

Optimization-Based System Identification and Moving Horizon Estimation Using Low-Cost Sensors for a Miniature Car-Like Robot

Sabrina Bodmer*, Lukas Vogel*, Simon Muntwiler, Alexander Hansson,
Tobias Bodewig, Jonas Wahlen, Melanie N. Zeilinger, and Andrea Carron



Fig. 1: (a) Chronos car based on a Kyosho AWD buggy model with custom electronics: IMU, wheel encoders with magnets in wheels, and Lighthouse positioning deck. (b) Track setup with Lighthouse base station. (c) Hardware experiment.

Abstract— This paper presents an open-source miniature car-like robot with low-cost sensing and a pipeline for optimization-based system identification, state estimation, and control. The overall robotics platform comes at a cost of less than \$700 and thus significantly simplifies the verification of advanced algorithms in a realistic setting. We present a modified bicycle model with Pacejka tire forces to model the dynamics of the considered all-wheel drive vehicle and to prevent singularities of the model at low velocities. Furthermore, we provide an optimization-based system identification approach and a moving horizon estimation (MHE) scheme. In extensive hardware experiments, we show that the presented system identification approach results in a model with high prediction accuracy, while the MHE results in accurate state estimates. Finally, the overall closed-loop system is shown to perform well even in the presence of sensor failure for limited time intervals. All hardware, firmware, and control and estimation software is released under a BSD 2-clause license to promote widespread adoption and collaboration within the community.

Code: <https://gitlab.ethz.ch/ics/crs>

Dataset: <https://gitlab.ethz.ch/ics/crs/-/tree/main/datasets>

Video: <https://youtu.be/vKF30Sol8Qk>

I. INTRODUCTION

Experimental work plays a crucial role in assessing the effectiveness and limitations of control and estimation methods. However, the execution of hardware experiments faces various challenges, even for relatively simple setups. These challenges include the high costs associated with procuring and setting up hardware platforms, the difficulty in identifying models and their parameters needed for model-based state estimation and control, the complexity involved in developing the control software architecture, and the scarcity of (advanced) open-source control and estimation

algorithms [1]–[3] that could be reused to speed up the deployment. To mitigate these limitations, we present an indoor low-cost localization¹ system, an optimization-based approach for system identification, and a moving horizon estimation (MHE) approach for an updated version of the miniature, low-cost car-like robot called Chronos [4].

Contribution: The contributions of this paper include:

- 1) The introduction of an open-source and low-cost hardware platform (Section II), enhancing the Chronos car introduced in [4] with custom-built wheel encoders and the off-the-shelf Lighthouse positioning deck [5], see Fig. 1. The presented robotics platform costs less than \$700. Fusing the information of the different sensors allows us to obtain accurate estimates of the state of Chronos. The presented hardware setup significantly lowers the bar for hardware experiments, which is paramount to testing algorithms in realistic settings in the control, robotics, and machine learning research and education communities. The presented hardware (electronic schematics and printed circuit board (PCB) designs), software, and firmware are open-source under a BSD 2-clause license. The platform is designed modularly, allowing for different system models, e.g., model rockets [6], and multi-agent applications, e.g., coverage control [7].
- 2) Improvements of the car modelling (Section III). In particular, we present an extension of the standard bicycle model by splitting the motor force to the front and rear axle such that one formulation can be used to model front-, rear-, and all-wheel drive cars. In addition, an approximation for the Pacejka tire forces is introduced to overcome singularities at zero longitudinal velocities.
- 3) A complete pipeline for advanced optimization-based system identification (Section IV), state estimation (Section V),

*Sabrina Bodmer and Lukas Vogel contributed equally to this work.

The work of Simon Muntwiler was supported by the Bosch Research Foundation im Stifterverband. The authors are members of the Institute for Dynamic Systems and Control, ETH Zurich, Switzerland.

¹By localization, we refer to estimating the configuration state (position and orientation) of a robotic system in three-dimensional space, while state estimation refers to estimating the full system state, e.g., including velocities and yaw rates.

and control (Section VI). In many practical applications, system identification and state estimation are limiting factors for advanced control design, e.g., due to the nonlinear nature of the considered systems and because the system state can not be measured directly. The presented pipeline of optimization-based algorithms aims to support the design and application of model learning and control with guarantees, e.g., stability and constraint satisfaction. An overview of the provided software frame work is given in [8, Appendix A].

4) Extensive hardware experiments (Section VII) show that the available low-cost sensors allow for accurate state estimation and for controlling the system in closed-loop, even in the presence of sensor failure for limited time intervals. Furthermore, we make the dataset used for system identification and the open-loop experiments available online. We believe that these datasets can become a useful benchmark for nonlinear system identification and estimation.

Related Work: Localization systems are crucial in enabling autonomous navigation for various robotic platforms. Possible approaches include motion capture, overhead camera systems, and onboard sensors. Motion capture systems, used in platforms like [9]–[12] offer unparalleled accuracy and precision. However, their widespread adoption is hindered by their high cost, making them impractical for low-cost applications. On the other hand, overhead camera systems, such as those employed in [13]–[17] offer a more cost-effective solution. However, their accuracy is typically lower and the software is not open-source, further limiting their accessibility and customization for specific applications. Alternatively, onboard sensor-based localization systems, like those utilized in [18], [19], offer a more versatile solution that does not require external infrastructure. However, they may not always be suitable for applications with stringent size limitations.

Car-like robots are often modeled using a bicycle model, with tire forces modeled with the Pacejka model [4], [20]. A modified model to account for all-wheel drive (AWD) configurations was introduced in [21]. Models obtained from first principles usually contain parameters that cannot be measured directly and, therefore, require parametric system identification methods relying on input/output data only. In general, these methods are scarce for nonlinear systems. Promising methods make use of (non-convex) optimization approaches to find sequences of system states and parameters which maximize the likelihood of available input/output data, compare, e.g., [22]–[24]. Similarly, sensor calibration can be performed in an optimization-based manner, as, e.g., done in [5] for the Lighthouse positioning system.

