

Mälardalen University Press Licentiate Theses
No. 336

CONTROL AND NAVIGATION OF AN AUTONOMOUS BICYCLE

Niklas Persson

2023



School of Innovation, Design and Engineering

Copyright © Niklas Persson, 2023
ISBN 978-91-7485-580-7
ISSN 1651-9256
Printed by E-Print AB, Stockholm, Sweden

Till Lina, Billy och Frans

“Life is like riding a bicycle. To keep your balance you must keep moving.” - Albert Einstein

Acknowledgements

At the end of my technical preparatory year back in 2014, I was convinced I would become a construction engineer. However, at an information meeting for further studies, a guy named Mikael Ekström was going on and on about the robotics program and all the exciting things the future holds for robots and what you got to do as a student in the robotics program. I was hooked! Of course, it was not until later that I realised that Mikeal forgot to mention that before you can do anything interesting with the robots, you have to complete about 4 out of 5 years of education. But the joke is on him, as he is now one of my supervisors. Mikael shares this burden with his brother Martin Ekström (yes, M. Ekström can cause some confusion in the author lists), and they are doing a great job which I'm very thankful for! You have taught me a great deal from my first year in the robotics program until this licentiate thesis and I am forever grateful that you are willing to share your knowledge and time with me. Both of you have made me feel very welcome and made the transition from a student to a PhD student silky smooth.

They say all good things come in threes, and I could not have asked for a better main supervisor. I am extremely grateful and thankful to have Alessandro Papadopolous filling this spot and believing in me. I have learnt so much from you, not only in the field of robotics, control, optimisation and how to conduct research but also in life and what it means to be a PhD student. You have a very humble approach to teaching and supervision, and you always have time for me. I feel that I can always ask you no matter the question and you have a lot of patience with me. In fact, many (100 or so) of my early questions you had already taught me at a total of three(?) control theory lectures (even though I sometimes find it hard to believe). I look forward to the upcoming years, getting guidance, laughing, gaining wisdom and working together with you and the Ekström brothers.

Moreover, I would like to thank my office roommate, Jonas, for being a good roommate and friend. The next-door office neighbours, Johan, Lennie and Carl for their support, friendship, and appetite for coffee (which they share with

the rest of the population of C2). A big thanks to Henrik for always helping me out, and of course, for letting me play with Bosse. The juggling man, Fredrik, for being a friend, problem solver and simply the best manager you can wish for. Nikola for reviewing my proposal. Christoph for trying to convince me that you can drink cold coffee and the rest of the staff of IDT for always being friendly and helpful.

There is also a world outside of work, which I tackle together with the love of my life, Lina, and our two superheroes, Billy and Frans. I am incredibly thankful for your support and for believing in me, and for supporting me to study in the first place. I will never take your love for granted and I look forward to spending the rest of my life with you. Moreover, a big thanks to my dad for teaching me that God made a set of perfect skulls, the rest he covered with hair. To my mum for all the laughs and love. To my brother for challenging me and believing in me.

Thanks, would not have made it without you!

Niklas Persson
Västerås, March 2023

Abstract

Autonomous control of mobile robots is a research topic that has received a lot of interest. There are several challenging problems associated with autonomous mobile robots, including low-level control, localisation, and navigation. Most research in the past has focused on developing algorithms for three or four-wheeled mobile robots, such as autonomous cars and differential drive robots, which are statically stable systems. In this thesis, autonomous two-wheeled robots are considered, such as autonomous bicycles, which are naturally unstable systems, and without proper actuation, they will lose balance and fall over. Thus, before developing algorithms for higher-level functionality such as localisation and navigation of an autonomous bicycle, the balance of the bicycle needs to be addressed. This is an interesting research problem as the bicycle is a statically unstable system that has proven difficult to control, but given adequate forward velocity, it is possible to balance a bicycle using only steering actuation. Moreover, given a sufficient forward velocity, the bicycle can even become self-stabilised.

In this thesis, the balance and trajectory tracking of an autonomous bicycle is investigated. First, we propose an extension of previously proposed bicycle models to capture the steering dynamics including the motor used for controlling the handlebar. Next, several control methods which can stabilise an autonomous bicycle by actuation of the steering axis and the forward velocity of the bicycle are developed. The controllers are compared in simulations on both a linear and nonlinear bicycle model. The simulation evaluation proceeds with experiments conducted on an instrumented bicycle running on a bicycle roller. Moreover, trajectory tracking of an autonomous bicycle is addressed using a model predictive controller approach where the reference lean angle is computed at every sample interval and is tracked by the balance controller in the inner loop. Finally, path planning in a static environment is considered where the proposed strategy realises a smooth path that adheres to the kinematic and dynamic constraints of the bicycle while avoiding obstacles and optimises the number of heading changes and the path distance. The results obtained from de-

tailed multibody simulations highlight the feasibility of the balance controller, trajectory tracking controller, and path planner.

Sammandrag

Autonom styrning av mobila robotar är ett forskningsämne som har fått stort intresse. Det finns flera utmanande problem förknippade med autonoma mobila robotar, inklusive lågnivåkontroll, lokalisering och navigering. Den mesta forskningen tidigare har fokuserat på att utveckla algoritmer för tre- eller fyrhjuliga mobila robotar, såsom autonoma bilar och differentialdrivna robotar, vilka är statiskt stabila system. I denna avhandling studeras autonoma tvåhjuliga robotar, som autonoma cyklar, vilka är naturligt instabila system, och utan korrekt reglering kommer de att tappa balansen och ramla omkull. Innan man utvecklar algoritmer för funktionalitet på högre nivå, såsom lokalisering och navigering av en autonom cykel, måste således balansen i cykeln lösas. Detta är ett intressant forskningsproblem eftersom cykeln är ett statiskt instabilt system som har visat sig vara svårt att kontrollera, men med tillräcklig hastighet framåt är det möjligt att balansera en cykel med enbart styret. Dessutom, givet en tillräcklig hastighet framåt, kan cykeln till och med bli självstabiliserad.

I denna avhandling undersöks balanskontroll och banspårningen av en autonom cykel ur ett regleringsperspektiv. Först föreslår vi en utökning av tidigare föreslagna modelleringar av cyklar för att fånga styrdynamiken inklusive motorn som används för att kontrollera styret. Därefter utvecklas flera reglermetoder som kan stabilisera en autonom cykel genom aktivering av styraxeln och hastigheten av cykeln. Regulatorerna jämförs i simuleringar på både en linjär och olinjär cykelmodell. Simuleringsutvärderingen efterföljs av experiment utförda på en instrumenterad cykel som körs på en cykelrulle. Dessutom adresseras banspårning av en autonom cykel med hjälp av en modellprediktiv reglering där referenslutningsvinkeln beräknas vid varje sampelintervall och följs av den inre balanskontrollen. Slutligen undersöks vägplanering i en statisk miljö där den föreslagna strategin realiserar en kontinuerlig väg som hörsammar cykelns kinematiska och dynamiska begränsningar samtidigt som man undviker hinder och optimerar antalet kursändringar och vägavståndet. Resultaten som erhållits från detaljerade flerkroppssimuleringar visar genom-

förbarheten av balanskontrollern, banspårningskontrollern och vägplaneraren.

List of Publications

Papers Included in This Thesis¹

Paper A: Tom Andersson, Niklas Persson, Anas Fattouh, Martin Ekström.
A Loop Shaping Method for Stabilising a Riderless Bicycle.
In European Conference on Mobile Robots (ECMR), 2019.

Paper B: Niklas Persson, Tom Andersson, Anas Fattouh, Martin Ekström, Alessandro V. Papadopoulos.
A Comparative Analysis and Design of Controllers for Autonomous Bicycles.
In European Control Conference (ECC), 2021.

Paper C: Niklas Persson, Martin Ekström, Mikael Ekström, Alessandro V. Papadopoulos.
Trajectory Tracking and Stabilisation of a Riderless Bicycle.
In International Conference on Intelligent Transportation Systems (ITSC), 2021.

Paper D: Niklas Persson, Martin Ekström, Mikael Ekström, Alessandro V. Papadopoulos.
On the Initialization Problem for Timed-Elastic Bands.
Submitted

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

Contents

I	Thesis	1
1	Introduction	3
2	Background	7
2.1	Model	7
2.2	Human Control	12
3	Related Work	15
3.1	Balance controller	15
3.2	Trajectory tracking	19
3.3	Path planning	21
4	Research Overview	23
4.1	Problem Formulation	23
4.2	Research Method	25
4.2.1	Simulation tools	25
4.2.2	Experimental setup	27
4.3	Threats to validity	27
5	Thesis Contributions	29
5.1	Contributions	29
5.2	Overview of Included Papers	32
5.2.1	Paper A	32
5.2.2	Paper B	33
5.2.3	Paper C	34
5.2.4	Paper D	34
6	Conclusions and Future Work	37
	Bibliography	38

II	Included Papers	46
7	Paper A	
	A Loop Shaping Method for Stabilising a Riderless Bicycle	49
7.1	Introduction	51
7.2	Related Works	51
7.3	Modelling of the Riderless Bicycle	53
	7.3.1 The Point-Mass Model	54
	7.3.2 Steering Response Matching	55
7.4	Balance Controller Design	56
7.5	Simulation Results	58
7.6	Application to the instrumented bicycle	59
	7.6.1 Control structure	60
	7.6.2 Experimental setups	61
	7.6.3 Results	63
7.7	Conclusion & future work	64
7.8	Acknowledgements	64
8	Paper B	
	A Comparative Analysis and Design of Controllers for Autonomous Bicycles	67
8.1	Introduction	69
8.2	Related Work	70
8.3	Modeling of instrumented bicycle	71
	8.3.1 Experimental platform	71
	8.3.2 Linear model	73
	8.3.3 Nonlinear model	76
8.4	Control strategies	76
	8.4.1 PID controllers	77
	8.4.2 Linear quadratic regulator	78
	8.4.3 Fuzzy controller	79
8.5	Results	80
	8.5.1 Simulation setup	80
	8.5.2 Simulation results	81
	8.5.3 Experimental setup	81
	8.5.4 Experimental results	82
8.6	Conclusion and Future Work	84
8.7	Acknowledgements	85

9 Paper C

Trajectory tracking and stabilisation of a riderless bicycle 89

- 9.1 Introduction 91
- 9.2 Related Work 92
 - 9.2.1 Modelling 92
 - 9.2.2 Path tracking 93
- 9.3 Bicycle model 94
- 9.4 Control design 97
 - 9.4.1 Balance controller 97
 - 9.4.2 Path tracking 100
- 9.5 Results 102
 - 9.5.1 Simulation setup 103
 - 9.5.2 Simulation results 104
 - 9.5.3 Discussion 105
- 9.6 Conclusion 106
- 9.7 Acknowledgements 108

10 Paper D

On the Initialization Problem for Timed-Elastic Bands 113

- 10.1 Introduction 115
- 10.2 Background 116
 - 10.2.1 Path planning 117
 - 10.2.2 Related work 119
- 10.3 Path optimisation 120
- 10.4 Evaluation 121
- 10.5 Results 122
 - 10.5.1 Path planning results 123
 - 10.5.2 Simulation results 123
 - 10.5.3 Discussion 124
- 10.6 Conclusion 126
- 10.7 Acknowledgements 127

I

Thesis

Chapter 1

Introduction

A bicycle is a popular means of transportation and has both health and environmental benefits. However, bicycles are often forced to share road segments with motorized vehicles, which places the unprotected cyclist at a higher risk of injuries. The safety of vulnerable road users, such as cyclists and pedestrians, can potentially be improved with the development of autonomous vehicles [1]. The vehicles are equipped with a variety of sensors for mapping the surrounding environment and detecting and classifying other road users, including vulnerable road users. Even though most cars are not fully autonomous yet, modern cars are often equipped with autonomous emergency braking (AEB) systems. When the AEB systems are evaluated by the European car safety performance assessment program, EuroNCAP, a bicycle is placed on a moving platform and tests are conducted using two different scenarios ¹. In the first scenario, the platform is moving in front of the vehicle, crossing its path, and the car should brake before the collision. In the second scenario, the platform and the vehicle are moving in the same direction and the vehicle is driving past the platform. In both scenarios, the platform is moving in a straight line and can not capture the realistic manoeuvres of a bicycle, nor the sometimes unpredictable behaviour of a cyclist. Thus, an autonomous bicycle that can manoeuvre realistically would resemble a cyclist to a larger extent compared to a bicycle on a moving platform and provide a more realistic test environment for the development of autonomous vehicles. Other potential areas of use for autonomous bicycles are within message delivery service or bike-sharing system [2, 3]. A common problem within the bike-sharing system is that the bicycles usually end up at a few popular sites and have to be manually distributed among the less popular sites. This is a time-consuming task that could be solved by using

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

autonomous bicycles which could distribute themselves, similar to what has already been proposed for autonomous scooters [4]. However, an autonomous bicycle that is only used at the test tracks is not restricted by the same rules and regulations as an autonomous bicycle in an urban environment would be, simplifying the problem.

The bicycle is an interesting vehicle that has been subject to research for more than a century [5]. On the one hand, a bicycle standing still is similar to an inverted pendulum and will fall over, making it a statically unstable system. On the other hand, given enough forward speed, the bicycle becomes self-stabilising. Thus, it is a dynamically stable system. Moreover, the bicycle is a non-minimum phase system at low velocities, thus it has a zero in the right half of the complex plane, which makes it more difficult to control from a control-theoretic point of view. Nevertheless, numerous controllers have been proposed for balancing an autonomous bicycle, but only a handful of the proposed strategies have been evaluated in real-life experiments.

In this licentiate thesis, the development and control design of an autonomous bicycle is investigated. An autonomous bicycle is similar to other wheeled robots such as cars and differential drive robots. However, a fundamental difference is that the bicycle is a statically unstable system, and without proper actuation, it will fall over. Therefore, a controller for stabilising a bicycle is investigated, and later evaluated and compared to different control methods including Proportional-Integral-Derivative (PID) controller, Linear Quadratic Regulator (LQR) and a Fuzzy controller, in both simulations and experiments conducted on an instrumented bicycle. Moreover, both linear and nonlinear models of a bicycle are considered in the thesis and we investigate how the models differ from each other and also how the dynamic models resemble the dynamics of a real instrumented bicycle. We extend previously proposed bicycle models with a steering step response matching procedure to capture the main dynamics of the steering system, including the DC motor used for controlling the handlebar. Moreover, by utilising a linear model and a balance controller, we formulate a solution for the trajectory tracking problem for an autonomous bicycle. The balance controller is incorporated into a bicycle model, which is used to formulate a Model Predictive Controller (MPC) for tracking a predefined reference path. Finally, the path planning problem for vehicles with non-holonomic constraints is considered. Techniques including grid searching methods, sampled-based methods, and hybrid path planning methods are used as inputs to an optimisation problem for smoothing the path while avoiding obstacles and adhering to the kinematic constraints of a bicycle. An initial path planned by Theta* and optimised using Time Elastic Bands (TEB) shows the most

promising results and the planned path is tracked using the MPC while the balance of the bicycle is controlled by a PID, highlighting the feasibility of the proposed method.

The rest of the thesis is organised as follows. Background on bicycle dynamics and modelling and a brief introduction to human control of a bicycle is given in Chapter 2. In Chapter 3 the related work is presented. Next, an overview of the conducted research is given in Chapter 4 which includes the problem formulation and the research method. Chapter 5 presents the research contributions of the thesis and a summary of the included papers. Finally, the thesis is concluded and future work is presented in Chapter 6.

Chapter 2

Background

The bicycle is a naturally unstable system. To develop and evaluate different control schemes for stabilisation, motion planning, and tracking for such a system it is often very beneficial and time-saving to use a model of the system. In this chapter, different bicycle models found in the literature are presented first. Next, some background on rider control of a bicycle is given which is important for understanding how a controller for a riderless bicycle can be developed.

2.1 Model

In the late 19th century Francis Whipple [6] and Emmanuel Carvallo [7], apparently independent of each other [8], derived equations for the dynamics of a bicycle. The bicycle is modelled with two wheels, a frame and a front fork with handlebars, each with an associated mass and inertia. The joint friction between the four rigid parts is neglected, also the effect of pedals, brakes, chains and other peripherals is disregarded. The contact between the wheels and the ground is modelled as a point contact, thus assuming the wheels to be thin. The friction between the ground and the wheels is neglected and the wheels are assumed to roll without side or longitudinal slip, i.e non-holonomic constraints. The rider is assumed to be a rigid part of the mainframe and thus follows the lean angle and velocity of the frame. Thus, for a riderless bicycle, the same model can be used by simply reducing the mass of the main frame. By linearising around small lean and steering angles and assuming a constant velocity, the model presented in the work of Mejiard et al. [5] is obtained as:

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

where $\mathbf{q} = [\varphi(t), \delta(t)]^\top$ are the lean angle and steering angle respectively, the symmetric invertible matrix \mathbf{M} represents the inertia and mass properties of the

bicycle. The damping matrix is denoted \mathbf{C} and is proportional to the forward velocity v . The stiffness matrix is made up of a constant part \mathbf{K}_0 and one part, \mathbf{K}_2 , which is proportional to the square of the forward velocity. The input, $\mathbf{f}(t) = [\tau_\varphi(t), \tau_\delta(t)]^\top$ are the lean torque and the torque applied to the handlebar respectively. Given this fourth-order model, known as the *Whipple model*, it is possible to form its characteristic fourth-degree polynomial:

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

which can be used to estimate the region of self-stability of bicycles [5]. In certain velocity ranges, depending on the physical and geometric parameters of a bicycle, all the real parts of the Eigenvalues, λ , in (2.2) become negative. In these velocity ranges, the system is stable, thus the bicycle can balance on its own. In Figure 2.1 the real part of the Eigenvalues are varying with the velocity, and in between the so-called *weave speed* (v_w), and *capsize speed* (v_c) all real parts are less than zero. To compute these Eigenvalues JBike6¹ is used, which requires 25 physical parameters to be measured and estimated on a bicycle. The parameters used to compute the velocity ranges in Figure 2.1 are measured from the instrumented bicycle used in papers A and B.

The characteristic polynomials in (2.2) were also used to experimentally validate the Whipple model in the work of Kooijman et al. [9]. With sensors attached to the bicycle the roll and yaw rate, steering angle, and velocity of the bicycle were measured. A human ran beside the bicycle to adequate speed and then released it into free coasting. When the human released the bicycle, he also perturbed the bicycle's roll angle. From the measurements, the Eigenvalues were extracted using curve fitting and compared to the Eigenvalues of the characteristic polynomial. In velocities between 3 and 6m/s the model and experimental data are similar, which suggests that the model is a good approximation of a real bicycle in this velocity span. In lower velocities, they found it difficult to get quality measures as the bicycle were too unstable at those speeds.

A number of extensions have been proposed for the Whipple model, for example, Plöchl et al. added a passive rider, frame compliance and lateral slipping tyres, where the lean angle of the rider was added as an additional degree of freedom to the model [10]. In the work of Schwab et al. [11] two passive rider models, with different postures, were investigated. It was shown that the posture of the rider impacts the controllability and the self-stability of the bicycle. The results also highlight the fact that it is much easier to control the bicycle using only steering actuation compared to upper body motions.

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

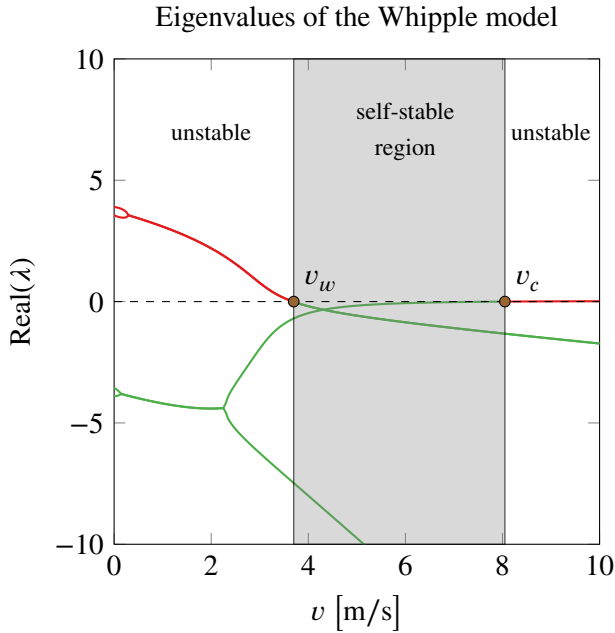


Figure 2.1: The Eigenvalues of the linearised Whipple model change with respect to the forward velocity. Between the weave speed (v_w) and capsize speed (v_c) all Eigenvalues have a negative real part, and the bicycle is self-stabilised.

The Whipple model is complex and to use it for modelling a physical bicycle there are 25 parameters to be measured or estimated [12, 9]. Simpler models require in general fewer physical parameters, at the cost of modelling accuracy. For example, the bicycle model presented in the work of Getz and Marsden [13, 14] only requires the vertical, (h), and horizontal position, (a), of the centre of mass with respect to the rear wheel, the wheelbase (b) and the total mass of the bicycle (m), lumped together in a point mass at the centre of mass (G), see Figure 2.2. In addition to the assumptions made in the Whipple model, this model also assumes negligible width, radii and inertial moments of the wheels. Furthermore, it also makes the assumption that the bicycle possesses zero trail which means that the steering axis is perpendicular to the ground plane, i.e $\lambda = 0$ in Figure 2.2. The control input for balance is the steering velocity, $\dot{\delta}$ in contrast to the steering torque which is the control input to the Whipple model.

A similar model as the one presented in the work of Getz and Marsden, is found in the work of Limebeer and Sharp [15], where they instead use the steering angle, δ , as control input for balance. An advantage of using the steering

position or the steering velocity as a control input, as compared to the steering torque, is that the steer dynamics can be neglected which simplifies the model [16]. The roll dynamics of Getz bicycle model are:

$$h\ddot{\varphi} = g \sin(\varphi) - ((1 - h\sigma \sin(\varphi))\sigma v^2 + b\ddot{\psi}) \quad (2.3)$$

where $g = 9.81m/s^2$ is the approximated acceleration due to gravity, σ is the curvature of the path and is defined as

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

and ψ is the yaw angle of the bicycle, obtained through

$$\dot{\psi} = v \frac{\tan(\delta)}{b} = v\sigma. \quad (2.5)$$

This yaw angle computation is also frequently used for approximating the orientation of a four-wheeled vehicle by exploiting the fact that the vehicle is symmetric around its x-axis. However, instead of computing the orientation with respect to the rear wheel as in (2.5) it is common to originate the yaw angle estimation from the centre of gravity instead [17]. This approximation of the curvature, and in extension the yaw angle, is valid as long as the roll angle is kept small, which is true in the case of a car, but not for bicycles and motorcycles. By including the roll angle in the curvature computation [15], equation (2.5) becomes:

$$\dot{\psi} = v \frac{\tan(\delta)}{b \cos(\varphi)} \quad (2.6)$$

which is defined for all practical lean angles, i.e $|\varphi| < \pi/2$.

