"From model-based to data-driven control: applications to self-balancing robots"

Soenen, Aurélien

ABSTRACT

Data-driven control has emerged as a promising paradigm for dynamical systems, enabling the construction of feedback controllers directly from historical data without the need for system model identification. This paper investigates the practical application of data-driven control by comparing it with traditional model-based control methods on a real balancing robot, the Pololu Balboa 32U4. The study confirms the unstable nature of the robot through mathematical modeling and underscores the need for active control to stabilize its motion. Model-based control methods, including LQR, Bessel, and ITAE pole placement, ensure stability across simulations and experiments. However, a noticeable steady-state error in reference tracking indicates potential for improvement, particularly in navigation-oriented applications. Data-driven control shows promise for self-balancing applications, producing promising results in simulations. In experimental contexts, these methods are effective in scenarios where data is gathered using slower controllers that manage natural disturbances. However, their effectiveness decreases with faster controllers due to inaccuracies in sensor data measurement and significant system noise, which compromise the integrity of the data-driven results. The research identifies avenues for future exploration to enhance stability and the quality of data-driven results. These include implementing sensor fusion, using model-based Kalman filters, or exploring alternative non-model-based filtering techniques. These strategies aim to advance the data-driven approach fo...

CITE THIS VERSION

UCLouvain

epl

École polytechnique de Louvain

From model-based to data-driven control

Applications to self-balancing robots

Author: **Aurélien Soenen**
Supervisor: **Gianluca Bianchin**
Readers: **Paul Fisette, Raphaël Jungers**
Academic year 2023–2024
Master [120] in Electro-mechanical Engineering

Acknowledgements

Completing a master's thesis is primarily seen as an individual process, but this work cannot be carried out without proper support. This is the reason why I want to express my gratitude to all the people who helped me to come to the end of this unique adventure.

Firstly, I would like to extend my deepest appreciation to Gianluca Bianchin for supervising this study and making this master's thesis possible. I thank him for his involvement, continuous support, and encouragement throughout this endeavor. His expertise in this field, along with his scientific publications, has been invaluable. I also appreciate his availability, constructive feedback, and attentive listening during the numerous presentations. Without his guidance and persistent assistance, this dissertation would not have been possible. Our discussions consistently set me on the right path and provided motivation during challenging times. I am profoundly grateful for the time he devoted to me and wish him the best in his professional life.

Additionally, I would like to thank the members of the jury—Prof. Paul Fisette and Prof. Raphaël Jungers—for dedicating their time to review this document and providing critical feedback.

Although not directly involved in my study activities, the support of other engineering students greatly contributed to the completion of this thesis. Their sharing of tips and knowledge was instrumental in writing and researching.

Lastly, I warmly thank my fantastic parents, my sister, and my brother for their unconditional support from the beginning to the end of my studies. I could not have accomplished this without them.

Abstract

Data-driven control has emerged as a promising paradigm for dynamical systems, enabling the construction of feedback controllers directly from historical data without the need for system model identification. This paper investigates the practical application of data-driven control by comparing it with traditional model-based control methods on a real balancing robot, the Pololu Balboa 32U4.

The study confirms the unstable nature of the robot through mathematical modeling and underscores the need for active control to stabilize its motion. Model-based control methods, including LQR, Bessel, and ITAE pole placement, ensure stability across simulations and experiments. However, a noticeable steady-state error in reference tracking indicates potential for improvement, particularly in navigation-oriented applications.

Data-driven control shows promise for self-balancing applications, producing promising results in simulations. In experimental contexts, these methods are effective in scenarios where data is gathered using slower controllers that manage natural disturbances. However, their effectiveness decreases with faster controllers due to inaccuracies in sensor data measurement and significant system noise, which compromise the integrity of the data-driven results.

The research identifies avenues for future exploration to enhance stability and the quality of data-driven results. These include implementing sensor fusion, using model-based Kalman filters, or exploring alternative non-model-based filtering techniques. These strategies aim to advance the data-driven approach for the Balboa 32U4, paving the way for enhanced stability and performance in real-world applications.