In practical control applications, state estimation plays a crucial role. Compared to the widely used extended Kalman filter (EKF) [25], an MHE approach [26, Chap. 4] is very promising for safety critical systems, as it does not rely on linearization of the system model and can provide robust stability of the resulting state estimate [27]. Outliers can be rejected within an MHE by neglecting the corresponding measurements [28] or using an appropriate objective function [29]. System parameters that cannot be identified

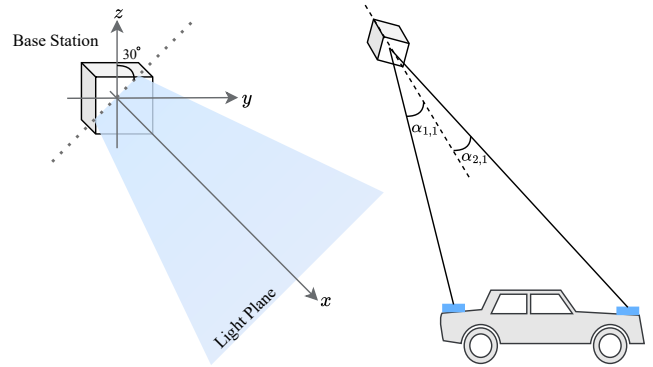


Fig. 2: The left figure demonstrates a single sweep of the light plane by the base station. Both light planes rotate around the z -axis. The second light plane is rotated by 60° compared to the first plane (depicted here in blue). The right image demonstrates the angles measured by the onboard sensors. The angles $\alpha_{1,1}$, $\alpha_{2,1}$ refer to the angle measurement of the first and second sensor for the first light plane sweep.

perfectly offline or are slowly time-varying (e.g., due to tire wear), can be jointly estimated with system states in an MHE if the online measurements are informative enough [30], while robustly stable state estimates can be obtained even if the data is not informative [31]. While MHE has been applied for (offline) estimation of car positions based on real-world vehicle test data [32] and joint state and friction estimation in simulation [33], applications of MHE to real-world dynamical systems are rare.

Notation: We use $\mathbb{I}_{[a,b]}$ to denote the set of integers $i \in \mathbb{N}$ with $a \leq i \leq b$ and $\mathbb{I}_{\geq a}$ to denote the set of integers greater or equal to a . The sequence (x_0, \dots, x_N) is denoted as $\{x_j\}_{j=0}^N$, or abbreviated as $\{x_j\}$. The weighted square norm of x is $\|x\|_Q^2 = x^\top Q x$. $R_\chi(\varphi) \in \text{SO}(3)$ are elementary rotation matrices of angle φ around the coordinate axis χ .

II. CHRONOS ONBOARD AND EXTERNAL SENSORS

In the following, we provide a brief hardware overview of the onboard sensors mounted on the Chronos car [4] and the external Lighthouse positioning system. For an in-depth overview of the CRS software framework, see [8, Appendix A] and [4].

A. Inertial Measurement Unit

The car is equipped with an inertial measurement unit (IMU) aligned with the body axes. It provides linear acceleration and angular velocity measurements at a rate of 250 Hz.

B. Wheel Encoders

Using rotary encoders as a source of odometry is an established practice in robotics [34], and the twist estimates from an encoder system can be used in dead reckoning scenarios. To obtain an estimate of the angular velocities of the wheels Ω_i , we mount eight small magnets in the wheel rims and a small PCB with a hall effect sensor close to each wheel axle. We sample the wheel speeds at 250 Hz.

C. Lighthouse Positioning System

The Lighthouse positioning system consists of (potentially) multiple base stations, each emitting two rotating light planes, and photodiode sensors mounted on an object to be tracked. The base stations modulate a data stream onto the emitted light planes, synchronized with their rotation. Using a third-party lightweight sensor board called Lighthouse deck, distributed by Bitcraze [5], the demodulated data stream can be used to determine the angle α of the light plane relative to the base station when the four sensors on the deck are hit. Through triangulation, the position of the sensors relative to the base station can be determined. Fig. 2 depicts the system's working principle.

We use version 2.0 of the base stations. In the planar 2D tracking case, a single base station above the racetrack is sufficient to cover an area of around 6 m by 6 m and provides updates at around 50 Hz. Additional base stations can be used to cover larger areas and increase the update rate.

III. SYSTEM AND SENSORS MODELLING

In this section, we describe the system and sensor models of the Chronos car used for system identification and state estimation. Furthermore, we introduce the model discretization.

A. System Model

We consider a dynamic bicycle model with a simplified Pacejka tire force model [4], [20], with adaptations similar to [21] to account for AWD configurations. The overall system state is denoted as $x = [x_p \ y_p \ \psi \ v_x \ v_y \ \omega]^T \in \mathbb{R}^{n_x}$ with $n_x = 6$. The x and y coordinates, as well as the yaw angle in world frame, are given by x_p, y_p and ψ . Longitudinal and lateral velocities, as well as the yaw rate, are denoted by v_x, v_y and ω , and are given in body frame. The model input $u = [\delta \ T]^T \in \mathbb{R}^{n_u}$ with $n_u = 2$ consists of the steering angle δ and the input torque T . The continuous-time system is governed by the following differential equations

$$\dot{x}_p = v_x \cos(\psi) - v_y \sin(\psi), \quad (1a)$$

$$\dot{y}_p = v_x \sin(\psi) + v_y \cos(\psi), \quad (1b)$$

$$\dot{\psi} = \omega, \quad (1c)$$

$$\dot{v}_x = \frac{1}{m} (F_{x,r} + F_{x,f} \cos(\delta) - F_{y,f} \sin(\delta) + m v_y \omega - F_{fr}), \quad (1d)$$

$$\dot{v}_y = \frac{1}{m} (F_{y,r} + F_{x,f} \sin(\delta) + F_{y,f} \cos(\delta) - m v_x \omega), \quad (1e)$$

$$\dot{\omega} = \frac{1}{I_z} (F_{y,f} l_f \cos(\delta) + F_{x,f} l_f \sin(\delta) - F_{y,r} l_r), \quad (1f)$$

where m is the mass of the car, I_z is the inertia along the z -axis, and l_f and l_r are the distance of the front and rear axis from the center of mass, respectively. The lateral tire forces $F_{y,f}$ and $F_{y,r}$ are modeled with the simplified Pacejka tire force model

$$F_{y,f} = D_f \sin(C_f \arctan(B_f \alpha_f)),$$

$$F_{y,r} = D_r \sin(C_r \arctan(B_r \alpha_r)),$$

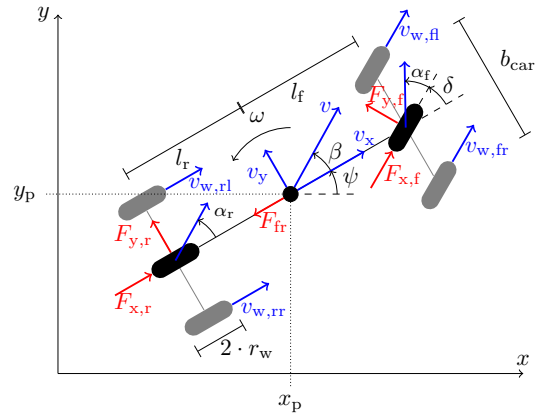


Fig. 3: Dynamic bicycle model with velocity in direction of each wheel used for the wheel encoder model (adapted from [35]).