In the work of Yi et al. [18, 19] a model of a motorcycle is presented which is an extension of the Getz model. As bicycles and motorcycles share many characteristics such as under-actuation, statically unstable, two inline wheels as well as numerous geometrical similarities, the model by Yi has successfully been used in bicycle research as well [20, 21]. An essential difference between the models is that the model proposed by Getz assumes that the steering axis is vertical, i.e a caster angle $\lambda = 0$. The model proposed by Yi considers the case when $\lambda \neq 0$ which provides a trail $c \neq 0$ 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 [8]. The caster angle will also have an impact on the orientation (ψ) and in extension the (x, y) position of the bicycle in the world frame which can be computed using the

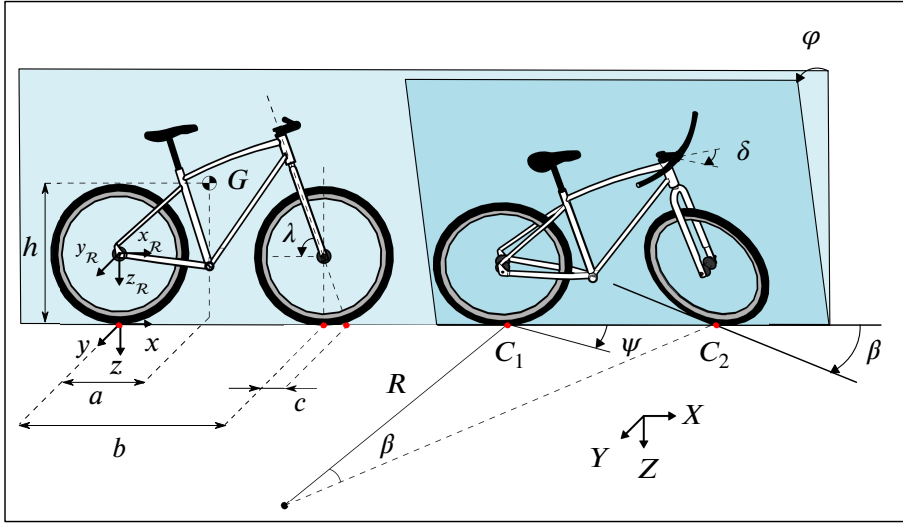


Figure 2.2: A bicycle riding on a flat horizontal plane.

following kinematic model:

$$\dot{x} = v_x \cos(\psi) \quad (2.7a)$$

$$\dot{y} = v_x \sin(\psi) \quad (2.7b)$$

$$\dot{\psi} = \frac{v_x \tan(\beta)}{b} \quad (2.7c)$$

where v_x is the forward velocity of the rear wheel, and β is the effective steering angle at the ground:

$$\beta = \arctan\left(\frac{\tan(\delta) \sin(\lambda)}{\cos(\varphi)}\right). \quad (2.8)$$

As a consequence of the non-zero caster angle, the roll dynamics become:

$$h^2 \ddot{\varphi} = g(h \sin(\varphi) + \frac{ca \sin(\lambda)}{b} \sigma \cos(\varphi)) - (1 - \frac{h\sigma}{b} \sin(\varphi)) \frac{h\sigma}{b} \cos(\varphi) v_x^2 - \frac{ah\sigma}{b} \cos(\varphi) \dot{v}_x - \frac{ah}{b} \cos(\varphi) v_x \omega_\sigma. \quad (2.9)$$

where ω_σ is the effective steering angular velocity.

On the one hand, the Getz model is a simple model which is straightforward but neglects important aspects of the complex dynamics of a bicycle. On the other hand, the more elaborated model by Yi, and later by Zhang and Yi,

is slightly more complicated but has been proven successful for the development of balance controllers [20, 22], for path tracking [18], and recently also for localisation [21]. The Whipple model has mainly been used for investigation of rider and bicycle stability [23, 9], but also to develop controllers for autonomous bicycles [24].

Numerous nonlinear models have also been proposed in the literature including analytic models [25] and multibody dynamics software models [24]. Even though a model often simplifies the process of designing a control scheme, it is also possible to develop successful balance controllers for a bicycle without any model using for example reinforcement learning [26]. The research on the modelling of bicycle dynamics stretches over more than a decade and a comprehensive survey of it is out of scope for this thesis. Instead, the interested reader is referred to [5, 27, 28] for a more detailed discussion of dynamic models of two-wheeled vehicles.

2.2 Human Control

With a combination of upper body lean, forward velocity and actuation of the handlebar a human steers into the fall of the bicycle to maintain balance. The upper body motion only has a small impact on balance, as revealed in the work of Kooijman [23] and later by Moore [29] using an in-depth motion capture analysis. The upper body motions were only visible at pedalling frequencies and hypothesized to be a reaction of pedalling and not crucial for balancing a bicycle. This was also evident in the work of Zhang et al. where a bicycle equipped with a flywheel was ridden by a human [22]. The flywheel was used to produce random lean torque disturbance on the bicycle without the awareness of the rider. They noticed that the rider responded to these random perturbations by actuation of the handlebar for the most part. This result agrees with the results from Kooijman and Moore which conclude that the handlebar motion is the preliminary control input for balancing and performing normal manoeuvres, even though at low speeds knee movements also play an important role. Thus, for stabilisation of a riderless bicycle riding at adequate forward speed, it is enough to use steering actuation.

Furthermore, to initiate a turn to the left, a small counter-steer to the right is first performed by the rider [30]. The small counter-steer to the right makes the bicycle lean to the left, and when the handlebar is turned left the bicycle continues to the left while maintaining balance. This is a fact that is not known by the common rider, and most people are performing it unconsciously. In the words of Mont Hubbard *"Everybody knows how to ride a bike, but nobody*

knows how we ride bikes" [31]. For a comprehensive survey on rider control, the reader is referred to the survey by Kooijman and Schwab [30].

Chapter 3

Related Work

In this chapter, previous work focusing on balancing a bicycle is presented first. This is followed by a section on trajectory tracking for a riderless bicycle, a topic that has received less attention compared to balance control, and the proposed solutions are usually restricted to simulations. At the end of the chapter, path planning of autonomous bicycles is discussed.

3.1 Balance controller

To balance a riderless bicycle a wide range of control approaches has been used, ranging from classic control methods such as the PID controller [32, 33], to more sophisticated methods such as Active Disturbance Rejection Controller (ACDR) [24] and fuzzy controller [34]. However, it is not only the control approaches that differ but also the actuation. Typically, there are three different approaches. The first approach is to attach a DC motor to the steering axis and use the regulation of the handlebar. This approach relies on the concept of steering the bicycle into the fall, similar to how a human balances a bicycle as discussed in Section 2.2. A second approach is to mount a flywheel or a motor-driven inverted pendulum on the bicycle and regulate the lean angle of the bicycle directly, similar to a human locking the handlebar in place and balancing solely by upper body motion. Finally, as a third option, a combination of the two former approaches can be used. Regardless of the actuation method, all these three approaches rely on sensing the roll angle of the bicycle. This is commonly accomplished by an Inertial Measurement Unit (IMU) or an Attitude and Heading Reference System (AHRS), but successful experiments have also been reported where a laser beam was utilised to compute the lean angle of a stationary bicycle [35]. However, such an approach is difficult to realise

outside a laboratory environment.

In 2005 Tanaka and Murakami presented experimental results from balancing a bicycle using only actuation of the forward velocity and the handlebar [32]. The error between the sensed lean angle and a reference of zero was fed to a PD controller which controlled the handlebar. The system was first evaluated in simulations, and then in experiments on a bicycle running on a bicycle roller. Actuation of the handlebar for balancing a bicycle was also used in the work of He et al. [20]. In their work, both experimental and simulation results of balancing a bicycle using feedback and feedforward control were presented. The experiments were conducted on an ordinary-sized bicycle where right and left turns were commanded to the bicycle through a radio controller. To change the heading of the bicycle, the lean angle and lean rates references were altered, which in extension adjusts the steering angle. The controller was designed using the model presented by Yi [18], linearised around its equilibrium and with a constant velocity. Nevertheless, using a lookup table for several different velocities the control parameters could vary with the velocity. However, it is not clear from the paper how the experiments were conducted and if varying velocities were considered in one experiment, or if several experiments were conducted where a different constant velocity was used for each experiment.

Defoort and Murakami relied on nonlinear control techniques for stabilising a bicycle [36]. More specifically, a second-order Sliding Mode Controller (SMC) in combination with a disturbance observer was used to balance the bicycle in both simulation and experiments by actuation of the handlebar. A comparison between the SMC with and without the disturbance observer, as well as a PD controller with the disturbance observer was conducted in simulation on a nonlinear bicycle model. The results from the simulation clearly show that the SMC with disturbance observer converges faster to the desired lean angle compared to the other two controllers. The same bicycle and similar experimental setup as in the work of Tanaka and Murakami [32] were utilised to evaluate the proposed solution. It was shown that the controller could balance the bicycle at several different velocities, ranging from 0.4 to 3m/s.

Baquero-Suárez et al. [24] used an active disturbance rejection controller to stabilise an ordinary-sized bicycle equipped with actuators for controlling the rear wheel velocity and the steering torque. By computing the characteristic equation of the Whipple model (2.2), similar analyses as in Figure 2.1 were conducted to investigate the velocity range for self-stabilising. However, it is questionable how accurate such a velocity analysis is since the self-stabilisation range assumes self-stabilisation without any external actuation of the handlebar. Obviously, a servo motor attached to the handlebar as in the case of their bicycle will violate this assumption. An analysis of the zeros of the system

from the steering torque to the lean angle was also performed. This analysis revealed that the system becomes a non-minimum phase, and is much more difficult to control, at velocities below $v_l = 0.78\text{m/s}$ and therefore the controller was designed for velocities higher than v_l . The control strategy was first evaluated in simulation on a detailed CAD model of their bicycle, imported to Adams View and controlled through co-simulation with Matlab. Next, experiments were conducted where both a constant velocity and a velocity profile between 2.3 and 3.1m/s were considered. The results clearly show that the bicycle could balance in both simulation and on straight asphalt tracks under varying forward velocities, but with noticeable oscillation in both the lean and steering angles in experiments. An ADRC scheme was also utilised in [37] to balance a bicycle in simulation. An Extended State Observer (ESO) was designed to estimate the uncertainties of the system such as unmeasured states, physical parameters and unmodeled dynamics. Based on the properties of the ESO, a feedback linearization control law was proposed to control the steering angle. The system was evaluated in a co-simulation between V-rep and Matlab and the outcome shows that the ESO was able to estimate the uncertainties, and the controller was able to balance the bicycle at several different velocities.

The bicycle is one of the most energy-efficient systems for transportation [33], though, the energy consumption of the control algorithms for riderless bicycles is rarely considered. In the work of Rodriguez-Rosa [38] an adaptive PI control scheme was proposed to address the issue of energy consumption. The steering angle was considered as the control signal. The integral and proportional gain of the PI controller was optimised with respect to energy consumption, steady-state error, settling time, and disturbance rejection. Furthermore, as the velocity was changing, so were the gains. The control energy was computed as the sum of the squared input signal. Not only does the bicycle model stabilise in simulation, but the adaptive controller consumes significantly less energy compared to a PI controller which had the same gains for all velocities.

Instead of mounting actuators and sensors to a bicycle, Huang et al. used a humanoid to pedal, steer, and in extension balance their miniature bicycle [39]. The bias in the roll angle measurements was estimated using an online estimator and used in an LQR with integral action to balance the bicycle. To model the system a linearised point mass model was utilised, similar to the one proposed by Åström et al. [40]. By inverse kinematics, the humanoid could apply the commanded steering angle and forward speed to the bicycle. Moreover, a weight was attached to the right side of the handlebar with the purpose of disturbing the system in experiments. To evaluate the performance of the proposed controller, it was compared to a PD controller. It is evident from the

results that the proposed controller was superior compared to the PD controller in straight-line balancing when the disturbance was present. This is not surprising, as the PD controller is lacking an integral action that is efficient for steady-state tracking, additionally, the biased lean angle measurements were used when the PD controller was evaluated. An extension of their work can be found in [41] where the proposed controller was shown to be superior to a full-state feedback controller without the estimator.

For a human, mastering the balance of the bicycle is typically learnt by performing numerous less successful attempts. A similar approach can be used for balancing a riderless bicycle, using reinforcement learning. Such an approach was used by Tuyen and Chung [26], where a nonlinear model of a bicycle was stabilized in simulation. The Deep Deterministic Policy Gradient (DDPG) [42] algorithm was used for learning how to balance the bicycle using steering actuation. Three different reward functions were considered, one where the agent was rewarded if the lean angle of the bicycle was kept close to the equilibrium for a full trial, which lasted for 60 seconds. The two other reward functions considered a weighting between the lean angle, rate and acceleration and these reward functions converge to a solution much faster compared to the function which only considered the lean angle. To evaluate the system, the nonlinear bicycle model in [43] was utilised. A LEGO bicycle was considered for further evaluation, however, the limited computational power of the hardware that was mounted on the bicycle was found to be insufficient for this control algorithm.

As mentioned before, another approach for stabilisation of a bicycle is by sole actuation of the lean angle, for example by using a moving mass or a flywheel. On the one hand, both these approaches share the advantage of being able to balance at zero velocity [44]. On the other hand, the approaches struggle to balance at higher velocities [20]. Nevertheless, balancing a bicycle using a spinning flywheel has shown several successful results [22, 45, 46] or using a moving mass as proposed by Keo et al. [47]. To control the moving mass, and in extension balance the bicycle, an output zeroing controller was utilised and the system was evaluated in simulations. However, in 2011, Keo et al. concluded that a flywheel balancer was superior to the moving mass in terms of balancing a bicycle [44]. By spinning the flywheel, a torque is generated to keep the bicycle stabilised. To control the direction of the torque, the flywheel is often used in a Control Moment Gyroscope (CMG) where one or more motorized gimbals are used to control the angle of the flywheel. Thanh et al. [45] used a CMG to maintain the balance of a bicycle at zero forward speed. To control the CMG a $\mathcal{H}_2/\mathcal{H}_\infty$ controller was optimised using a Particle Swarm Optimization (PSO) algorithm. The controller was compared to another $\mathcal{H}_2/\mathcal{H}_\infty$ optimised using the Genetic Algorithm, as well as a traditional PD controller. They concluded

that the $\mathcal{H}_2/\mathcal{H}_\infty$ controller optimised using PSO shows better performance in terms of step response in simulation and also in terms of balancing the bicycle in experiments. Furthermore, a fuzzy sliding mode controller was used in the work of Hsieh et al. [46] to balance a stationary bicycle equipped with a CMG. The bicycle managed to balance even though it was subject to lateral forces during experiments.

Finally, a third alternative for balancing a riderless bicycle is to use a combination of actuation on the lean and steering angle. By using the lean torque actuation as the main actuator in low velocities and the steering actuation in high velocities the bicycle can balance in a wide range of velocities as shown in simulations by Garcia et al. [48]. Moreover, experimental results were reported in the work of Wang et al. [49], where a flywheel was used for the stationary balance of the bicycle and up till $v = 1.2\text{m/s}$. At $v = 1.2\text{m/s}$ steering actuation was used for balancing the bicycle. From the results, it is clear that it was more difficult to maintain balance in the transfer from moving to stationary, compared to going from stationary to moving.

3.2 Trajectory tracking

Trajectory tracking of an autonomous bicycle is more complex than for many other wheeled robots. This is mainly due to the fact that a bicycle requires active actuation to maintain stability. Moreover, if the autonomous bicycle is stabilised through steering actuation, it is not straightforward to track the desired path using the steering input as well. Instead, a desired lean angle can be used which an inner balance controller can track. Such a solution is proposed in the work of Dao and Chen [50] where a multi-loop control was used to realise both balance and trajectory tracking. In the inner loop, an SMC was used to balance the bicycle and realise tracking of a desired lean angle. The outer loop consists of a fuzzy logic controller with gain scheduling and integral action. Small tracking errors were reported when the proposed solution was evaluated in simulation. Baquero-Suárez et al. [24] also considered an extension of their ADRC approach to realise trajectory tracking. They established a mathematical relationship between the lean angle of the bicycle and its yaw angle, thus by manipulating the desired lean angle they could track the desired yaw angle. The solution was evaluated in multi-body simulations for low velocities, 1.5m/s , and only for wide circular paths and straights. Shafiei and Emami instead use a numerical approach to establish a relationship between the desired yaw angle and the reference lean angle in their work [51]. First, an inner control law is derived which stabilises the bicycle by using the steering input to track a desired lean angle. Next, a relationship between the desired lean angle and

the steady-state steering angle is numerically established by means of numerous simulations using different desired lean angles at varying velocities. This relationship is fitted with a fifth-order polynomial and is then used to compute the desired roll angle needed for tracking a reference path. The presented result from simulations shows good tracking performance while maintaining balance.

A nonlinear control approach for trajectory tracking is proposed by Yi et al. [18] for a motorcycle. As a motorcycle is considered instead of a bicycle, the forward velocity was higher. The controller was evaluated in numerical simulations with noise on the measured variables. The velocity ranges approximately from 5m/s to 14m/s. Because of the high velocities and wide curves, the reported steering angles were small, around 2° . Thus, it is not clear if it would be possible to apply such a controller for an autonomous bicycle which in general operates at lower velocities. A nonlinear controller was also used in the work of Turnwald et al. [52], where a passivity-based trajectory tracking controller was designed. The reported results from the simulation were promising and the controller was also planned to be used in experiments [53], but because of problems with sensor accuracy, it is yet to be evaluated in experiments. An asymptotic trajectory tracking approach for autonomous bicycles is presented by Cui et al. [3]. An optimal control algorithm is used to balance the bicycle and backstepping is utilised for trajectory tracking. A comparison is made with the external-internal convertible (EIC) approach of Getz [54], and it is highlighted that the EIC approach manages to approximately track the desired trajectory, while the proposed method can asymptotically track the trajectory when evaluated on an analytic model of a bicycle. Experimental results of a trajectory tracking controller for an autonomous bicycle are reported in the work of Wang et al. [55]. A feedback controller is designed by examining the EIC structure of the autonomous bicycle dynamics, inspired by the previous work of Getz. The autonomous bicycle is equipped with three actuators, one propulsion motor, one steering motor, and a gyro balancer for direct regulation of the lean angle. Two controllers are compared, in both controllers the gyro balancer is used for lower velocities and stationary balance of the bicycle. However, the first controller only uses steering actuation at higher velocities to track a given trajectory, while the second controller also incorporates the gyro balancer to improve the tracking performance at higher velocities. From the results, it is suggested that the second controller is superior to the first one. The results are also compared to a human riding a bicycle, and the results are similar in terms of tracking performance and control.

3.3 Path planning

Path planning deals with the problem of planning a feasible route on a map from an initial point to a goal point. By discretising the map into cells graph searching algorithms such as Dijkstra's [56], A* [57] or Theta* [58] can be used to find a feasible path. Each cell represents either free space or occupied space. It is also possible to use a sample-based path-searching algorithm, such as Rapidly Exploring Random Tree (RRT) [59], and hybrid methods, such as Hybrid A* [60], which does not require a discretized map. Hybrid A* relies on a set of manoeuvres that are constrained by the minimum turning radius of the vehicle. Another advantage of using Hybrid A* is the fact that it can plan smooth paths which adhere to the non-holonomic constraints of a bicycle. If grid search methods such as A* or Theta* are used, the path needs to be smoothed using methods such as path optimisation, polynomial interpolation, B-splines or Dubin's Curves [61].

In the work of von Wissel and Nikoukhah [62] an approach similar to Hybrid A* was used. A set of bicycle manoeuvres were pre-computed, and at the end of each manoeuvre, or path segment, a neutral position of the bicycle was considered with zero lean and steering angle. By enforcing this constraint, the case of turning right when leaning to the left can be avoided as this would make the bicycle lose balance. The result from numerical simulations shows that the proposed solution can plan a path, feasible for a bicycle, through a map with static obstacles.

Yuan et al. also focused on planning trajectories for a bicycle in their work [63]. The trajectory is created from a curve between two points in the XY plane, satisfying initial and final constraints on the yaw angle and the x and y position of the bicycle on the plane. Furthermore, the curve is parameterized by two third-order polynomials, thus there is one parameter in each polynomial that can be chosen freely to optimise the trajectory with respect to some quantity. In their work, they focus on minimising the roll angle of the bicycle, and the two parameters of the polynomials are found using PSO. However, they do not consider any obstacles when the proposed method is evaluated in simulation. This shortcoming is addressed in later work by Yuan et al. [64] where the optimisation is now solved by means of the Gauss pseudospectral method, where the cost function is the square of the roll angle. From the simulation results, it is clear that the proposed method can plan a smooth path while avoiding obstacles, however, there seem to be no constraints on the minimum turning radius, thus the planned paths sometimes have very narrow curves which would be difficult to track with an autonomous bicycle. Moreover, no simulation or experimental results are reported where a bicycle follows the planned path, thus

the proposed strategy is yet to be validated. Moreover, to compute the curves, the Getz bicycle mode [13] was used. However, as discussed previously, this model does not include the trail of the bicycle which is important for planning realistic paths for bicycles, as revealed in the work of Turnwald and Liu when they investigated motion planning for an autonomous bicycle [53]. In their work, they compare motion planning using models with and without trail. To compute the desired motion for a specific trajectory, the models are used to find the reference lean angle, steering angle and forward velocity throughout the trajectory. For evaluation, the resulting motion sets are tested on an instrumented bicycle. They conclude that the trailed model produces more natural motions compared to the model that assumes a vertical steering axis. However, there is no ground truth reported in their experiment which makes it difficult to evaluate how the different models differ from the desired trajectory.

Obstacle detection and avoidance for autonomous bicycles are considered in the work of Zhao et al. [65]. To detect obstacles, a Lidar is mounted in the front of the bicycle, replacing the position of the headlight. Moreover, an obstacle avoidance strategy consisting of four phases is used to avoid static obstacles. In the first phase, the bicycle tracks a global path. If an obstacle is detected the second phase is initialised, and the bicycle either turns right or left around the obstacle. The choice of direction is not only based on the position of the obstacle with respect to the bicycle but the current roll angle of the bicycle is also taken into consideration. This creates a smoother and more stable route for the bicycle. In phase three, the bicycle is passing the obstacle. Finally, in phase four, after passing the obstacle, the global path is tracked once again. The proposed solution is experimentally evaluated on an instrumented bicycle and successful results are reported.

The research presented in this thesis considers the balancing problem of an autonomous bicycle, the trajectory tracking problem, and path planning. Many of the previously proposed balance controllers are only evaluated in simulations and there are rarely any comparisons made to other control algorithms. Thus, we perform a comparative comparison of different control methods in simulations and experiments. Moreover, previous contributions in trajectory tracking for an autonomous bicycle have considered simplified paths. In this thesis, the trajectory tracking controller is evaluated using narrow curves and different velocities in detailed multibody simulations. Furthermore, we propose a path planning strategy that results in smooth paths that adhere to the non-holonomic constraints and the minimum turning radius of the bicycle. Finally, the instrumented bicycle used in this thesis does not consider direct lean angle regulation, instead, the actuation of the handlebar is used in combination with forward speed.

Chapter 4

Research Overview

In this chapter, the problem formulation is presented and a hypothesis is formulated. From the hypothesis, four research questions are defined. Next, the research method used in the thesis is presented with details of the experimental setup and simulation software. The chapter ends with a discussion on the problem of generalising the presented results.

4.1 Problem Formulation

An essential difference between an autonomous bicycle and many other autonomous wheeled vehicles is the fact that the bicycle needs actuation to maintain stability. With its two in-line wheels, it is a naturally unstable system. Therefore, the design and functionality of the low-level controller are prerequisites for any higher-level controller to function properly. At the same time, the performance of the low-level balance controller is dependent on how the system is modelled, i.e. what assumptions are made, model accuracy etc. Moreover, an autonomous bicycle should not only be able to balance, but it is also important to investigate frameworks and algorithms for localisation, trajectory tracking, and path planning and how they can be adapted to an autonomous bicycle. A hypothesis is formulated as:

Hypothesis: *By extending previous bicycle models, a controller can be designed for balancing an instrumented bicycle in real-time by actuation of the rear wheel and steering axis, meanwhile following a reference trajectory that is planned in a realistic environment with static obstacles.*

From the hypothesis and the problem formulation, the following research questions (RQ) are defined:

RQ 1. *How can an autonomous bicycle's dynamics and kinematics characteristics be modelled, and how can the models be used for control design?*

The first research question investigates what models have been proposed in the literature for modelling the main dynamics and kinematics of a bicycle, and the possible benefits and limitations of the models. It also investigates if the models had previously been used for control design.

