Trajectory tracking and stabilisation of a riderless bicycle*

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

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.**

I. 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 car[1].

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-simulation between Adams[2] 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 II the related work is presented. The bicycle model is derived in Section III followed by Section IV where the control design is discussed. Section V presents the results and the paper is concluded in Section VI which also includes future research.

II. 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.

A. 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

The authors are affiliated with the academy of Innovation, Design and Technique, Mälardalens University, Västerås, Sweden e-mail: Firstname.Surname@mdh.se.

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

[2]https://www.mscsoftware.com/it/product/adams

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].

B. 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.

III. 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 $\mathcal{R}(x_\mathcal{R}, y_\mathcal{R}, z_\mathcal{R})$, as shown in Fig. 1. The lean angle, $\varphi(t)$, and steering angle, $\delta(t)$, are positive following the right-hand rule.

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

$$\tan(\beta(t)) = \frac{\tan(\delta(t))\sin(\lambda)}{\cos(\varphi(t))}, \qquad (1)$$

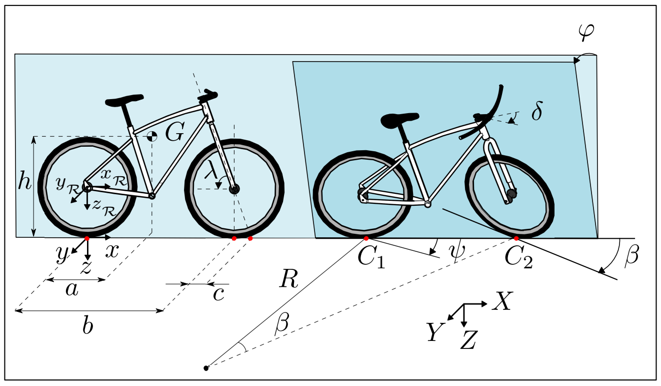

Fig. 1. A bicycle riding on a flat horizontal plane.

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

$$\sigma(t) = \frac{b}{R(t)} = \tan(\beta(t)) \tag{2}$$

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

$$\sigma(t) \approx \delta(t)\sin(\lambda), \tag{3}$$

and the respective curvature rate of change:

$$\omega_\sigma(t) = \dot{\sigma}(t) \approx \dot{\delta}(t)\sin(\lambda). \tag{4}$$

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

$$\begin{aligned} \dot{x} &= v(t)\cos(\psi(t)) \\ \dot{y} &= v(t)\sin(\psi(t)) \\ \dot{\psi}(t) &= \frac{\sigma(t)v(t)}{b} = \frac{\tan(\delta(t))\sin(\lambda)}{b\cos(\varphi(t))}v(t). \end{aligned} \tag{5}$$

The roll dynamics of the bicycle can be expressed as:

$$\begin{aligned} h^2\ddot{\varphi}(t)m = gm\Big(h\sin\big(\varphi(t)\big) + \frac{ca}{b}\sin(\lambda)\sigma(t)\cos\big(\varphi(t)\big)\Big) - \\ \Big(1 - \frac{h}{b}\sigma(t)\sin\big(\varphi(t)\big)\Big)\frac{h}{b}\sigma(t)\cos\big(\varphi(t)\big)v^2(t)m \\ - \frac{ahm}{b}\cos\big(\varphi(t)\big)\Big(\sigma(t)\dot{v}(t) - v(t)\omega_\sigma(t)\Big). \end{aligned} \tag{6}$$

Here, the mass, m, cancels out and using the relationships in Eq. (1) and Eq. (2) the roll dynamics can be simplified as:

$$\begin{aligned} h^2\ddot{\varphi}(t) = g\left(h\sin\big(\varphi(t)\big) + \frac{cap^2}{b}\tan\big(\delta(t)\big)\right) - \\ \left(1 - \frac{hp}{b}\tan\big(\delta(t)\big)\tan\big(\varphi(t)\big)\right)\frac{hp}{b}\tan\big(\delta(t)\big)v(t)^2 \\ - \frac{ahp}{b}\tan\big(\delta(t)\big)\dot{v}(t) - \frac{ah}{b}\cos\big(\varphi(t)\big)v(t)\omega_\sigma(t). \end{aligned} \tag{7}$$