where B_f, B_r, C_f, C_r, D_f , and D_r are the Pacejka tire model parameters and α_f, α_r refer to the front and rear slip angles which depend on v_x, v_y , and ω . In the classical Pacejka model [4], [20], the slip angles α_f, α_r are discontinuous as $v_x \rightarrow 0$, which poses a significant limitation when modeling the dynamics at low speeds or standstill. To overcome this, we fit polynomials of third-degree (p_f, p_r) to express the slip angles in a low-velocity regime $|v_x| \leq \epsilon$, with small constant $\epsilon \in \mathbb{R}_{>0}$. Therefore, the front and rear slip angles are computed as

$$\alpha_f = \begin{cases} \hat{\alpha}_f(v_x) = \arctan\left(\frac{-\omega l_f - v_y}{v_x}\right) + \delta, & \text{if } |v_x| \geq \epsilon, \\ p_f(v_x) = b_f v_x + c_f v_x^3, & \text{otherwise,} \end{cases}$$

$$\alpha_r = \begin{cases} \hat{\alpha}_r(v_x) = \arctan\left(\frac{\omega l_r - v_y}{v_x}\right), & \text{if } |v_x| \geq \epsilon, \\ p_r(v_x) = b_r v_x + c_r v_x^3, & \text{otherwise,} \end{cases}$$

where b_f, c_f, b_r , and c_r depend on v_y, ω, l_f , and l_r and are determined such that the slip angle dynamics α_f, α_r are continuously differentiable by solving

$$\hat{\alpha}_i(\epsilon) = p_i(\epsilon), \quad \left. \frac{\partial}{\partial v_x} (\hat{\alpha}_i(v_x)) \right|_{v_x=\epsilon} = b_i + 3c_i \epsilon^2,$$

for $i = \{f, r\}$. We propose a modification of the standard model [4], [20], as similarly done in [21], where the longitudinal forces acting in the direction of the rear and front wheels, $F_{x,r}$ and $F_{x,f}$, respectively, are modeled as $F_m = (C_{m,1} - C_{m,2} v_x) T$, $F_{x,r} = \gamma F_m$, $F_{x,f} = (1 - \gamma) F_m$, where $C_{m,1}$, and $C_{m,2}$ are model parameters, and the factor γ is used to split the force between rear and front axle. Values of $\gamma = 1$, $\gamma = 0$ and $0 < \gamma < 1$ can model rear-, front-, and all-wheel drive vehicles, respectively. We model friction effects in a parameterized Taylor approximation, inspired by the physical effects of roll resistance and viscous drag as $F_{fr} = \text{sgn}(v_x)(C_{d,2} v_x^2 + C_{d,1} v_x + C_{d,0})$, where $C_{d,i}$ are the expansion parameters.

The Pacejka tire model, physical model, friction, and motor parameters are given as $\theta_p = [D_f \ D_r \ C_f \ C_r \ B_f \ B_r]$, $\theta_{car} = [m \ I_z \ l_f \ l_r]$, $\theta_{fr} = [C_{d,0} \ C_{d,1} \ C_{d,2}]$, and $\theta_T = [C_{m,1} \ C_{m,2}]$ respectively. The total dynamic model parameters are denoted as $\theta_m = [\theta_p \ \theta_{car} \ \theta_{fr} \ \theta_T]$, with $n_{\theta_m} = 15$.

B. Sensor Models

In the following, we will describe the sensor models of the inertial measurement unit (IMU), wheel encoders, and Lighthouse positioning system.

1) *Inertial measurement unit*: The IMU provides measurements of linear acceleration and angular velocity as

$$y_{\text{IMU}} = \begin{bmatrix} \frac{1}{m} (F_{x,r} + F_{x,f} \cos(\delta) - F_{y,f} \sin(\delta) - F_{fr}) + v_y \omega \\ \frac{1}{m} (F_{y,r} + F_{x,f} \sin(\delta) + F_{y,f} \cos(\delta)) - v_x \omega \\ \omega \end{bmatrix}.$$

The number of measurements is given as $n_{\text{IMU}} = 3$.

2) *Wheel encoders*: Custom-built wheel encoders are used to measure the angular velocity of each of the four wheels of the car. Assuming no skid, the measurement model of the wheel encoders can be obtained as

$$y_{\text{we}} = \frac{1}{r} [v_{w,fl} \quad v_{w,fr} \quad v_{w,rl} \quad v_{w,rr}]^\top = \frac{1}{r} \begin{bmatrix} \cos(\delta)(v_x - 0.5b_{\text{car}}\omega) + \sin(\delta)(v_y + l_f\omega) \\ \cos(\delta)(v_x + 0.5b_{\text{car}}\omega) + \sin(\delta)(v_y + l_f\omega) \\ v_x - 0.5b_{\text{car}}\omega \\ v_x + 0.5b_{\text{car}}\omega \end{bmatrix}, \quad (5)$$

where r is the wheel radius, b_{car} is the width of the car, and $v_{w,fl}$, $v_{w,fr}$, $v_{w,rl}$, $v_{w,rr}$ are the velocities in the direction of the front left, front right, rear left, and rear right wheels, respectively. The parameters of the wheel encoders are denoted as $\theta_{\text{we}} = [r \quad b_{\text{car}}]^\top \in \mathbb{R}^{n_{\theta_{\text{we}}}}$, with $n_{\theta_{\text{we}}} = 2$. The number of measurements is given as $n_{\text{we}} = 4$.

3) *Lighthouse positioning system*: The Lighthouse positioning system is used to measure the light plane impact angles $\alpha_{k,l}$, where $k \in \{1, 2, 3, 4\}$ refers to the sensor and $l \in \{1, 2\}$ determines the light plane. The position and orientation of the vehicle can be determined from these measurements. In the following, we provide the sensor model and relevant equations for a single base station. However, these can be extended to account for multiple base stations. The measurement model can be obtained as

$$y_{\text{lh}} = [\alpha_{1,1} \quad \alpha_{2,1} \quad \alpha_{3,1} \quad \alpha_{4,1} \quad \alpha_{1,2} \quad \alpha_{2,2} \quad \alpha_{3,2} \quad \alpha_{4,2}],$$

where $\alpha_{k,l}$ is the impact angle for sensor k and light plane l . The impact angles are given as

$$\alpha_{k,l} = \tan^{-1} \left(\frac{y_{\text{lh},k}}{x_{\text{lh},k}} \right) + \sin^{-1} \left(\frac{z_{\text{lh},k} \tan(t_i)}{\sqrt{x_{\text{lh},k}^2 + y_{\text{lh},k}^2}} \right), \quad (6)$$

where $t_1 = -\pi/6 - \delta t_1$ and $t_2 = \pi/6 - \delta t_2$ are the tilt angles of the two light planes of each base station, and δt_1 and δt_2 are factory calibrated offsets. Finally, the position of each sensor $\mathcal{L}p_{\text{lh},k} = [x_{\text{lh},k} \quad y_{\text{lh},k} \quad z_{\text{lh},k}]^\top$, in the Lighthouse base station frame, is related to the position of the car via the transformation