RQ 2. *How to design a controller for balancing an autonomous instrumented bicycle by actuation of the rear wheel and the steering axis and how to evaluate such a controller?*

The second research question addresses the problem of balancing a bicycle using no direct lean angle regulation. Furthermore, it will also investigate how different control techniques compare to each other in terms of balancing an instrumented bicycle and what performance metrics can be used for the evaluation of the different control methods.

RQ 3. *How can an autonomous bicycle track a trajectory with narrow curves and at velocities common for a bicycle, and maintain balance?*

The third research question investigates how a trajectory-tracking controller can be designed for an autonomous bicycle to track a reference trajectory. While tracking the reference trajectory, the bicycle needs to remain stable, given the actuation of the steering axis and forward velocity.

RQ 4. *How can a smooth path be planned for an autonomous bicycle in an environment with static obstacles?*

The final research question considers the problem of path planning for an unmanned bicycle. The planned path should adhere to the physical constraints of a bicycle, such as velocity and acceleration limits and the minimum turning radius. Moreover, due to the non-holonomic constraints of a bicycle, it is crucial that the planned path is smooth and avoids static obstacles.

4.2 Research Method

In this thesis, an iterative research approach is used. The process 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. The knowledge gained from the literature review is used to propose a solution to the identified problem, and the solution is then implemented in software. The solution is first evaluated in simulations and if the results are promising, the solution may be implemented on an embedded system where experiments are conducted on an experimental platform, see Figure 4.2. For example, the balance controllers in papers A and B are first evaluated in simulation on different models, and later in experiments on the instrumented bicycle. In both experiments and in simulations, a quantitative analysis is performed on the collected data. In experiments, sensors are utilised to measure the relevant parameters of the bicycle and two motors are used to control the motion of the instrumented bicycle. The trajectory tracking controller and the path planning algorithm have only been evaluated in simulations due to the lack of reliable localisation data of the bicycle in experiments. If promising results are obtained, the literature review, the implementation details, and the evaluation are collected in an article and submitted to a relevant venue for reviews and possibly a publication.

4.2.1 Simulation tools

As experiments are often time-consuming, and sometimes difficult to reproduce, the ideas are first implemented and tested in simulations using different software. First, the analytical models are implemented in Matlab ¹ and Simulink ². Moreover, the instrumented bicycle is dismantled and each component is measured, weighted, and a model of the bicycle is designed in SolidWorks ³. The SolidWorks model is then exported to two different multibody dynamic simulation tools and represents two nonlinear models of the bicycle. In papers B and C, Adams View ⁴ is used and co-simulations between Adams View and Simulink are performed. The relevant control loops are implemented in Simulink and the nonlinear graphical model is realised in Adams View. In paper D, Adams View is replaced with Simscape ⁵ which is an extension of Simulink and offers better performance in terms of execution time compared to

¹<https://mathworks.com/products/matlab.html>

²<https://se.mathworks.com/products/simulink.html>

³<https://www.solidworks.com/>

⁴<https://hexagon.com/products/adams-student-edition>

⁵<https://mathworks.com/products/simscape.html>

Adams View. The discrete time control schemes are implemented in Simulink and controls the graphical nonlinear Simscape model. Matlab is used to post-process the simulation data.

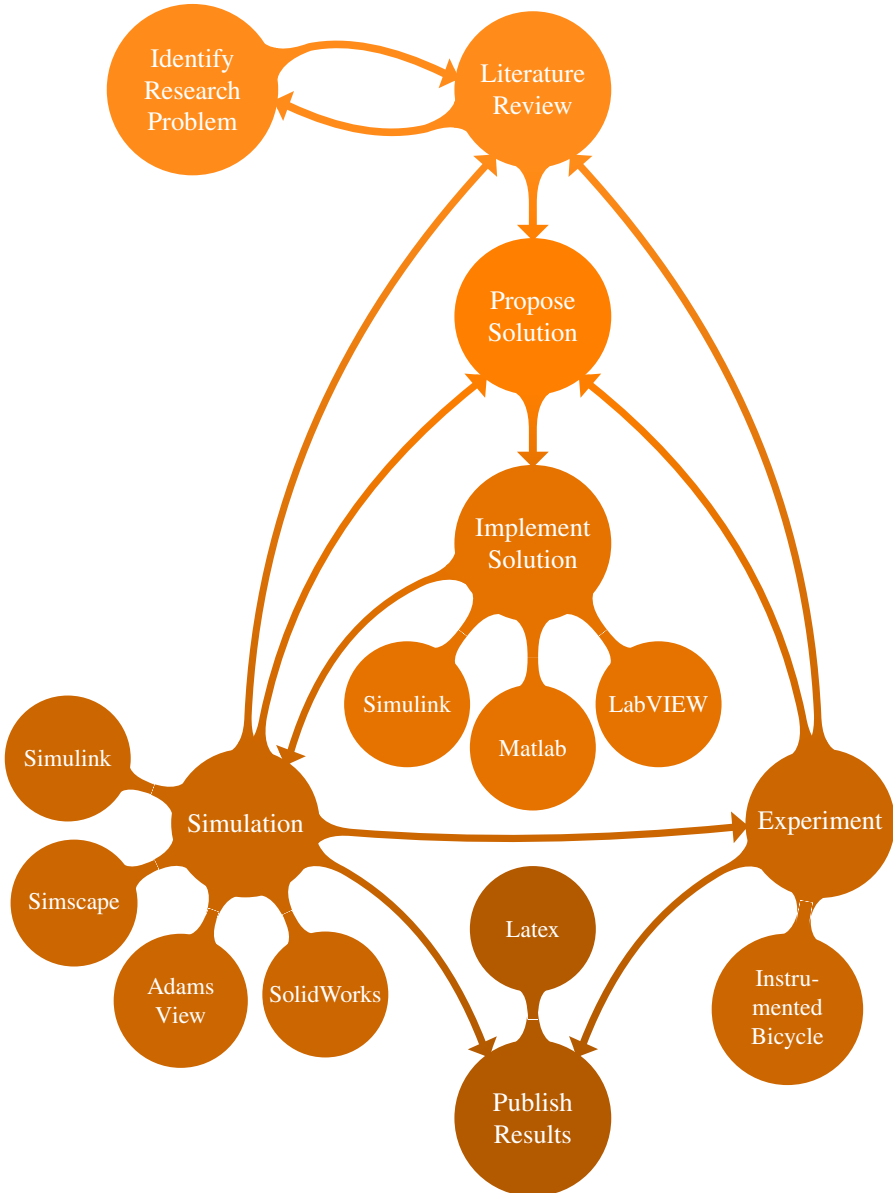


Figure 4.1: Research approach used in this thesis.

4.2.2 Experimental setup

An instrumented bicycle, based on a regular-sized electric bicycle of a male model is used in this thesis as an experimental platform, see Figure 4.2. The design of the bicycle was first developed in a student project at Mälardalens University [66]. The propulsion motor and battery are located in the rear wheel and on the mainframe of the bicycle respectively. The VN-100 AHRS is used for sensing the orientation of the bicycle and is mounted underneath the bottom bracket shell of the bicycle. The handlebar is actuated by a DCX32L Maxon motor together with a gear hub and encoder for sensing the position of the handlebar with respect to the mainframe, i.e the steering angle. A Hall sensor together with 12 evenly distributed magnets on the rear wheel is used for measuring the forward velocity of the bicycle. 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 LabVIEW⁶, which is a visual programming environment based on G-code and is specialised for tests and measurements. The sensor measurements and the actuation commands are processed on the FPGA. On the real-time target, the deterministic control loop and the logging of signals are performed. Communication between the real-time target and the FPGA target is realised through first in first out (FIFO) queues. The logged data is later post-processed in Matlab.

4.3 Threats to validity

An identified threat to the external validity is the fact that real-life experiments are only conducted on one bicycle. A similar problem is related to the multi-body simulations which are based on a Solidworks model of the instrumented bicycle. To address these issues, and generalise the proposed solutions, a fleet of instrumented bicycles would be needed, where the bicycle model, size, and instrumentation are altered.

⁶<https://www.ni.com/sv-se/shop/labview.html>



Hardware used on the instrumented bicycle








①	Electric Speed Controller Phoenix edge HV 60 AMP	
②	Hall effect sensor 103SR13A-9	
③	Bafang RM G040.250.DC Motor	
④	AHRS VN-100	
⑤	Battery 11.6 Ah 36V	
⑥	Junus motor controller	
⑦	Maxon DCX 32 L motor	
⑧	roboRIO (on opposite side)	

Figure 4.2: Instrumented bicycle used as the experimental platform in this thesis.

Chapter 5

Thesis Contributions

The research contributions of the thesis are presented in this chapter. A mapping between the four research questions and the four contributions is given in Table 5.1. The chapter is wrapped up with an overview of the papers included in the thesis.

5.1 Contributions

This thesis includes four contributions that address the research questions in Section 4.1. The research began with a literature review to understand how bicycles have been modelled in past. This is crucial as the design of a controller is highly dependent on the model, and how accurately the model can capture the main dynamics of a bicycle. The first contribution (**C1**) extends previous models with a transfer function that represents the steering dynamics, including the steering motor, of an instrumented bicycle. The second contribution (**C2**) is within applied control theory and is focusing on finding controllers for balancing an autonomous bicycle and important metrics for evaluating the performance of the controllers. As the controllers are evaluated on both different models and an instrumented bicycle, the comparison also evaluates the models. The third contribution (**C3**) makes use of both a bicycle model and a balance controller and integrates them into a tracking controller for solving the trajectory tracking of an autonomous bicycle while maintaining balance through the actuation of the steering axis. The fourth and final contribution (**C4**) of this thesis is within path planning for non-holonomic constrained vehicles, such as the autonomous bicycle. A mapping between the research questions (**RQ1-RQ4**) and the contributions (**C1-C4**) are given in Table 5.1, while the contributions with respect to a control hierarchy are illustrated in Figure 5.1.

Table 5.1: Mapping between research questions and contributions.

	RQ1	RQ2	RQ3	RQ4
C1	X	X		
C2	X	X		
C3		X	X	
C4				X

C1 - The steering dynamics of an autonomous bicycle are often neglected, for example, in [19] and [20] assumptions are made that the motor attached to the handlebar can instantly move from one position to another without any delay. Others assume a human operator, which of course is a valid assumption when studying cyclist behaviour. However, for an autonomous bicycle where the steering is motorized, it is important to consider not only the steering dynamics but also the dynamics of the motor. To this end, an extension of previous models was proposed. To model the steering dynamics, including the steering motor attached to the instrumented bicycle, a step response matching procedure was used. The bicycle was held in an upright position by a human, while a desired steering position or steering velocity was commanded to the motor controller. The output of the system was sampled and visualised in Matlab.

Furthermore, in Matlab, the sampled data was matched with a transfer function, $P(s)$, which could then be used to model the steering dynamics. The transfer function was put in series with previous models found in the literature. In paper A, $P(s)$ was put in series with the linear point mass model [40] for designing a controller which was evaluated in experiments on an instrumented bicycle. The same model was then used in paper B for designing different controllers. In both papers A and B, the controllers could successfully be transferred from simulation to real-life experiments without any further tuning, highlighting the usefulness of the modelling approach. Moreover, in paper C, $P(s)$ was put in series with the model proposed by Yi et al. [19]. Additionally, $P(s)$ was also successfully used in multi-body simulations using Adams View, in papers B and C, and Simscape in paper D. Thus, the proposed method of modelling the steering dynamics and steering motor using a step response matching method has successfully been used in papers A, B, C, and D. Furthermore, the simple, yet the useful procedure of step response matching can be generalised for similar autonomous bicycles as well. Finally, the Simscape multibody model has been made publicly available online ¹.

¹<https://github.com/NiklasPerssonMDU/On-the-Initial-of-Timed-Elastic-Bands.git>

C2 - The second contribution includes a comparative quantitative analysis between several different controllers and tuning techniques and the results are presented in papers A and B. The modelling approach in **C1** was used for tuning the different controllers. Moreover, the model was used as a first step to evaluate the performance of the controllers. For further evaluation, simulations were conducted on the nonlinear model in Adams View and experiments on an instrumented bicycle. To evaluate the controllers in simulations, the integrated squared error of the lean angle error and a step disturbance in the lean angle measurements were used. The step disturbance was meant to simulate a gentle push on the bicycle. Furthermore, on the instrumented bicycle the balancing time on a narrow roller was considered as well as the execution time of the controllers implemented on the embedded system. All metrics can easily be transferred to others controllers for evaluation. Based on the conclusion drawn in paper B, a PID controller was also used for balancing in paper C and paper D.

C3 - The third contribution is within trajectory tracking for an autonomous bicycle. A cascade control scheme was utilised, where the outer control loop computed a desired lean angle by considering the position and orientation of the bicycle and comparing it to a reference trajectory. Constraints on both dynamic and kinematic properties, such as the lean angle, steering angle and maximum tracking error were incorporated into an MPC framework. In the inner control loop, the aforementioned PID controller is utilised for stabilising the bicycle and tracking the reference lean angle. It is shown that the control approach can track a reference trajectory in high velocities and narrow curves, as in comparison to previous work where either wide curves or low velocities have been considered. By using the Adams model of the instrumented bicycle and inducing disturbance and noise in the measurements the solution was evaluated. The small Hausdorff distance and the low mean squared error between the reference trajectory and the actual trajectory highlighted the feasibility of the approach. The trajectory tracking controller was also used in paper D to show that the autonomous bicycle could track the reference trajectory from the proposed path planner. Furthermore, as different sampling rates are used in the different control loops, the trajectory tracker could be implemented on the instrumented bicycle.

C4 - The fourth and final contribution is related to the problem of path planning for a non-holonomic constrained vehicle such as a bicycle. More specifically, a strategy is proposed that is not only smooth but also tries to minimise the number of heading changes and the path distance in an environment with static obstacles. An initial path is planned using Theta* and then smoothed and optimised using TEB. Moreover, the optimisation problem takes into account

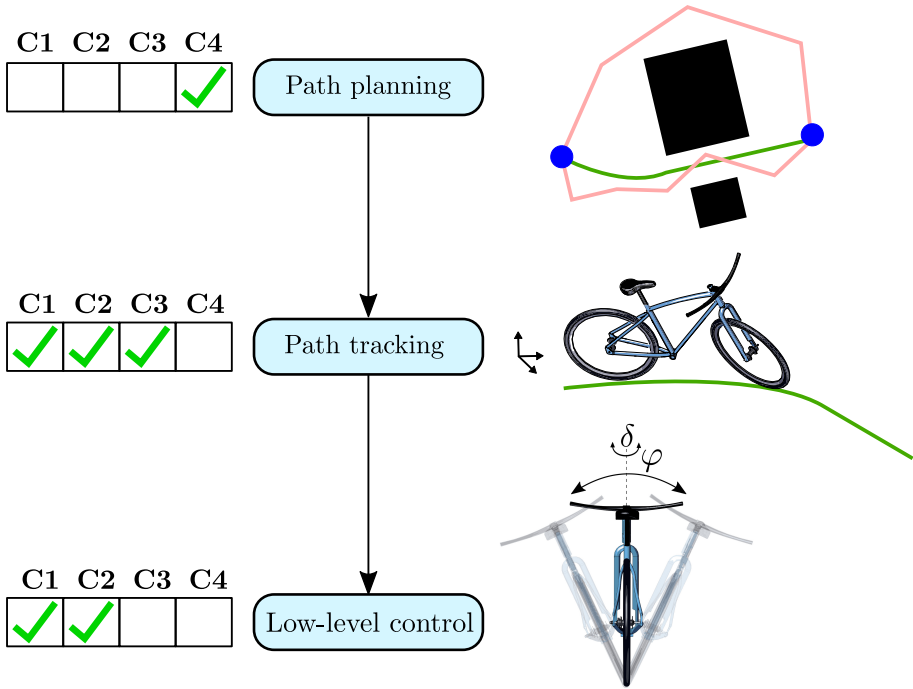


Figure 5.1: Contributions matched to a control hierarchy of an autonomous bicycle.

the geometric and dynamic constraints of the vehicle, such as the minimum turning radius and the maximum velocity while keeping a safe distance from the obstacles. We compare our approach to several other methods, and it is clear from the results that the proposed approach with Theta* and TEB have fewer heading changes and a shorter path distance compared to the other methods.

5.2 Overview of Included Papers

In this section, the four papers and how they relate to the contributions are given. Moreover, the personal contribution of the author of this thesis is also highlighted for each paper.

5.2.1 Paper A

Title: A Loop Shaping Method for Stabilising a Riderless Bicycle

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

Status: Published at European Conference on Mobile Robots (ECMR), 2019.

Abstract: Several control methods have been proposed to stabilise riderless bicycles but they do not have sufficient simplicity for practical applications. This paper proposes a practical approach to model an instrumented bicycle as a combination of connected systems. Using this model, a PID controller is designed by a loop shaping method to stabilise the instrumented riderless bicycle. The initial results show that the bicycle can be stabilised when running on a roller. The work presented in this paper shows that it is possible to self stabilise a riderless bicycle using cascade PI/PID controllers.

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

Personal Contributions Me and Tom Andersson were the main drivers of the paper and contributed equally to the idea, implementation, and writing of most of the manuscript. During the work, the other co-authors came with constructive feedback on the implementation and improved the manuscript in collaboration with me and Tom Andersson.

5.2.2 Paper B

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 at 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** and **C2**.

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 me and Tom Andersson. The other co-authors contributed with feedback and improved the manuscript in collaboration with me.

5.2.3 Paper C

Title: Trajectory tracking and stabilisation of a riderless bicycle

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

Status: Published at 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 **C1**, **C2**, and **C3**.

Personal Contributions I was the main author of the paper, contributing to the idea, the implementation, and the simulation setup. Also, I wrote the major part of the paper.

5.2.4 Paper D

Title: On the Initialization Problem for Timed-Elastic Bands

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

Status: Submitted

Abstract: Path planning is an important part of navigation for mobile robots. Several approaches have been proposed in the literature based on a discretisation of the map, including A^* , Θ^* , and RRT^* . While these approaches have been widely adopted also in real applications, they tend to generate non-smooth paths, which can be difficult to follow, based on the kinematic and dynamic constraints of the robot. Time-Elastic-Bands (TEB) have also been used in the literature, to deform an original path in

real-time to produce a smoother path, and to handle potential local changes in the environment, such as the detection of an unknown obstacle. This work analyses the effects on the overall path for different choices of initial paths fed to TEB. In particular, the produced paths are compared in terms of total distance, curvature, and variation in the desired heading. The optimised version of the solution produced by Theta* shows the highest performance among the considered methods and metrics, and we show that it can be successfully followed by an autonomous bicycle.

Paper contirbutions This paper relates to contribution **C3** and **C4**.

Personal Contributions I was the main author of the paper, contributing to the idea, the implementation, and the simulation setup. Also, I wrote the major part of the paper.

Chapter 6

Conclusions and Future Work

This thesis investigates a framework for an autonomous bicycle in a bottom-up approach. First, an extension for previous bicycle models is proposed where a step response matching procedure is used for capturing the steering dynamics, including the motor controlling the handlebar. The proposed model is then used to design balance controllers that can stabilise the bicycle by controlling the position of the handlebar and the forward velocity. The controllers are evaluated in both simulations, and later in experiments without any need for future tuning. In comparing different controllers for stabilising the riderless bicycle, the simple PID controller shows promising results and is recurring as the balance controller when trajectory tracking and path planning are investigated for an autonomous bike. A linear model that considers a tilted steering axis and captures the steering dynamics is used in an MPC framework to realise trajectory tracking. Constraints on both the dynamics and kinematics of the bike are included in the formulation of the controller, where the reference lean angle and velocity are the output. The reference lean angle is tracked by the aforementioned PID controller. Narrow curves and predefined varying reference paths are tracked initially. Next, we consider the problem of planning a smooth path for a non-holonomic constrained vehicle while adhering to the physical and dynamic constraints of the bicycle. The path is initially planned by the any angle path planning algorithm, Theta*, and optimised and smoothed using TEB. Finally, we show that we can control and guide a realistic multibody dynamic modelled bicycle in an environment with static obstacles by utilising the linear model, the balance controller, the trajectory tracking controller, and the path planner.

In the future, the trajectory tracking controller and the path planner should be evaluated in experiments on an instrumented bicycle. However, this calls for reliable localisation of the bicycle. Nonlinear filtering algorithms, such as

Extended Kalman filters or particle filters could be utilised in combination with the models in chapter 2 and data from GPS, Lidars, and cameras mounted on the bicycle. Furthermore, for the functionality of an autonomous bicycle operating on test tracks, the bicycle should be redesigned such that the components mounted on the bicycle are leaving a minimum footprint on the appearance of the bicycle. Moreover, the impact on the balance controller when a dummy is mounted on top of the bicycle, to mimic a rider, should be investigated. Finally, dynamic obstacle avoidance and capabilities of handling unknown environments should be investigated to improve path planning.

Bibliography

- [1] Boniphace Kutela, Subasish Das, and Bahar Dadashova. Mining patterns of autonomous vehicle crashes involving vulnerable road users to understand the associated factors. *Accident Analysis & Prevention*, page 106473, 2021.
- [2] Naroa Coretti Sanchez, Luis Alonso Pastor, and Kent Larson. Autonomous bicycles: A new approach to bicycle-sharing systems. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.
- [3] Leilei Cui, Shuai Wang, Zhengyou Zhang, and Zhong-Ping Jiang. Asymptotic trajectory tracking of autonomous bicycles via backstepping and optimal control. *IEEE Control Systems Letters*, 6:1292–1297, 2022.
- [4] Dániel Kondor, Xiaohu Zhang, Malika Meghjani, Paolo Santi, Jinhua Zhao, and Carlo Ratti. Estimating the potential for shared autonomous scooters. *IEEE Transactions on Intelligent Transportation Systems*, 23(5):4651–4662, 2022.
- [5] Jaap P Meijaard, Jim M Papadopoulos, Andy Ruina, and Arend 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*, 463(2084):1955–1982, 2007.
- [6] Francis Whipple. Stability of the motion of a bicycle. *Quarterly Journal of Pure and Applied Mathematics*, 30:312–348, 1899.
- [7] Emmanuel Carvallo. Theorie du mouvement du monocycle et de la bicyclette. *Journal de L'Ecole Polytechnique*, 5:119–188, 1900.
- [8] Jodi D.G Kooijman, Jaap P Meijaard, Jim M Papadopoulos, Andy Ruina, and Arend L Schwab. A bicycle can be self-stable without gyroscopic or caster effects. *Science*, 332(6027):339–342, 2011.
- [9] Jodi D.G Kooijman, Arend L Schwab, and Jaap P Meijaard. Experimental validation of a model of an uncontrolled bicycle. *Multibody System Dynamics*, 19:115–132, 2008.
- [10] Manfred Plöchl, Johannes Edelmann, Bernhard Angrosch, and Christoph Ott. On the wobble mode of a bicycle. *Vehicle system dynamics*, 50(3):415–429, 2012.

