{tr} \left((\mathbf{Q} + \mathbf{G}^\top \mathbf{U}_0^\top \mathbf{R} \mathbf{U}_0 \mathbf{G}) \Sigma_{\mathbf{K}} \right) \\ \text{subject to} \quad & \Sigma_{\mathbf{K}} = \mathbf{I}_n + \mathbf{X}_1 \mathbf{G} \Sigma_{\mathbf{K}} \mathbf{G}^\top \mathbf{X}_1^\top, \\ & \mathbf{X}_0 \mathbf{G} = \mathbf{I}_n. \end{aligned} \quad (11.14)$$

The optimal feedback gain is then given by $\mathbf{K} = \mathbf{U}_0\mathbf{G}$.

In [16], the authors introduce an alternative policy parametrization based on the sample covariance of the data, defined as:

$$\Phi \triangleq \frac{1}{t} \mathcal{D} \mathcal{D}^\top = \begin{bmatrix} \mathbf{U}_0 \mathcal{D}^\top / t \\ \mathbf{X}_0 \mathcal{D}^\top / t \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{U}_0} \\ \overline{\mathbf{X}_0} \end{bmatrix}. \quad (11.15)$$

Defining $\mathbf{V} \in \mathbb{R}^{(n+m) \times n}$ as the solution to:

$$\begin{bmatrix} \mathbf{K} \\ \mathbf{I}_n \end{bmatrix} = \Phi \mathbf{V}, \quad (11.16)$$

and following steps similar to (11.11)–(11.13), the data-driven LQR optimization problem can be reformulated as:

$$\begin{aligned} \min_{\Sigma_V \geq 0, V} \quad & C(V) = \text{tr} \left(\left(Q + V^\top \bar{U}_0^\top R \bar{U}_0 V \right) \Sigma_V \right) \\ \text{subject to} \quad & \Sigma_V = I_n + \bar{X}_1 V \Sigma_V V^\top \bar{X}_1^\top, \\ & \bar{X}_0 V = I_n, \end{aligned} \tag{11.17}$$

where $\bar{X}_1 = X_1 D^\top / t$. Since the dimension of V is independent of the number of samples, t , this formulation is particularly advantageous in adaptive design strategies where the sample size grows linearly.

In the DeePO algorithm, starting from an initial feasible solution K_t , the feedback gain evolves iteratively via a gradient descent approach to reach the optimal solution K^* . The following lemma provides a methodology for computing $\widehat{V_V C}$ solely based on data.

Lemma 11.2 ([16]). *Let P_V be the unique solution for the Lyapunov equation*

$$P_V = Q + V^\top \bar{U}_0^\top R \bar{U}_0 V + V^\top \bar{X}_1^\top P_V \bar{X}_1 V,$$

then

$$\widehat{V_V C} = 2 \left(\bar{U}_0^\top R \bar{U}_0 + \bar{X}_1^\top P_V \bar{X}_1 \right) V \Sigma_V.$$

■

Algorithm 11.1 outlines the steps required to execute the DeePO algorithm.

In summary, to ensure the convergence of Algorithm 11.1, the optimization problem (11.17) must be feasible, U_0 must be persistently exciting, and both ω_k and $\mathbf{x}_k$ must remain bounded to maintain stability.

11.2.2 Why is probing noise added to u_i in Algorithm 11.1?

To begin, we present a lemma demonstrating that state feedback control of the form $\mathbf{u}_k = K \mathbf{x}_k$ is incapable of generating a persistently exciting sequence U_0 .

Lemma 11.3. *State feedback control of the form $\mathbf{u}_k = K \mathbf{x}_k$, where K is a gain matrix and $\mathbf{x}_k$ is the state vector at time step k , cannot generate a persistently exciting sequence U_0 .*

Algorithm 11.1 Data-Enabled Policy Optimization (DeePO) [16]

Require: $\mathbf{U}_0, \mathbf{X}_0, \mathbf{X}_1, \mathbf{K}_t$, a probing noise signal $\{\mathbf{e}\}$, and a learning rate η .

Start

$i = t$.

while the stop criterion is not satisfied, do:

Apply $\mathbf{u}_i = \mathbf{K}_i \mathbf{x}_i + \mathbf{e}_i$ and observe $\mathbf{x}_{i+1}$.

Update $\mathbf{X}_0$ by $\mathbf{X}_0 = [\mathbf{X}_0, \mathbf{x}_i]$.

Update $\mathbf{X}_1$ by $\mathbf{X}_1 = [\mathbf{X}_1, \mathbf{x}_{i+1}]$.

Update $\mathbf{U}_0$ by $\mathbf{U}_0 = [\mathbf{U}_0, \mathbf{u}_i]$.

$$\mathbf{V}_{i+1} = \Phi_{i+1}^{-1} \begin{bmatrix} \mathbf{K}_i \\ \mathbf{I}_n \end{bmatrix}.$$

$$\mathbf{V}'_{i+1} = \mathbf{V}_{i+1} - \eta \Pi_{\bar{\mathbf{X}}_0} \widehat{\nabla_{\mathbf{V}} C}.$$

Update the control gain by $\mathbf{K}_{i+1} = \bar{\mathbf{U}}_0 \mathbf{V}'_{i+1}$.

$i = i + 1$.

End while

End

Proof. From Lemma 11.1, we construct the matrix $\mathcal{D}$ as follows:

$$\mathcal{D} = \begin{bmatrix} \sum_{j=1}^n \mathbf{k}^{1,j} \mathbf{x}_0^j & \sum_{j=1}^n \mathbf{k}^{1,j} \mathbf{x}_1^j & \cdots & \sum_{j=1}^n \mathbf{k}^{1,j} \mathbf{x}_{t-1}^j \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{j=1}^n \mathbf{k}^{m,j} \mathbf{x}_0^j & \sum_{j=1}^n \mathbf{k}^{m,j} \mathbf{x}_1^j & \cdots & \sum_{j=1}^n \mathbf{k}^{m,j} \mathbf{x}_{t-1}^j \\ \mathbf{x}_0^1 & \mathbf{x}_1^1 & \cdots & \mathbf{x}_{t-1}^1 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_0^n & \mathbf{x}_1^n & \cdots & \mathbf{x}_{t-1}^n \end{bmatrix}, \quad (11.18)$$

where first m rows of $\mathcal{D}$ can be expressed as linear combinations of the last n rows. Consequently, $\text{rank}(\mathcal{D}) = n$. From Lemma 11.1, we observe that the condition $\text{rank}(\mathcal{D}) \neq n + m$ implies that the sequence $\mathbf{U}_0$ is not persistently exciting. This completes the proof. ■

In addition, the following lemma provides a more transparent perspective on the subject.

Lemma 11.4. Let $\mathcal{D}$ be defined as the horizontal concatenation of two matrices, $\mathcal{D} = [\mathcal{D}_1, \mathcal{D}_2]$,

where:

- $\mathcal{D}_1 \in \mathbb{R}^{(n+m) \times t_1}$ with $\text{rank}(\mathcal{D}_1) = n + m$, and

- $D_2 \in \mathbb{R}^{(n+m) \times t_2}$ with $\text{rank}(D_2) < n + m$.

If $t_2 \rightarrow \infty$, then $\lim_{t_2 \rightarrow \infty} \underline{\sigma}(\Phi) \rightarrow 0$.