$$\mathcal{L}p_{\text{lh},k} = R_{\text{bs}}^\top \left(\begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \\ 0 & 0 \end{bmatrix} \mathcal{B}p_{\text{lh},k} + \begin{bmatrix} x_p \\ y_p \\ 0 \end{bmatrix} - p_{\text{bs}} \right),$$

where $\mathcal{B}p_{\text{lh},k}$ is the position of the k -th sensor in body frame and $R_{\text{bs}}, p_{\text{bs}}$ are the rotation and

position of the Lighthouse base station. Thereby, R_{bs} is parametrized by the three rotation angles $\Phi = (\varphi_1, \varphi_2, \varphi_3)^\top$ as $R_{\text{bs}} = R_z(\varphi_3)R_y(\varphi_2)R_x(\varphi_1)$. The parameters of the Lighthouse model are denoted as $\theta_{\text{lh}} = [\Phi \quad p_{\text{bs}} \quad \mathcal{B}p_{\text{lh},k} \quad \delta t_1 \quad \delta t_2]^\top \in \mathbb{R}^{n_{\theta_{\text{lh}}}}$, with $n_{\theta_{\text{lh}}} = 16 \cdot n_{\text{bs}}$ and n_{bs} is the number of base stations. The number of measurements is given as $n_{\text{lh}} = 8$.

C. Discrete-time System Model

By numerically integrating (1) and combining all sensor models, we obtain a discrete-time system of the form

$$x_{t+1} = f(x_t, u_t, w_t, \theta), \quad (7a)$$

$$y_t = h(x_t, u_t, w_t, \theta), \quad (7b)$$

where $y := [y_{\text{IMU}}^\top \quad y_{\text{we}}^\top \quad y_{\text{lh}}^\top]^\top \in \mathbb{Y} \subseteq \mathbb{R}^{n_y}$ with $n_y := n_{\text{IMU}} + n_{\text{we}} + n_{\text{lh}}$ and $\theta := [\theta_m^\top \quad \theta_{\text{we}}^\top \quad \theta_{\text{lh}}^\top]^\top \in \mathbb{R}^{n_\theta}$ with $n_\theta := n_{\theta_m} + n_{\theta_{\text{we}}} + n_{\theta_{\text{lh}}}$. Additional process and measurement noise is denoted by $w_t \in \mathbb{W} \subseteq \mathbb{R}^{n_w}$. Note that w_t appears in the dynamics (7a) and measurement model (7b) and hence can also model separate process disturbances and measurement noise. We assume to know an estimate \bar{x}_0 of the initial state of the system (7) at time step $t = 0$. While the parameter θ is unknown, we assume to have access to a prior estimate $\bar{\theta}_0$ and additional information in the form of constraints

$$\theta \in \Theta \subseteq \mathbb{R}^{n_\theta}. \quad (8)$$

Note that if some parameters are easy to measure (e.g., the mass m), this can be enforced in the constraints (8). Additional information and physical state limits in the form of constraints \mathbb{X}_{ID} can be used during system identification, while it is often crucial to consider safety constraints on states \mathbb{X} and inputs \mathbb{U} during online operation.

IV. SYSTEM IDENTIFICATION

To use the system model (7), the model parameters θ need to be identified. In the following, we describe the calibration procedure for the Lighthouse positioning system and an optimization-based approach to identify the parameters θ .

A. Lighthouse Calibration

For practical reasons, we decouple determining the parameters θ_{lh} from the system model identification. The parameters $\delta t_1, \delta t_2$ are factory-calibrated offsets of the light planes, provided in the base station's data stream. The position of the k sensors in body frame, $\mathcal{B}p_{\text{lh},k}$, can readily be measured. To determine the position and rotation of the base station, p_{bs} and R_{bs} , with respect to the world reference frame, we measure the Lighthouse angles received at n_{cal} static known positions p_i , $i = 1, \dots, n_{\text{cal}}$, and average them as $\bar{\alpha}_{i,l} = (\sum_k \alpha_{i,k,l})/k$, for both light planes $l \in \{1, 2\}$. We then solve an optimization problem minimizing the mean squared error between the angle measurements $\bar{\alpha}_i = [\bar{\alpha}_{i,1} \quad \bar{\alpha}_{i,2}]^\top$ and the expected measurements from (6),

$$\min_{(\Phi, p_{\text{bs}})} \sum_{i=1}^{n_{\text{cal}}} \|\bar{\alpha}_i - \alpha_i(\Phi, p_{\text{bs}}, p_i)\|^2. \quad (9)$$

The obtained Lighthouse parameters are denoted as $\hat{\theta}_{\text{lh}}$.

B. Optimization-based System Identification

To estimate the parameters θ of the system (7) from noisy output measurements, we rely on an optimization-based approach similar to [22]–[24]. The (unknown) state trajectory of the system is denoted by $\{\hat{x}_j\}_{j=0}^{L-1}$. Assuming a set of input-output data of length L , $\mathbb{D} = \{u_j, y_j\}_{j=0}^{L-1}$, and an initial parameter estimate $\hat{\theta}_0$ are available, the objective is chosen as

$$V_{\text{SysID}}(\hat{\theta}, \hat{x}_0, \{\hat{w}_j\}_{j=0}^{L-1}) = \|\hat{\theta} - \bar{\theta}_0\|_{P_\theta}^2 + \|\hat{x}_0 - \bar{x}_0\|_{P_x}^2 + \sum_{j=0}^{L-1} \|\hat{w}_j\|_Q^2, \quad (10)$$

where the weighting matrices $P_x, P_\theta, Q \succ 0$ are tuning matrices. A natural choice is the inverse of the covariance matrices of the initial state, parameter, and process/measurement noise, respectively. An estimate of the system parameters is then obtained by solving the following nonlinear program offline

$$\begin{aligned} \min_{\hat{\theta}, \{\hat{x}_j\}, \{\hat{w}_j\}} V_{\text{SysID}}(\hat{\theta}, \hat{x}_0, \{\hat{w}_j\}) & \quad (11a) \\ \text{s.t. } \hat{x}_{j+1} = f(\hat{x}_j, u_j, \hat{w}_j, \hat{\theta}), \quad j \in \mathbb{I}_{[0, L-1]}, & \quad (11b) \\ y_j = h(\hat{x}_j, u_j, \hat{w}_j, \hat{\theta}), \quad j \in \mathbb{I}_{[0, L-1]}, & \quad (11c) \\ \hat{w}_j \in \mathbb{W}, \hat{x}_j \in \mathbb{X}_{\text{ID}}, \quad j \in \mathbb{I}_{[0, L-1]}, & \quad (11d) \\ \hat{\theta} \in \Theta. & \quad (11e) \end{aligned}$$

A (non-unique) minimizer of (11) is denoted as $\hat{\theta}^*, \{\hat{x}_j\}^*, \{\hat{w}_j\}^*$ and the resulting parameter estimate as $\hat{\theta} = \hat{\theta}^*$. Note that θ_{lh} obtained above can be enforced within (11e). The optimization problem (11) is non-convex, even for linear system dynamics. Therefore, the solution is highly sensitive to the initialization of the solver. To overcome this, it is essential to either run the optimization problem from multiple different initialization points or to warm-start the problem with a good initial guess, e.g., by first running a state estimator based on the prior parameter value $\bar{\theta}_0$ to initialize the sequence of states $\{\hat{x}_j\}$.