Nomenclature

Abbreviations

DC	Direct Current
HPCB	High Power Carbon Brushes
ICR	Input Capture Register
IMU	Inertial Measurement Unit
ITAE	Integral of Time-weighted Absolute Error
LQR	Linear-Quadratic Regulator
OCR	Output Compare Register
PWM	Pulse Width Modulation

Alpha numeric symbols

ℓ	Distance between pendulum and wheel centre of mass	[m]
$\mathcal{L}$	Pole locations in control system design	[-]
$\overline{u}_a$	Mean absolute voltage	[V]
A	System matrix defining the state dynamics	[-]
a	Coefficient linking torque to voltage	[V/Nm]
B	Input matrix defining the control input impact	[-]
b	Coefficient linking rotational speed to voltage	[V/(rad/s)]
C	Output matrix defining the measurement output	[-]
c	Voltage offset due to friction	[V]

C_{em}	Electromagnetic torque of DC motor	[Nm]
C_r	Resistive torque of DC motor	[Nm]
d	Depth of pendulum	[m]
E	Matrix of rotational and translational dynamics parameters	[-]
F	Applied force / Force vector influencing system dynamics	[N] / [-]
G	Gravity vector affecting the system	[-]
g	Gravitational acceleration	[m/s^2]
GR	Gear ratio from motor to wheel	[-]
h	Height of pendulum	[m]
I	Moment of inertia about the rotation axis	[kgm^2]
i_a	Armature current of DC motor	[A]
J	Cost function for LQR optimization	[-]
K	Feedback gain matrix	[-]
$k\phi$	Torque constant of DC motor	[Nm/A]
K_ν	Viscous damping coefficient of DC motor	[Nm/(rad/s)]
L_a	Armature inductance of DC motor	[H]
M	Transformation matrix used in control algorithms	[-]
m	Mass	[kg]
MRE	Mean relative error	[-]
n	System noise vector / Number of state variables	[rad] / [-]
N_u	Precompensator gain for control input	[-]
N_x	Precompensator gain for state variable	[-]
Q	State weighting matrix in LQR	[-]
$r(t)$	Reference trajectory for wheel angle	[rad]

R	Control input weighting matrix in LQR	[-]
r	Wheel radius	[m]
R_a	Armature resistance of DC motor	[Ω]
r_i	Internal wheel radius	[m]
T	Total number of measurements	[-]
t_s	Settling time of the system response	[-]
U_0	Control input measurements	[-]
u_a	Armature voltage of DC motor	[V]
$v(t)$	External input of the dynamical system	[-]
$x(t)$	State vector of the dynamical system	[-]
x	Horizontal position	[m]
X_0	State vector measurements	[-]
X_1	State vector measurements	[-]
$y(t)$	Output variable of the dynamical system	[-]
y	Vertical position	[m]

Greek letters

Δt	Sampling interval for measurements	[s]
ϵ_φ	Mean wheel position error	[rad]
Λ	Diagonal matrix containing system poles	[-]
λ	Individual pole locations	[-]
σ_θ	Standard deviation of angular position	[rad]
σ_φ	Standard deviation of wheel position	[rad]
σ_n	Standard deviation of system noise	[rad]
σ_v	Standard deviation of external input	[rad]

τ_0	Total torque exerted by the two motors	[Nm]
τ_m	Torque provided by each motor	[Nm]
θ	Pendulum angle relative to the vertical	[rad]
φ	Wheel rotational angle	[rad]

Operators

$\_^{\dagger}$	Moore-Penrose pseudoinverse
$\dot{\_}$	Derivative with respect to time

Subscripts

$\cdot_0$	Initial component
$\cdot_f$	Friction-related aspects
$\cdot_N$	Normal component
$\cdot_p$	Pendulum component
$\cdot_r$	Related to reference tracking
$\cdot_T$	Tangential component
$\cdot_w$	Wheel component
$\cdot_x$	Related to the horizontal axis / state vector
$\cdot_y$	Related to the vertical axis
$\cdot_{Bessel}$	Relating to Bessel pole placement method
$\cdot_{DD}$	Relating to data-driven method
$\cdot_{dt}$	Discrete-time variable
$\cdot_{ITAE}$	Relating to ITAE pole placement method
$\cdot_{LQR}$	Relating to Linear-Quadratic Regulator control
$\cdot_{MB}$	Relating to model-based method
$\cdot_{ss}$	Steady-state value