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

$$\begin{aligned} h^2\ddot{\varphi}(t) = g\left(h\varphi(t) + \frac{cap^2}{b}\delta(t)\right) - \\ \left(1 - \frac{hp}{b}\delta(t)\varphi(t)\right)\frac{hp}{b}\delta(t)v^2 - \frac{ahp}{b}v\dot{\delta}(t) \end{aligned} \tag{8}$$

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

$$\ddot{\varphi}(t) = \frac{g}{h}\varphi(t) + \frac{gcap^2}{bh^2}\delta(t) - \frac{p}{bh}v^2\delta(t) - \frac{ap}{bh}v\dot{\delta}(t). \tag{9}$$

Eq. (9) can be rewritten in state-space form, with the input $\mathbf{u}_\varphi = \dot{\delta}$, output $\mathbf{y}_\varphi = \varphi$, and the state $\mathbf{x}_\varphi = [\dot{\varphi}, \varphi, \delta]^\top$ as:

$$\begin{aligned} \mathbf{A}_\varphi &= \begin{bmatrix} 0 & \frac{g}{h} & \frac{p}{bh}\left(\frac{gcap}{h} - v^2\right) \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{B}_\varphi &= \begin{bmatrix} -\frac{ap}{bh}v \\ 0 \\ 1 \end{bmatrix} \\ \mathbf{C}_\varphi &= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}, & \mathbf{D}_\varphi &= \begin{bmatrix} 0 \end{bmatrix}. \end{aligned} \tag{10}$$

TABLE I

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

The relevant model parameters are obtained through measurements on the instrumented bicycle in [21] and presented in Table I. 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 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. 2 and the matched response is represented by a state-space model as:

$$\begin{aligned} \mathbf{A}_{\dot{\delta}} &= \begin{bmatrix} -100 \end{bmatrix}, & \mathbf{B}_{\dot{\delta}} &= \begin{bmatrix} 100 \end{bmatrix}, \\ \mathbf{C}_{\dot{\delta}} &= \begin{bmatrix} 1 \end{bmatrix}, & \mathbf{D}_{\dot{\delta}} &= \begin{bmatrix} 0 \end{bmatrix}, \end{aligned} \tag{11}$$

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:

$$\begin{aligned} \mathbf{A}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{A}_{\dot{\delta}} & \mathbf{0} \\ \mathbf{B}_\varphi\mathbf{C}_{\dot{\delta}} & \mathbf{A}_\varphi \end{bmatrix}, & \mathbf{B}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{B}_{\dot{\delta}} \\ \mathbf{B}_\varphi\mathbf{D}_{\dot{\delta}} \end{bmatrix}, \\ \mathbf{C}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{D}_\varphi\mathbf{C}_{\dot{\delta}} & \mathbf{C}_\varphi \end{bmatrix}, & \mathbf{D}_{\dot{\delta}\varphi} &= \begin{bmatrix} \mathbf{D}_\varphi\mathbf{D}_{\dot{\delta}} \end{bmatrix}, \end{aligned} \tag{12}$$

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

IV. 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.

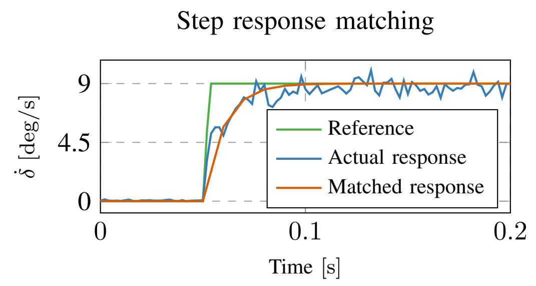

Fig. 2. Step response matching of the steering system. The matched response is used in simulation to model the dynamics of the steering system.