Proof. First, notice that Φ defined in (11.15) is only a function of t_2 , since t_1 is fixed:

$$\Phi(t_2) = \frac{1}{t_1 + t_2} (D_1 D_1^\top + D_2 D_2^\top). \quad (11.19)$$

Since D_2 is not full row rank for all t_2 , there exists a non-zero vector $\mathbf{a} \in \mathbb{R}^{n+m}$ such that $\mathbf{a}^\top D_2 = 0$. Thus

$$\begin{aligned} \mathbf{a}^\top \Phi(t_2) \mathbf{a} &= \frac{1}{t_1 + t_2} (\mathbf{a}^\top D_1 D_1^\top \mathbf{a} + \mathbf{a}^\top D_2 D_2^\top \mathbf{a}) \\ &= \frac{1}{t_1 + t_2} \mathbf{a}^\top D_1 D_1^\top \mathbf{a}. \end{aligned} \quad (11.20)$$

From this, it is clear that

$$\lim_{t_2 \rightarrow \infty} \mathbf{a}^\top \Phi(t_2) \mathbf{a} = 0. \quad (11.21)$$

Since $\underline{\sigma}(\Phi(t_2))$ is such that

$$\underline{\sigma}(\Phi(t_2)) \|\mathbf{a}\|^2 \leq \mathbf{a}^\top \Phi(t_2) \mathbf{a}, \quad \forall t_2 \in \mathbb{N}, \quad (11.22)$$

then the limit (11.21) implies

$$\lim_{t_2 \rightarrow \infty} \underline{\sigma}(\Phi(t_2)) \rightarrow 0.$$

This concludes the proof. ■

The results from Lemmas 11.3 and 11.4 reveal critical limitations for applying conventional control laws, such as $\mathbf{u}_i = \mathbf{K}_i \mathbf{x}_i$, to the system:

- As $\mathbf{x}_i \rightarrow 0$, the columns of D also approach zero, and D can be partitioned such that the second matrix is zero. By Lemma 11.4, $\lim_{t_2 \rightarrow \infty} \underline{\sigma}(\Phi) \rightarrow 0$, jeopardizing the full rank of Φ needed to ensure (11.16).
- As $\mathbf{K}_i \rightarrow \mathbf{K}$, Lemma 11.3 indicates that adding columns to D is equivalent to appending a singular matrix to D_1 , pushing Φ toward singularity.

To address this undesired outcome, which compromises DeePO's performance, the authors in [16] propose adding a probing noise to the input. This noise ensures that the input remains sufficiently rich, thereby preserving the full rank of Φ and maintaining persistent excitation.

11.2.3 Undesired State Perturbations by Adding Probing Noise to $\mathbf{u}_i$

In asymptotically stable LTI systems, the system naturally drives the state $\mathbf{x}_i$ towards equilibrium as time progresses. However, when probing noise is added to the input signal to maintain persistent excitation, the noise introduces high-frequency components into the control input. These high-frequency components can interact with the feedback dynamics, causing rapid oscillations or fluctuations in the control signal and, consequently, the system state. States perturbations is particularly problematic in practical implementations, as it can lead to actuator wear, increased energy consumption, and degraded overall system performance.

11.3 Perturbations-free DeePO (PFDeePO)

As discussed in Section 11.2, the convergence of $\mathbf{x}_i \rightarrow 0$ and $\mathbf{K}_i \rightarrow \mathbf{K}$ can compromise the full rank of Φ . In Algorithm 11.2, we present our proposed method, PFDeePO, which serves as an alternative to DeePO, ensuring the full rank of Φ is preserved while avoiding performance degradation caused by state perturbations.

The main idea behind PFDeePO is to prevent conditions that may compromise the full rank of Φ . To achieve this, we introduce two additional positive scalars, $\gamma > 0$ and $\delta > 0$.

At each iteration, the first “if-else” block ensures that arbitrarily scaling the control signal by a random gain, v_i , where $\underline{v} \leq v_i \leq \bar{v}$, occurs only when necessary. Specifically, this scaling is applied only if the control gain, $\mathbf{K}_i$, has already converged to its optimal value, yet the system states remain relatively far from the equilibrium point.

Next, in the second “if-else” block, we evaluate the convergence of the states. The condition $\|\mathbf{x}_i\| < \gamma$ represents a scenario where the states are close to the equilibrium point, causing the control signal to approach zero. In this situation, the data gathered is not sufficiently informative to update the controller. Conversely, if the states have already reached near-equilibrium, further efforts to improve the control law are unnecessary, as the states have effectively converged, and any additional effort is unlikely to yield meaningful improvements in classical control performance metrics.

To formalize PFDeePO, we first need to ensure that the matrix Φ_i retains full rank when Algorithm 11.2 is applied.

Theorem 11.5. *Let Φ_i be the matrix constructed at time step i during the execution of PFDeePO. By implementing Algorithm 11.2, the matrix Φ_i attains*

full rank, i.e., $\text{rank}(\Phi_i) = n + m$, and as a result, $\underline{\sigma}(\Phi_i) > 0$.

Proof. From (11.15) we see that $\text{rank}(\Phi_i) = \text{rank}(\mathcal{D})$. We proceed by contradiction. Assume that $\text{rank}(\mathcal{D}) < n + m$. This implies the existence of scalars $\lambda_i, \mu_j \in \mathbb{R}$ such that:

$$\sum_{i=1}^n \lambda_i \mathbf{X}_0^i = \sum_{j=1}^m \mu_j \mathbf{U}_0^j, \quad (11.23)$$

where $\mathbf{X}_0^i$ denotes the i -th row of $\mathbf{X}_0$, and $\mathbf{U}_0^j$ denotes the j -th row of $\mathbf{U}_0$. Note that the second if-else block, avoids presence of all zeros columns in $\mathbf{X}_0$ and $\mathbf{U}_0$ and recall that at each time step k , $\mathbf{U}_0^{j,k} = \sum_{i=1}^n \mathbf{k}_k^{j,i} \mathbf{x}_k^i$, which holds for all $k = 0, \dots, t-1$, we substitute into (11.23) to obtain:

$$\sum_{i=1}^n \lambda_i \mathbf{x}_k^i = \sum_{j=1}^m \sum_{i=1}^n \mu_j \mathbf{k}_k^{j,i} \mathbf{x}_k^i. \quad (11.24)$$

For (11.24) to hold for all $k = 0, \dots, t-1$ and $i = 1, \dots, n$, we must have:

$$\lambda_i = \sum_{j=1}^m \mu_j \mathbf{k}_k^{j,i}. \quad (11.25)$$

However, this condition (11.25) is violated under both cases defined by Algorithm 11.2:

- When $\mathbf{K}_k \neq \mathbf{K}_{k-1}$, the matrices $\mathbf{K}_k$ are distinct, and it is impossible for λ_i to be expressed as a consistent linear combination of $\mu_j \mathbf{k}_k^{j,i}$. In other words, since the i -th column of the controller changes at each time step, the value of λ_i obtained at time step k as a linear combination of $\mu_j \mathbf{k}_k^{j,i}$ will differ from the values obtained at other time steps and it is not feasible to find a consistent λ_i for all $k = 0, \dots, t-1$.
- When $\mathbf{K}_k = v_k \mathbf{K}$, where v_k is randomly sampled from an arbitrary distribution, the randomness in v_k ensures that λ_i will be different value at each time step in $\lambda_i = \sum_{j=1}^m v_k \mu_j \mathbf{k}^{j,i}$.

Thus, the assumption that $\text{rank}(\mathcal{D}) < n + m$ leads to a contradiction. Therefore, we conclude that:

$$\text{rank}(\mathcal{D}) = n + m.$$

■

Another critical aspect to address is ensuring that randomly scaling the control signal by v_i does not compromise the closed-loop stability. First recall that from Lemma 1 in [16], the converged control gain $\mathbf{K}$ obtained by DeePO is equivalent to indirect LQR solution. Therefore,

$$\mathbf{K} = \left(\hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} + \mathbf{R} \right)^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}}, \quad (11.26)$$

where $\mathbf{H}$ is the solution of the following Discrete Algebraic Riccati Equation (DARE)

$$\mathbf{H} = \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{A}} + \mathbf{Q} - \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{B}} \left(\hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} + \mathbf{R} \right)^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}}. \quad (11.27)$$

and $\mathbf{Q} \in \mathbb{R}^{n \times n}$ and $\mathbf{R} \in \mathbb{R}^{m \times m}$ are given symmetric positive definite matrices. In the following, we establish the conditions that $\bar{v}$ and $\underline{v}$ must satisfy to guarantee stability in Algorithm 11.2. We begin by the following lemma.