- [11] Arend L Schwab, Jaap P Meijaard, and Jodi D.G Kooijman. Lateral dynamics of a bicycle with a passive rider model: stability and controllability. *Vehicle System Dynamics*, 50(8):1209–1224, 2012.
- [12] Jason K Moore, Mont Hubbard, Arend L Schwab, and Jodi D.G Kooijman. Accurate measurement of bicycle parameters. In *Bicycle and Motorcycle Dynamics 2010 Symposium on the Dynamics and Control of Single Track Vehicles*, 2010.
- [13] Neil H Getz. Control of balance for a nonlinear nonholonomic non-minimum phase model of a bicycle. In *Proceedings of 1994 American Control Conference (ACC)*, volume 1, pages 148–151. IEEE, 1994.
- [14] Neil H Getz and Jerrold E Marsden. Control for an autonomous bicycle. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 2, pages 1397–1402, 1995.
- [15] David J.N Limebeer and Robin S Sharp. Bicycles, motorcycles, and models. *IEEE Control Systems Magazine*, 26(5):34–61, 2006.
- [16] Dylan E Meehan and Andy L Ruina. Linear and nonlinear controllers of an autonomous bicycle have almost identical basins of attraction in a nonlinear simulation. In *Bicycle and Motorcycle Dynamics 2019. Symposium on the Dynamics and Control of Single Track Vehicles*, 2020.
- [17] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.
- [18] Jingang Yi, Dezhen Song, Anthony Levandowski, and Suhada Jayasuriya. Trajectory tracking and balance stabilization control of autonomous motorcycles. In *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2583–2589. IEEE, 2006.
- [19] Jingang Yi, Yizhai Zhang, and Dezhen Song. Autonomous motorcycles for agile maneuvers, part i: Dynamic modeling. In *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, pages 4613–4618. IEEE, 2009.
- [20] Jiarui He, Mingguo Zhao, and Sotirios Stasinopoulos. Constant-velocity steering control design for unmanned bicycles. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 428–433, 2015.

- [21] Shahjahan Miah, Efstathios Milonidis, Ioannis Kaparias, and Nicholas Karcaniyas. An innovative multi-sensor fusion algorithm to enhance positioning accuracy of an instrumented bicycle. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1145–1153, 2020.
- [22] Yizhai Zhang, Pengcheng Wang, Jingang Yi, Dezhen Song, and Tao Liu. Stationary balance control of a bikebot. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6706–6711. IEEE, 2014.
- [23] Jodi D.G Kooijman, Arend L Schwab, and Jason K Moore. Some observations on human control of a bicycle. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 4, pages 2021–2028, 2009.
- [24] Mauro Baquero-Suárez, John Cortés-Romero, Jaime Arcos-Legarda, and Horacio Coral-Enriquez. A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle. *Multibody System Dynamics*, 45(1):7–35, 2019.
- [25] Everett X. Wang, Juncheng Zou, Gengping Xue, Yijun Liu, Yang Li, and Qun Fan. Development of efficient nonlinear benchmark bicycle dynamics for control applications. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):2236–2246, 2015.
- [26] Le P Tuyen and TaeChoong Chung. Controlling bicycle using deep deterministic policy gradient algorithm. In *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 413–417, 2017.
- [27] Arend L Schwab and Jaap P Meijaard. A review on bicycle dynamics and rider control. *Vehicle System Dynamics*, 51(7):1059–1090, 2013.
- [28] Camilo A Manrique-Escobar, Carmine M Pappalardo, and Domenico 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*, 8(1):153–181, 2021.
- [29] Jason K Moore, Jodi D.G Kooijman, Arend L Schwab, and Mont Hubbard. Rider motion identification during normal bicycling by means of principal component analysis. *Multibody System Dynamics*, 25(2):225–244, 2011.

- [30] Jodi D.G Kooijman and Arend L Schwab. A review on bicycle and motorcycle rider control with a perspective on handling qualities. *Vehicle System Dynamics*, 51(11):1722–1764, 2013.
- [31] Brendan Borrell. The bicycle problem that nearly broke mathematics. *Nature*, 535(7612):338–342, 2016.
- [32] Yasuhito Tanaka and Toshiyuki Murakami. Self sustaining bicycle robot with steering controller. In *The 8th IEEE International Workshop on Advanced Motion Control (AMC)*, pages 193–197. IEEE, 2004.
- [33] David J MacKay. *Sustainable Energy-without the hot air*. UIT cambridge, 2008.
- [34] Chih-Keng Chen and Thanh-Son Dao. Fuzzy control for equilibrium and roll-angle tracking of an unmanned bicycle. *Multibody system dynamics*, 15(4):321–346, 2006.
- [35] Maciej Rózewicz and Adam Piłat. Robust controller based on kharitonov theorem for bicycle with cmg. In *Advanced, Contemporary Control*, pages 411–423. Springer International Publishing, 2020.
- [36] Michael Defoort and Toshiyuki Murakami. Sliding-mode control scheme for an intelligent bicycle. *IEEE Transactions on Industrial Electronics*, 56(9):3357–3368, 2009.
- [37] Kanghui He, Chaoyang Dong, An Yan, Qingyuan Zheng, Bin Liang, and Qing 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*, pages 2766–2773. IEEE, 2020.
- [38] David Rodriguez-Rosa, Ismael Payo-Gutierrez, Fernando J Castillo-Garcia, Antonio Gonzalez-Rodriguez, and Sergio Perez-Juarez. Improving energy efficiency of an autonomous bicycle with adaptive controller design. *Sustainability*, 9(5):866, 2017.
- [39] Chun-Feng Huang, Yen-Chun Tung, and Ting-Jen Yeh. Balancing control of a robot bicycle with uncertain center of gravity. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5858–5863. IEEE, 2017.
- [40] Karl J Åström, Richard E Klein, and Anders Lennartsson. Bicycle dynamics and control: adapted bicycles for education and research. *IEEE Control Systems Magazine*, 25(4):26–47, 2005.

- [41] Chun-Feng Huang, Yen-Chun Tung, Hao-Tien Lu, and T-J Yeh. Balancing control of a bicycle-riding humanoid robot with center of gravity estimation. *Advanced Robotics*, 32(17):918–929, 2018.
- [42] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [43] Jette Randsløv and Preben Alstrøm. Learning to drive a bicycle using reinforcement learning and shaping. In *Proceedings of the Fifteenth International Conference on Machine Learning*, volume 98, pages 463–471. Morgan Kaufmann Publishers Inc., 1998.
- [44] Lychee Keo, Kiyoshi Yoshino, Masahiro Kawaguchi, and Masaki Yamakita. Experimental results for stabilizing of a bicycle with a flywheel balancer. In *2011 IEEE international conference on robotics and automation*, pages 6150–6155. IEEE, 2011.
- [45] Bui Trung Thanh and Manukid Parnichkun. Balancing control of bicyrobo by particle swarm optimization-based structure-specified mixed H_2/H_∞ control. *International Journal of Advanced Robotic Systems*, 5(4):39, 2008.
- [46] Ming-Hung Hsieh, Yen-Ting Chen, Cheng-Hung Chi, and Jui-Jen 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*, pages 13–18. IEEE, 2014.
- [47] Lychee Keo and Yamakita Masaki. Trajectory control for an autonomous bicycle with balancer. In *2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 676–681. IEEE, 2008.
- [48] M Ramos García, Daniel A Mántaras, Juan C Álvarez, and David Blanco F. Stabilizing an urban semi-autonomous bicycle. *IEEE Access*, 6:5236–5246, 2018.
- [49] Pengcheng Wang, Yongbin Gong, Jingang Yi, and Tao Liu. An integrated stationary/moving balance control of an autonomous bikebot. In *2019 American Control Conference (ACC)*, pages 3273–3278. IEEE, 2019.
- [50] Trung-Kien Dao and Chih-Keng Chen. Path-tracking control of a riderless bicycle via road preview and speed adaptation. *Asian Journal of Control*, 15(4):1036–1050, 2013.

- [51] Mohammad H Shafiei and M. Emami. Design of a robust path tracking controller for an unmanned bicycle with guaranteed stability of roll dynamics. *Systems Science & Control Engineering*, 7(1):12–19, 2019.
- [52] Alen Turnwald, Matthias Schäfer, and Steven Liu. Passivity-based trajectory tracking control for an autonomous bicycle. In *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, pages 2607–2612, 2018.
- [53] Alen Turnwald and Steven Liu. Motion planning and experimental validation for an autonomous bicycle. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 3287–3292. IEEE, 2019.
- [54] Neil H Getz. *Dynamic inversion of nonlinear maps with applications to nonlinear control and robotics*. PhD thesis, University of California, Berkeley, 1995.
- [55] Pengcheng Wang, Jingang Yi, Tao Liu, and Yizhai Zhang. Trajectory tracking and balance control of an autonomous bikebot. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2414–2419, 2017.
- [56] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
- [57] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4:100–107, 1968.
- [58] Alex Nash, Kenny Daniel, Sven Koenig, and Ariel Felner. Theta*: Any-angle path planning on grids. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, volume 2, pages 1177–1183, 2007.
- [59] Steven M LaValle. *Planning Algorithms*. Cambridge university press, 2006.
- [60] Janko Petereit, Thomas Emter, Christian W. Frey, Thomas Kopfstedt, and Andreas Beutel. Application of Hybrid A* to an autonomous mobile robot for path planning in unstructured outdoor environments. *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, 2012.
- [61] Abhijeet Ravankar, Ankit A Ravankar, Yukinori Kobayashi, Yohei Hoshino, and Chao-Chung Peng. Path smoothing techniques in robot

- navigation: State-of-the-art, current and future challenges. *Sensors*, 18(9):3170, 2018.
- [62] Dirk von Wissel and Ramine Nikoukhah. Maneuver-based obstacle-avoiding trajectory optimization: Example of a riderless bicycle. *Journal of Structural Mechanics*, 23(2):223–255, 1995.
- [63] Jing Yuan, Huan Chen, Fengchi Sun, and Yalou Huang. Trajectory planning and tracking control for autonomous bicycle robot. *Nonlinear Dynamics*, 78(1):421–431, 2014.
- [64] Jing Yuan, Jinhe Zhang, and Song Ding. Pseudospectral motion planning for autonomous bicycles. In *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 482–487, 2015.
- [65] Mingguo Zhao, Sotirios Stasinopoulos, and Yongchao Yu. Obstacle detection and avoidance for autonomous bicycles. *IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1310–1315, 2017.
- [66] Mikael Ekström and Therese Eriksson. Student report - auto-bike 2018. <http://www.es.mdh.se/publications/5454->, February 2019. [Online; accessed 25-Jan-2023].

II

Included Papers

Chapter 7

Paper A

A Loop Shaping Method for Stabilising a Riderless Bicycle

Tom Andersson, Niklas Persson, Anas Fattouh, Martin C. Ekström.
In European Conference on Mobile Robots, 2019.

Abstract

Several control methods have been proposed to stabilise riderless bicycles but they do not have sufficient simplicity for practical applications. This paper proposes a practical approach to model an instrumented bicycle as a combination of connected systems. Using this model, a PID controller is designed by a loop shaping method to stabilise the instrumented riderless bicycle. The initial results show that the bicycle can be stabilised when running on a roller. The work presented in this paper shows that it is possible to self stabilise a riderless bicycle using cascade PI/PID controllers.

7.1 Introduction

A bicycle is a well-known and popular means of transportation. The bicycle is often configured to have two inline wheels which make the system inherently unstable and difficult to control. The rider must train how to balance the bicycle. Balancing a bicycle is based on the simple concept, steering into the fall direction, but it is hard to be applied automatically as many features of the bicycle should be considered such as the trail, the gyroscopic torque, the mass distribution, and the forward velocity.

Two common models which represent, to some extent, the dynamics of a bicycle are the Whipple model and the point-mass model. The Whipple model, developed by Francis John Welsh Whipple in 1899, is the first analytic model that in a correct way describes the dynamics of a bicycle [1]. Linearised equations were derived from the Whipple model by Meijaard et al. [2] and they have been used in many implementations regarding control of a riderless bicycle, such as in the work by Baquero-Suarez et al. and Shafiekhani et al. [3, 4]. A less complex dynamic model of a bicycle is the point mass model, as described by Liembeer and Sharp [5], where a set of assumptions allows the bicycle to be modelled as an inverted pendulum, with its mass as a point mass.

Several control methods were proposed to stabilise the riderless bicycle using the point mass model [6, 7]. However, the proportional integral derivative (PID) is still attractive from an industrial point of view [8] and more investigation is needed to come up with an effective tuning method for PIDs.

In this paper, the mass-point model is considered to model the riderless bicycle. A robust PID controller design method is proposed to come up with the simplifications on the considered model. It is then used to stabilise an instrumented bicycle.

The remaining of this paper is structured as follows. Section II gives a brief survey of related works. The simulation model of the bicycle is explained in Section III. The proposed robust PID designed method is developed in Section IV. Section V is devoted to the application of the robust PID control design method on an instrumented riderless bicycle and the obtained results. Finally, concluding remarks and future works are given in Section VI.

7.2 Related Works

Different types of controllers have been proposed in the literature to control and balance riderless bicycles. From the more traditional ones, such as the Proportional-Integral-Derivative (PID) controllers, to more complex ones which rely on various Artificial Intelligence (AI) controllers.

Tanaka and Murakami [9] presented one of the first riderless bicycles which managed to keep its balance while riding on a roller. To control the bicycles steering axis, and by extension the balance, a PD controller was utilised along with disturbance observers. From the results, it is possible to see how the steering angle follows the lean angle which makes the bicycle balanced. A PD controller was also implemented in the work by Suebsomran [10], where a bicycle, equipped with a reaction wheel was kept stable. The controller regulates the angle of the reaction wheel to produce a necessary torque for the bicycle to be able to balance. The PD controller manages to stabilise the bicycle, however, only in a simulated environment. In the work by Wang *et al.* a cascade controller for balance and directional control of a bicycle-type two-wheeled vehicle is presented [11]. In the inner loop, a PD controller was used for balancing the vehicle by sensing the lean angle and output a steering torque to the plant. The outer loop, which composed of the directional control, also relied on a PD controller, but by sensing the yaw angle it outputs a reference lean angle which was fed to the inner loop. Experiments were made in simulation and on a real bicycle-type two-wheeled vehicle. Both the balance and the directional controller showed promising results. However, the platform used for testing was small and was some sort of hybrid between a kid bicycle model and small scooter. A regular sized bicycle have a greater height and mass compared to small bicycle, and also the center of gravity is generally at a greater distance from the ground which makes the regular sized bicycles more sensitive to disturbance and harder to control. Because of the size and the structure of the platform, the presented results cannot be generalised to a regular sized bicycle.

Since many systems, including bicycles, are simplified when modelled, there are some uncertainties present. A robust controller is specially designed to tune a system to have the desired behaviour, even with some uncertainties in the models [12]. Many of the proposed controllers in the related works are only evaluated in simulation and often experiments conducted on a real bicycle is missing. For example, in the work by Chen and Dao [13], a Sliding-Mode Controller (SMC) was proposed to track the bicycles roll angle. The benefits of using SMC to control a bicycle is that the uncertainties in the Whipple model's velocity can be compensated. The results showed that the bicycle was stable in 15km/h in a simulation environment.

Anjumol and Jisha [14] proposed an LQR controller to control a second-degree bicycle model. The results obtained from the controller was compared with a posture controlled proposed by Tanaka and Murakami [15]. The result from the comparison shows that an LQR controller performs better. However, the posture controller uses a PD controller with a disturbance observer which was better for disturbance rejection. An adaptive self-tuning regulator is pro-

posed by Al-Buraiki and Ferik [16]. The controller uses an estimation stage and a construction stage for the input signal, the first stage uses a weighted recursive least squares approach to estimate the models state-space, which is used in the second stage to construct the input signal. Additionally, in the second stage, an LQR controller is adapted for the on-line estimation. The proposed controller was only evaluated in simulation, where it successfully managed to balance the bicycle.

An AI-controller is used in the work by Sharma, where a fuzzy controller is presented to balance a bicycle[6]. The developed controller takes two inputs, the lean angle, and the steering angle, and outputs a correction lean. Thus, the controller relies on direct regulation of the lean angle which would require a reaction wheel, an inverted pendulum or something similar. In this paper, regulation of the steering angle and the rear wheel speed is used to keep the bicycle stable. Sharmas' controller manages to stabilise the bicycle in a simulated environment but was never used in any real-life experiments.

Real-life experiments are conducted in the work of Shafiekhani et al. [4] where an adaptive critic-based neuro-fuzzy controller was developed. A comparison was made between the neuro-fuzzy controller and a Fuzzy Inference System (FIS) controller, and it was concluded that the neuro-fuzzy offers more accurate performance in term of tracking the lean angle in both simulation and reality. Additionally, Abdolmalaki [17] presented a control system that relies on a FIS in combination with a PID controller to balance a bicycle. Results from experiments showed that the bicycle was able to balance along with a straight line, however with oscillations of $\pm 5^\circ$. The oscillations were also present when a sinusoidal trajectory was followed, despite this, the proposed control was still able to keep the bicycle from falling over in real-life experiments.

In 2018, Baquero-Suarez et al. [3] presented promising results where a regular sized bicycle, of a male model, equipped with sensors and actuators, manage to balance itself. It was able to follow a path using steering torque and forward velocity as the only control outputs. The results showed that the proposed system can balance even under small accelerations, however, the control structure is complex.

7.3 Modelling of the Riderless Bicycle

The riderless bicycle consists of three main parts, the bicycle, the steering unit, and the moving unit. The point-mass model is used in this paper to model the bicycle where a direct relationship between the lean angle and the steering angle are described. The steering motor is internally controlled to get the desired

second-order system dynamic. The rear wheel motor is controlled such that the bicycle can move in a constant speed and it is not considered in this paper.

7.3.1 The Point-Mass Model

The point-mass model is one of the more basic analytic bicycle models, it's a simple second-order linear model with a set of simplifications. The model assumes that both the front and rear wheel along with the front frame is massless, giving them inertia of zero. However, their masses are lumped together forming a point-mass, hence the models' name. To simplify the bicycle model further, it is assumed that the forward speed is constant and the heading angle λ and the trail distance are zero. Consider the bicycle shown in Fig. 7.1, with x -axis in the forward direction of the bicycle and the z -axis in the vertical direction. The two points, P1 and P2 are the contact points between the ground and the rear wheel and front wheel respectively. P3 is the point where the steering axis and the horizontal plane intersects with each other. The parameters a and h describe the distance from the rear wheel to the centre of gravity (CoG) in the x - and z -axis. λ is the head angle, c is the trail distance, and the b is the wheel base [18].

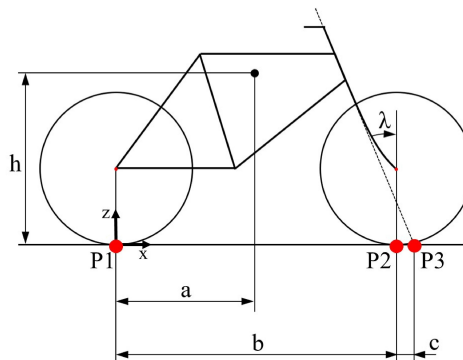


Figure 7.1: The a and h corresponds to the position of the CoG. The wheelbase is given by b , λ describes the head angle, and c is the trail.

Following the procedure described in [18], we get the following transfer function from steer angle δ to lean angle φ

$$\begin{aligned}
G_{\phi\delta}(s) &= \frac{v(Ds + mVh)}{b(Js^2 - mgh)} \\
&= \frac{vD}{bJ} \frac{s + \frac{mVh}{D}}{s^2 - \frac{mgh}{J}} \approx \frac{av}{bh} \frac{s + \frac{v}{a}}{s^2 - \frac{g}{h}}.
\end{aligned} \tag{7.1}$$

As can be seen from the transfer function, it will behave differently for different velocities. In this paper, a constant forward velocity of 14km/h and the bicycles physical parameters presented in Table 7.1 are considered.

Table 7.1: Instrumented bicycle parameters.

Design parameters			
Parameter	Symbol	Unit	Value
CoG with respect to O (x)	a	[m]	0.486
CoG with respect to O (z)	h	[m]	0.519
Gravity	g	[m/s ²]	9.820
Wheel base	b	[m]	1.080

7.3.2 Steering Response Matching

The internal structure of the position controller is composed of one PD steering angle controller followed by a speed and a current PI controller as shown in Fig. 7.7. Instead of modelling the three closed loop controllers, handlebar mass, friction and the motor characteristics, a step matching response method is applied where the recorded step response from the instrumented bicycle is matched with a timed delayed second-degree transfer function. To record the step response from the instrumented bicycle, it is held in an upright position with the wheels on the ground and a step of 3 degrees are commanded. The input to the transfer function is the desired steering angle and the output is the actual steering angle.

$$P(s) = e^{-d \cdot s} \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \tag{7.2}$$

Matching the response in Fig. 7.2 gives a transfer function with damping factor $\zeta = 0.6$, $\omega_n = 33.9$, and the time delay $d = 0.015$.

By coupling the system (2) and (1), i.e expanding the point-mass model with the dynamics captured from the steering system, the instrumental bicycle

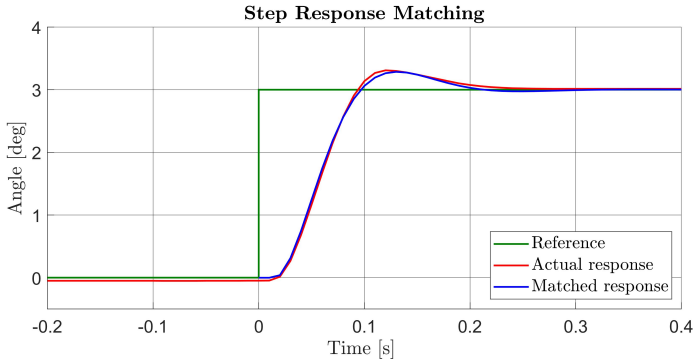


Figure 7.2: Recorded response along with the matched transfer function.

is modelled. Converted into discrete time, using zero-order hold as the discretisation method, with a sampling time of 0.01 seconds, the complete model is given by the following transfer function. The input to the system is the desired steering angle and the output is the current lean angle.

$$P(z) = z^{-2} \times \frac{0.000461z^3 + 0.00198z^2 - 0.00186z - 0.000324}{z^4 - 3.574z^3 + 4.813z^2 - 2.905z + 0.6658} \quad (7.3)$$

7.4 Balance Controller Design

In this section, the design problem of a PID controller is formulated as an optimisation problem that can be solved using an appropriate algorithm. The goal of the balance controller is to track a lean angle, by outputting a desired steering angle to the steering position controller. Consider the feedback control system in Fig. 7.3, where $G(z)$ is the plant, $K(z)$ is the controller, $r(t)$ is the reference signal, $y(t)$ is the system output signal, $u(t)$ is the control signal, $d(t)$ is the disturbance signal, and $n(t)$ is the noise signal.

The objectives of the feedback control loop are to ensure the stability of the closed loop system, good tracking performance, robustness against the plant uncertainties, and rejection of the disturbances affecting the system. These objectives can be formulated as constrained on the frequency response of the loop transfer function as follows [19]:

- The open loop frequency response should cross the 0 dB once with a constraint on the phase margin (to ensure the stability and the performance of the closed loop system).
- The gain of the open loop frequency response should be high below the desired bandwidth (to ensure the rejection of the disturbances).
- The gain of the open loop frequency response should be low above the desired bandwidth (to ensure the robustness against the plant uncertainties).