A. 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. 3 where $R(s)$ is a real PID controller in parallel form

$$R(s) = K_p + \frac{K_i}{s} + \frac{K_d N}{1 + \frac{N}{s}}. \quad (13)$$

In the figure, $G(s)$ is the transfer function of the steering dynamics in Eq. (11), and $H(s)$ is the transfer function associated with Eq. (9), with input $\dot{\delta}$ and output φ, and it is equal to:

$$H(s) = \frac{\Phi(s)}{\dot{\Delta}(s)} = \frac{(gcap^2 - hpv^2) - ahpvs}{bh^2 s^3 - bhgs}. \quad (14)$$

The roll dynamics, including the actuation, can be therefore expressed as the transfer function $P(s) = G(s)H(s)$. As illustrated in Fig. 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 $v = 14$km/h, is formulated as a nonlinear optimisation problem:

$$\begin{aligned} \min_{K_p, K_i, K_d, N} \quad & (\omega_c^{\text{des}} - \omega_c)^2 + w_1 \int_0^\infty tq(t)^2 \mathrm{d}t + \\ & + w_2 \int_0^\infty t\ell(t)^2 \mathrm{d}t \\ \text{s.t.} \quad & K_p, K_i, K_d \in [-200, 0], \\ & N \in [10, 1000], \\ & \omega_c^{\text{des}} = 60\text{rad/s}. \end{aligned} \quad (15)$$

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. (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. (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 a 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:

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{(\varphi^2 - 4\varphi)}|}. \quad (16)$$

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 II.

The state-space matrices of the PID controller in (13) are defined as:

$$\begin{aligned} \mathbf{A}_R &= \begin{bmatrix} -N & 0 \\ 1 & 0 \end{bmatrix}, & \mathbf{B}_R &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \\ \mathbf{C}_R &= \begin{bmatrix} K_i - K_d N^2 & K_i N \end{bmatrix}, & \mathbf{D}_R &= \begin{bmatrix} K_p + K_d N \end{bmatrix}, \end{aligned} \quad (17)$$

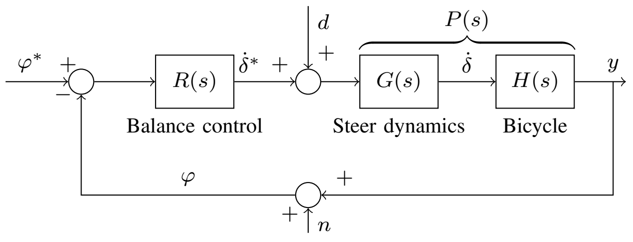

Fig. 3. Balance controller $R(s)$ and the linear model $P(s)$.

TABLE II
COMPUTED OPTIMAL PID PARAMETERS.

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

where the state vector is $\mathbf{x}_R = [e_1, e_2]^\top$, the output $\mathbf{y} = \dot{\delta}^*$, and the input $\mathbf{u} = e$ with $e = \varphi^* - \varphi$. Now, the open loop system in Fig 3 can be written as:

$$\begin{aligned} \mathbf{A}_O &= \begin{bmatrix} \mathbf{A}_R & \mathbf{0}^{(2\times4)} \\ \mathbf{B}_{\dot{\delta}\varphi}\mathbf{C}_R & \mathbf{A}_{\dot{\delta}\varphi} \end{bmatrix} & \mathbf{B}_O &= \begin{bmatrix} \mathbf{B}_R \\ \mathbf{B}_{\dot{\delta}\varphi}\mathbf{D}_R \end{bmatrix} \\ \mathbf{C}_O &= \begin{bmatrix} \mathbf{D}_{\dot{\delta}\varphi}\mathbf{C}_R & \mathbf{C}_{\dot{\delta}\varphi} \end{bmatrix} & \mathbf{D}_O &= \begin{bmatrix} \mathbf{D}_{\dot{\delta}\varphi}\mathbf{D}_R \end{bmatrix} \end{aligned} \tag{18}$$