Lemma 11.6. *Consider matrices $\hat{\mathbf{B}} \in \mathbb{R}^{n \times m}$ and $\mathbf{K} \in \mathbb{R}^{m \times n}$, and symmetric positive definite matrices $\mathbf{Q} \in \mathbb{R}^{n \times n}$ and $\mathbf{R} \in \mathbb{R}^{m \times m}$. Then, there exists an interval $\mathcal{V} = [\underline{v}, \bar{v}]$, where $0 < \underline{v} < 1 < \bar{v}$, such that*

$$\mathbf{Q} - \mathbf{K}^\top ((v-1)^2 \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} + (1-2v)\mathbf{R}) \mathbf{K} \geq 0, \quad \forall v \in [\underline{v}, \bar{v}]. \quad (11.28)$$

Proof. Note first that for $v = 1$, we have

$$\begin{aligned} & \left(\mathbf{Q} - \mathbf{K}^\top ((v-1)^2 \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} + (1-2v)\mathbf{R}) \mathbf{K} \right)_{v=1} \\ &= \mathbf{Q} + \mathbf{K}^\top \mathbf{R} \mathbf{K} > 0. \end{aligned} \quad (11.29)$$

Since $\mathbf{Q} > 0$ by assumption and $\mathbf{K}^\top \mathbf{R} \mathbf{K} \geq 0$, the matrix in the left-hand side of (11.28) is positive definite at $v = 1$. Recall that the eigenvalues of a square matrix, whose components depend continuously on a parameter, are also continuous functions of that parameter. As a result, there exists an open neighborhood $(\underline{v}, \bar{v})$ around $v = 1$ where all eigenvalues of the matrix remain strictly positive, ensuring that the matrix is positive definite.

At the endpoints $\underline{v}$ and $\bar{v}$, the matrix becomes positive semidefinite, meaning some of its eigenvalues reach zero. Consequently, the inequality (11.28) holds for all $v \in [\underline{v}, \bar{v}]$, completing the proof. ■

Remark 5. *In Lemma 11.6, we show that there exists an interval $\mathcal{V} = [\underline{v}, \bar{v}]$ such that (11.28) holds for all $v \in \mathcal{V}$. A practical approach to determine $\mathcal{V}$ is to start with a small interval around 1 and iteratively narrow it, checking Lemma 11.6 at each step, until the inequality is satisfied. □*

Now, we are ready to present our main theorem.

Theorem 11.7. *Consider a system controlled by Algorithm 11.2. Let $\mathbf{Q} \in \mathbb{R}^{n \times n}$ and $\mathbf{R} \in \mathbb{R}^{m \times m}$ be given symmetric and positive definite matrices, and let $\beta \in (0, 1)$.*

Let's represent the converged control gain by the matrix $\mathbf{K} \in \mathbb{R}^{m \times n}$, since $\mathbf{K}$ is equal to a certainly equivalent LQR gain, we have

$$\mathbf{K} = (\mathbf{R} + \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}})^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}}, \quad (11.30)$$

where $\mathbf{H} \in \mathbb{R}^{n \times n}$ is the unique solution of the modified DARE:

$$\hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{A}} - \beta^2 \mathbf{H} - \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{B}} (\mathbf{R} + \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}})^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}} + \mathbf{Q} = \mathbf{0}_{n \times n}. \quad (11.31)$$

Consider the interval $[\underline{v}, \bar{v}]$, as defined in Lemma 11.6, such that the inequality (11.28) holds. Then, the time-varying state feedback control law, $\mathbf{u}_k = -v_k \mathbf{K} \mathbf{x}_k$, where v_k is any sequence taking values in the interval $[\underline{v}, \bar{v}]$, ensures that the origin of the closed-loop system

$$\mathbf{x}_{k+1} = (\hat{\mathbf{A}} - v_k \hat{\mathbf{B}} \mathbf{K}) \mathbf{x}_k, \quad k = 0, 1, \dots \quad (11.32)$$

is exponentially stable. Specifically, the state satisfies the bound

$$\|\mathbf{x}_k\| \leq \beta^k \sqrt{\frac{h_{\max}}{h_{\min}}} \|\mathbf{x}_0\|, \quad k = 0, 1, \dots, \quad (11.33)$$

where $h_{\max} = \lambda_{\max}(\mathbf{H})$, and $h_{\min} = \lambda_{\min}(\mathbf{H})$.

Proof. For a compact notation, the gain $\mathbf{K}$ defined in (11.30) will be written as

$$\mathbf{K} = \mathbf{S}^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}}, \quad \text{where} \quad \mathbf{S} = \mathbf{R} + \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}}, \quad (11.34)$$

so that the term $\hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{B}} (\mathbf{R} + \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}})^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}}$ in the modified DARE (11.31) can be equivalently written as

$$\hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{B}} \mathbf{S}^{-1} \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}} = \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{B}} \mathbf{K} = \mathbf{K}^\top \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}} = \mathbf{K}^\top \mathbf{S} \mathbf{K}, \quad (11.35)$$

and the modified DARE (11.31) can be rewritten as

$$\hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{A}} - \beta^2 \mathbf{H} - \mathbf{K}^\top \mathbf{S} \mathbf{K} + \mathbf{Q} = \mathbf{0}_{n \times n}. \quad (11.36)$$

From this, we get the identity

$$\hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{A}} - \beta^2 \mathbf{H} = -\mathbf{Q} + \mathbf{K}^\top \mathbf{S} \mathbf{K}. \quad (11.37)$$

Now, consider the Lyapunov function candidate $V(\mathbf{x}) = \mathbf{x}^\top \mathbf{H} \mathbf{x}$. We will prove that

$$V(\mathbf{x}_{k+1}) \leq \beta^2 V(\mathbf{x}_k), \quad k = 0, 1, \dots, \quad (11.38)$$

independently of the sequence v_k , provided that $v_k \in \mathcal{V}$, so that

$$V(\mathbf{x}_k) \leq \beta^{2k} V(\mathbf{x}_0), \quad k = 0, 1, \dots \quad (11.39)$$

From this, recalling that $h_{\min} \|\mathbf{x}\|^2 \leq \mathbf{x}^\top \mathbf{H} \mathbf{x} \leq h_{\max} \|\mathbf{x}\|^2$, we easily get

$$h_{\min} \|\mathbf{x}_k\|^2 \leq \beta^{2k} h_{\max} \|\mathbf{x}_0\|^2, \quad (11.40)$$

from which, taking the square roots of both terms of the inequality, the inequality (11.33) follows.

Thus, to prove the Theorem, i.e., the inequalities (11.33), it is sufficient to prove the inequalities (11.38). These can be rewritten as

$$V(\mathbf{x}_{k+1}) - \beta^2 V(\mathbf{x}_k) \leq 0, \quad k = 0, 1, \dots, \quad (11.41)$$

or

$$\mathbf{x}_{k+1}^\top \mathbf{H} \mathbf{x}_{k+1} - \beta^2 \mathbf{x}_k^\top \mathbf{H} \mathbf{x}_k \leq 0, \quad k = 0, 1, \dots \quad (11.42)$$

Exploiting the system equation (11.32), these inequalities change to the following

$$\mathbf{x}_k^\top ((\hat{\mathbf{A}} - v_k \hat{\mathbf{B}} \mathbf{K})^\top \mathbf{H} (\hat{\mathbf{A}} - v_k \hat{\mathbf{B}} \mathbf{K}) - \beta^2 \mathbf{H}) \mathbf{x}_k \leq 0, \quad (11.43)$$

where $v_k \in [\underline{v}, \bar{v}]$. It is clear that all inequalities (11.43) are verified if the following inequality is true:

$$(\hat{\mathbf{A}} - v \hat{\mathbf{B}} \mathbf{K})^\top \mathbf{H} (\hat{\mathbf{A}} - v \hat{\mathbf{B}} \mathbf{K}) - \beta^2 \mathbf{H} \leq 0, \quad \forall v \in [\underline{v}, \bar{v}]. \quad (11.44)$$