Contents

Chapter 1

Introduction

1.1 Context

The field of robotics has undergone transformative growth, extending its influence far beyond its origins in industrial automation. From the assembly lines of the 20th century to the complex social interactions of the 21st century, robotics has become integral to everyday environments such as homes, offices, and even outdoor spaces, assuming roles ranging from security and assistance to entertainment and companionship. This broad integration demands the development of specialized mechanisms tailored to enhance robotic performance across a variety of applications. Achieving stable and reliable operation in these diverse contexts is essential, especially when robots are designed to interact dynamically within public and domestic environments.

Intelligent mobile robots represent a transformative domain within robotics. These robots evolve from simple sensor-based entities to sophisticated agents that can perform tasks such as recognizing features, detecting patterns, learning from experiences, localizing themselves, and constructing maps for navigation. This field of study disrupts the prevailing trend of increasing specialization in science by necessitating a comprehensive approach that fuses various disciplines.

This thesis explores a category of intelligent mobile robots, specifically those utilizing a two-wheeled, inverted pendulum system. Despite their operational challenges, those robots offer substantial advantages over more traditional mobile platforms. Their ability to pivot in place and compact design makes them extremely maneuverable and well-suited for navigating in confined spaces, making them particularly valuable in crowded or intricately designed indoor environments.

However, mastering the complex dynamics and inherent non-linearities of the wheeled inverted pendulum system is a significant control challenge. The academic community has developed numerous models and control systems aimed at stabilizing these mechanisms, each with varying degrees of success and application. In response to the observed limitations of traditional model-based control strategies, there is a growing trend toward adopting data-driven control approaches. This emerging methodology does not rely on a predetermined model and offers a flexible framework capable of adapting to the robot's dynamic environment. Data-driven control promises to enhance the robot's ability to handle unexpected scenarios and complex dynamics, marking a significant step forward in the evolution of robotic control systems.

1.2 Literature review

Two-wheeled balancing robots present a sophisticated challenge in control theory, characterized by their nonlinear dynamics. These robots serve as a bridge between theoretical exploration and practical application, making them a focal point in control system research. Historical milestones in this area include the development by Ha and Yuta in 1994 of a mobile balancing robot capable of autonomous high-speed navigation while maintaining balance [1]. In 2001, Dean Kamen's introduction of the Segway Human Transporter marked a significant advancement by employing a self-balancing mechanism for personal transportation with zero emissions [2]. Further, in 2002, Grasser et al. presented JOE, a scaled-down prototype that used additional weights to simulate the dynamics of a human rider [3]. Subsequent research has introduced various models and control strategies, each tailored to meet specific operational challenges.

This section provides a comprehensive overview of established models and control systems for two-wheeled balancing robots. Drawing primarily on the review by Chan et al. (2013) [4], it highlights significant findings from the literature that are relevant and potentially applicable to this project. Furthermore, it identifies opportunities for further research, specifically focusing on the advancement of data-driven control strategies.

1.2.1 Balancing robot model

Various methods can derive the non-linear dynamic model of a two-wheeled robot, including the Euler-Lagrange equation, Newton's laws of motion, and Kane's method. The Newton method, although comprehensive, often involves computationally inefficient calculations of irrelevant forces. In contrast, the Euler-Lagrange method, despite its rigorous formulation, suffers from computational inefficiency due to complex solutions for the Lagrange multipliers. Kane's method, which is based on partial velocities, offers a more streamlined approach as it avoids unnecessary force calculations and the use of multipliers.

The simplest model confines the robot to straight-line motion in a vertical plane, characterized by two degrees of freedom: longitudinal displacement and tilt of the pendulum. Examples of 1-dimensional models include those developed using the Euler-Lagrange equation by Ha and Yuta (1994)[1], Åkesson et al. (2006)[5], and others [6, 7], as well as those using Newton's Law by Ooi (2003)[8] and Li et al. (2007)[9].