and by closing the loop we obtain

$$\begin{aligned} \mathbf{A}_C &= \mathbf{A}_O - \mathbf{B}_O\mathbf{C}_O \\ \mathbf{B}_C &= \mathbf{B}_O \\ \mathbf{C}_C &= \mathbf{C}_O \\ \mathbf{D}_C &= \mathbf{D}_O. \end{aligned} \tag{19}$$

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

B. 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. (5), and given small angle approximation and a constant velocity, it can be written in state space form as:

$$\begin{aligned} \mathbf{A}_K &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ v_c & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_K = \begin{bmatrix} \frac{p}{b}v_c & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\ \mathbf{C}_K &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{D}_K = \begin{bmatrix} 0 \end{bmatrix}, \end{aligned} \tag{20}$$

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. (20) to the closed loop dynamics in Eq. (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:

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} \mathbf{A}_K & \mathbf{0}^{(3\times5)} & \mathbf{B}_{K1} \\ \mathbf{0}^{(6\times3)} & \mathbf{A}_C & \end{bmatrix}, & \mathbf{B} &= \begin{bmatrix} \mathbf{B}_{K2} & \mathbf{0}^{(3\times1)} \\ \mathbf{0}^{(6\times1)} & \mathbf{B}_C \end{bmatrix}, \\ \mathbf{C} &= \begin{bmatrix} \mathbf{C}_K & \mathbf{C}_C \end{bmatrix}, & \mathbf{D} &= \begin{bmatrix} \mathbf{0}^{(4\times2)} \end{bmatrix}, \end{aligned} \tag{21}$$

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 $\mathbf{x} = [\psi, x, y, e_1, e_2, \dot{\delta}, \dot{\varphi}, \varphi, \delta]^\top$, the input $\mathbf{u} = [v, \varphi^*]$, and the output $\mathbf{y} = [\psi, x, y, \varphi, \delta]^\top$.

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. (21) is linearsed 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:

$$\begin{aligned} \Delta\psi &= \psi_r - \psi \\ \Delta x &= \cos(\psi - \Delta\psi)(x_r - x) + \sin(\psi - \Delta\psi)(y_r - y) \\ \Delta y &= -\sin(\psi - \Delta\psi)(x_r - x) + \cos(\psi - \Delta\psi)(y_r - y), \end{aligned} \tag{22}$$

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:

$$\begin{aligned} \min_{\mathbf{u}_k} \quad & \sum_{k=0}^{H_p} \|\mathbf{y}_k - \mathbf{r}_k\|^2_{\mathbf{Q}} + \sum_{k=0}^{H_u} \|\mathbf{u}_k - \mathbf{u}_{r_k}\|^2_{\mathbf{R}} + \|\Delta\mathbf{u}_k\|^2_{\mathbf{S}} \\ \text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \\ & \mathbf{y}_k = \mathbf{C}\mathbf{x}_k, \\ & \mathbf{y}_{\min} \le \mathbf{y}_k \le \mathbf{y}_{\max} \quad k = 0, \dots, H_p, \\ & \mathbf{u}_{\min} \le \mathbf{u}_k \le \mathbf{u}_{\max}, \\ & \Delta\mathbf{u}_{\min} \le \Delta\mathbf{u}_k \le \Delta\mathbf{u}_{\max} \end{aligned} \tag{23}$$

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

TABLE III
MPC PARAMETERS AND CONSTRAINTS.

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

V. 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[3] 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. (21) used for the MPC is updated for each nominal velocity, however, the PID parameters reported in Table II remains the same for all simulations.