Upon carrying out the multiplications in the left-hand side term of the inequality and considering the identities (11.35) and (11.37), we have

$$\begin{aligned} & (\hat{\mathbf{A}} - v \hat{\mathbf{B}} \mathbf{K})^\top \mathbf{H} (\hat{\mathbf{A}} - v \hat{\mathbf{B}} \mathbf{K}) - \beta^2 \mathbf{H} \\ &= \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{A}} - \beta^2 \mathbf{H} - v \mathbf{K}^\top \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{A}} - v \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{B}} \mathbf{K} \\ & \quad + v^2 \mathbf{K}^\top \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} \mathbf{K} \\ &= \hat{\mathbf{A}}^\top \mathbf{H} \hat{\mathbf{A}} - \beta^2 \mathbf{H} - 2v \mathbf{K}^\top \mathbf{S} \mathbf{K} + v^2 \mathbf{K}^\top \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} \mathbf{K} \\ &= -\mathbf{Q} + \mathbf{K}^\top \mathbf{S} \mathbf{K} - 2v \mathbf{K}^\top \mathbf{S} \mathbf{K} + v^2 \mathbf{K}^\top \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} \mathbf{K} \\ &= -\mathbf{Q} + \mathbf{K}^\top ((1 - 2v) \mathbf{S} + v^2 \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}}) \mathbf{K}. \end{aligned} \quad (11.45)$$

Recalling that $\mathbf{S} = \mathbf{R} + \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}}$, we get (11.44):

$$\begin{aligned} & (\hat{\mathbf{A}} - v \hat{\mathbf{B}} \mathbf{K})^\top \mathbf{H} (\hat{\mathbf{A}} - v \hat{\mathbf{B}} \mathbf{K}) - \beta^2 \mathbf{H} \\ &= -\mathbf{Q} + \mathbf{K}^\top ((v - 1)^2 \hat{\mathbf{B}}^\top \mathbf{H} \hat{\mathbf{B}} + (1 - 2v) \mathbf{R}) \mathbf{K}. \end{aligned} \quad (11.46)$$

Thus, the inequality (11.44) can be written equivalently in the form (11.28) of Lemma 11.6, from which we know that there exists a nonempty interval $[\underline{v}, \bar{v}]$ such that the matrix (11.46) is negative semidefinite. It follows that all inequalities (11.43) are satisfied as long as $v_k \in [\underline{v}, \bar{v}]$, and this implies inequality (11.33), and the Theorem is proved. $\blacksquare$

Remark 6. As discussed in Theorem 11.7, $\hat{\mathbf{B}}$ is essential for verifying (11.28) and determining $v \in [\underline{v}, \bar{v}]$ to ensure stability. If $\hat{\mathbf{B}}$ is unknown, it can be estimated from collected data using

$$[\hat{\mathbf{B}} \quad \hat{\mathbf{A}}] = \mathbf{X}_1 \begin{bmatrix} \mathbf{U}_0 \\ \mathbf{X}_0 \end{bmatrix}^\dagger. \quad (11.47)$$

Since any interval $(v_m, v_M) \subseteq [\underline{v}, \bar{v}]$ with $v_m < 1 < v_M$ can replace $\mathcal{V}$ in Algorithm 11.2, choosing a narrower interval is preferable, as it reduces control deviations from the optimum and improves stability margins. $\square$

11.4 Simulations

To evaluate Algorithm 11.2, the open loop and controllable LTI system in [16] is utilized in numerical simulations, with

$$\mathbf{A} = \begin{bmatrix} -0.13 & 0.14 & -0.29 & 0.28 \\ 0.48 & 0.09 & 0.41 & 0.30 \\ -0.01 & 0.04 & 0.17 & 0.43 \\ 0.14 & 0.31 & -0.29 & -0.10 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1.63 & 0.93 \\ 0.26 & 1.79 \\ 1.46 & 1.18 \\ 0.77 & 0.11 \end{bmatrix}. \quad (11.48)$$

First, a persistently exciting input, $\mathbf{U}_0$, is used to generate the offline state data $\mathbf{X}_0$ and $\mathbf{X}_1$ from the system dynamics in (11.1). The noise is sampled from a normal distribution, $\boldsymbol{\omega}_k = \mathcal{N}(0, \sigma_\omega)$ and the offline input is sampled from $\mathbf{u}_k = \mathcal{N}(0, \sigma_u)$ with $\sigma_\omega = \sigma_u = 0.01$. The offline data consists of 8 time samples. Since we consider an open-loop stable system, the initial policy is kept at zero, i.e., $\mathbf{K}_t = \mathbf{0}_{m \times n}$. The parameters of PFDDeePO are chosen as $\gamma = 0.1$, $\delta = 0.1, \eta = 10^{-4}$, and the interval of v is chosen as $\underline{v} = 0.5$ and $\bar{v} = 1.5$, which satisfies (11.28). DeePO, introduced in Algorithm 11.1, is used as a benchmark, with $\eta = 10^{-4}$ and a probing noise signal $\mathbf{e}$ sampled from a zero mean normal distribution with $\sigma_e = 0.1$. At sample $k = 15$, a disturbance is induced in the states using a uniform random value.

In Fig. 11.1, we present the results of running both algorithms on the system. As observed, once the states reach equilibrium, PFDDeePO does not introduce further perturbations. In contrast, the probing noise in DeePO continuously disturbs the states, inducing oscillations that increase control effort.

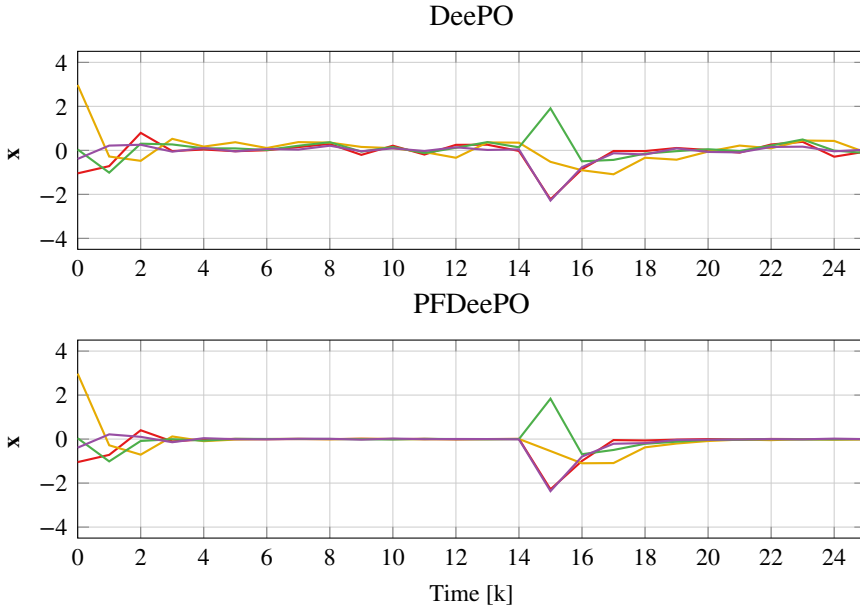


Figure 11.1: Evolution of the states $\mathbf{x}$, using DeePO (top) and PFDeePO (bottom).

Fig. 11.2 illustrates evolution of $\underline{\sigma}(\Phi_i)$. It is evident that in DeePO, probing noise is essential to maintaining the full rank of Φ , as its minimum singular value otherwise approaches zero. However, with PFDeePO, $\underline{\sigma}(\Phi_i)$ remains consistently above zero throughout the control period, ensuring the rank condition is met without perturbing the system states.

11.5 Conclusion

Data-driven control design has gained significant attention, but the field remains in its early stages, with several open challenges. Among recent approaches for LTI systems, DeePO is a promising method that refines control gains online through adaptive updates. However, its reliance on probing noise to ensure persistency of excitation can degrade performance and induce state perturbations. This article introduces PFDeePO, which alleviates these issues. Simulations show that PFDeePO outperforms conventional DeePO in terms of control performance. Future work includes extending PFDeePO to LTV and nonlinear systems and optimizing its design parameter selection.