A controller that meets the above constraints can be designed by optimising the following objective function [20]:

$$\begin{aligned}
 J = & \omega_1(\omega_b - \omega_t)^2 \\
 & + \omega_2 \sum_{\omega > \omega_b} 20 \log |K(j\omega)P(j\omega)| \\
 & - \omega_3 \sum_{\omega \leq \omega_b} 20 \log |K(j\omega)P(j\omega)|
 \end{aligned} \tag{7.4}$$

where ω_t is the target bandwidth, $\omega_i, i = 1, 2, 3$, are weights, and ω_b is the bandwidth of the loop transfer function defined by:

$$\omega_b = \inf_{\omega} |K(j\omega)P(j\omega)| \leq 1 \tag{7.5}$$

The weights $\omega_i, i = 1, 2, 3$ have to be selected properly in order to approximately satisfy the design requirements. The following weights are suggested

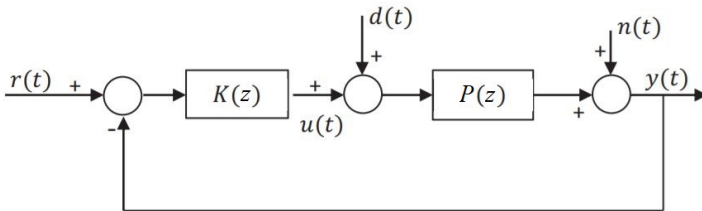


Figure 7.3: A standard feedback control system.

by [20]

$$\omega_1 = \frac{1}{\omega_t^2}, \omega_2 = \omega_3 = \frac{1}{2000} \quad (7.6)$$

The above optimisation problem is solved for the bicycle model in eq. 7.3 using *fmincon*, which is a nonlinear programming solver in MATLAB. The solution yields the following PID controller:

$$K(z) = K_P(1 + K_I T_s \frac{1}{z-1} + \frac{K_D}{T_s} \frac{(z-1)}{z}) \quad (7.7)$$

where $K_P = 2.5117$, $K_I = 1.5431$, $K_D = 0.075$, and $T_s = 0.01s$.

Fig. 7.4 shows the sensitivity function of the designed system. It shows that the largest value of the sensitivity function is 2.02 which is in the range of the recommended values and the system will have a good rejection of the disturbance in lean angle.

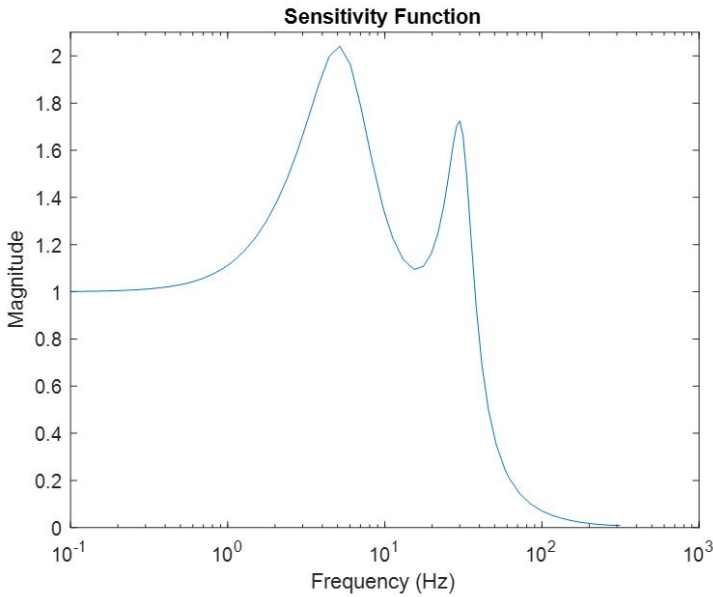


Figure 7.4: Sensitivity function of the designed system.

7.5 Simulation Results

The complete system is modelled in Simulink along with the noise of the lean angle sensor which has a measured variance of 0.0001 and a standard deviation of 0.01. In the Simulink model, the balance controller has a sampling

time of 100 Hz, the transfer function which represents the steering system executes in 600 Hz and the point-mass model is implemented in continuous time. The result of running the complete Simulink model with a disturbance on the lean angle is shown in Fig. 7.5. The simulation shows that the bicycle keeps balancing after disturbing the lean angle.

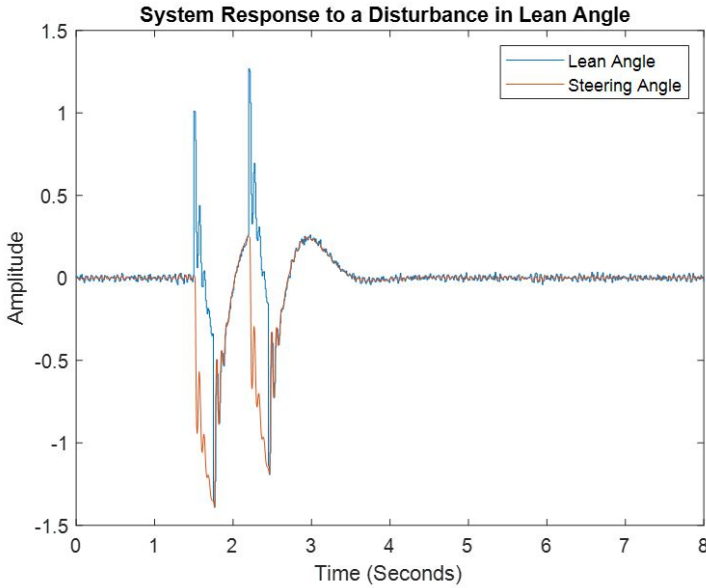


Figure 7.5: The steering and lean angles from simulation.

7.6 Application to the instrumented bicycle

The bicycle is a modified electrical bicycle of a regular sized male model with a motor in the rear wheel. The bicycle is equipped with a brushed DC motor for controlling the steering angle which is measured with an encoder. To control the steering motor a Junus motor controller is utilised. A VN-100 is used for measuring the lean angle of the bicycle and is mounted underneath the bottom bracket shell. To be able to send remote commands to the bicycle, a receiver is mounted on the bicycle. As the main processing unit a National Instruments roboRIO is utilised and the software is written using LabVIEW, the instrumented bicycle is shown in Fig. 7.6.



Figure 7.6: The instrumented bicycle with its rear wheel motor inside the green square. The blue box indicates the position of the VN-100, used for measuring the lean angle. Inside the yellow box is the electrical speed controller. National Instruments roboRIO is mounted on the bicycle and is highlighted by the purple box. To be able to send remote commands to the bicycle, the receiver inside the red box is utilised. The motor used to control the steering angle is highlighted by the orange rectangle.

7.6.1 Control structure

The control structure on the roboRIO is devised of an inner and an outer loop forming a cascade controller as shown in Fig. 7.7

The steering angle controller tracks the position of the handlebar and the outer loop tracks the desired lean angle of the bicycle. The steering angle controller uses a PD controller running on the roboRIO FPGA target with a loop speed of 600Hz, while the outer balancing PID controller is implemented on the real-time OS running with a loop frequency of 100Hz. The output of the steering angle controller is a PWM signal which is fed to the motor controller. Inside the motor controller, the PWM signal is mapped to the desired motor velocity and compared to the current motor velocity. The error is inputted to a PI controller and the resulting output is forwarded as the desired current to the last control loop which regulates the voltage going to the motor by using another PI controller.

Using software which accompanied the motor controller, the current controller is auto-tuned. The velocity controller is manually tuned using the same software and their respective control gains can be seen in Table 7.2 along with their saturation limits. The two controllers which reside on the roboRIO use the values shown in Table 7.3.

To control the forward velocity of the bicycle, a PI controller is imple-

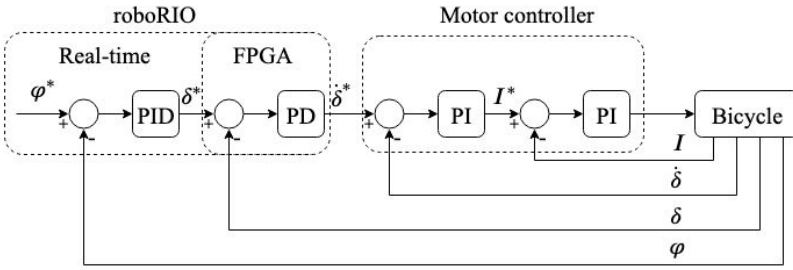


Figure 7.7: The complete control structure where φ^* , δ^* , $\dot{\delta}^*$, I^* represents the desired values and φ , δ , $\dot{\delta}$, I are the measured values from the bicycle. The balancing controller and steering angle controller are implemented on the roborRIO while the velocity and current controller reside on the motor controller.

Table 7.2: The gains used for the two PI controllers which reside on the motor controller.

Junus controllers			
Velocity controller		Current controller	
C_p	70	V_p	700
C_i	81	V_i	200
Limit	4.12 A	Limit	7000 RPM
Speed	4 kHz	Speed	20 kHz

mented on the FPGA target of the roborRIO. To measure the speed of the rear wheel, 12 magnets are mounted on the rear wheel, and a Hall sensor measures the pulses from the magnets and converts it to speed. The output from the PI controller is fed to the rear wheel motor through an electrical speed controller.

7.6.2 Experimental setups

To evaluate the proposed controller, the bicycle is placed on a bicycle roller with the front wheel pointing forward and is kept in an upright position by a human. Riding a bicycle on a roller is similar to riding a bicycle in an outdoor environment [21].

An experiment begins with a small calibration phase where the steering angle is initialised to zero and the IMU is powered on. During the calibration phase, it is important that the front wheel is pointing forward. Using a small offset, the IMU angle is also initialised to approximately zero. Next, the control loop is deployed and the rear wheel is powered on and accelerates to approximately 14km/h. When the bicycle rides at a nearly constant velocity the human

Table 7.3: Controller gains for the PD controller implemented on the FPGA target of the roboRIO and the PID controller which executes on the real-time target.

roboRIO controllers		
	Steering angle controller	Balance controller
K_p	0.10	2.5117
K_i	0	1.5431
K_d	0.04	0.0750
Output range	± 50	± 45
Speed	600 Hz	100 Hz
Filter coefficient	0.80	-

is releasing the bicycle and the control loop is fully in-charge of keeping the bicycle stable, as shown in Fig. 7.8.



Figure 7.8: Self stabilising bicycle on a roller running with a forward speed of 14 km/h. The width of the rollers is 37 cm.

Two experiments are conducted, in the first one the bicycle is simply riding on the bicycle roller and the signals are logged until a human has to interfere with the bicycle. To further evaluate the controller and its robustness, lean angle disturbance is injected in the lean angle measurements. The amplitude is 1 degree and its present for 0.25 seconds, and the disturbance is injected two times. Another possibility is to induce disturbance manually by introducing some lateral forces on the bicycle. However, by inducing the disturbance directly in the measurements, the experiment can easily be reproduced.

7.6.3 Results

In Fig. 7.9 and Fig. 7.10 the result from the bicycle experiment is presented. The signals are only logged while the bicycle self-stabilising, as soon as a human interacts with the bicycle the logging is turned off. Fig. 7.9 plots the lean and steering angle of the bicycle as well as the desired steering angle which is outputted from the balance controller. The forward speed of the bicycle is given in Fig. 7.10. In the experiment, the gains of the PID controller are taken from the tuning process done in simulation without any modifications and can be found in Table 7.3. The results from the disturbance rejection experiment are shown in Fig. 7.11.

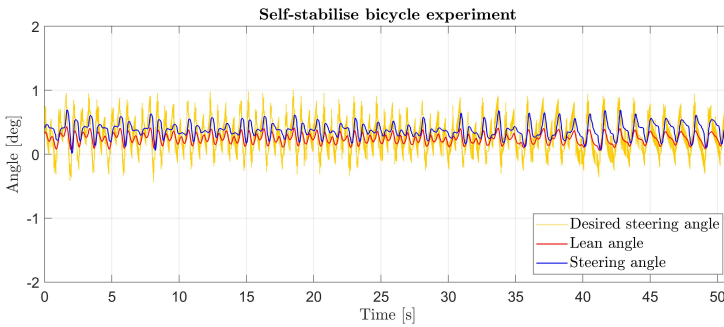


Figure 7.9: The result from the bicycle experiment where the bicycle is placed on a roller and a commanded forward speed of 14km/h is used. The signals are logged when the bicycle is self-stabilising on the roller. The PID controller, employed for balancing the bicycle, is first tuned in simulation and then implemented on a roboRIO.

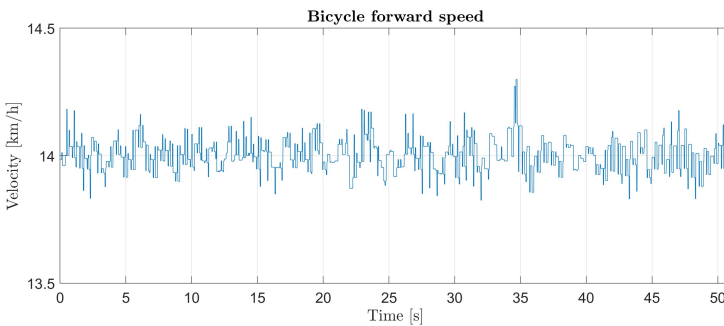


Figure 7.10: The forward speed of the bicycle during the experiment. To measure the speed of the bicycle, 12 magnets are mounted on the rear wheel and its pulses are measured with a Hall sensor. By calculating the time between the pulses and knowing the radius of the rear wheel it is possible to calculate the speed of the bicycle.

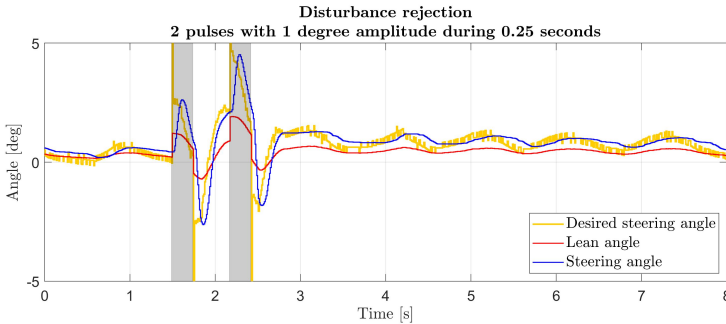


Figure 7.11: A disturbance is injected at approximately 1.5 seconds and another one at 2.2 seconds. Each disturbance period is present for 0.25 seconds and each period highlighted by a grey rectangle in the figure. The amplitude of the disturbance, which is injected in the lean angle measurements, is 1 degree.

7.7 Conclusion & future work

The work presented in this paper showed that it is possible to self stabilise a riderless bicycle using a cascade PID controller. By modelling a dynamic bicycle model, step response matching, and sensor noise a controller can be tune in simulation and adopted on an instrumented bicycle. The roller experiment showed that the instrumented bicycle is self-stabilising for 51 seconds or approximately 200 meters. Additionally, from the disturbance experiment, it is possible to see that the proposed controller can reject disturbance on the lean angle and still maintain to balance the bicycle. Both experiments are cancelled when a human has to interfere with the bicycle, this is due to the width limitations of the bicycle roller and the lack of trajectory tracking which makes bicycle drift to the sides. This could also be the result of asymmetric mass distribution of the bicycle or that the testing ground is not entirely flat. Additionally, in reality, the lean and steering angle might not be initialised to the absolute zero position. This needs to be investigated further, and an outer loop for trajectory tracking should be implemented. Furthermore, to avoid the limitations of the roller, future experiments should be conducted on a flat surface without the confined limitations of the roller.

7.8 Acknowledgements

This work is part of an ongoing research project between Mälardalen University, Chalmers University, Volvo Cars, AstaZero, and Cycleurope supported by Vinnova.

Bibliography

- [1] Francis Whipple. Stability of the motion of a bicycle. *Quarterly Journal of Pure and Applied Mathematics*, 30:312–348, 1899.
- [2] Jaap P Meijaard, Jim M Papadopoulos, Andy Ruina, and Arend 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*, 463(2084):1955–1982, 2007.
- [3] Mauro Baquero-Suárez, John Cortés-Romero, Jaime Arcos-Legarda, and Horacio Coral-Enriquez. A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle. *Multibody System Dynamics*, 45(1):7–35, 2019.
- [4] Ali Shafiekhani, Mohammad J Mahjoob, and Mahdi Akraminia. Design and implementation of an adaptive critic-based neuro-fuzzy controller on an unmanned bicycle. *Mechatronics*, 28:115–123, 2015.
- [5] David J.N Limebeer and Robin S Sharp. Bicycles, motorcycles, and models. *IEEE Control Systems Magazine*, 26(5):34–61, 2006.
- [6] Himanshu Dutt Sharma and Nagarajan Umashankar. A Fuzzy Controller Design for an Autonomous Bicycle System. *2006 IEEE International Conference on Engineering of Intelligent Systems*, pages 1–6, 2006.
- [7] Neil H Getz and Jerrold E Marsden. Control for an autonomous bicycle. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 2, pages 1397–1402, 1995.
- [8] Aidan O’Dwyer. *Handbook of PI and PID controller tuning rules*. World Scientific, 2009.
- [9] Yasuhito Tanaka and Toshiyuki Murakami. Self sustaining bicycle robot with steering controller. In *The 8th IEEE International Workshop on Advanced Motion Control (AMC)*, pages 193–197. IEEE, 2004.
- [10] Anan Suebsomran. Dynamic compensation and control of a bicycle robot. *2014 International Electrical Engineering Congress, iEECON*, pages 1–4, 2014.
- [11] Louis X Wang, Mikael J Eklund, and Vijyat Bhalla. Simulation & road test results on balance and directional control of an autonomous bicycle. *2012 25th IEEE Canadian Conference on Electrical and Computer Engineering: Vision for a Greener Future (CCECE)*, pages 1–5, 2012.

- [12] Khaled Halbaoui, Djamel Boukhetala, and Fares Boudjema. Introduction to Robust Control Techniques. *Introduction to Robust Control*, (May 2014):23, 2007.
- [13] Chih Keng Chen and Trung Kien Dao. A study of bicycle dynamics via system identification. *2010 International Symposium on Computer, Communication, Control and Automation (3CA)*, 2:204–207, 2010.
- [14] 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)*, pages 1–6, 2014.
- [15] Yasuhito Tanaka and Toshiyuki Murakami. A study on straight-line tracking and posture control in electric bicycle. *IEEE Transactions on Industrial Electronics*, 56(1):159–168, 2009.
- [16] Omar Al-Buraiki and Sami El Ferik. Adaptive control of autonomous bicycle kinematics. In *2013 13th International Conference on Control, Automation and Systems (ICCAS)*, pages 783–787, 2013.
- [17] Reza Yazdanpanah Abdolmalaki. Implementation of fuzzy inference engine for equilibrium and roll-angle tracking of riderless bicycle. *arXiv e-prints*, 2017.
- [18] Karl J Åström, Richard E Klein, and Anders Lennartsson. Bicycle dynamics and control: adapted bicycles for education and research. *IEEE Control Systems Magazine*, 25(4):26–47, 2005.
- [19] Ioan D Landau and Alireza Karimi. Robust digital control using pole placement with sensitivity function shaping method. *International Journal of Robust and Nonlinear Control*, 8(2):191–210, 1998.
- [20] Dennis Bruijnen, René van de Molengraft, and Maarten Steinbuch. Optimization aided loop shaping for motion systems. In *IEEE Int. Conf. on Contr. Appl.*, pages 255–260. IEEE, 2006.
- [21] Stephen M Cain, James A Ashton-Miller, and Noel C Perkins. On the skill of balancing while riding a bicycle. *PLoS one*, 11(2):e0149340, 2016.

Chapter 8

Paper B

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.

8.1 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/>

8.2 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. Vatansevanopakorn 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.

8.3 Modeling of instrumented bicycle

8.3.1 Experimental platform

The instrumented bicycle, illustrated in Fig. 8.1 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

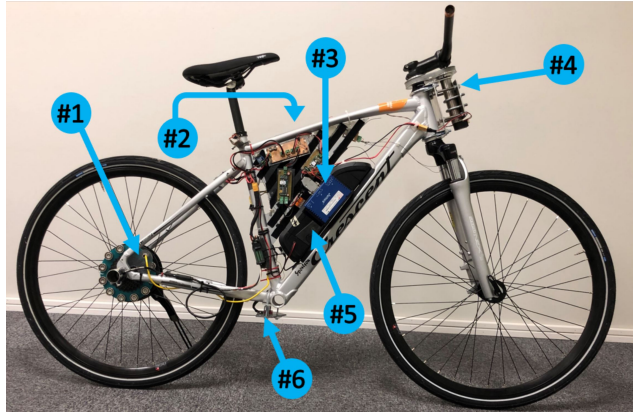


Figure 8.1: Instrumented bicycle. (#1) Propulsion motor; (#2) NI roboRIO; (#3) Motor controller; (#4) Steering motor; (#5) Battery; (#6) IMU.

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 8.1. 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

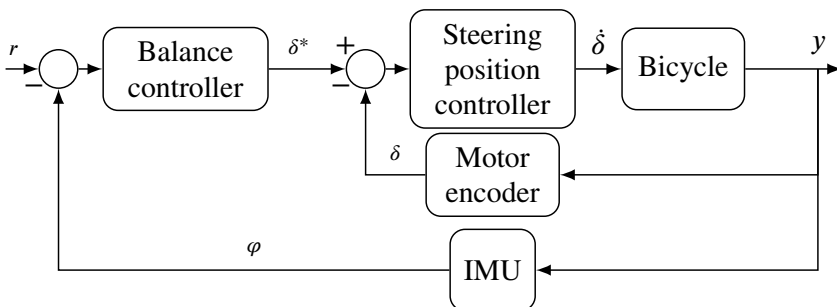


Figure 8.2: Control structure of the instrumented bicycle where a PD controller is used in an inner loop to control the steering position.

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. 8.2. 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. 8.3 and given in Table 8.1 along with constraints on the lean angle, lean rate, steering angle, and steering rate.

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
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

8.3.2 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]:

$$\ddot{\varphi} - mgh\varphi = \frac{v}{b}\dot{\delta} + \frac{mv^2h}{b}\delta, \quad (8.1)$$

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

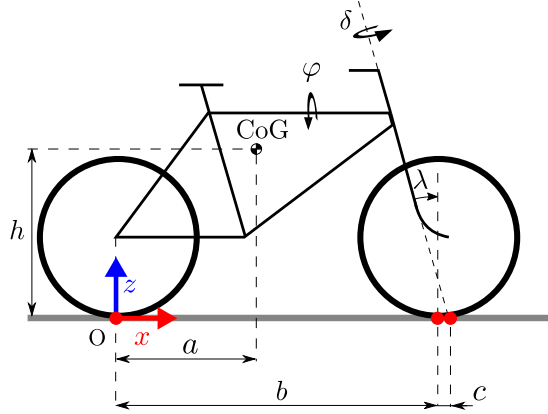


Figure 8.3: Geometrical features of a bicycle, where a and h corresponds to the measures horizontal and vertical measure of the CoG. The wheelbase is denoted by b .

where v is the forward velocity of the bicycle, and the physical parameters are reported in Table 8.1. 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

$$G(s) = \frac{\Phi(s)}{\Delta(s)} = \frac{av}{bh} \frac{s + \frac{v}{a}}{s^2 - \frac{g}{h}}. \quad (8.2)$$

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. 8.4.A, and matched with a Second-Order Plus Dead Time (SOPDT) transfer function

$$H(s) = \frac{\Delta(s)}{\Delta^*(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} e^{-ds}, \quad (8.3)$$

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 (8.1) is

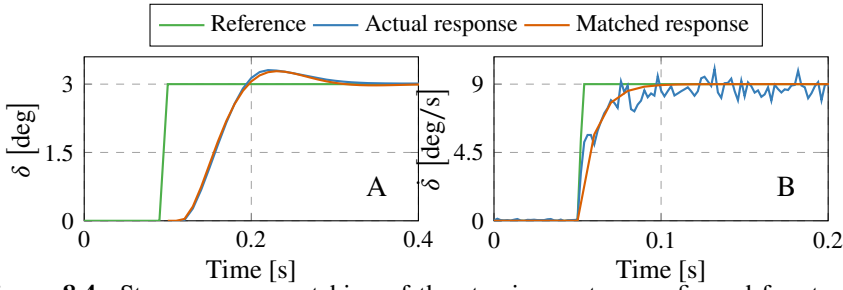


Figure 8.4: Step response matching of the steering system configured for steering position (A), and steering velocity (B). The matched responses are used in simulation to model the steering system including the motor and friction.

utilised in its state-space representation, with:

$$\mathbf{A}_1(v) = \begin{bmatrix} 0 & 1 & 0 \\ \frac{g}{h} & 0 & -\frac{v^2}{bh} \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_1(v) = \begin{bmatrix} 0 \\ -\frac{av}{bh} \\ 1 \end{bmatrix}, \quad (8.4)$$

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

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 (8.4) 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. 8.4.B.