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

$$MSE = \frac{\sum_{k=0}^{N}(\|\mathbf{x}_k^{\text{bike}} - \mathbf{x}_{k-1}^{\text{ref}}\|^2)T_s}{t} \tag{24}$$

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

$$d_H(A,B) = \max\left\{\sup_{a\in A}\inf_{b\in B} d(a,b), \sup_{b\in B}\inf_{a\in A} d(a,b)\right\}, \tag{25}$$

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.

A. Simulation setup

The ordinary-sized male bicycle in [21] was dismantled and each component was weighed, measured, and designed in SolidWorks[4] 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 wheels and the ground is modelled as a Coulomb friction force with the dynamic

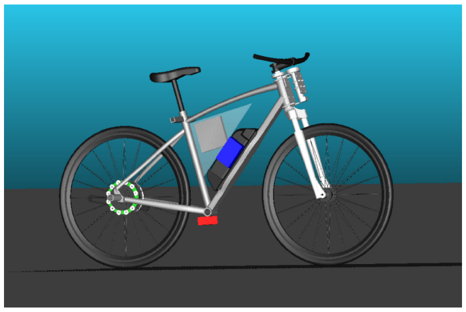
Fig. 4. The instrumented bicycle designed in SolidWorks and imported to Adams is used as the plant in the co-simulation between Adams and Matlab.

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. 4.

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

B. 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. 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. 7 and in Fig. 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 reaching the finish line in all ten repetitions. In Fig. 9

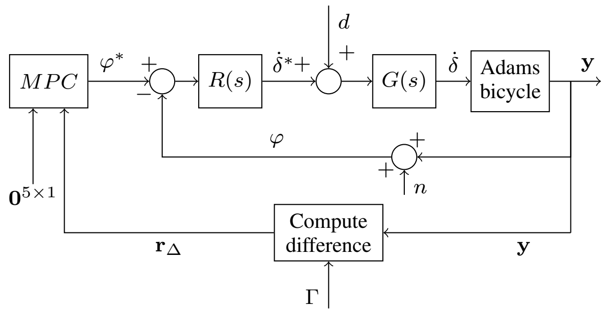

Fig. 5. Simulation setup of the control system.

[3]https://www.openstreetmap.org/#map=18/59.47979/17.82856
[4]https://www.solidworks.com/

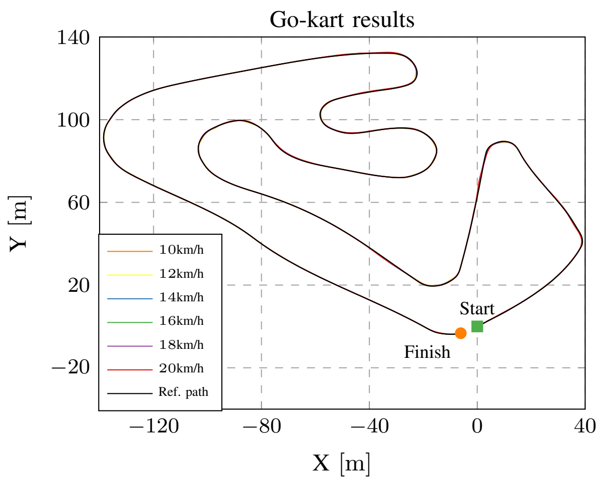

Fig. 6. The Adams bicycle riding on a go-kart track.

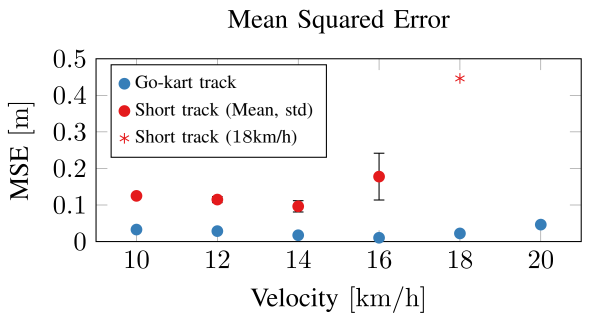

Fig. 7. The Mean Squared Error between the reference path and the bicycle paths at each nominal velocity. The blue dots correspond to the MSE and computed for the go-kart track. The red dots and error bars correspond to the mean and standard deviation of the MSE for ten runs at each nominal velocity 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.

the short track trajectory is presented for the nominal velocity of 14km/h[5]. The variance, due to noise and disturbances, between the ten repetitions is represented in blue, and the reference path is highlighted in red.