V. STATE ESTIMATION

Given the discrete system dynamics (7), as well as the parameter estimate $\hat{\theta}$ obtained through system identification in Section IV, we introduce an online approach to obtain an estimate \hat{x}_t of the system state at each time step t given past input and output data $\{u, y\}$. In particular, we rely on a moving horizon estimation (MHE) approach, which is an optimization-based estimation scheme that considers the past state estimate \hat{x}_{t-M_t} , obtained at time step $t - M_t$, as well as input and output data $\{u_j, y_j\}_{j=t-M_t}^{t-1}$ within a window M_t . To account for the initialization phase with limited data available, we define $M_t = \min\{t, M\}$, where $M \in \mathbb{I}_{\geq 0}$ is a fixed (bounded) horizon length. The MHE approach optimizes over the initial state estimate $\hat{x}_{t-M_t|t}$ and a sequence of M_t noise estimates $\hat{w}_{\cdot|t} = \{\hat{w}_j|t\}_{j=t-M_t}^{t-1}$ and output estimates $\hat{y}_{\cdot|t} = \{\hat{y}_j|t\}_{j=t-M_t}^{t-1}$. Given $\hat{\theta}$, $\hat{x}_{t-M_t|t}$ as well as $\hat{w}_{\cdot|t}$, the sequence of state estimates along the

horizon can be computed from the system dynamics (7a). The objective of the MHE problem is chosen as

$$V_{\text{MHE}}(\hat{x}_{t-M_t|t}, \hat{w}_{\cdot|t}, \hat{y}_{\cdot|t}, t) = \eta^{M_t} \|\hat{x}_{t-M_t|t} - \hat{x}_{t-M_t}\|_P^2 + \sum_{j=1}^{M_t} \eta^{j-1} (\|\hat{w}_{t-j|t}\|_Q^2 + \|\hat{y}_{t-j|t} - y_{t-j}\|_R^2), \quad (12)$$

where P, Q and R are appropriate covariance matrices. The discount factor $\eta \in (0, 1)$ ensures that more recent measurements have a greater impact on the optimization problem. The state estimate at time step t is then obtained by solving the following nonlinear program (NLP)

$$\begin{aligned} \min_{\hat{x}_{t-M_t|t}, \hat{w}_{\cdot|t}} V_{\text{MHE}}(\hat{x}_{t-M_t|t}, \hat{w}_{\cdot|t}, \hat{y}_{\cdot|t}, t) & \quad (13a) \\ \text{s.t. } \hat{x}_{j+1|t} = f(\hat{x}_j|t, u_j, \hat{w}_j|t, \hat{\theta}), \quad j \in \mathbb{I}_{[t-M_t, t-1]}, & \quad (13b) \\ \hat{y}_j|t = h(\hat{x}_j|t, u_j, \hat{w}_j|t, \hat{\theta}), \quad j \in \mathbb{I}_{[t-M_t, t-1]}, & \quad (13c) \\ \hat{w}_j|t \in \mathbb{W}, \hat{y}_j|t \in \mathbb{Y}, \quad j \in \mathbb{I}_{[t-M_t, t-1]}, & \quad (13d) \\ \hat{x}_j|t \in \mathbb{X}, \quad j \in \mathbb{I}_{[t-M_t, t]}, & \quad (13e) \end{aligned}$$

where (13b), (13c) are the measurement and dynamic model constraints, (13d) refers to noise and output bounds and (13e) are state constraints. A (non-unique) minimizer of (13) is denoted as $\hat{x}_{t-M_t|t}^*, \hat{w}_{\cdot|t}^*$, and the resulting state estimate as

$$\hat{x}_t = \hat{x}_{t|t}^*. \quad (14)$$

Note that a particular benefit of MHE compared to other nonlinear estimation approaches, e.g., the classical EKF [25], in the context of safety-critical applications is the ability to establish theoretical properties for the resulting state estimate (14). In particular, if the true parameters are obtained during system identification, the system (7) is detectable [27, Ass. 1], and $\eta < 1$ and the horizon length M are chosen sufficiently large, then the estimation error, i.e., $x_t - \hat{x}_t$, resulting from the MHE approach (13) is robustly stable [27, Cor. 1], which implies that it is upper bounded at all time steps by decaying terms involving the initial state estimation error and the noise acting on the system. For bounded uncertainties, this allows to robustify a control algorithm to state estimation errors, compare [36]. If the true system parameters are not recovered during system identification, MHE allows to obtain stable state estimates even if the available online data is not sufficiently informative [31]. Additionally, states and parameters can be estimated jointly when the data is informative [30].

VI. CONTROL

In the following, we first provide an overview of a high-level control scheme to compute a control input based on the current state estimate \hat{x}_t , and then introduce the low-level controllers onboard the car.

A. Model Predictive Contouring Control

Model predictive contouring control (MPCC) has been used extensively in planning and control for autonomous racing [4], [16], [35]. It takes into account the track boundaries as constraints and optimizes the progress along a reference

path while trading off path following and performance. It can be formulated as the NLP

$$\min_{\{u_i\}_{i=0}^N} \sum_{i=0}^N \|\varepsilon(x_i)\|_Q^2 - Q_{\text{adv}}\gamma(x_i) + \|u_i\|_R^2 \quad (15a)$$

$$\text{s.t. } x_{i+1} = f(x_i, u_i, 0, \hat{\theta}), \quad x_0 = \hat{x}_t, \quad i \in \mathbb{I}_{[0, N]}, \quad (15b)$$

$$x_i \in \mathbb{X}, \quad u_i \in \mathbb{U}, \quad \gamma(x_i) > 0, \quad i \in \mathbb{I}_{[0, N]}, \quad (15c)$$

where $\varepsilon(x) = [\varepsilon_l(x), \varepsilon_c(x)]^\top$ denotes the lag and contour error, defined as longitudinal and lateral error from the reference trajectory [4]. The matrices $Q \in \mathbb{R}^2$, $Q_{\text{adv}} \in \mathbb{R}$, and $R \in \mathbb{R}^2$ are tuning parameters. The function $\gamma(x)$ describes the progress along the reference path. State and input constraints are specified by the sets \mathbb{X} and \mathbb{U} . The MPCC problem (15) is initialized with the state estimate \hat{x}_t obtained from the MHE (13), while the model parameters $\hat{\theta}$ are found through the methods from Section IV. Note that (15) constitutes a nominal MPCC for system (7), which has proven to be inherently robust to small model perturbations [37].