Certain models include turning mechanisms by introducing two additional states associated with the yaw angle, allowing the robot's position to be tracked in Cartesian coordinates instead of just longitudinal axis. The non-linear dynamics of such models are often simplified by ignoring variations in moment of inertia on the yaw axis as the tilt angle changes, and by decoupling the yaw motions from other states. Notable research in this area includes work by Grasser et al. (2002) [3], Takei et al. (2009) [10], and others [11, 12].

Some models consider changes in the moment of inertia around the yaw axis with varying tilt angles. Pathak et al. (2005) [13] utilized a moment of inertia tensor matrix to fully account for these dynamics. Simplifications using Kane's method have also been proposed, where minor rotational effects are ignored, but significant changes due to the shifting center of mass are included [14, 15, 16].

In addition to physical models, black-box models, which do not detail the internal dynamics but aim to accurately replicate the system behavior, have been explored. Alarfaj and Kantor (2010) [17] and Jahaya et al. (2011) [18] have developed such data-driven models in the discrete-time domain. A particularly sophisticated approach is the Takagi–Sugeno fuzzy model, utilized by Qin et al. (2011) [19]. This model employs a linear combination of linear state-space models, each weighted by a membership function, to approximate the nonlinear dynamics effectively. This method offers a powerful means of handling non-linearities by blending multiple linear models according to fuzzy logic rules.

The various research efforts documented in the literature have been organized and summarized in Figure 1.1. This table categorizes each study based on the type of dynamic equations used and whether the models consider longitudinal only or both longitudinal and yaw motions. The classification further distinguishes between decoupled and coupled approaches for models that incorporate yaw motion, with some using simplified methods or a full inertia tensor to describe the dynamics.

Figure 1.1: Summary of balancing robot models.

Dynamic equations	**Longitudinal only**	**Longitudinal and yaw motion**		
		Decoupled	Coupled	
			Simplified	Inertia tensor
Newton's equations of motion	Ooi (2003) [8] Li et al. (2007) [9]	Grasser et al. (2002) [3] Takei et al. (2009) [10] Wu et al. (2011) [20]		
Euler-Lagrange equations	Ha & Yuta (1994) [1] Åkesson et al. (2006) [5] Lien et al. (2006) [6] Han et al. (2008) [7]	Tsai & Hu (2007) [11] Hu & Tsai (2008) [12]		Pathak et al. (2005) [13]
Kane's method			Kim et al. (2005) [14] Nawawi et al. (2007) [15] Muhammad et al. (2011) [16]	
Black box model				
Data-driven model	Alarfaj & Kantor (2010) [17] Jahaya et al. (2011) [18]			
Takagi-Sugeno fuzzy models	Qin et al. (2011) [19]			

1.2.2 Control systems

Linear control methods

Linear state feedback control using a linearized model of the self-balancing robot is prevalent due to its simplicity and effectiveness, especially for small tilt angles where the system behaves more linearly. Various linearization techniques have been employed, including Jacobian linearization, which has been widely adopted for its computational simplicity [3, 5, 10, 14].

Once the model is linearized, designing a linear controller to track a reference state involves simply determining an appropriate feedback gain matrix. One common method is pole placement, which allows for the adjustment of system response characteristics like rise time and overshoot, ensuring satisfactory performance [3, 15, 7, 21, 22].

Linear Quadratic Regulation (LQR) is another popular method for designing optimal control systems. LQR aims to minimize a cost function, providing a balance between system performance and energy usage, and has been applied in numerous studies [1, 14, 5, 10, 17].

Comparisons between LQR and pole placement reveal mixed results. Lien et al. (2006) [6], Ghani et al. (2010) [23] and Wu and Zhang (2011) [24] found advantages in pole placement for certain performance metrics. In contrast, Ooi (2003) [8] highlights LQR's superior robustness. These divergent findings emphasize the need to tailor control strategies to specific system requirements.

Additionally, the integration of a Kalman filter with an LQR results in a Linear Quadratic Gaussian (LQG) controller, which optimizes performance under the assumption of Gaussian noise and disturbances [25]. In the realm of advanced linear controllers, Hu and Tsai (2008) [12] and Ruan and Chen (2010) [26] developed H_∞ controllers, known for their robustness against model inaccuracies and disturbances. Similarly, Kanada et al. (2011) [27] designed an H_2 controller and demonstrated its effectiveness in comparison to LQR.