The matched step response in Fig. 8.4.B is converted to its state-space, and its matrices are:

$$\mathbf{A}_2 = [-100], \quad \mathbf{B}_2 = [1], \quad \mathbf{C}_2 = [100], \quad \mathbf{D}_2 = [0], \quad (8.5)$$

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 (8.4) and (8.5) are combined

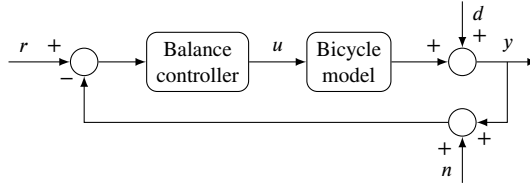


Figure 8.5: Reference feedback control scheme for balancing a bicycle.

in series as:

$$\begin{aligned} \mathbf{A}(v) &= \begin{bmatrix} \mathbf{A}_2 & \mathbf{0} \\ \mathbf{B}_1 \mathbf{C}_2 & \mathbf{A}_1 \end{bmatrix}, & \mathbf{B}(v) &= \begin{bmatrix} \mathbf{B}_2 \\ \mathbf{B}_1 \mathbf{D}_2 \end{bmatrix}, \\ \mathbf{C} &= [\mathbf{D}_1 \mathbf{C}_2 \quad \mathbf{C}_1], & \mathbf{D} &= [\mathbf{D}_1 \mathbf{D}_2]. \end{aligned} \quad (8.6)$$

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

8.3.3 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.

8.4 Control strategies

The closed-loop system in Fig. 8.5 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/>

8.4.1 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

$$U(z) = K_p \left(1 + K_I \cdot T_s \frac{1}{z-1} + K_D \cdot \frac{1}{T_s} \frac{z-1}{z} \right), \quad (8.7)$$

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:

$$\begin{aligned} J = & w_1 (\omega_b - \omega_t)^2 \\ & + w_2 \int_{\omega_b}^{\pi/T_s} 20 \log |K(j\omega)P(j\omega)| d\omega \\ & - w_3 \int_0^{\omega_b} 20 \log |K(j\omega)P(j\omega)| d\omega, \end{aligned} \quad (8.8)$$

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

$$\omega_b = \inf_{\omega} |K(j\omega)P(j\omega)| \leq 1, \quad (8.9)$$

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 8.2.

Table 8.2: 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

8.4.2 Linear quadratic regulator

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

$$J(\mathbf{u}) = \sum_{k=1}^{\infty} (\mathbf{x}(k)^{\top} \mathbf{Q} \mathbf{x}(k) + \mathbf{u}(k)^{\top} \mathbf{R} \mathbf{u}(k)) \quad (8.10)$$

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

$$\mathbf{u}(k) = -\mathbf{K} \mathbf{x}(k-1) \quad (8.11)$$

where \mathbf{K} is computed as

$$\mathbf{K} = (\mathbf{R} + \mathbf{B}^{\top} \mathbf{P} \mathbf{B})^{-1} \mathbf{B}^{\top} \mathbf{P} \mathbf{A} \quad (8.12)$$

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

$$\begin{aligned} \mathbf{P}(k-1) = & -\mathbf{A}^{\top} \mathbf{P}(k) \mathbf{B} (\mathbf{B}^{\top} \mathbf{P}(k) \mathbf{B} + \mathbf{R})^{-1} \mathbf{B}^{\top} \mathbf{P}(k) \mathbf{A} \\ & + \mathbf{Q} + \mathbf{A}^{\top} \mathbf{P}(k) \mathbf{A}. \end{aligned} \quad (8.13)$$

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 (8.6), converted to discrete time with a sampling time $T_s = 0.01$ 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

$$Q_{ii} = \frac{1}{\text{Max value of } x_i^2} \quad R_{jj} = \frac{1}{\text{Max value of } u_j^2} \quad (8.14)$$

with the constraints found in table 7.1. Now, by utilising equation (8.12) the following state-feedback gain, \mathbf{K} , is obtained:

$$\mathbf{K} = [22.46 \quad -37.35 \quad -4.91 \quad 8.77]^{\top}. \quad (8.15)$$

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

$$\dot{\delta}(t) \approx \frac{\delta(t) - \delta(t - T_s)}{T_s}, \quad T_s = 0.01\text{s}. \quad (8.16)$$

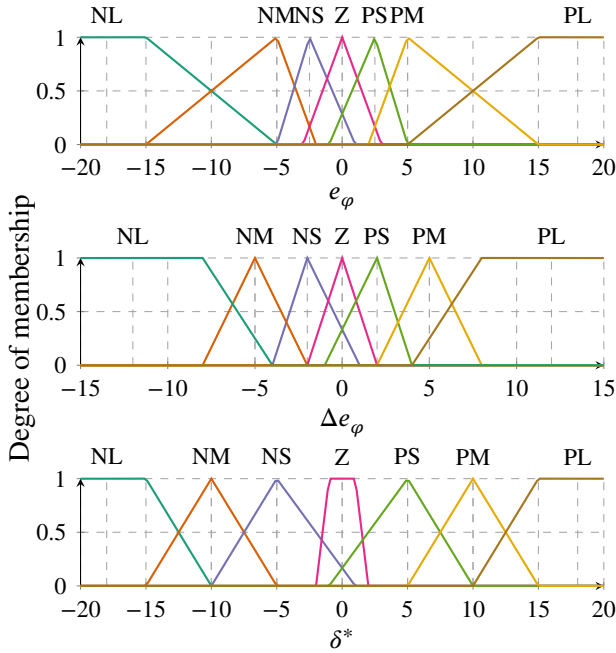


Figure 8.6: The fuzzy controllers membership functions, the first membership function is the lean angle error e_φ and the second membership function is the lean angle error difference Δe_φ . The output membership function presented in the bottom graph correlates to the desired steering angle δ^* .

8.4.3 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. 8.6 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 8.3, using the same abbreviations as in Fig. 8.6. 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 8.3: 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

8.5 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.

8.5.1 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>

8.5.2 Simulation results

The lean and steering angle of both the nonlinear and the linear models are shown in Fig. 8.7. 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 8.4.

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 8.4: 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

8.5.3 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. 8.8. There were no disturbance injected in the lean angle measurements in the experiments, mainly due to the experimental setup using a bicycle roller.

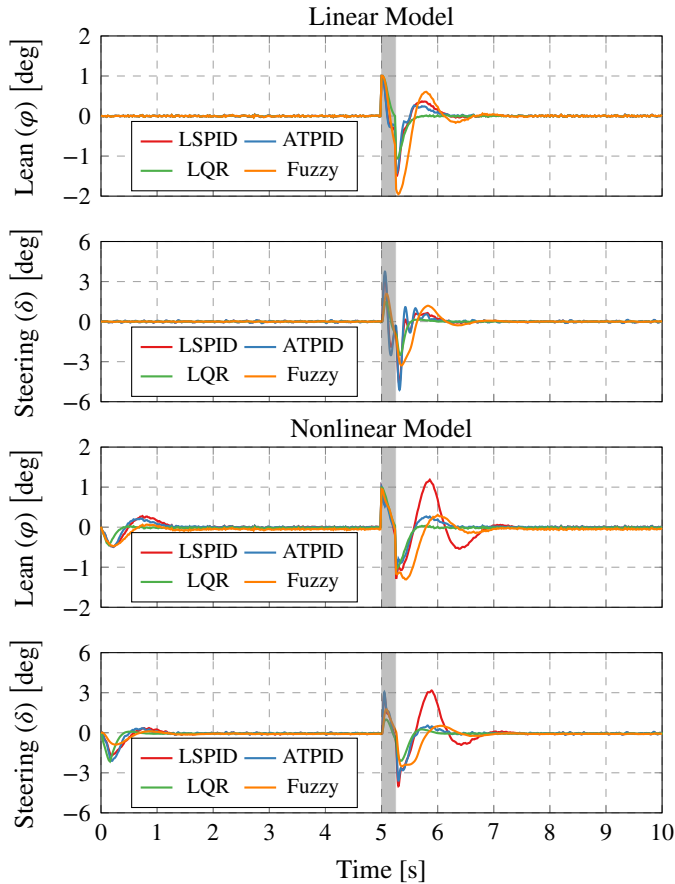


Figure 8.7: Results of the different control strategies when controlling the linear model and the nonlinear model in simulation.

8.5.4 Experimental results

The outcome of the experiments conducted on the roller is presented in Fig. 8.8. 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.

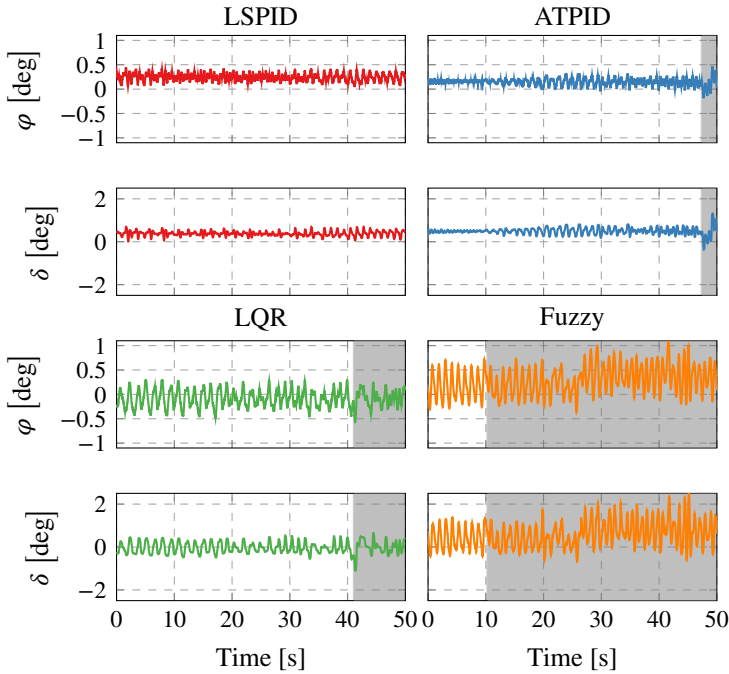


Figure 8.8: Results of the experiments conducted on the bicycle roller. The top row presents the lean angle of the bicycle while steering angle is reported in the second row for LSPID and ATPID. The third and the fourth row presents the lean angle and steering angle of the LQR and Fuzzy controller respectively. 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 8.5 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 ($T_s = 0.01$ 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.

8.6 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

Table 8.5: Execution time of the different control strategies

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

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.

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 Fabrikförening.

Bibliography

- [1] P. Fairley. Self-driving cars have a bicycle problem [news]. *IEEE Spectrum*, 54(3):12–13, 2017.
- [2] Neil H Getz and Jerrold E Marsden. Control for an autonomous bicycle. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 2, pages 1397–1402, 1995.
- [3] Karl J Åström, Richard E Klein, and Anders Lennartsson. Bicycle dynamics and control: adapted bicycles for education and research. *IEEE Control Systems Magazine*, 25(4):26–47, 2005.
- [4] Arend L Schwab and Jaap P Meijaard. A review on bicycle dynamics and rider control. *Vehicle System Dynamics*, 51(7):1059–1090, 2013.
- [5] Jie Tan, Yuting Gu, C. Karen Liu, and Greg Turk. Learning bicycle stunts. *ACM Trans. Graph.*, 33(4):1–12, 2014.
- [6] Mauro Baquero-Suárez, John Cortés-Romero, Jaime Arcos-Legarda, and Horacio Coral-Enriquez. A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle. *Multibody System Dynamics*, 45(1):7–35, 2019.
- [7] Ali Shafiekhani, Mohammad J Mahjoob, and Mahdi Akraminia. Design and implementation of an adaptive critic-based neuro-fuzzy controller on an unmanned bicycle. *Mechatronics*, 28:115–123, 2015.
- [8] Francis Whipple. Stability of the motion of a bicycle. *Quarterly Journal of Pure and Applied Mathematics*, 30:312–348, 1899.

- [9] Jaap P Meijaard, Jim M Papadopoulos, Andy Ruina, and Arend 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*, 463(2084):1955–1982, 2007.
- [10] Himanshu Dutt Sharma and Nagarajan Umashankar. A Fuzzy Controller Design for an Autonomous Bicycle System. *2006 IEEE International Conference on Engineering of Intelligent Systems*, pages 1–6, 2006.
- [11] John Hauser, Alessandro Saccon, and Ruggero Frezza. Aggressive motorcycle trajectories. *IFAC Proceedings Volumes*, 37(13):949–954, 2004.
- [12] Dylan E Meehan and Andy L Ruina. Linear and nonlinear controllers of an autonomous bicycle have almost identical basins of attraction in a nonlinear simulation. In *Bicycle and Motorcycle Dynamics 2019*. Symposium on the Dynamics and Control of Single Track Vehicles, 2020.
- [13] M Ramos García, Daniel A Mántaras, Juan C Álvarez, and David Blanco F. Stabilizing an urban semi-autonomous bicycle. *IEEE Access*, 6:5236–5246, 2018.
- [14] Arend L Schwab, Jodi D.G Kooijman, and Jaap P Meijaard. Some recent developments in bicycle dynamics and control. In *European Conf. on Structural Control (ECSC)*, 2007.
- [15] Jiarui He, Mingguo Zhao, and Sotirios Stasinopoulos. Constant-velocity steering control design for unmanned bicycles. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 428–433, 2015.
- [16] Yasuhito Tanaka and Toshiyuki Murakami. Self sustaining bicycle robot with steering controller. In *The 8th IEEE International Workshop on Advanced Motion Control (AMC)*, pages 193–197. IEEE, 2004.
- [17] Sorawuth Vatanashevanopakorn and Manukid Parnichkun. Steering control based balancing of a bicycle robot. In *2011 IEEE International Conference on Robotics and Biomimetics*, pages 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)*, pages 1–6, 2014.
- [19] Yasuhito Tanaka and Toshiyuki Murakami. A study on straight-line tracking and posture control in electric bicycle. *IEEE Transactions on Industrial Electronics*, 56(1):159–168, 2009.

- [20] Tom Andersson, Niklas Persson, Anas Fattouh, and Martin C Ekström. A loop shaping method for stabilising a riderless bicycle. In *2019 European Conference on Mobile Robots (ECMR)*, pages 1–6, 2019.
- [21] Dennis Bruijnen, René van de Molengraft, and Maarten Steinbuch. Optimization aided loop shaping for motion systems. In *IEEE Int. Conf. on Contr. Appl.*, pages 255–260. IEEE, 2006.
- [22] Vinodh Kumar E, Jovitha 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)*, pages 1–6, 2014.
- [23] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Pearson, 8th edition, 2019.
- [24] Reza Yazdanpanah Abdolmalaki. Implementation of fuzzy inference engine for equilibrium and roll-angle tracking of riderless bicycle. *arXiv e-prints*, 2017.

Chapter 9

Paper C

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.

9.1 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 9.2 the related work is presented. The bicycle model is derived in Section 9.3 followed by Section 9.4 where the control design is discussed. Section 9.5 presents the results and the paper is concluded in Section 9.6 which also includes future research.

9.2 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.

9.2.1 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>

9.2.2 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.

9.3 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. 9.1. The lean angle, $\varphi(t)$, and steering angle, $\delta(t)$, are positive following the right-hand rule.

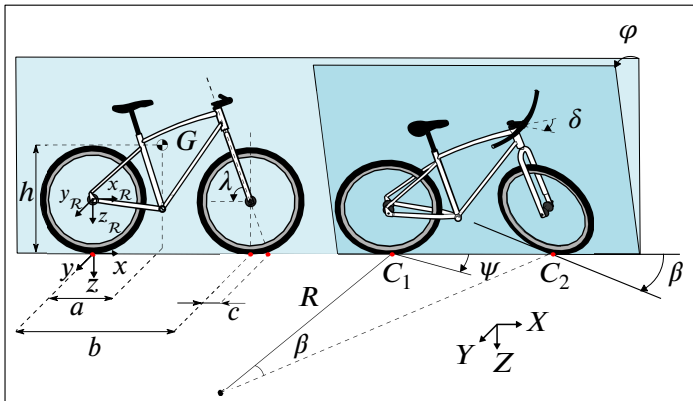


Figure 9.1: A bicycle riding on a flat horizontal plane.

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

$$\tan(\beta(t)) = \frac{\tan(\delta(t)) \sin(\lambda)}{\cos(\varphi(t))}, \quad (9.1)$$

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

$$\sigma(t) = \frac{b}{R(t)} = \tan(\beta(t)) \quad (9.2)$$

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

$$\sigma(t) \approx \delta(t) \sin(\lambda), \quad (9.3)$$

and the respective curvature rate of change:

$$\omega_\sigma(t) = \dot{\sigma}(t) \approx \dot{\delta}(t) \sin(\lambda). \quad (9.4)$$

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

$$\begin{aligned} \dot{x} &= v(t) \cos(\psi(t)) \\ \dot{y} &= v(t) \sin(\psi(t)) \\ \dot{\psi}(t) &= \frac{\sigma(t)v(t)}{b} = \frac{\tan(\delta(t)) \sin(\lambda)}{b \cos(\varphi(t))} v(t). \end{aligned} \quad (9.5)$$

The roll dynamics of the bicycle can be expressed as:

$$\begin{aligned} h^2 \ddot{\varphi}(t)m &= gm \left(h \sin(\varphi(t)) + \frac{ca}{b} \sin(\lambda) \sigma(t) \cos(\varphi(t)) \right) - \\ &\left(1 - \frac{h}{b} \sigma(t) \sin(\varphi(t)) \right) \frac{h}{b} \sigma(t) \cos(\varphi(t)) v^2(t)m \\ &- \frac{ahm}{b} \cos(\varphi(t)) \left(\sigma(t) \dot{v}(t) - v(t) \omega_\sigma(t) \right). \end{aligned} \quad (9.6)$$

Here, the mass, m , cancels out and using the relationships in Eq. (9.1) and Eq. (9.2) the roll dynamics can be simplified as:

$$\begin{aligned} h^2 \ddot{\varphi}(t) &= g \left(h \sin(\varphi(t)) + \frac{cap^2}{b} \tan(\delta(t)) \right) - \\ &\left(1 - \frac{hp}{b} \tan(\delta(t)) \tan(\varphi(t)) \right) \frac{hp}{b} \tan(\delta(t)) v(t)^2 \\ &- \frac{ahp}{b} \tan(\delta(t)) \dot{v}(t) - \frac{ah}{b} \cos(\varphi(t)) v(t) \omega_\sigma(t). \end{aligned} \quad (9.7)$$

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

$$h^2 \ddot{\varphi}(t) = g \left(h\varphi(t) + \frac{cap^2}{b} \delta(t) \right) - \left(1 - \frac{hp}{b} \delta(t)\varphi(t) \right) \frac{hp}{b} \delta(t)v^2 - \frac{ahp}{b} v\dot{\delta}(t) \quad (9.8)$$

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

$$\ddot{\varphi}(t) = \frac{g}{h} \varphi(t) + \frac{gcap^2}{bh^2} \delta(t) - \frac{p}{bh} v^2 \delta(t) - \frac{ap}{bh} v \dot{\delta}(t). \quad (9.9)$$

Eq. (9.9) 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:

$$\mathbf{A}_\varphi = \begin{bmatrix} 0 & \frac{g}{h} & \frac{p}{bh} \left(\frac{gcap}{h} - v^2 \right) \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_\varphi = \begin{bmatrix} -\frac{ap}{bh} v \\ 0 \\ 1 \end{bmatrix} \quad (9.10)$$

$$\mathbf{C}_\varphi = [0 \quad 1 \quad 0], \quad \mathbf{D}_\varphi = [0].$$

The relevant model parameters are obtained through measurements on the instrumented bicycle in [21] and presented in Table 9.1. 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 9.1: 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. 9.2 and the matched response is represented by a state-space model as:

$$\begin{aligned} \mathbf{A}_{\delta} &= [-100], & \mathbf{B}_{\delta} &= [100], \\ \mathbf{C}_{\delta} &= [1], & \mathbf{D}_{\delta} &= [0], \end{aligned} \quad (9.11)$$

with the desired steering velocity, $\mathbf{u}_{\delta} = \dot{\delta}^*$, as input and the actual steering velocity, $\mathbf{y}_{\delta} = \mathbf{x}_{\delta} = \dot{\delta}$, 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:

$$\begin{aligned} \mathbf{A}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{A}_{\delta} & \mathbf{0} \\ \mathbf{B}_{\varphi} \mathbf{C}_{\delta} & \mathbf{A}_{\varphi} \end{bmatrix}, & \mathbf{B}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{B}_{\delta} \\ \mathbf{B}_{\varphi} \mathbf{D}_{\delta} \end{bmatrix}, \\ \mathbf{C}_{\dot{\delta}\varphi} &= [\mathbf{D}_{\varphi} \mathbf{C}_{\delta} \quad \mathbf{C}_{\varphi}], & \mathbf{D}_{\dot{\delta}\varphi} &= [\mathbf{D}_{\varphi} \mathbf{D}_{\delta}], \end{aligned} \quad (9.12)$$

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

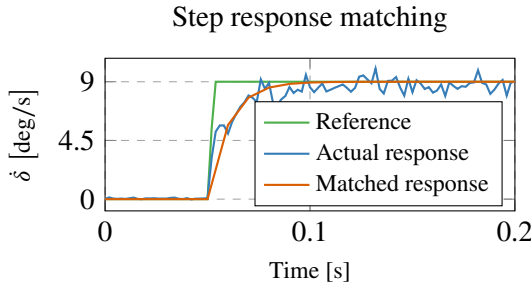


Figure 9.2: Step response matching of the steering system. The matched response is used in simulation to model the dynamics of the steering system.

9.4 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.

9.4.1 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. 9.3 where $R(s)$ is a real PID controller in parallel form

$$R(s) = K_p + \frac{K_i}{s} + \frac{K_d N}{1 + \frac{N}{s}}. \quad (9.13)$$

In the figure, $G(s)$ is the transfer function of the steering dynamics in Eq. (9.11), and $H(s)$ is the transfer function associated with Eq. (9.9), with input $\dot{\delta}$ and output φ , and it is equal to:

$$H(s) = \frac{\Phi(s)}{\dot{\Delta}(s)} = \frac{(gcap^2 - hpv^2) - ahpv s}{bh^2 s^3 - bhgs}. \quad (9.14)$$

The roll dynamics, including the actuation, can be therefore expressed as the transfer function $P(s) = G(s)H(s)$. As illustrated in Fig. 9.3, 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

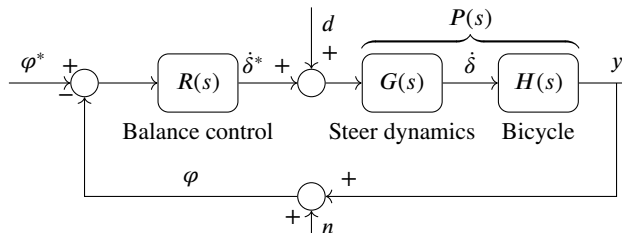


Figure 9.3: Balance controller $R(s)$ and the linear model $P(s)$.

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

$$\begin{aligned}
 & \min_{K_p, K_i, K_d, N} \quad (\omega_c^{\text{des}} - \omega_c)^2 + w_1 \int_0^\infty tq(t)^2 dt + w_2 \int_0^\infty t\ell(t)^2 dt \\
 & \text{s.t.} \quad K_p, K_i, K_d \in [-200, 0], \\
 & \quad N \in [10, 1000], \\
 & \quad \omega_c^{\text{des}} = 60\text{rad/s}.
 \end{aligned} \tag{9.15}$$

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. (9.15) 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. (9.15) 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:

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{(\varphi^2 - 4\varphi)}|}. \tag{9.16}$$

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 9.2.