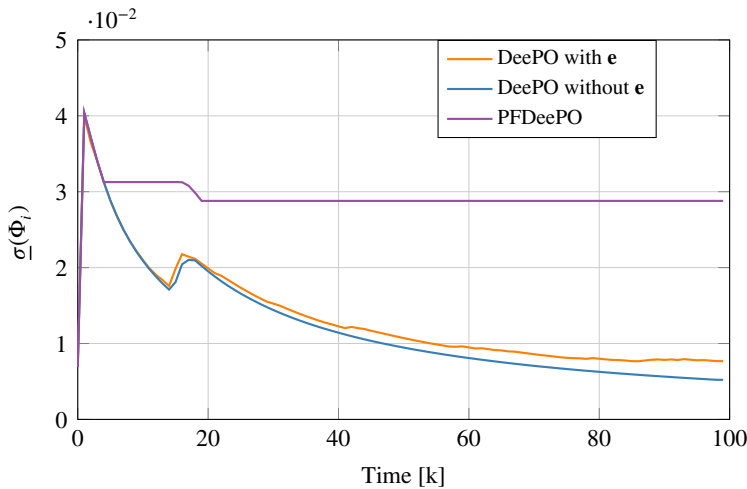


Figure 11.2: $\sigma(\Phi_i)$ for PFDDeePO and DeePO, with and without probing noise.

Bibliography

- [1] 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.
- [2] C.-T. Chen, *Linear System Theory and Design*. Oxford University Press, 3rd ed., 1999.
- [3] A. Cohen, T. Koren, and Y. Mansour, “Learning linear-quadratic regulators efficiently with only $\sqrt{T}$ regret,” in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 1300–1309, June 2019.
- [4] H. Mania, S. Tu, and B. Recht, “Certainty equivalence is efficient for linear quadratic control,” in *33rd International Conference on Neural Information Processing Systems (NIPS)*, pp. 10154 – 10164, December 2019.
- [5] M. Ferizbegovic, J. Umenberger, H. Hjalmarsson, and T. B. Schön, “Learning robust LQ-controllers using application oriented exploration,” *IEEE Control Systems Letters*, vol. 4, no. 1, pp. 19–24, 2020.
- [6] L. Sforni, G. Carnevale, I. Notarnicola, and G. Notarstefano, “On-policy data-driven linear quadratic regulator via combined policy iteration and recursive least squares,” in *62nd IEEE Conference on Decision and Control (CDC)*, pp. 5047–5052, 2023.

- [7] C. De Persis and P. Tesi, “Formulas for data-driven control: Stabilization, optimality, and robustness,” *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 909–924, 2020.
- [8] H. Mohammadi, M. Soltanolkotabi, and M. R. Jovanović, “On the linear convergence of random search for discrete-time LQR,” *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 989–994, 2021.
- [9] C. De Persis and P. Tesi, “Low-complexity learning of linear quadratic regulators from noisy data,” *Automatica*, vol. 128, p. 109548, 2021.
- [10] F. Dörfler, P. Tesi, and C. De Persis, “On the role of regularization in direct data-driven LQR control,” in *IEEE 61st Conference on Decision and Control (CDC)*, pp. 1091–1098, 2022.
- [11] V. G. Lopez, M. Alsalti, and M. A. Müller, “Efficient off-policy Q-learning for data-based discrete-time LQR problems,” *IEEE Transactions on Automatic Control*, vol. 68, no. 5, pp. 2922–2933, 2023.
- [12] F. Dörfler, P. Tesi, and C. De Persis, “On the certainty-equivalence approach to direct data-driven LQR design,” *IEEE Transactions on Automatic Control*, vol. 68, no. 12, pp. 7989–7996, 2023.
- [13] W. Fan and J. Xiong, “Q-Learning methods for LQR control of completely unknown discrete-time linear systems,” *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 5933–5943, 2025.
- [14] 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)*, pp. 114–120, 2024.
- [15] 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)*, pp. 6160–6165, 2023.
- [16] 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.
- [17] F. Zhao, R. Leng, L. Huang, H. Xin, K. You, and F. Dörfler, “Direct adaptive control of grid-connected power converters via output-feedback data-enabled policy optimization,” *arXiv preprint arXiv:2411.03909*, 2024.

- [18] N. Persson, F. Zhao, M. Kaheni, F. Dörfler, and A. V. Papadopoulos, “An adaptive data-enabled policy optimization approach for autonomous bicycle control,” *arXiv preprint arXiv:2502.13676*, 2025.
- [19] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.
- [20] Y. Sattar, Z. Du, D. A. Tarzanagh, S. Oymak, L. Balzano, and N. Ozay, “Certainty equivalent quadratic control for markov jump systems,” in *American Control Conference (ACC)*, pp. 2871–2878, 2022.
- [21] J. Pilipovsky and P. Tsiotras, “Data-driven covariance steering control design,” in *62nd IEEE Conference on Decision and Control (CDC)*, pp. 2610–2615, 2023.
- [22] W. Liu, G. Wang, J. Sun, F. Bullo, and J. Chen, “Learning robust data-based LQG controllers from noisy data,” *IEEE Transactions on Automatic Control*, vol. 69, no. 12, pp. 8526–8538, 2024.
- [23] I. R. Shafarevich and A. O. Remizov, *Linear Algebra and Geometry*. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 1 ed., 2013.

Algorithm 11.2 Perturbations-free DeePO (PFDeePO)

Require: $\mathbf{U}_0, \mathbf{X}_0, \mathbf{X}_1, \mathbf{K}_t, \gamma > 0, \delta > 0$, and $\eta > 0$.

Start

$i = t$.

$\Delta \mathbf{K} = (\delta + 1) \cdot \mathbf{1}_{m \times n}$.

while the stop criterion is not satisfied, do:

if $\|\Delta \mathbf{K}\| > \delta$ **or** $\|\mathbf{x}_i\| \leq \gamma$

 Apply $\mathbf{u}_i = \mathbf{K}_i \mathbf{x}_i$ and observe $\mathbf{x}_{i+1}$.

else

 Find $\bar{v}$ and $\underline{v}$ as in Theorem 11.7 for $\mathbf{K}_i$.

 Randomly select $\underline{v} \leq v_i \leq \bar{v}$.

 Apply $\mathbf{u}_i = v_i \mathbf{K}_i \mathbf{x}_i$ and observe $\mathbf{x}_{i+1}$.

End if

if $\|\mathbf{x}_i\| > \gamma$

 Update $\mathbf{X}_0$ by $\mathbf{X}_0 = [\mathbf{X}_0, \mathbf{x}_i]$.

 Update $\mathbf{X}_1$ by $\mathbf{X}_1 = [\mathbf{X}_1, \mathbf{x}_{i+1}]$.

 Update $\mathbf{U}_0$ by $\mathbf{U}_0 = [\mathbf{U}_0, \mathbf{u}_i]$.

$$\mathbf{V}_{i+1} = \Phi_{i+1}^{-1} \begin{bmatrix} \mathbf{K}_i \\ \mathbf{I}_n \end{bmatrix}.$$

$$\mathbf{V}'_{i+1} = \mathbf{V}_{i+1} - \eta \Pi_{\bar{\mathbf{X}}_0} \widehat{\mathbf{V}}_{\mathbf{V}} \mathbf{C}.$$

 Update the control gain by $\mathbf{K}_{i+1} = \bar{\mathbf{U}}_0 \mathbf{V}'_{i+1}$.

else

 Update the control gain by $\mathbf{K}_{i+1} = \mathbf{K}_i$.

End if

$$\Delta \mathbf{K} = \mathbf{K}_{i+1} - \mathbf{K}_i.$$

$i = i + 1$.