PID controllers remain a staple in control engineering due to their simplicity and the ease of tuning their three parameters. They are often calibrated through trial and error [28] or using conventional tuning methods such as Ziegler–Nichols [29].

Non-linear control methods

Non-linear control strategies address the dynamic complexities of two-wheeled robots, each tailored to specific system requirements. Fuzzy control adapts to system state variations with less mathematical complexity, using fuzzy logic to modify control actions effectively. This adaptability is advantageous in systems with multiple states, such as those described by Chiu and Peng (2006) [30] and further refined in the Takagi–Sugeno fuzzy model by Qin et al. (2011) [19].

Sliding mode control stands out for its robustness against disturbances and model uncertainties. By forcing system states towards a desired sliding surface, it simplifies dynamic management, highlighted in studies by Wu et al. (2011) [20] and complemented by the proportional integral sliding mode approach of Nawawi et al. (2006) [31], which reduces overshoot and enhances tracking accuracy compared to traditional methods. Backstepping, for example, is particularly useful for sequential control strategies, effectively integrating with other controllers to enhance overall system stability as demonstrated by Thao et al. (2010) [32].

Gain scheduling dynamically adjusts control gains based on system states, maintaining performance across varying dynamics, a strategy illustrated by Wu, Ma, and Wang (2010) [33]. Lyapunov-based control strategies provide theoretical stability assurances, offering a robust alternative to traditional control methods. Kausar et al. (2011) [34] demonstrated its effectiveness, showcasing superior performance over conventional LQR methods in maintaining stability. Reinforcement learning, as implemented by Sun and Gan (2011)[35], enables controllers to learn optimal strategies through trial-and-error interactions with the environment, optimizing performance without a predefined model.

Artificial neural networks offer powerful adaptive control capabilities, learning to handle non-linearities effectively within dynamic environments. This approach is particularly beneficial in systems where model uncertainties are prevalent, as shown in implementations by Tanaka et al. (2010) [36] and Ruan and Chen (2010b) [37], where neural networks manage complex control tasks in real-time. Similarly, adaptive control strategies, such as those developed by Li et al. (2010) [38], optimize control parameters in real-time to match changing system conditions, enhancing the precision and robustness of the control system. Finally, model predictive control, developed by Azimi and Koofigar (2013) [39], forecasts future system states to calculate optimal control actions, enhancing decision-making under constraints.

The diagram in Figure 1.2 organizes the range of control methods applied to two-wheeled balancing robots, as documented in the literature. It categorizes

the methods into linear, non-linear strategies, listing key research papers for each approach. Notably, some methods are often combined to leverage complementary strengths and achieve better overall system performance. This summary serves both as a quick reference and a guide to the diverse control techniques utilized in robotic applications.