The state-space matrices of the PID controller in (9.13) are defined as:

$$\begin{aligned}
 \mathbf{A}_R &= \begin{bmatrix} -N & 0 \\ 1 & 0 \end{bmatrix}, & \mathbf{B}_R &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \\
 \mathbf{C}_R &= [K_i - K_d N^2 \quad K_i N], & \mathbf{D}_R &= [K_p + K_d N],
 \end{aligned} \tag{9.17}$$

where the state vector is $\mathbf{x}_R = [e_1, e_2]^\top$, the output $\mathbf{y} = \delta^*$, and the input $\mathbf{u} = e$ with $e = \varphi^* - \varphi$. Now, the open loop system in Fig 9.3 can be written as:

$$\begin{aligned} \mathbf{A}_O &= \begin{bmatrix} \mathbf{A}_R & \mathbf{0}^{(2 \times 4)} \\ \mathbf{B}_{\delta\varphi} \mathbf{C}_R & \mathbf{A}_{\delta\varphi} \end{bmatrix} & \mathbf{B}_O &= \begin{bmatrix} \mathbf{B}_R \\ \mathbf{B}_{\delta\varphi} \mathbf{D}_R \end{bmatrix} \\ \mathbf{C}_O &= \begin{bmatrix} \mathbf{D}_{\delta\varphi} \mathbf{C}_R & \mathbf{C}_{\delta\varphi} \end{bmatrix} & \mathbf{D}_O &= \begin{bmatrix} \mathbf{D}_{\delta\varphi} \mathbf{D}_R \end{bmatrix} \end{aligned} \quad (9.18)$$

and by closing the loop we obtain

$$\begin{aligned} \mathbf{A}_C &= \mathbf{A}_O - \mathbf{B}_O \mathbf{C}_O \\ \mathbf{B}_C &= \mathbf{B}_O \\ \mathbf{C}_C &= \mathbf{C}_O \\ \mathbf{D}_C &= \mathbf{D}_O. \end{aligned} \quad (9.19)$$

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

Table 9.2: Computed optimal PID parameters.

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

9.4.2 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. (9.5), and given small angle approximation and a constant velocity, it can be written in state space form as:

$$\begin{aligned} \mathbf{A}_K &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ v_c & 0 & 0 \end{bmatrix}, & \mathbf{B}_K &= \begin{bmatrix} \frac{l}{b}v_c & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\ \mathbf{C}_K &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, & \mathbf{D}_K &= [0], \end{aligned} \quad (9.20)$$

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. (9.20) to the closed loop dynamics in Eq. (9.19). 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:

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} \mathbf{A}_K & \mathbf{0}^{(3 \times 5)} & \mathbf{B}_{K1} \\ \mathbf{0}^{(6 \times 3)} & \mathbf{A}_C & \mathbf{0} \end{bmatrix}, & \mathbf{B} &= \begin{bmatrix} \mathbf{B}_{K2} & \mathbf{0}^{(3 \times 1)} \\ \mathbf{0}^{(6 \times 1)} & \mathbf{B}_C \end{bmatrix}, \\ \mathbf{C} &= [\mathbf{C}_K \quad \mathbf{C}_C], & \mathbf{D} &= [\mathbf{0}^{(4 \times 2)}], \end{aligned} \quad (9.21)$$

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. (9.21) 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:

$$\begin{aligned} \Delta\psi &= \psi_r - \psi \\ \Delta x &= \cos(\psi - \Delta\psi)(x_r - x) + \sin(\psi - \Delta\psi)(y_r - y) \\ \Delta y &= -\sin(\psi - \Delta\psi)(x_r - x) + \cos(\psi - \Delta\psi)(y_r - y), \end{aligned} \quad (9.22)$$

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:

$$\begin{aligned}
\min_{\mathbf{u}_k} \quad & \sum_{k=0}^{H_p} \|\mathbf{y}_k - \mathbf{r}_k\|_{\mathbf{Q}}^2 + \sum_{k=0}^{H_u} \|\mathbf{u}_k - \mathbf{u}_{r_k}\|_{\mathbf{R}}^2 + \|\Delta\mathbf{u}_k\|_{\mathbf{S}}^2 \\
\text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \\
& \mathbf{y}_k = \mathbf{C}\mathbf{x}_k, \\
& \mathbf{y}_{\min} \leq \mathbf{y}_k \leq \mathbf{y}_{\max} \quad k = 0, \dots, H_p, \\
& \mathbf{u}_{\min} \leq \mathbf{u}_k \leq \mathbf{u}_{\max}, \\
& \Delta\mathbf{u}_{\min} \leq \Delta\mathbf{u}_k \leq \Delta\mathbf{u}_{\max}
\end{aligned} \tag{9.23}$$

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. (9.21) discretized using zero-order hold and a sampling time of $T_s = 0.1\text{s}$. The design parameters for the MPC and their corresponding values are presented in Table 9.3. To estimate the states of the model, a Kalman filter is utilised and integral terms are used for estimating the output [24].

Table 9.3: 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
$\Delta\mathbf{u}_{\max}$	$\left[0.2, \frac{\pi}{3}\right]^\top$		

9.5 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. (9.21) used for the MPC is updated for each nominal velocity, however, the PID parameters reported in Table 9.2 remains the same for all simulations.

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

$$MSE = \frac{\sum_{k=0}^N (\|\mathbf{x}_k^{\text{bike}} - \mathbf{x}_{k-1}^{\text{ref}}\|^2) T_s}{t} \quad (9.24)$$

where \mathbf{x}^{bike} and \mathbf{x}^{ref} are the x and y coordinates of the bicycle and the reference trajectory respectively, $T_s = 0.1$ 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}^{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:

$$d_H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b) \right\}, \quad (9.25)$$

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.

9.5.1 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. 9.4.



Figure 9.4: 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 $\mathbf{u} = [v, \delta]^T$ and the output $\mathbf{y} = [\psi, x, y, v, \delta, \varphi]^T$. 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. 9.5. 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.

9.5.2 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. 9.6. 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. 9.7 and in Fig. 9.8 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. 9.9 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.

9.5.3 Discussion

The results in Fig. 9.6 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. 9.7. The Hausdorff distances in Fig. 9.8 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. 9.9 is a more challenging track with narrow s-curves and short straights compared to the go-kart track in Fig. 9.6 or the tracks considered in [7, 12]. As a result, both the MSE and the Hausdorff distance in Fig. 9.7 and Fig. 9.8 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. 9.9 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=Wq6SvOX7erA>

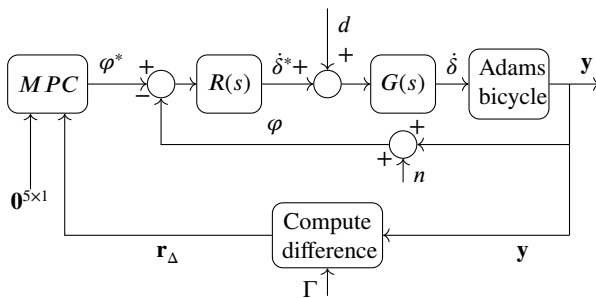


Figure 9.5: Simulation setup of the control system.

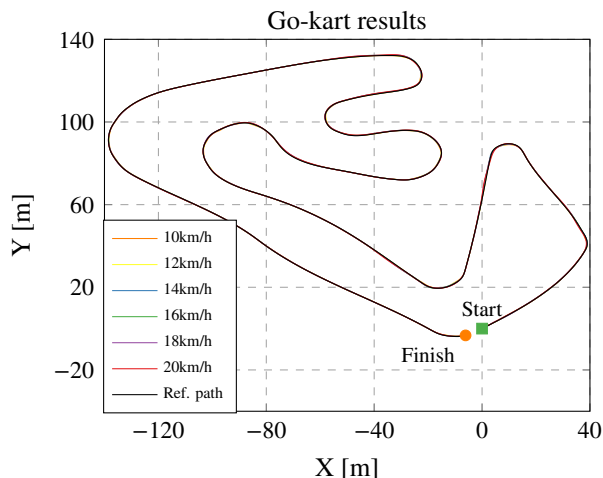


Figure 9.6: The Adams bicycle riding on a go-kart track.

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.

9.6 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-

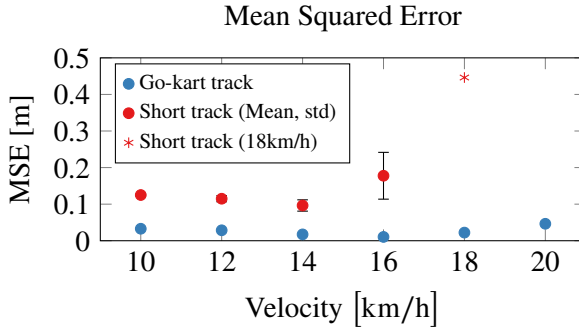


Figure 9.7: The Mean Squared Error between the reference path and the bicycle paths at each nominal velocity. The blue dots correspond to the MSE and computed for the go-kart track. The red dots and error bars correspond to the mean and standard deviation of the MSE for ten runs at each nominal velocity 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.

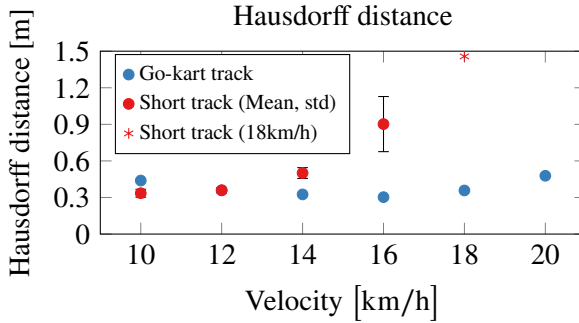


Figure 9.8: The Hausdorff distance at respective velocity for the long go-kart track is represented by the blue dots. Similarly, the red dots represent the mean Hausdorff distance, with error bars indicating the standard deviation, when riding on the short track with narrow curves. At the nominal velocity 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.

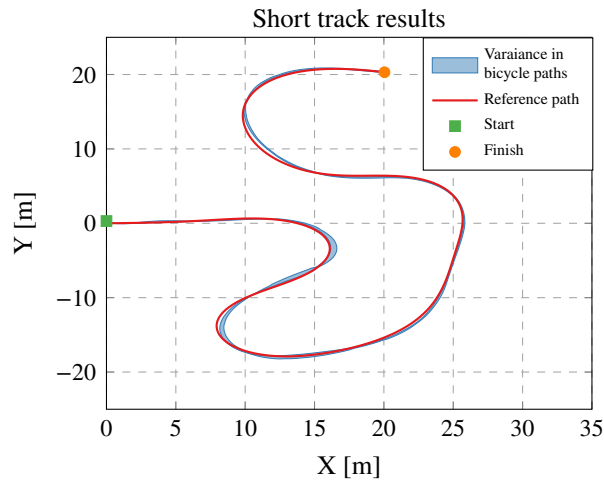


Figure 9.9: Simulations results for the co-simulation on the short track at the nominal velocity of 14km/h. The simulations are repeated ten times and the variance in bicycle path for these ten repetitions are highlighted in blue.

9.7 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] Karl J Åström, Richard E Klein, and Anders Lennartsson. Bicycle dynamics and control: adapted bicycles for education and research. *IEEE Control Systems Magazine*, 25(4):26–47, 2005.
- [2] Arend L Schwab and Jaap P Meijaard. A review on bicycle dynamics and rider control. *Vehicle System Dynamics*, 51(7):1059–1090, 2013.
- [3] Naroa Coretti Sanchez, Luis Alonso Pastor, and Kent Larson. Autonomous bicycles: A new approach to bicycle-sharing systems. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.
- [4] P. Fairley. Self-driving cars have a bicycle problem [news]. *IEEE Spectrum*, 54(3):12–13, 2017.

-
- [5] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.
- [6] Noor Hafizah Amer, Hairi Zamzuri, Khisbullah Hudha, and Zulkifli Abdul 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*, 86(2):225–254, 2017.
- [7] Mauro Baquero-Suárez, John Cortés-Romero, Jaime Arcos-Legarda, and Horacio Coral-Enriquez. A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle. *Multibody System Dynamics*, 45(1):7–35, 2019.
- [8] Trung-Kien Dao and Chih-Keng Chen. Path-tracking control of a riderless bicycle via road preview and speed adaptation. *Asian Journal of Control*, 15(4):1036–1050, 2013.
- [9] Francis Whipple. Stability of the motion of a bicycle. *Quarterly Journal of Pure and Applied Mathematics*, 30:312–348, 1899.
- [10] Neil H Getz. Control of balance for a nonlinear nonholonomic non-minimum phase model of a bicycle. In *Proceedings of 1994 American Control Conference (ACC)*, volume 1, pages 148–151. IEEE, 1994.
- [11] Neil H Getz and Jerrold E Marsden. Control for an autonomous bicycle. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 2, pages 1397–1402, 1995.
- [12] Jingang Yi, Dezhen Song, Anthony Levandowski, and Suhada Jayasuriya. Trajectory tracking and balance stabilization control of autonomous motorcycles. In *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2583–2589. IEEE, 2006.
- [13] Jingang Yi, Yizhai Zhang, and Dezhen Song. Autonomous motorcycles for agile maneuvers, part i: Dynamic modeling. In *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, pages 4613–4618. IEEE, 2009.
- [14] Jodi D.G Kooijman, Jaap P Meijaard, Jim M Papadopoulos, Andy Ruina, and Arend L Schwab. A bicycle can be self-stable without gyroscopic or caster effects. *Science*, 332(6027):339–342, 2011.

- [15] Jiarui He, Mingguo Zhao, and Sotirios Stasinopoulos. Constant-velocity steering control design for unmanned bicycles. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 428–433, 2015.
- [16] Yizhai Zhang, Pengcheng Wang, Jingang Yi, Dezhen Song, and Tao Liu. Stationary balance control of a bikebot. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6706–6711. IEEE, 2014.
- [17] Shahjahan Miah, Efstathios Milonidis, Ioannis Kaparias, and Nicholas Karcianas. An innovative multi-sensor fusion algorithm to enhance positioning accuracy of an instrumented bicycle. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1145–1153, 2020.
- [18] Jaap P Meijaard, Jim M Papadopoulos, Andy Ruina, and Arend 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*, 463(2084):1955–1982, 2007.
- [19] Ruggero Frezza, Alessandro Beghi, and Alessandro 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)*, volume 1, pages 767–772. IEEE, 2004.
- [20] Mingguo Zhao, Sotirios Stasinopoulos, and Yongchao Yu. Obstacle detection and avoidance for autonomous bicycles. *IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1310–1315, 2017.
- [21] Tom Andersson, Niklas Persson, Anas Fattouh, and Martin C Ekström. A loop shaping method for stabilising a riderless bicycle. In *2019 European Conference on Mobile Robots (ECMR)*, pages 1–6, 2019.
- [22] Aidan O’Dwyer. *Handbook of PI and PID Controller Tuning Rules*. Imperial College Press, 3rd edition, 2009.
- [23] Maurice Clerc and James Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.
- [24] Alberto Zenere and Mattia Zorzi. Model predictive control meets robust kalman filtering. *IFAC-PapersOnLine*, 50(1):3774–3779, 2017.

- [25] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.
- [26] Jason K Moore, Jodi D.G Kooijman, Arend L Schwab, and Mont Hubbard. Rider motion identification during normal bicycling by means of principal component analysis. *Multibody System Dynamics*, 25(2):225–244, 2011.

Chapter 10

Paper D

On the Initialization Problem for Timed-Elastic Bands

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

Abstract

Path planning is an important part of navigation for mobile robots. Several approaches have been proposed in the literature based on a discretisation of the map, including A*, Theta*, and RRT*. While these approaches have been widely adopted also in real applications, they tend to generate non-smooth paths, which can be difficult to follow, based on the kinematic and dynamic constraints of the robot. Time-Elastic-Bands (TEB) have also been used in the literature, to deform an original path in real-time to produce a smoother path, and to handle potential local changes in the environment, such as the detection of an unknown obstacle. This work analyses the effects on the overall path for different choices of initial paths fed to TEB. In particular, the produced paths are compared in terms of total distance, curvature, and variation in the desired heading. The optimised version of the solution produced by Theta* shows the highest performance among the considered methods and metrics, and we show that it can be successfully followed by an autonomous bicycle.

10.1 Introduction

Planning a path between two points in a known, partially known or completely unknown environment is called path planning. A map is commonly divided into cells and graph-based search methods, such as Dijkstra's algorithm [1], can be used to find the shortest path between two given cells. Other popular search methods include A^* , Θ^* , D^* Lite and Rapidly-exploring Random Tree (RRT). A^* is an extension of Dijkstra's, by using a heuristic to focus its search towards the goal [2]. Moreover, D^* Lite and Θ^* are extensions of A^* where D^* Lite is intended to be used in an unknown environment [3]. As long as the path is planned from the start position to the goal position, it only has to re-plan parts of the path when obstacles are encountered. Θ^* belongs to any angle path planning algorithms and is not constrained by the edges of the cells which is the case of A^* , D^* Lite, and Dijkstra's [4].

The initial path planned by A^* , Θ^* , D^* Lite and its variants, can be tracked by many different types of robots, such as differential drive robots or omni-wheeled robots, which can rotate around their centre axis without forward or backward motion, thus making sharp turns. The algorithms are also popular in computer games where a low execution time is desirable [5]. However, many other vehicles such as cars and bicycles that are subject to non-holonomic constraints can not follow the paths planned by A^* , Θ^* , or D^* Lite. Instead, a continuous path is required. To address this issue, there are several ways of post-smoothing the path, such as utilising B-splines, Dubin's Curve, or polynomial interpolation [6]. As an alternative, it is also possible to define the problem as an optimisation problem. Time-Elastic-Bands (TEB) [7] is formulated as a nonlinear optimisation problem with constraints on parameters such as the maximum velocity and acceleration, minimum turning radius, and minimum distance to obstacles in the optimisation problem. The TEB has been used for trajectory planning for numerous different robots, such as differential drive robots [8], carlike robots [9], and mobile base platforms [10]. However, how the initial condition of the nonlinear optimisation problem is often neglected or assumed known in beforehand.

In this paper, we investigate four different path-finding algorithms and compare the results by providing them as initial paths for the TEB. A basic A^* path-finding algorithm, an any angle path finder represented by Θ^* , a smooth path finder represented by Hybrid A^* , and finally, RRT^* as a sample-based path-finder are considered. To evaluate the performance of the optimised and non-optimised paths, the length of the path, the integrated absolute value of the heading derivative, and the curvature of the path are taken into account. Furthermore, to demonstrate the feasibility of the approach an autonomous bicycle

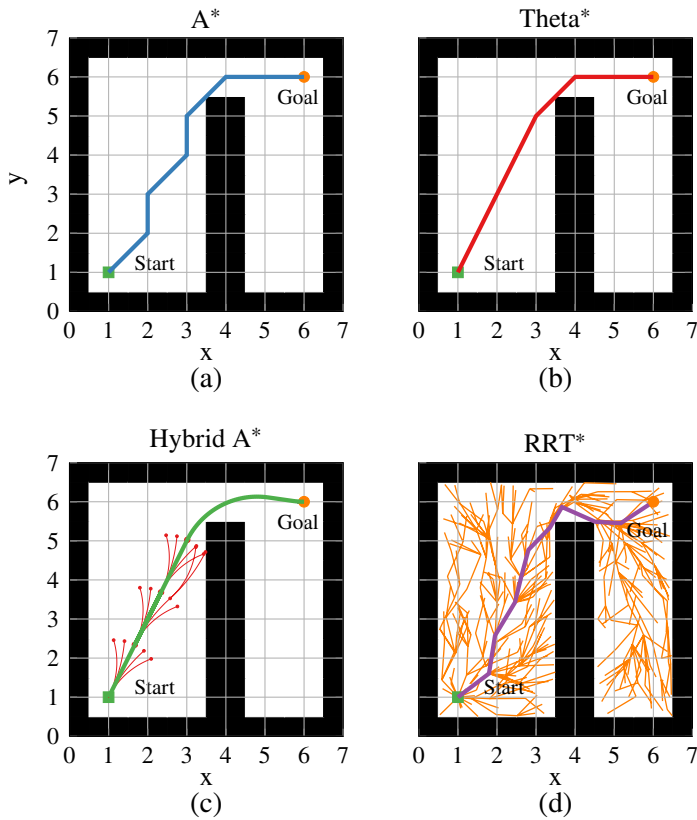


Figure 10.1: The path planned by A*, Theta*, Hybrid A*, and RRT* respectively.

is tracking the paths in a realistic multi-body simulation using a previously designed Model Predictive Controller (MPC) [11].

The paper is structured in the following way, first, background on the different path planners and related work is presented in Section 10.2. The optimisation of the paths using the TEB is described in Section 10.3. Next, the measurements used for the evaluation of the different path planners are presented in Section 10.4 followed by the results in Section 10.5. Concluding remarks and future work are outlined in Section 10.6.

10.2 Background

In this section, the background and details of the path planners considered in this paper are presented. Next, work conducted using TEB and related work in terms of path planning for autonomous bicycles are given.

10.2.1 Path planning

In this paper, we consider a map, $M^{m \times n}$, which is represented by a 2-dimensional binary occupancy grid, and each cell, m_i , is either free $m_i = 0$ or occupied $m_i = 1$. Four different path planners are used to find a path between the start and goal position. A* is a graph search algorithm where a heuristic is used to focus the search towards the goal. It was the first path planning algorithm that combined the cost, $f(n)$, from the start node to the current node, $g(n)$, with the heuristic, $h(n)$, between the current node and the goal:

$$f(n) = g(n) + h(n). \quad (10.1)$$

In this paper, the euclidean distance is chosen as the heuristic. Moreover, A* is a complete path planner, meaning that if there is a path of free cells between the start and goal node, it will be found. In the A* algorithm, each node has eight neighbouring nodes, i.e. its horizontal, vertical and diagonal neighbours. In Figure 10.1(a) it is clear that A* is constrained to movements in the directions of these neighbours.

Theta* works similarly to A* and in fact, they are sharing the same main loop. The difference is how the parent to a node is computed. In the case of A*, the parent will be within the neighbours of its parent node. However, the parent to the current node in Theta* is not constrained to the neighbours of the current node. Instead, Theta* checks if the current node and the parent node lie within Line Of Sight (LOS) of each other [4], i.e the two nodes do not need to be connected. Thus, the resulting path of Theta* is made up of several line segments which have an arbitrary angle and in general result in a path with fewer turns and shorter paths compared to a path planned by for example A* where the heading is constrained [4]. However, it is important to note that Theta* requires longer execution time compared to A*, due to the LOS check, which is an important consideration in some applications [12]. From Figure 10.1(b), it is clear that the path planned by Theta* is shorter, includes fewer heading changes compared to both A* and RRT*, and can have an arbitrary angle between two nodes. As in the case of A*, Theta* considers eight nodes as neighbours and searches the grid, i.e the edges of the cells.

In this paper, we also consider the Hybrid A* algorithm which is a path planner designed for creating smooth paths. The paths planned by Hybrid A* are constrained by the minimum turning radius of the vehicle, the length of the motions, and the number of motion primitives generated [13]. Instead of planning on a grid as in the case of A* and Theta*, Hybrid A* generates N number of smooth motion primitives from the current node and the planner is constrained to these motion primitives. As a consequence, Hybrid A* is not

constrained to only search on the grid or the centre of the cells, which is the case of A^* and Θ^* . Instead, the nodes of Hybrid A^* can be placed anywhere within a free cell. This is illustrated in Figure 10.1(c), where the five motion primitives, in red, generated by Hybrid A^* are not constrained by the edges or the centre of the cells. The resulting path is visualised in green.