End while

End

Chapter 12

Paper F

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

Niklas Persson, Mojtaba Kaheni, Alessandro V. Papadopoulos.
Submitted to European Control Conference (ECC), Nov. 2025

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.

12.1 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 have 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.

12.1.1 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 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.

12.1.2 Organization

The remainder of this paper is organized as follows. Section 12.2 details the derivation of the MIMO FL controller. Section 12.3 presents the DeePO and DeePC algorithms, including the proposed modifications. Simulation results and comparative evaluations are provided in Section 12.4, followed by concluding remarks in Section 12.5.

12.1.3 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 $Q \geq 0$. Continuous-time signals are $x(t)$, while discrete-time signals are x_k . Normalized quantities are indicated by a tilde (e.g., $\tilde{x}$).

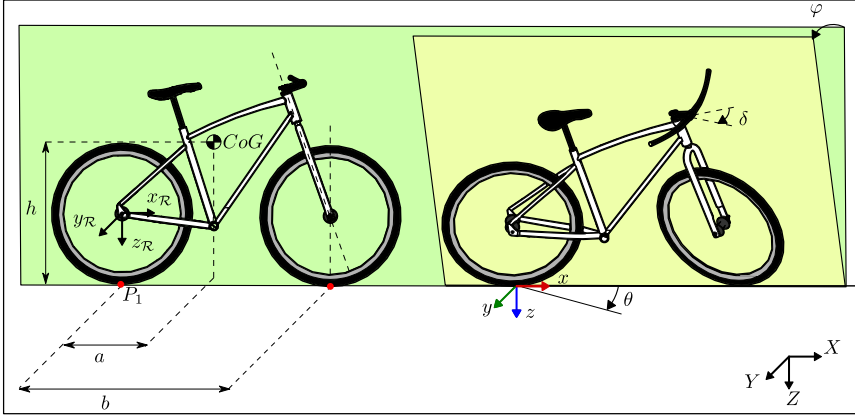


Figure 12.1: Illustration of the parameters used in the bicycle model.

12.2 Bicycle stablization

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. 12.1, the point-mass model of a bicycle is formulated as

$$\ddot{\varphi}(t) = \frac{g}{h} \sin(\varphi(t)) + \frac{a}{bh} \cos(\varphi(t)) v(t) \dot{\delta}(t) - \left(\frac{1}{bh} - \frac{1}{b^2} \tan(\delta(t)) \tan(\varphi(t)) \right) \tan(\delta(t)) v(t)^2, \quad (12.1)$$

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

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

represents the lean angle, lean rate, steer angle, and forward velocity of the bicycle while $\dot{\delta}(t)$ represent 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

$$\dot{\mathbf{x}} = \begin{bmatrix} \frac{g}{h} \sin(x_1) - \left(\frac{1}{bh} - \frac{1}{b^2} \tan(x_3) \tan(x_1) \right) \tan(x_3) x_4^2 \\ x_2 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{a}{bh} \cos(x_1) x_4 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad (12.2)$$

and more compact as

$$\dot{\mathbf{x}} = f(\mathbf{x}) + G(\mathbf{x})\mathbf{u}^{FL}, \quad \mathbf{y}^{FL} = h(\mathbf{x}), \quad (12.3)$$

where

$$\begin{aligned} y_1^{FL} &= h_1(\mathbf{x}) = x_1 = \varphi \\ y_2^{FL} &= h_2(\mathbf{x}) = x_4 = v, \end{aligned} \quad (12.4)$$

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

$$E(\mathbf{x}) = \begin{bmatrix} L_{g_1} L_f^{r_1-1} h_1 & L_{g_2} L_f^{r_1-1} h_1 \\ L_{g_1} L_f^{r_2-1} h_2 & L_{g_2} L_f^{r_2-1} h_2 \end{bmatrix} = \begin{bmatrix} \frac{a}{bh} \cos(x_1) x_4 & 0 \\ 0 & 1 \end{bmatrix} \quad (12.5)$$

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

$$\mathbf{u}_t^{FL} = E(\mathbf{x})^{-1}(\mathcal{L} - \alpha(\mathbf{x})), \quad (12.6)$$

where

$$\begin{aligned} \alpha(\mathbf{x}) &= \begin{bmatrix} L_f^2 h_1 \\ L_f h_2 \end{bmatrix} = \\ &= \begin{bmatrix} \frac{g}{h} \sin(x_1) - \left(\frac{1}{bh} - \frac{1}{b^2} \tan(x_3) \tan(x_1) \right) \tan(x_3) x_4^2 \\ 0 \end{bmatrix} \end{aligned} \quad (12.7)$$

and $\mathcal{L}$ is the linearized input $l_1 = \ddot{\varphi}_d(t) + k_{1_\varphi} \dot{\varphi}_e(t) + k_{2_\varphi} \varphi_e(t) + k_{3_\varphi} \sum_0^t \varphi_e(t)$, and $l_2 = \dot{v}_d(t) + k_{1_v} v_e$ where the subscript e denotes the error between the desired value and the measured value. The model parameters and the control gains are given in Table 12.1.

12.3 Data-driven control

While the FL controller stabilizes and partially cancels the nonlinearities of the system, it relies on a rather simple continuous time model of a bicycle (12.2) and can not capture all nonlinearities and time varying dynamics of a bicycle. To address these shortcomings, we enhance the input from (12.6) using DeePO as can be seen in Fig. 12.2. DeePO learns a LQR policy directly from data obtained by persistently excite the input to the system, and bypasses the need of a model [13]. Thanks to the adaptive nature of DeePO, it can handle time-varying systems, and we enhance this property by incorporating a forgetting factor in the algorithm. Moreover, for realizing trajectory tracking, the DeePC algorithm is implemented in an outer control loop. To address the nonlinearities in the outer loop, regularization and terminal cost is included in the DeePC formulation.

12.3.1 DeePO

Consider the closed-loop data from the bicycle, controlled through FL, where $\mathbf{x}_t^e = [\varphi_t^e, \dot{\varphi}_t^e, v_t^e]$, $\mathbf{u}_t^o = [\delta_t, \dot{v}_t]$ denotes the discrete time error state and control input data respectively. Assuming a t long time series of data from the system

Table 12.1: Model and control parameters for FL control

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 ²
k_{1_φ}	-	2	-
k_{2_φ}	-	10	-
k_{3_φ}	-	0.1	-
k_{1_v}	-	25	-

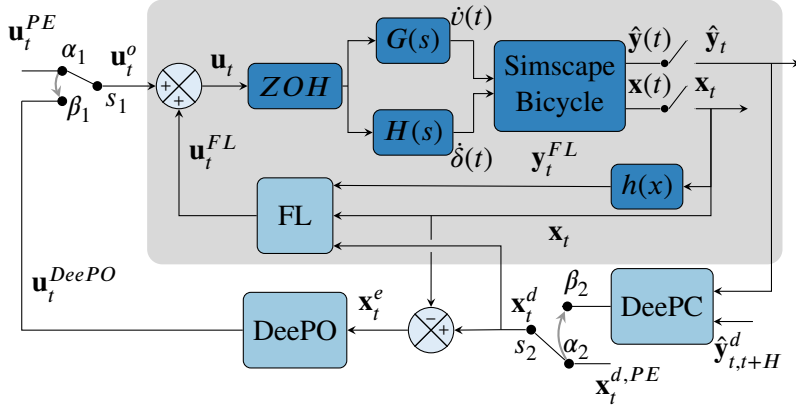


Figure 12.2: The FL-DeePO-DeePC control setup. Here, the blocks $H(s)$ and $G(s)$ are transfer functions representing the dynamics of the steering system and the rear wheel respectively. The switches s_1, s_2 in the figure switches between persistently excited input signal ($s_i = \alpha_i$) and the output of DeePO and DeePC ($s_i = \beta_i$), respectively. Furthermore, $\hat{\mathbf{y}}$ is the pose of the bicycle and is defined in (12.23).