B. Low-level Control

Onboard Chronos, low-level controllers track the reference given by the solution of (15). The input to the low-level controller is either the pair $\{\delta, T\}$ or $\{\delta, v_x\}$, where δ is the steering angle and T the input torque; as an alternative, the reference longitudinal velocity v_x can be specified. All low-level controllers run at 250 Hz.

1) *Steering Angle Control*: Given a map from the onboard steering potentiometer's voltage V_δ to the resulting steering angle $\delta(V_\delta)$, we estimate the current steering angle and track it using a PID controller that outputs a PWM signal to the motor chip controller of the steer DC motor.

2) *Longitudinal Velocity Control*: From the wheel encoder sensor model (5), we can obtain an onboard estimate of the longitudinal velocity from the rear wheel velocities as $v_x = (\Omega_{r1} + \Omega_{r2}) / (2 \cdot r)$. A PID controller closes the loop to track a given reference velocity.

VII. HARDWARE EXPERIMENTS

This section presents the results of real-world experiments performed on the hardware described in Section II. The Lighthouse base station was mounted approximately over the center of the track on a tripod, with the base station pointing straight down to cover the largest area of the track. Where applicable, we use Qualisys, a high-quality motion capture system, as a ground-truth reference. Note that all relevant parameters can be found in the code available online.

A. Lighthouse Calibration

For calibration, we are interested in two key figures: what is the expected accuracy of the Lighthouse positioning system in static scenarios and how many point-angle correspondences are needed to achieve a sufficient estimate of θ_{lh} . We use the calibration procedure detailed in Section IV-A and collect angle measurements in 24 static locations. We then solve (9) for increasing point-angle correspondences and calculate the resulting residual positions.

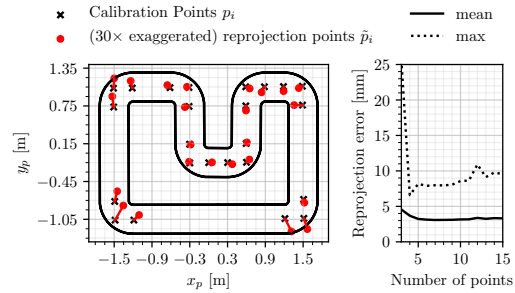


Fig. 4: Left: Reprojection error (scaled 30 \times) of calibration points. Right: Mean/maximum calibration residual by number of points.

As a measure of calibration fidelity, the angles measured during calibration are reprojected onto the plane using the crossing beams method detailed in [5], resulting in residuals with respect to the ground truth calibration points p_i . Fig. 4 shows the reprojection errors displayed on the calibration points, where the error has been scaled 30-fold. Also, mean and maximum residuals are evaluated for a varying number of calibration points n_{cal} , arranged so that the density between points increases approximately equally when adding points. We find that by using only 5–6 well-distributed points, a mean residual of 3 mm can be achieved in static conditions. A small number of points may have residuals up to 11 mm, in areas where the angle averaging described in Section IV-A yields a worse approximation, particularly in shallow angles. Note that the lower bound for achievable residuals here is given by the motion capture system's 2 mm–3 mm residuals for the (ground-truth) calibration points.

B. Optimization-based System Identification

We implement the optimization problem (11) using CasADi [38] and solve it using the interior point solver Ipopt [39]. Using a model-free controller (PID) and an EKF, data is recorded as the car drives around the track. Based on the recorded control inputs and measurements, we perform system identification. The prior parameter estimate θ_0 is chosen from estimates available for a related car model.

From the solution of (11), $\hat{\theta}$ is obtained. We then predict open-loop trajectories $\tilde{x}(t; \theta)$, for both $\hat{\theta}_0$ and $\hat{\theta}$ on an unseen validation dataset $\{\tilde{u}_j, \tilde{y}_j\}_{j=0}^{L-1}$ by simulating (7) over a fixed horizon M for $j \in \mathbb{I}_{[0, L-M-1]}$. In Fig. 5, the contrast between the open-loop predictions of the prior model, parametrized by $\hat{\theta}_0$, to the model using the identified parameters $\hat{\theta}$ is shown. As an improvement metric, we compare to the measured system trajectory. We achieve a reduction in root-mean-squared error (RMSE) from 0.24 m to 0.09 m over an open loop prediction horizon of 2 s.

C. Estimation Analysis

To evaluate the effect of various sensors for the different estimation architectures, an EKF and MHE were implemented. For the MHE, we used Acados [40] as a solver. The respective EKF and MHE RMSE for each state² averaged over one run of twenty seconds are reported in Table I.

²Note that we do not provide the RMSE in yaw rates because we do not have accurate estimates of the ground truth for this state.

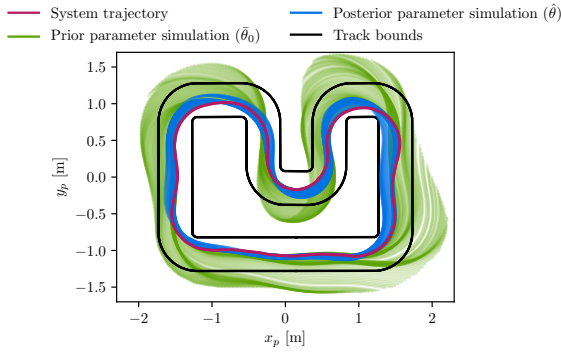


Fig. 5: Open-loop predicted trajectories from inputs $\{\tilde{u}_j\}_{j=t}^{t+M}$ based on the prior parameter estimate $\hat{\theta}_0$ and posterior identified system model $\hat{\theta}$.

TABLE I: RMSE and standard deviation for different estimators

Type	Sensors			x [mm]	y [mm]	ϕ [mrad]	v_x [mm/s]	v_y [mm/s]
	LH	IMU	WE					
EKF	✓	✗	✗	31 ± 0.03	21 ± 0.02	42 ± 0.02	69 ± 0.07	67 ± 0.05
MHE	✓	✗	✗	29 ± 0.03	18 ± 0.02	71 ± 0.05	99 ± 0.10	83 ± 0.08
EKF	✓	✓	✗	25 ± 0.01	16 ± 0.01	44 ± 0.02	375 ± 0.38	290 ± 0.24
MHE	✓	✓	✗	23 ± 0.02	15 ± 0.01	62 ± 0.06	262 ± 0.26	241 ± 0.24
EKF	✓	✗	✓	30 ± 0.03	20 ± 0.02	49 ± 0.03	31 ± 0.03	42 ± 0.04
MHE	✓	✗	✓	32 ± 0.03	21 ± 0.02	74 ± 0.04	34 ± 0.03	44 ± 0.04
EKF	✓	✓	✓	30 ± 0.03	20 ± 0.02	45 ± 0.02	26 ± 0.03	36 ± 0.04
MHE	✓	✓	✓	32 ± 0.03	21 ± 0.02	45 ± 0.03	25 ± 0.02	32 ± 0.03

The estimators are evaluated on open-loop data collected using a joystick controller to evaluate the performance of the MHE without the effects of closed-loop feedback. We collect ground-truth data for all states of the car using Qualisys and record all sensor readings (Lighthouse (LH), Wheel Encoders (WE), IMU). Note that running Qualisys and the Lighthouse positioning system at the same time can result in decreased performance since the infrared flashes of the Qualisys motion capture system can interfere with the demodulation of the Lighthouse frames – both systems occupy a similar spectrum. To limit this effect, the Qualisys system frequency is reduced to 35 Hz. The estimators are then evaluated in an offline setting but process the data in real-time to ensure successful deployment to hardware. The RMSE is calculated using nearest neighbor interpolation based on the measurement times from the Qualisys system.