There are also sampling-based path planners, such as RRT and RRT*. A tree structure is obtained by repeatedly sampling a new randomly selected node in space and connecting this node with the closest node already in the tree. The advantage of the sampling-based algorithms compared to graph-based search methods is that they can effectively find feasible paths in large state spaces and are not constrained to discrete cells in the map [14]. Another advantage of the RRT* is that even if the original planned path is found unfeasible due to some unknown obstacle, a new path can quickly be planned by using the already generated tree structure [15]. Similarly to Hybrid A^* , RRT* is not constrained to discrete cells either, and the nodes can be anywhere within the free space of the map. Furthermore, as in the case of A^* and Θ^* , RRT* is a complete path planner if a sufficient number of iterations are performed. In fact, RRT* will converge towards the optimal path as the number of nodes approaches infinity as it continues to optimise the path after the goal is reached. This is the main difference between RRT and RRT* [15]. However, an infinite number of nodes is impracticable. Instead, it is up to the designer to determine the maximum number of iterations and the maximum number of nodes. Another important parameter for RRT* is the maximum distance between a new sample and the nodes in the tree as this choice will have a high impact on the convergence time. In Figure 10.1(d), the tree of RRT* after 500 iterations and a maximum connection distance of 1m is illustrated together with the shortest path in purple.

However, both Hybrid- A^* and RRT* share the drawback of often producing jagged paths where an increasing number of heading changes is required, as compared to straight paths. This will lead to increased energy consumption and longer reference paths. Moreover, it might increase the complexity of the path tracker. In the case of an autonomous bicycle, the change of heading can be slow, especially if the bicycle is only equipped with a propulsion motor and a steering motor, as the steering regulation would also be in charge of balancing the bicycle [16]. Furthermore, the resulting path from Θ^* , A^* , and RRT* have sharp turns which would require the vehicle tracking the path to turn around its own axis. This is a manoeuvre that is not possible for vehicles which adhere to non-holonomic constraints, such as an autonomous bicycle, instead a smooth path is desirable.

10.2.2 Related work

One approach for smoothing the path is to use TEB [7] which are based on the Elastic Bands proposed by Quinlan and Khatib [17]. An advantage of the TEB is that constraints on the kinodynamic properties of the vehicle can easily be included in the nonlinear optimisation problem while keeping a safe distance from obstacles and minimising travel time. In the work of Deray et al. [10] the Timed-Elastic Smooth Curve (TESC), an extension of TEB that relies on Lie groups, is proposed and compared to the TEB. Both TESC and TEB are given the task to plan between the start position and randomly selected goal position, i.e no initial path is planned. Both planners fail repeatedly in environments with static obstacles, something that could have been avoided if an initial path was planned with a complete path planner such as A* or Theta*. In the work of Yongzhe et al. a car-like robot used TEB to park the robot in a parking lot [9]. The initial path was planned by A* and the strategy was evaluated in both simulations and experiments with promising results. A* was also used in the work of Ma et al. where the number of heading changes in the planned path was reduced by minimising the snap of the trajectory [18]. Next, TEB was used as a local path planner to find a local optimal path.

Planning a feasible path for an autonomous bicycle requires the path to be smooth due to the non-holonomic constraints of the bicycle. This has been solved using different methods such as in the work of Wissel and Nikoukhah [19], where a path is planned based on the manoeuvres which can be performed by an autonomous bicycle, similar to Hybrid-A*. The manoeuvres are optimised to minimise the time of travel of the bicycle. The trajectory for a bicycle is also considered in the work of Yuan et al. [20], where the trajectory is optimised by means of Particle Swarm Optimisation (PSO). A curve in the XY plane is parameterized by two third-order polynomials while satisfying initial and final constraints on the yaw angle and the x,y position. This leaves two free parameters, one for each polynomial which can be used by PSO to minimise the maximum lean angle of the bicycle. By using a higher order of polynomials, initial and final constraints of more parameters, such as the steering angle, could be included. However, the initial path is assumed known is neglected, moreover, it is not clear how the proposed method would handle obstacles between the two points. In the work of Turnwald et al. motion planning of a bicycle is investigated [21]. Three different models are used to construct three different nonlinear optimisation problems. Two of the models assume a zero trail, i.e the distance between the contact point of the front wheel with the ground and the intersection of the steering axis with the ground. The third model includes a positive trail.

Simulation and experimental results are compared and it is concluded that the nonlinear model which includes trail, produces paths best suitable for an autonomous bicycle.

10.3 Path optimisation

Since we are interested in smooth paths which can be tracked by an autonomous bicycle, the planned paths are smoothed by TEB [7]. The TEB can be visualised as putting an elastic band on top of the previously planned path, then tightening the band between the start and goal position to remove any slack and create a smooth path, while keeping a safe distance to obstacles, \mathbf{o}_{min} , and adhere to a minimum turning radius, r_{min} . Moreover, the algorithm minimises the time to travel from the start pose, \mathbf{x}_s , to the goal pose, \mathbf{x}_g while considering constraints on the velocity, acceleration, angular velocity and angular acceleration. Thus, it is a multi-objective optimisation problem where the states of the vehicle, $\mathcal{X} = \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ and time of travel between states, $T = \sum_{k=1}^{n-1} \Delta T_1, \Delta T_2, \dots, \Delta T_{n-1}$, are the objectives and collected as:

$$\mathcal{M} := \{\mathbf{x}_1, \Delta T_1, \mathbf{x}_2, \Delta T_2, \dots, \mathbf{x}_{n-1}, \Delta T_{n-1}, \mathbf{x}_n\}. \quad (10.2)$$

Following the approach in the work of [22], the optimisation problem can be formulated as:

$$\begin{aligned} \min_{\mathcal{M}} \quad & \sum_{k=1}^{n-1} \Delta T_k^2 \\ \text{s.t.} \quad & \mathbf{x}_1 = \mathbf{x}_s, \\ & \mathbf{x}_n = \mathbf{x}_g, \\ & \mathbf{h}_k(\mathbf{x}_{k+1}, \mathbf{x}_k) = \mathbf{0}, \\ & r_k - r_{min} \geq 0, \\ & \mathbf{o}_k(\mathbf{x}_k) - \mathbf{o}_{min} \geq \mathbf{0}, \\ & |v| \leq v_{max}, |a| \leq a_{max}, \\ & |\omega| \leq \omega_{max}, |\alpha| \leq \alpha_{max} \end{aligned} \quad (10.3)$$

where $\mathbf{h}_k(\mathbf{x}_{k+1}, \mathbf{x}_k) = \mathbf{0}$ iff two consecutive poses $\mathbf{x}_k, \mathbf{x}_{k+1}$ are located on a common arc of constant curvature. Thus, this constraint affects the smoothness of the resulting path. $\mathbf{o}_k(\mathbf{x}_k)$ is the distance to a set of obstacles in the proximity of \mathbf{x}_k . Moreover, v_{max} , a_{max} , ω_{max} , and α_{max} define the maximum velocity, acceleration, angular velocity and angular acceleration respectively.

Table 10.1: Constraints and weights for TEB

Constraint	Value	w	Constraint	Value	w
v_{max}	5m/s	1	r_{min}	3m	10
a_{max}	2m/s ²	1	ΔT	0.1s	-
ω_{max}	0.3rad/s	1	$\sum_{k=1}^{n-1} \Delta T_k^2$	-	20
α_{max}	0.5rad/s ²	1	\mathbf{h}	-	1000
\mathbf{o}_{min}	1m	3			

The nonlinear program in equation (10.3) is solved by means of Levenberg-Marquard solver by approximating the problem as a nonlinear least square problem where the constraints are used as penalty terms in the objective. Moreover, each penalty term is weighted with a weight to express the importance of each constraint. The constraints and the corresponding weights are presented in Table 10.1.

As the problem is a nonlinear program, there is no guarantee for converging to the optimal solution. The solution is heavily dependent on the initial conditions, which in this case are the initial path and the initial velocities and accelerations. The initial path is the path planned by the A*, Theta*, Hybrid A*, and RRT* respectively. Moreover, the initial velocity, acceleration, angular velocity, and angular acceleration are all set to 0.

10.4 Evaluation

The paths planned by Theta*, A*, Hybrid-A*, RRT* and their optimised versions are compared in 300 randomised maps. The size of each map is 100×100 m with a resolution of 1 cell per meter. For each map, a maze is randomised with a wall thickness of 3m and a passage width of 8m. Three different scenarios are used, in the first scenario, the passages in the maze are made up of free space. In the second scenario, the free space is cluttered with 50 randomly positioned obstacles and in the third scenario 100 randomly positioned obstacles are used. The start and goal positions are placed randomly on the map, with a minimum distance of 70m apart. Moreover, to realise a safety distance to the obstacles and the walls of the mazes, the obstacles and walls are inflated by a radius of 1m. To evaluate the performance of the different path planners the euclidean distance of the paths, the curvature of the paths and the integrated absolute value of the heading derivative (IAT) are considered. A shorter path length is desirable as it can save both time and energy for the vehicle tracking

the path. The number of heading changes metric is also related to energy efficiency as a vehicle consumes more energy when it has to change its heading a lot. It is also related to the comfort of the ride, as constantly changing the steering direction will make an uncomfortable ride. However, the metric is better suited to be used in noncontinuous paths where sharp heading changes are applied such as those produced by A^* or Θ^* . In the case of continuous paths, there may be small variations in the heading even on paths that appear straight. Moreover, small and large heading changes would count the same which makes the metric favouring large changes which are rarely found on continuous paths. Instead, the IAT is considered and is defined as:

$$IAT = \sum_k^{n-1} \left| \frac{\theta_{k+1} - \theta_k}{T_s} \right|, \quad (10.4)$$

where T_s is the sampling time θ_k is the heading in sample k . This metric combines the magnitude of the heading changes with the frequency of heading changes. Before IAT is computed each path is interpolated over 1000 samples. The resulting value is normalised with respect to Optimised Θ^* for each map and the mean and standard deviation of 100 iterations for mazes with 0, 50, and 100 obstacles are computed. The curvature of the paths is only computed for the interpolated paths planned by the optimised versions of the path planners and the Hybrid A^* , as the paths planned by Θ^* , A^* , and RRT^* are made up of line segments and thus are not smooth. As in the case of the path distance and the IAT , the curvature is normalised with respect to the Optimised Θ^* for each map and the mean and the standard deviation are computed.

10.5 Results

In this section, the results from the different path planning algorithms are presented. In the comparison, the maximum distance between a new node and the tree in RRT^* is set to 10m, and 10^5 iterations are performed with a maximum of 3×10^4 nodes in the tree. The Hybrid A^* uses a minimum turning radius of 3m, a motion primitive length of 1.5m and 15 motion primitives are sampled at each node. Moreover, only forward motion is considered for all path planners. The resulting paths planned by A^* , Θ^* , Hybrid A^* , and RRT^* are optimised using the TEB as described in Section 10.3. Moreover, the path tracking results are presented where an autonomous bicycle is tracking a path planned by Optimised Θ^* . The code for the comparisons and simulation, together with a video of the simulation, are available online ¹. The section is

¹<https://github.com/NiklasPerssonMDU/On-the-Initial-of-Timed-Elastic-Bands.git>

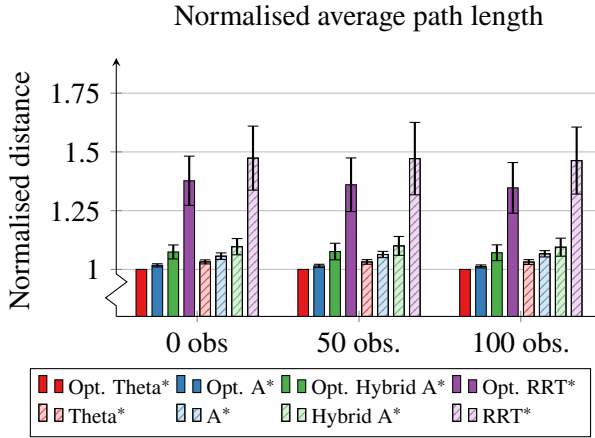


Figure 10.2: Average and standard deviation of the normalised path lengths for 100 iterations with zero obstacles, 100 iterations with 50 randomised obstacles, and 100 iterations with 100 randomised obstacles in a randomised maze. The path lengths are normalised with respect to Optimised Theta*.

wrapped up with a discussion of the results.

10.5.1 Path planning results

In Figure 10.2, the mean and standard deviation of the normalised path lengths are presented. The mean and standard deviation of the curvature is presented in Figure 10.3. Moreover, the mean and standard deviation for the *IAT* value for all paths are given in Figure 10.4.

10.5.2 Simulation results

As the Optimised Theta* produces the most promising results when compared to the other path planners, it is used to plan a path for an autonomous bicycle in a realistic multi-body dynamics simulation using Simscape. A randomised maze of size 50 × 50m with a resolution of 1 cell per meter with no further obstacles is considered. A minimum turning radius of 3m is used and a safety distance of 1m is considered. The 2-dimensional map is generated as a 3-dimensional environment in Simscape and a SolidWorks model of an ordinary-sized bicycle is imported and controlled through Simulink. Based on previous work [11], an MPC is used to track the reference trajectory and a PID controller is used for balancing the bicycle. The inner loop in charge of balancing the bicycle by steering the bicycle into the fall is executing at 100Hz and the outer trajectory tracking loop is running at 10Hz, while the bicycle model is simulated

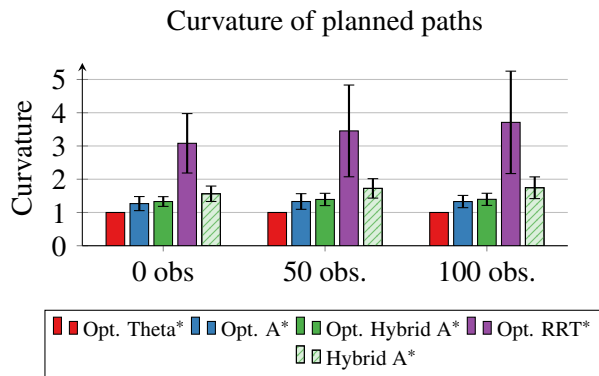


Figure 10.3: The mean and the standard deviation for the curvature of the planned paths, normalised with respect to Optimised Theta* in every iteration.

in continuous time. The control strategy is illustrated in Figure 10.5. To ensure a uniform sampling time of the optimised trajectory it is re-sampled with a sampling time of $T_s = 0.1s$ before the simulation starts. The planned path and path-tracking performance of the autonomous bicycle are presented in Figure 10.6.

10.5.3 Discussion

From Figure 10.2, 10.3, 10.4 it is clear that the paths optimised using the TEB are shorter, have less curvature, and do not require as much heading regulation as compared to their non-optimised counterparts in general. However, RRT* is actually performing worse when optimised using TEB in terms of IAT for mazes which are cluttered with obstacles. Due to the cluttered environments RRT* tends to plan paths which have lots of nodes on a short distance which makes it difficult for the TEB to respect some constraints such as the minimum turning radius. An increased number of iterations and nodes allowed in the solution could improve the results of RRT* but at the cost of the execution time which is already high compared to the other path planners. Moreover, tuning of the maximum distance could have a positive effect on the results.

The paths length are decreased with 4.4%, 3.1%, 2.1% and 7.3% for A*, Theta*, Hybrid A*, and RRT* respectively when computing the average of the three different obstacle scenarios. Furthermore, the Optimised Theta* produces the shortest paths which were expected as Theta*, in general, produces short paths, by smoothing the path using an elastic band the slack at the curves can be minimised. However, the TEB perform worse on paths which are already smooth, as in the case of Hybrid A*. This can be explained by the ratio of the

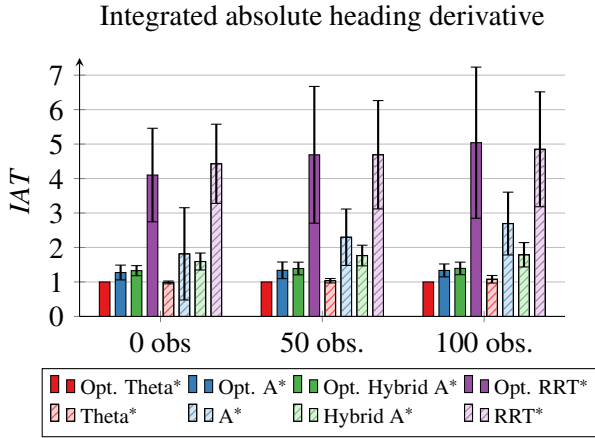


Figure 10.4: Integrated absolute value of the heading derivative normalised with respect to Optimised Theta*.

weights and that the TEB favours smoothing the path and gets stuck in a locally optimal solution. The results highlights the importance of the initial conditions given to the NLP in equation (10.3). For the initial condition of the TEB, a discrete path with sharp turns, but with a low number of heading changes and short path length is performing better compared to an already smooth path with a longer path length and an increasing number of heading changes such as in the case of Hybrid A*. The results also suggests that TEB performs better on paths planned by grid search algorithms compared to sampled based algorithms and hybrid search algorithms. Moreover, Figure 10.6 shows that the path planned by Optimised Theta* successfully can be tracked by an autonomous bicycle in an environment with static obstacles.

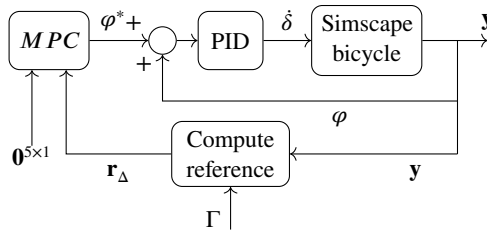


Figure 10.5: Simulation setup of the control system.

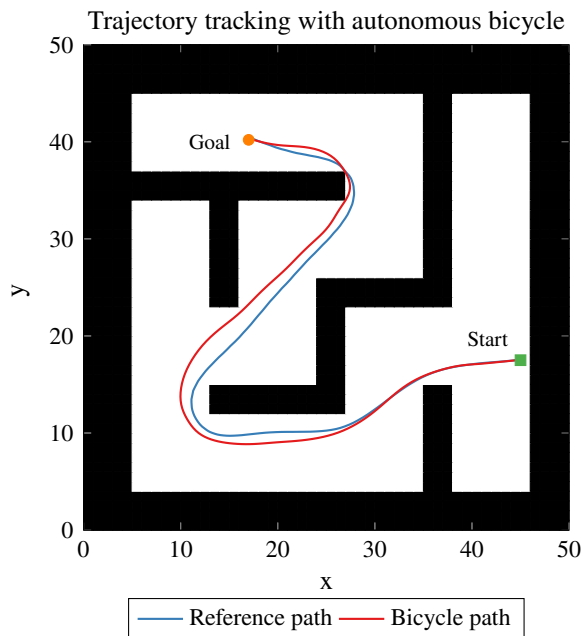


Figure 10.6: Autonomous bicycle tracking the reference path planned by Optimised Theta*.

10.6 Conclusion

In this paper, four different path planners are compared in 300 different maps. The resulting paths are optimised using Timed-Elastic-Bands which creates smooth paths that adhere to a number of different constraints, including maximum velocity, acceleration, and minimum turning radius. The results highlight the importance of the initial path fed to the TEB. Moreover, the results show that the resulting paths from the optimised Theta* have the shortest path length, the lowest curvature, and the lowest *IAT*. The optimisation does not only smooth the paths but in general improves the path planned by all algorithms in terms of all path lengths, heading changes, and curvature. This emphasizes the importance of optimisation when it comes to path planning for non-holonomic constrained vehicles. Furthermore, to demonstrate that it is possible to track the resulting path of the Optimised Theta*, an autonomous bicycle is used in a multi-body dynamic simulation. An MPC is utilised as a path tracker and a PID is used to keep the bicycle balanced by steering the bicycle into the fall. In the future, a TEB formulation with a bicycle model could be investigated to constrain the maximum lean angle of the bicycle and include dynamic and

unknown obstacles in the path planning.

10.7 Acknowledgements

The work is partly funded by Eskilstuna kommun och Eskilstuna Fabriksförening.

Bibliography

- [1] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
- [2] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4:100–107, 1968.
- [3] Sven Koenig and Maxim Likhachev. D* lite. In *Eighteenth National Conference on Artificial Intelligence*, pages 476–483. American Association for Artificial Intelligence, 2002.
- [4] Alex Nash, Kenny Daniel, Sven Koenig, and Ariel Felner. Theta*: Any-angle path planning on grids. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, volume 2, pages 1177–1183, 2007.
- [5] Peter Yap, Neil Burch, Robert C Holte, and Jonathan Schaeffer. Any-angle path planning for computer games. In *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, page 201–207, 2011.
- [6] Abhijeet Ravankar, Ankit A Ravankar, Yukinori Kobayashi, Yohei Hoshino, and Chao-Chung Peng. Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges. *Sensors*, 18(9):3170, 2018.
- [7] Christoph Rösmann, Wendelin Feiten, Thomas Wösch, Frank Hoffmann, and Torsten Bertram. Trajectory modification considering dynamic constraints of autonomous robots. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, 2012.
- [8] Christoph Rösmann, Wendelin Feiten, Thomas Wösch, Frank Hoffmann, and Torsten Bertram. Efficient trajectory optimization using a sparse

- model. In *2013 European Conference on Mobile Robots*, pages 138–143, 2013.
- [9] Zhang Yongzhe, Benjamin Ma, and Chan Kit Wai. A practical study of time-elastic-band planning method for driverless vehicle for auto-parking. In *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*, pages 196–200, 2018.
- [10] Jérémie Deray, Bence Magyar, Joan Solà, and Juan Andrade-Cetto. Timed-elastic smooth curve optimization for mobile-base motion planning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3143–3149, 2019.
- [11] Niklas Persson, Martin C. Ekström, Mikael Ekström, and Alessandro V. Papadopoulos. Trajectory tracking and stabilisation of a riderless bicycle. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1859–1866, 2021.
- [12] Tansel Uras and Sven Koenig. An empirical comparison of any-angle path-planning algorithms. In *International Symposium on Combinatorial Search*, volume 6, 2015.
- [13] Janko Petereit, Thomas Emter, Christian W. Frey, Thomas Kopfstedt, and Andreas Beutel. Application of Hybrid A* to an autonomous mobile robot for path planning in unstructured outdoor environments. *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, 2012.
- [14] Steven M LaValle. *Planning Algorithms*. Cambridge university press, 2006.
- [15] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.
- [16] Mingguo Zhao, Sotirios Stasinopoulos, and Yongchao Yu. Obstacle detection and avoidance for autonomous bicycles. *IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1310–1315, 2017.
- [17] Sean Quinlan and Oussama Khatib. Elastic bands: connecting path planning and control. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 802–807, 1993.

- [18] Zhengwei Ma, Hailun Qiu, Huaizhi Wang, Longhao Yang, Lihong Huang, and Rengao Qiu. A* algorithm path planning and minimum snap trajectory generation for mobile robot. In *2021 4th International Conference on Robotics, Control and Automation Engineering (RCAE)*, pages 284–288, 2021.
- [19] Dirk von Wissel and Ramine Nikoukhah. Maneuver-based obstacle-avoiding trajectory optimization: Example of a riderless bicycle. *Journal of Structural Mechanics*, 23(2):223–255, 1995.
- [20] Jing Yuan, Huan Chen, Fengchi Sun, and Yalou Huang. Trajectory planning and tracking control for autonomous bicycle robot. *Nonlinear Dynamics*, 78(1):421–431, 2014.
- [21] Alen Turnwald and Steven Liu. Motion planning and experimental validation for an autonomous bicycle. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 3287–3292. IEEE, 2019.
- [22] Christoph Rösmann, Frank Hoffmann, and Torsten Bertram. Kinodynamic trajectory optimization and control for car-like robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5681–5686, 2017.