C. Discussion

The results in Fig. 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. 7. The Hausdorff distances in Fig. 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 is a more challenging track with narrow s-curves and short straights compared to the go-kart track in Fig. 6 or the tracks considered in [7], [12]. As a result, both the MSE and the Hausdorff distance in Fig. 7 and Fig. 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 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 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

[5]`https://www.youtube.com/watch?v=Wq6SvOX7erA`

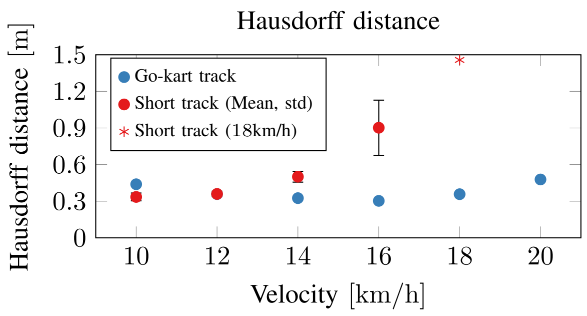

Fig. 8. The Hausdorff distance at respective velocity for the long go-kart track is represented by the blue dots. Similarly, the red dots represent the mean Hausdorff distance, with error bars indicating the standard deviation, when riding on the short track with narrow curves. At the nominal velocity 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.

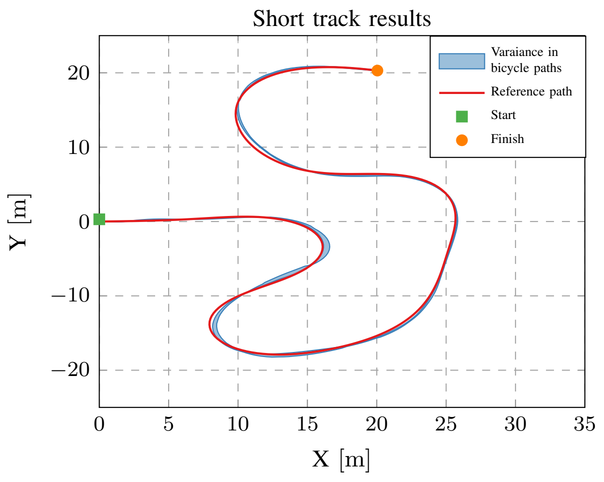

Fig. 9. Simulations results for the co-simulation on the short track at the nominal velocity of 14km/h. The simulations are repeated ten times and the variance in bicycle path for these ten repetitions are highlighted in blue.

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.

VI. 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 successfully 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.

VII. Acknowledgments

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.

References

[1] K. J. Åström, R. E. Klein, and A. Lennartsson, "Bicycle dynamics and control: adapted bicycles for education and research," *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, 2005.

[2] A. L. Schwab and J. P. Meijaard, "A review on bicycle dynamics and rider control," *Vehicle System Dynamics*, vol. 51, no. 7, pp. 1059–1090, 2013.

[3] N. C. Sanchez, L. A. Pastor, and K. Larson, "Autonomous bicycles: A new approach to bicycle-sharing systems," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2020.

[4] P. Fairley, "Self-driving cars have a bicycle problem [news]," *IEEE Spectrum*, vol. 54, pp. 12–13, March 2017.