A closer analysis of the results in Table I shows that the MHE and EKF performed similarly. Interestingly, the sole addition of the IMU sensor leads to improved position estimates at the expense of less accurate body velocities. Including the wheel encoders (which provide a measurement of the forward body velocity) mitigates these effects. While the exact cause is unknown, we attribute this effect to the high noise levels and potential bias on the IMU measurements, which can lead to large velocity errors if these quantities are not directly observed. Note that generally, adding sensors tends to improve the estimator performance. While MHE allows us to establish theoretical guarantees, investigating settings where MHE results in an increased estimation performance compared to, e.g., an EKF remains an interesting question for future work.

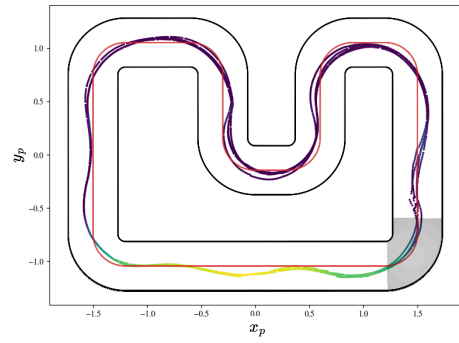


Fig. 6: Closed-loop MPCC using the MHE with the Lighthouse positioning system, IMU, and wheel encoders. High velocities are shown in lighter colors and low velocities in darker colors, respectively. In the shaded gray area, no positioning measurements were available. The track centerline is shown in red.

D. Closed-loop Control

In this experiment, we achieve reliable small-scale autonomous racing with the proposed low-cost and readily available sensor setup. The closed-loop experiments were performed using the MPCC and the MHE described in Section VI and V, respectively. The MHE and MPCC both use a horizon length of 40. The MHE is run at 60 Hz, while the MPCC is run at 30 Hz. Fig. 6 demonstrates the results of the closed-loop system using the Lighthouse positioning system, as well as the IMU and wheel encoders. Additionally, to verify the robustness of the closed-loop estimation and control architecture, we temporarily blinded the Lighthouse positioning system (Fig. 6 sections highlighted in grey) such that the MHE had to rely only on IMU and wheel encoder measurements. The Lighthouse positioning system measurements were missing for a maximum of 1.127 sec and 0.753 sec on average. At the MHE update rate of 60 Hz, positional information was missing for up to 68 MHE calls. We note that the closed-loop system behavior was unaffected by these measurement dropouts and the car successfully completed the lap, showcasing the system can perform well even in the presence of sensor failure for limited time intervals.

VIII. CONCLUSIONS

In this work, we present a low-cost and readily available sensor setup and optimization pipeline for miniature car-like robots where hardware, firmware, and software have been open-sourced. Our method allows for optimization-based system identification, state estimation (MHE), and controls (MPCC). To this end, we introduce analytical sensor models for the Lighthouse positioning system, inertial measurement unit, and wheel encoders, as well as a modified bicycle dynamic model. Finally, we validate and demonstrate that our setup allows for real-time control and estimation by deploying it on miniature race cars.

ACKNOWLEDGMENTS

We would like to thank Karl Marcus Aaltonen, Jerome Sieber, Shengjie Hu, Joshua Naef, Maria Krinner, Griffin Norris, Moritz Hüsser, Marvin Harms, and Andrea Zanelli for the support with the development of Chronos and CRS.