is available:

$$\begin{aligned} X_{0,t} &:= [\mathbf{x}_0^e \quad \mathbf{x}_1^e \quad \dots \quad \mathbf{x}_{t-1}^e] \in \mathbb{R}^{n \times t}, \\ U_{0,t} &:= [\mathbf{u}_0^o \quad \mathbf{u}_1^o \quad \dots \quad \mathbf{u}_{t-1}^o] \in \mathbb{R}^{m \times t}, \\ X_{1,t} &:= [\mathbf{x}_1^e \quad \mathbf{x}_2^e \quad \dots \quad \mathbf{x}_t^e] \in \mathbb{R}^{n \times t}, \end{aligned} \quad (12.8)$$

which is assumed to be persistently exciting, i.e., the matrix

$$D_{0,t} := \begin{bmatrix} U_{0,t} \\ X_{0,t} \end{bmatrix} \quad (12.9)$$

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

$$\Phi_t := \frac{1}{t} D_{0,t} D_{0,t}^\top = \begin{bmatrix} U_0 D_{0,t}/t \\ X_0 D_{0,t}/t \end{bmatrix} = \begin{bmatrix} \bar{U}_0 \\ \bar{X}_0 \end{bmatrix}, \quad (12.10)$$

which enables the covariance-based policy parameterization of the feedback gain

$$\begin{bmatrix} K \\ I \end{bmatrix} = \Phi_t V. \quad (12.11)$$

The matrix V represents the parametrized policy, updated online through a single step of a projected gradient descent of the covariance-parameterized LQR cost [14].

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

$$\begin{aligned}
 & \min_{W, S, L, Z \geq 0} \text{Tr}(\mathcal{Q}_{\text{DeePO}} S) + \text{Tr}(\mathcal{R}_{\text{DeePO}} L) + \gamma \text{Tr}(Z \Phi_t), \\
 & \text{s.t. } S = \bar{X}_{0,t} W, \\
 & [S - I, \bar{X}_{1,t} W; W^\top \bar{X}_{1,t}^\top, S] \geq 0, \\
 & [L, \bar{U}_{0,t} W; W^\top \bar{U}_{0,t}^\top, S] \geq 0, \\
 & [Z, W; W^\top, S] \geq 0,
 \end{aligned} \tag{12.12}$$

where γ is a regularization parameter promoting robustness to noise, unmodeled dynamics, and uncertainties. The matrices $\mathcal{R}_{\text{DeePO}}$ and $\mathcal{Q}_{\text{DeePO}}$ are positive definite weighting matrices. Solving (12.12) yields the initial policy

$$K_0 = U_{0,t} W S^{-1}, \quad V_{\text{prim}} = \Phi_t^{-1} \begin{bmatrix} K_{t_0} \\ I \end{bmatrix}, \tag{12.13}$$

which serves as the starting gain for the online DeePO recursion. DeePO then updates the policy iteratively from new input–state data and adjusts the feedback gain online.

12.3.2 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

$$w_{t+1} = \lambda_f w_t + 1, \quad w_0 = 0. \tag{12.14}$$

The exponentially weighted covariance update then becomes

$$\Phi_{t+1} = \left(\frac{w_{t+1} - 1}{w_{t+1}} \right) \Phi_t + \frac{1}{w_{t+1}} \phi_t \phi_t^\top, \tag{12.15}$$

where $\phi_t = [\mathbf{u}_t^\top \mathbf{x}_t^\top]^\top$. When $\lambda_f = 1$, the recursion in (12.15) reduces to the standard unweighted form used in [14]. Furthermore, the inverse of Φ_t can be efficiently updated via the Sherman–Morrison formula [20],

$$\Phi_{t+1}^{-1} = \frac{w_{t+1}}{w_{t+1} - 1} \left(\Phi_t^{-1} - \frac{\Phi_t^{-1} \phi_t \phi_t^\top \Phi_t^{-1}}{w_{t+1} - 1 + \phi_t^\top \Phi_t^{-1} \phi_t} \right), \tag{12.16}$$

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

$$V_{t+1} = V_t + \frac{1}{w_{t+1}} \Phi_{t+1}^{-1} \phi_t \phi_t^\top (I - \Phi_t) V_t. \quad (12.17)$$

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.

12.3.3 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 μ_z and standard deviation σ_z are computed and z-score normalization is then applied to the data as

$$\tilde{\mathbf{z}} = \frac{\mathbf{z} - \mu_z}{\sigma_z}. \quad (12.18)$$

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

$$\tilde{Q} = S_x^{-\top} Q_{\text{DeePO}} S_x^{-1}, \quad \tilde{R} = S_u^{-\top} R_{\text{DeePO}} S_u^{-1}, \quad (12.19)$$

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

$$K = S_u^{-1} \tilde{K} S_x. \quad (12.20)$$

This ensures that the controller is applied to the unnormalized physical states of the bicycle. The complete algorithm is summarized in Algorithm 12.1, where the gradient of the covariance-parameterized LQR cost $J_t(V_t)$ is computed as

$$\nabla J_t(V_t) = 2(\tilde{U}_{0,t}^\top \tilde{R} \tilde{U}_{0,t} + \tilde{X}_{1,t}^\top P_t \tilde{X}_{1,t}) V_t \Sigma_t, \quad (12.21)$$

where P_t and Σ_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 η 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 12.1 DeePO (normalized, with forgetting factor)

Require: Offline data $(X_{0,t_0}, U_{0,t_0}, X_{1,t_0})$, scaling (μ_x, S_x, μ_u, S_u) , initial $\tilde{K}_{t_0}$, stepsize η , λ_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 (12.14); update $\tilde{\Phi}_{t+1}^{-1}$ via (12.16).
- 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 (12.15).

- 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**
-

12.3.4 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 ℓ as

$$\mathcal{H}_\ell(\mathbf{u}^{\text{DeePC}}) := \begin{bmatrix} \hat{\mathbf{u}}_0 & \hat{\mathbf{u}}_1 & \cdots & \hat{\mathbf{u}}_{T-\ell} \\ \hat{\mathbf{u}}_1 & \hat{\mathbf{u}}_2 & \cdots & \hat{\mathbf{u}}_{T-\ell+1} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\mathbf{u}}_{\ell-1} & \hat{\mathbf{u}}_\ell & \cdots & \hat{\mathbf{u}}_{T-1} \end{bmatrix}, \quad (12.22)$$

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

$$\hat{\mathbf{y}}_t = [x_t, y_t, \theta_t], \quad (12.23)$$

represents the pose of the bicycle and

$$\hat{\mathbf{u}}_t = [\varphi_t^d, v_t^d]^\top, \quad (12.24)$$

provides the desired lean angle and forward velocity for the inner control loop. By selecting an initialization horizon T_{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:

$$\begin{aligned} \min_{\mathbf{g}, \mathbf{u}, \mathbf{y}} & \|\hat{\mathbf{y}} - \hat{\mathbf{y}}_d\|_{\hat{Q}}^2 + \|\hat{\mathbf{u}} - \hat{\mathbf{u}}_d\|_{\hat{R}}^2 + \lambda_{\text{gd}} \|\mathbf{g} - \mathbf{g}_d\|_2^2 \\ & + \lambda_{\text{ini}} \|Y_q \mathbf{g} - \hat{\mathbf{y}}_{\text{ini}}\|_2^2 + (\hat{\mathbf{y}}_H - \hat{\mathbf{y}}_H^d)^\top P (\hat{\mathbf{y}}_H - \hat{\mathbf{y}}_H^d), \\ \text{s.t.} & \begin{bmatrix} U_q \\ Y_q \\ U_f \\ Y_f \end{bmatrix} \mathbf{g} = \begin{bmatrix} \hat{\mathbf{u}}_{\text{ini}} \\ \hat{\mathbf{y}}_{\text{ini}} \\ \hat{\mathbf{u}} \\ \hat{\mathbf{y}} \end{bmatrix}, \end{aligned} \quad (12.25)$$