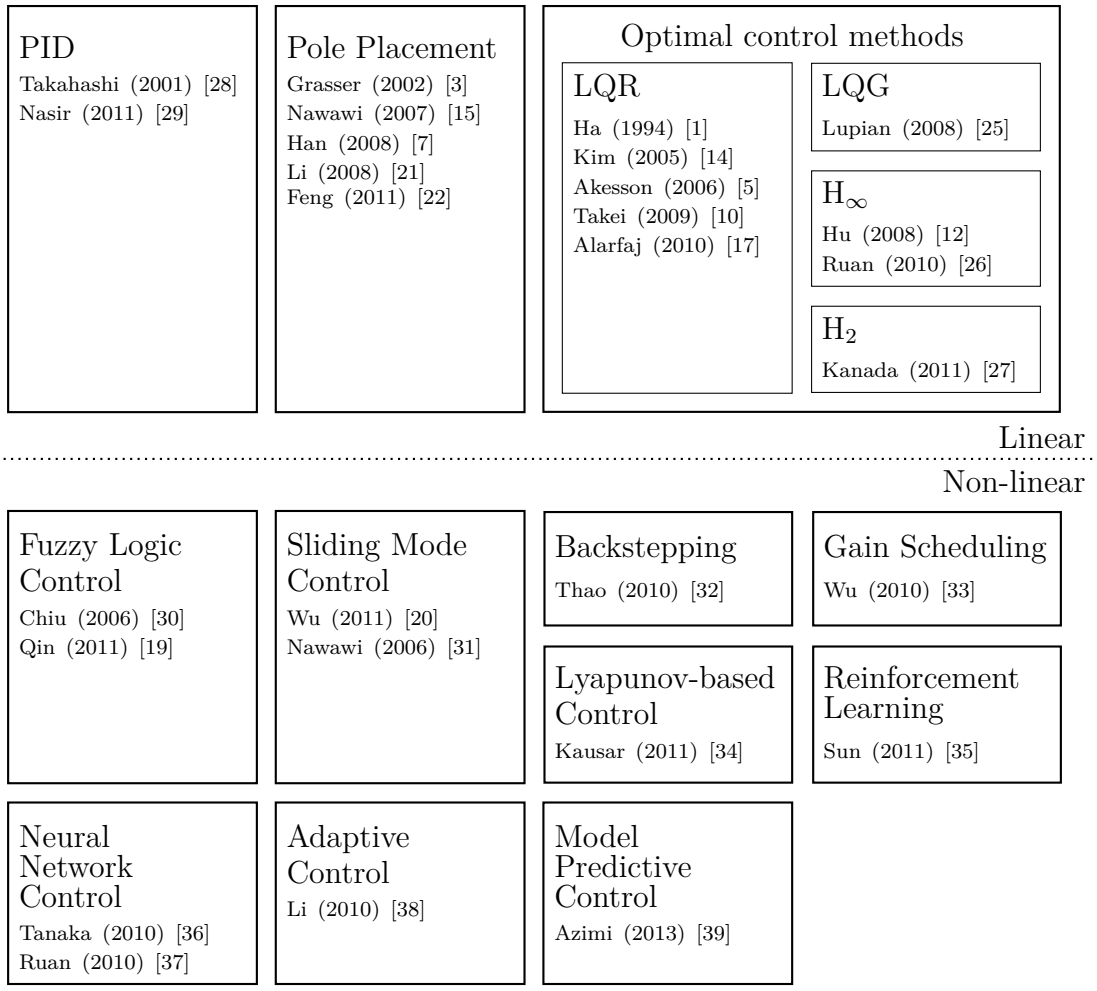

Figure 1.2: Diagram of control methods sorted by linearity and non-linearity.

1.2.3 Data-driven control

Data-driven control methods facilitate the development of feedback controllers directly from operational data, eliminating the necessity for detailed system models. This advantage is crucial in situations where deriving accurate first-principle models is arduous or leads to unreliable parameter estimations, as noted by Krishnan and Dorfler (2021) and Dorfler et al. (2023) [40, 41]. These methods not only streamline controller synthesis by utilizing historical data but also avoid the propagation of uncertainties inherent in model parameters during control design.

Recent developments have introduced various methodologies for synthesizing data-driven controllers without system model identification. Static feedback control and linear quadratic regulators were notably explored by Maupong and Rapisarda (2016) [42], and De Persis and Tesi (2020) [43], respectively. Furthermore, model predictive control and minimum-energy control laws have been significantly developed by Coulson et al. (2019) and Baggio et al. (2019) [44, 45].

Particularly noteworthy is the contribution of Bianchin (2023), who advanced the understanding of data-driven pole placement and eigenstructure assignment [46]. His research demonstrates the feasibility of setting closed-loop eigenvalues precisely at predetermined locations directly from data. This research trajectory is complemented by the work of Mukherjee and Sayak (2022), who focused on the placement of closed-loop poles within specified linear matrix inequality regions, thus enhancing the precision and adaptability of data-driven control systems [47].

1.3 Motivations and objectives

The literature review highlights significant advancements in understanding data-driven control, an innovative paradigm that enables the creation of feedback controllers directly from historical data without preliminary system modeling. This method is poised to outperform traditional model-based control approaches. While numerous numerical comparisons between traditional and data-driven control methods have been documented, there remains a significant gap in their experimental application, particularly regarding the data-driven pole placement method.