[5] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[6] N. H. Amer, H. Zamzuri, K. Hudha, and Z. A. Kadir, "Modelling and control strategies in path tracking control for autonomous ground vehicles: a review of state of the art and challenges," *Journal of intelligent & robotic systems*, vol. 86, no. 2, pp. 225–254, 2017.

[7] M. Baquero-Suárez, J. Cortés-Romero, J. Arcos-Legarda, and H. Coral-Enriquez, "A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle," *Multibody System Dynamics*, no. 45, pp. 1–29, 2018.

[8] T.-K. Dao and C.-K. Chen, "Path-tracking control of a riderless bicycle via road preview and speed adaptation," *Asian Journal of Control*, vol. 15, no. 4, pp. 1036–1050, 2013.

[9] F. Whipple, "Stability of the motion of a bicycle," *Quarterly Journal of Pure and Applied Mathematics*, vol. 30, pp. 312–348, 1899.

[10] N. Getz, "Control of balance for a nonlinear nonholonomic non-minimum phase model of a bicycle," in *Proceedings of 1994 American Control Conference-ACC'94*, vol. 1, pp. 148–151, IEEE, 1994.

[11] N. H. Getz and J. E. Marsden, "Control for an autonomous bicycle," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1397–1402 vol.2, 1995.

[12] J. Yi, D. Song, A. Levandowski, and S. Jayasuriya, "Trajectory tracking and balance stabilization control of autonomous motorcycles," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2583–2589, IEEE, 2006.

[13] J. Yi, Y. Zhang, and D. Song, "Autonomous motorcycles for agile maneuvers, part i: Dynamic modeling," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 4613–4618, IEEE, 2009.

[14] J. Kooijman, J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. Schwab, "A bicycle can be self-stable without gyroscopic or caster effects," *Science*, vol. 332, no. 6027, pp. 339–342, 2011.

[15] J. He, M. Zhao, and S. Stasinopoulos, "Constant-velocity steering control design for unmanned bicycles," *2015 IEEE International Conference on Robotics and Biomimetics, IEEE-ROBIO 2015*, pp. 428–433, 2015.

[16] Y. Zhang, P. Wang, J. Yi, D. Song, and T. Liu, "Stationary balance control of a bikebot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6706–6711, IEEE, 2014.

[17] S. Miah, E. Milonidis, I. Kaparias, and N. Karcanias, "An innovative multi-sensor fusion algorithm to enhance positioning accuracy of an instrumented bicycle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1145–1153, 2020.

[18] J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, "Linearized dynamics equations for the balance and steer of a bicycle: a benchmark and review," *Proceedings of the Royal society A: mathematical, physical and engineering sciences*, vol. 463, no. 2084, pp. 1955–1982, 2007.

[19] R. Frezza, A. Beghi, and A. Saccon, "Model predictive for path following with motorcycles: application to the development of the pilot model for virtual prototyping," in *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, vol. 1, pp. 767–772, IEEE, 2004.

[20] M. Zhao, S. Stasinopoulos, and Y. Yu, "Obstacle detection and avoidance for autonomous bicycles," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pp. 1310–1315, 2017.

[21] T. Andersson, N. Persson, A. Fattouh, and M. C. Ekström, "A loop shaping method for stabilising a riderless bicycle," in *2019 European Conference on Mobile Robots (ECMR)*, pp. 1–6, Sep. 2019.

[22] A. O'Dwyer, *Handbook of PI and PID Controller Tuning Rules*. Imperial College Press, 3rd ed., 2009.

[23] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[24] A. Zenere and M. Zorzi, "Model predictive control meets robust kalman filtering," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3774–3779, 2017.

[25] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*, vol. 317. Springer Science & Business Media, 2009.

[26] J. K. Moore, J. Kooijman, A. Schwab, and M. Hubbard, "Rider motion identification during normal bicycling by means of principal component analysis," *Multibody System Dynamics*, vol. 25, no. 2, pp. 225–244, 2011.