REFERENCES

- [1] S. Zhou, L. Brunke, A. Tao, A. W. Hall, F. P. Bejarano, J. Panerati, and A. P. Schoellig, "What is the impact of releasing code with publications?: Statistics from the machine learning, robotics, and control communities," *IEEE Control Systems Magazine*, vol. 44, no. 4, pp. 38–46, 2024.
- [2] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Lariviere, A. Beygelzimer, F. d'Alche Buc, E. Fox, and H. Larochelle, "Improving reproducibility in machine learning research (a report from the neurips 2019 reproducibility program)," *Journal of Machine Learning Research*, vol. 22, no. 164, pp. 1–20, 2021.
- [3] J. P. How, "Control systems reproducibility challenge [from the editor]," *IEEE Control Systems Magazine*, vol. 38, no. 4, pp. 3–4, 2018.
- [4] A. Carron, S. Bodmer, L. Vogel, R. Zurbrüg, D. Helm, R. Rickenbach, S. Muntwiler, J. Sieber, and M. N. Zeilinger, "Chronos and CRS: Design of a miniature car-like robot and a software framework for single and multi-agent robotics and control," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 1371–1378, 2023.
- [5] A. Taffanel, B. Rousset, J. Danielsson, K. McGuire, K. Richardsson, M. Eliasson, T. Antonsson, and W. Hönig, "Lighthouse Positioning System: Dataset, Accuracy, and Precision for UAV Research," *arXiv preprint arXiv:2104.11523*, 2021.
- [6] L. Spannagl, E. Hampp, A. Carron, J. Sieber, C. A. Pascucci, A. U. Zraggen, A. Domahidi, and M. N. Zeilinger, "Design, optimal guidance and control of a low-cost re-usable electric model rocket," in *Proc. Int. Conf. Intell. Robots Syst. (IROS)*, pp. 6344–6351, 2021.
- [7] R. Rickenbach, J. Köhler, A. Scampicchio, M. N. Zeilinger, and A. Carron, "Active learning-based model predictive coverage control," *IEEE Trans. Autom. Contr.*, pp. 1–16, 2024.
- [8] S. Bodmer, L. Vogel, S. Muntwiler, A. Hansson, T. Bodewig, J. Wahlen, M. N. Zeilinger, and A. Carron, "Optimization-based system identification and moving horizon estimation using low-cost sensors for a miniature car-like robot," *arXiv preprint arXiv:2404.08362*, 2024.
- [9] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The robotarium: A remotely accessible swarm robotics research testbed," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 1699–1706, 2017.
- [10] B. Chalaki, L. E. Beaver, A. I. Mahbub, H. Bang, and A. A. Malikopoulos, "A research and educational robotic testbed for real-time control of emerging mobility systems: From theory to scaled experiments [applications of control]," *IEEE Control Systems Magazine*, vol. 42, no. 6, pp. 20–34, 2022.
- [11] N. Buckman, A. Hansen, S. Karaman, and D. Rus, "Evaluating autonomous urban perception and planning in a 1/10th scale minicity," *Sensors*, vol. 22, p. 6793, 2022.
- [12] N. Hyldmar, Y. He, and A. Prorok, "A fleet of miniature cars for experiments in cooperative driving," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 3238–3244, 2019.
- [13] M. Kloock, P. Scheffe, J. Maczajewski, A. Kampmann, A. Mokhtarian, S. Kowalewski, and B. Alrifae, "Cyber-physical mobility lab: An open-source platform for networked and autonomous vehicles," in *Proc. Europ. Contr. Conf. (ECC)*, pp. 1937–1944, 2021.
- [14] S. Wilson, R. Gamos, M. Sheely, M. Lin, K. Dover, R. Gevorkyan, M. Haberland, A. Bertozzi, and S. Berman, "Pheeno, a versatile swarm robotic research and education platform," *IEEE Robot. Autom. Lett.*, vol. 1, no. 2, pp. 884–891, 2016.
- [15] J. Dong, Q. Xu, J. Wang, C. Yang, M. Cai, C. Chen, Y. Liu, J. Wang, and K. Li, "Mixed cloud control testbed: Validating vehicle-road-cloud integration via mixed digital twin," *IEEE Trans. Intell. Veh.*, vol. 8, no. 4, pp. 2723–2736, 2023.
- [16] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale RC cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.
- [17] A. Carron and E. Franco, "Receding horizon control of a two-agent system with competitive objectives," in *Proc. Am. Control Conf. (ACC)*, pp. 2533–2538, 2013.
- [18] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, D. Hoehener, S.-Y. Liu, M. Novitzky, I. F. Okuyama, J. Papis, G. Rosman, V. Varricchio, H.-C. Wang, D. Yershov, H. Zhao, M. Benjamin, C. Carr, M. Zuber, S. Karaman, E. Frazzoli, D. Del Vecchio, D. Rus, J. How, J. Leonard, and A. Censi, "Duckietown: An open, inexpensive and flexible platform for autonomy education and research," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 1497–1504, 2017.
- [19] M. O'Kelly, V. Sukhil, H. Abbas, J. Harkins, C. Kao, Y. V. Pant, R. Mangharam, D. Agarwal, M. Behl, P. Burgio, and M. Bertogna, "F1/10: An open-source autonomous cyber-physical platform," *arXiv preprint arXiv:1901.08567*, 2019.
- [20] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.
- [21] A. Raji, A. Liniger, A. Giove, A. Toschi, N. Musiu, D. Morra, M. Verucchi, D. Caporale, and M. Bertogna, "Motion Planning and Control for Multi Vehicle Autonomous Racing at High Speeds," *Proc. Conf. Intell. Transp. Syst. (ITSC)*, pp. 2775–2782, 2022.
- [22] H. G. Bock, "Recent advances in parameter identification techniques for ODE," pp. 95–121, Birkhäuser Boston, 1983.
- [23] J. Valluru, P. Lakhmani, S. C. Patwardhan, and L. T. Biegler, "Development of moving window state and parameter estimators under maximum likelihood and bayesian frameworks," *Journal of Process Control*, vol. 60, pp. 48–67, 2017.
- [24] L. Simpson, A. Ghezzi, J. Asprion, and M. Diehl, "An efficient method for the joint estimation of system parameters and noise covariances for linear time-variant systems," in *Proc. IEEE Conf. Decis. Control (CDC)*, pp. 4524–4529, 2023.
- [25] L. A. McGee and S. F. Schmidt, "Discovery of the kalman filter as a practical tool for aerospace and industry," *NASA Technical Memorandum*, p. 21, 1985.
- [26] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*. Santa Barbara, CA, USA: Nob Hill Publish., LLC, 2nd ed., 2020. 3rd printing.
- [27] J. D. Schiller, S. Muntwiler, J. Köhler, M. N. Zeilinger, and M. A. Müller, "A Lyapunov function for robust stability of moving horizon estimation," *IEEE Trans. Autom. Control*, 2023.
- [28] A. Alessandri and M. Awawdeh, "Moving-horizon estimation with guaranteed robustness for discrete-time linear systems and measurements subject to outliers," *Automatica*, vol. 67, pp. 85–93, 2016.
- [29] M. Gharbi, F. Bayer, and C. Ebenbauer, "Proximity moving horizon estimation for discrete-time nonlinear systems," *IEEE Control Systems Letters*, vol. 5, pp. 2090–2095, 2021.
- [30] J. D. Schiller and M. A. Müller, "A moving horizon state and parameter estimation scheme with guaranteed robust convergence," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 6759–6764, 2023.
- [31] S. Muntwiler, J. Köhler, and M. N. Zeilinger, "MHE under parametric uncertainty – Robust state estimation without informative data," *arXiv preprint arXiv:2312.14049*, 2023.
- [32] J. Brembeck, "Nonlinear constrained moving horizon estimation applied to vehicle position estimation," *Sensors*, vol. 19, no. 10, 2019.
- [33] M. Zanon, J. V. Frasch, and M. Diehl, "Nonlinear moving horizon estimation for combined state and friction coefficient estimation in autonomous driving," in *Proc. Europ. Contr. Conf. (ECC)*, pp. 4130–4135, 2013.
- [34] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE access*, vol. 8, pp. 58443–58469, 2020.
- [35] L. P. Fröhlich, C. Küttel, E. Arcari, L. Hewing, M. N. Zeilinger, and A. Carron, "Contextual tuning of model predictive control for autonomous racing," in *Proc. Int. Conf. Intell. Robots Syst. (IROS)*, pp. 10555–10562, 2022.
- [36] J. Köhler, M. A. Müller, and F. Allgöwer, "Robust output feedback model predictive control using online estimation bounds," *arXiv preprint arXiv:2105.03427*, 2021.
- [37] L. Numerow, A. Zanelli, A. Carron, and M. N. Zeilinger, "Inherently robust suboptimal mpc for autonomous racing with anytime feasible sqp," *IEEE RAL*, vol. 9, no. 7, pp. 6616–6623, 2024.
- [38] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, pp. 1–36, 2019.
- [39] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2005.
- [40] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. v. Duijkeren, A. Zanelli, B. Novosel'nik, T. Albin, R. Quirynen, and M. Diehl, "acados—a modular open-source framework for fast embedded optimal control," *Math. Program. Comput.*, vol. 14, no. 1, pp. 147–183, 2022.