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 $\dot{\phi}_t^d$ and $\dot{v}_t^d$ are computed, yielding the complete desired state vector $\mathbf{x}_t^d = [\phi_t^d, \dot{\phi}_t^d, v_t^d, \dot{v}_t^d]^\top$. Moreover, $\mathbf{g}$ is the coefficient vector describing a feasible trajectory, λ_{gd} and λ_{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 12.2. The optimization problem in (12.25) 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 :

$$d_H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b) \right\}, \quad (12.26)$$

where $d(a, b)$ is the Euclidean distance between the points a and b . To find the parameters of the DeePC problem in (12.25), we formulate another optimization problem aiming to minimize the Hausdorff distance between the desired trajectory and the bicycle trajectory:

$$\begin{aligned}
 & \min_{\hat{Q}, \hat{R}, \lambda_{\text{gd}}, \lambda_{\text{ini}}} d_H(\hat{\mathbf{y}}_d, \hat{\mathbf{y}}(\hat{Q}, \hat{R}, \lambda_{\text{gd}}, \lambda_{\text{ini}})) \\
 & \text{s.t.} \quad 100 \leq \hat{Q}_{ii} \leq 600, \\
 & \quad \quad 0.001 \leq \hat{R}_{jj} \leq 50, \\
 & \quad \quad 10 \leq \lambda_{\text{gd}} \leq 500, \\
 & \quad \quad 10^3 \leq \lambda_{\text{ini}} \leq 10^{10}.
 \end{aligned} \tag{12.27}$$

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

12.4 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.

12.4.1 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

$$G(s) = \frac{1}{0.5s + 1}, \tag{12.28}$$

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

$$H(s) = \frac{100 + s}{100} \tag{12.29}$$

is put in series with the bicycle model as illustrated in Fig 12.2. The steering rate control signal is saturated at $\pm 4 \text{ rad/s}$, and has a resolution of 0.024 rad/s . Zero-mean Gaussian noise was added to all measured and estimated signals in

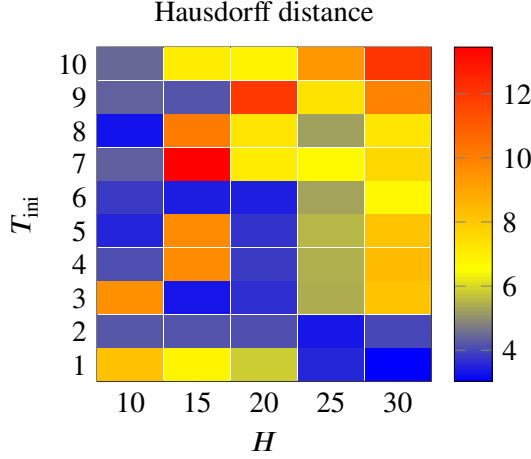


Figure 12.3: Optimal value solving (12.27) for different values of H and T_{ini} .

both the inner and outer control loops. Each noise component ξ_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 (12.12) 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. 12.2 is moved from α_1 to β_1 , while s_2 is placed in α_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 φ_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 φ_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 (12.27) is solved for each combination using Matlabs `patternsearch` routine, and each set of parameters is evaluated by tracking the desired trajectory given in Fig. 12.4, using the control system given in Fig. 12.2.

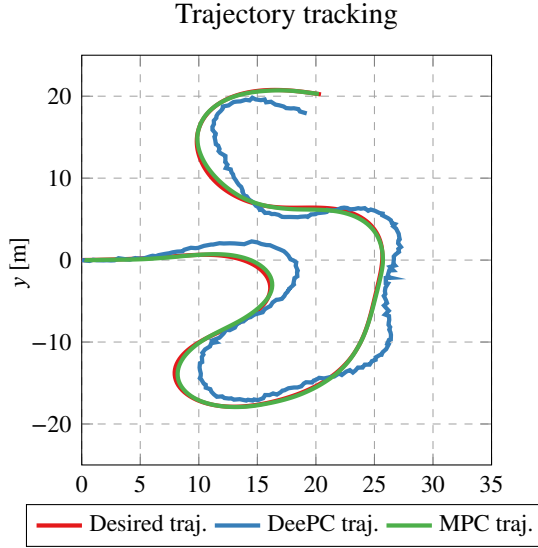


Figure 12.4: Trajectory tracking using the proposed FL-DeePO-DeePC scheme, compared to a PID-MPC scheme [23].

12.4.2 Simulation results

The results of solving the problem in (12.27) are given in Fig. 12.3 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. 12.4. The tracking performance of the DeePO and FL controller for the desired states, $\mathbf{x}_t^d$, is given in Fig. 12.5. 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 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. 12.4, where the PID-MPC scheme resulted in a $d_H = 0.359\text{m}$.

12.4.3 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,

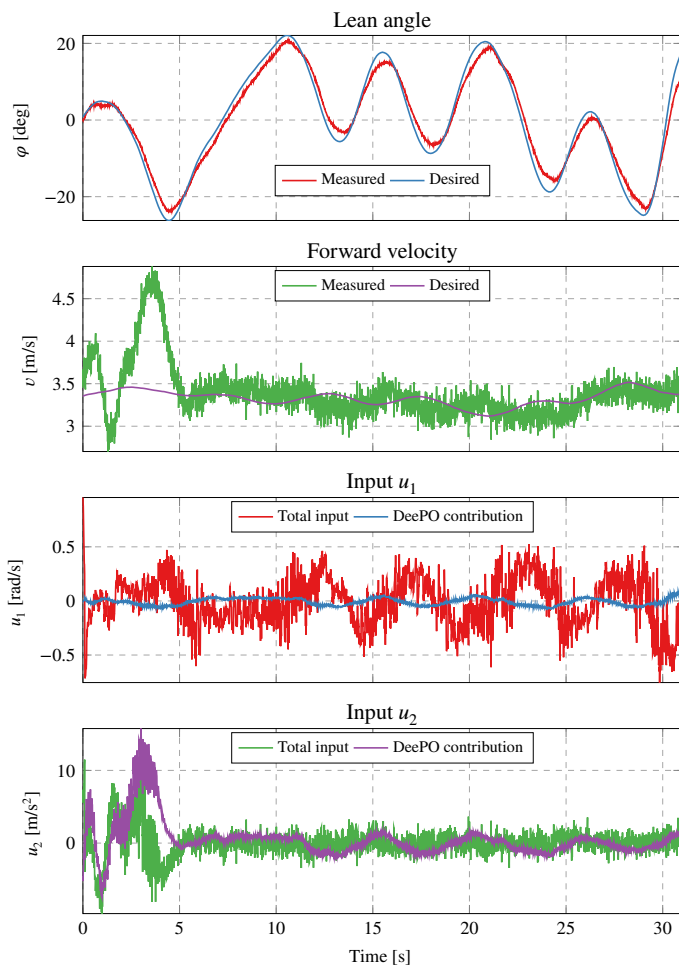


Figure 12.5: The tracking performance of the FL-DeePO controller. The first two plots show the tracking of the desired lean angle and velocity, respectively, while the last two plots highlight the DeePO contribution in the total input.

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 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. 12.3 suggest trends in the Hausdorff distance with respect to the initialization and prediction horizons. Smaller T_{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.

12.5 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.

Bibliography

- [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 *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, 2020.
- [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)*, pp. 1570–1576, 2021.
- [5] Y. Tanaka and T. Murakami, “Self sustaining bicycle robot with steering controller,” in *The 8th IEEE International Workshop on Advanced Motion Control (AMC)*, pp. 193–197, IEEE, 2004.
- [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, pp. 148–151 vol.1, 1994.
- [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)*, pp. 413–417, 2017.
- [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)*, pp. 307–312, 2019.
- [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)*, pp. 6160–6165, 2023.
- [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)*, pp. 114–120, 2024.
- [19] J. Slotine and W. Li, *Applied Nonlinear Control*. 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*, vol. 317. Springer Science & Business Media, 2009.
- [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)*, pp. 1859–1866, 2021.