This thesis shifts focus towards empirical experimentation to deepen our understanding and application of theoretical concepts in real-world scenarios. The motivation is to test both model-based and data-driven control methods on a real balancing robot, exploring how data-driven control strategies can be applied to complex systems.

1.4 Structure of the manuscript

In order to achieve the objectives, the manuscripts are divided into 3 chapters:

- **Chapter 2 : Self-balancing system**

  This chapter revisits the mathematical model of the self-balancing robot, beginning with a comprehensive overview of the robot's components and functionalities, which is elaborated further in Appendix A. The chapter progresses to detail the dynamics of the self-balancing system, focusing on the derivation of equations that describe its behavior and refining the model of motor torque. A stability analysis is also conducted to emphasize the necessity of a robust control system. Overall, this chapter establishes a foundational understanding of the robot's dynamics, setting the stage for deeper analyses in subsequent sections.

- **Chapter 3 : Model-based control**

  This chapter investigates model-based control strategies, focusing on the Linear-Quadratic Regulator (LQR), and pole placement methods such as Bessel and ITAE. It evaluates their efficacy through numerical simulations that analyze system dynamics and performance metrics to identify the most optimal feedback gains. The chapter also assesses how system noise influences control performance. Furthermore, it details practical experiments conducted on the Balboa 32U4 robot to validate these control strategies and to compare their experimental feedback performances against simulation outcomes.

- **Chapter 4 : Data-driven control**

  This chapter delves into deriving feedback gains using data-driven approaches. It introduces methods such as data-driven pole placement and system identification. The chapter details numerical simulations to evaluate these methods under varying conditions, such as changes in data acquisition periods and chosen pole locations. Additionally, practical experiments conducted on the Balboa 32U4 robot are described to validate these data-driven strategies, focusing on the influence of different controller responses and the effectiveness of various filtering techniques to enhance outcomes.

Chapter 2

Self-balancing system

2.1 Introduction

This chapter revisits the mathematical model of the self-balancing robot, specifically the Pololu Balboa 32U4. It begins with an overview of the robot, highlighting its key components and functionalities, with a more detailed description provided in Appendix A.

The main focus is on refining the mathematical model, particularly the representation of motor torque to ensure it accurately reflects the relationship between the input voltage signals and the robot's response.

The chapter concludes with a stability analysis of the refined model, underscoring the challenges of maintaining balance and the necessity for a robust control system. This analysis sets the foundation for understanding the robot's dynamics, paving the way for further explorations in chapter 3 and 4.

2.2 System description

2.2.1 Pololu Balboa 32U4

As illustrated in Figure 2.1, the Pololu Balboa 32U4 is a compact and versatile two-wheeled balancing robot, primarily designed for educational and research applications. This robot integrates a powerful microcontroller with motor drivers and a suite of onboard sensors, such as encoders, accelerometers, and gyroscopes. It operates on six AA batteries, which supply a nominal voltage of 7.2V.

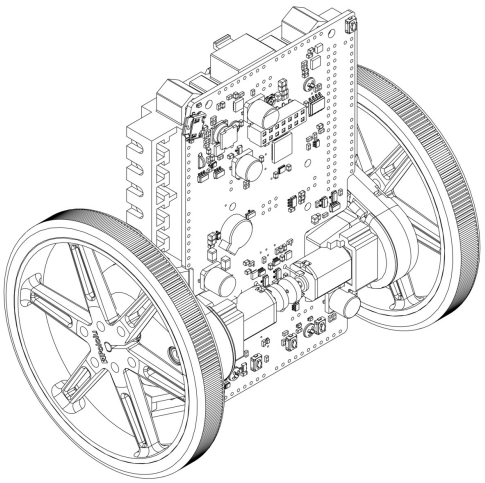

Figure 2.1: Self-balancing robot: Pololu Balboa 32U4.

The Balboa 32U4 boasts sophisticated motor control and sensor feedback systems, establishing it as an exemplary platform for control theory education. Its integral sensors facilitate precise real-time tracking of position and orientation, which are critical for the development and validation of diverse control algorithms.

Utilizing the Pololu Balboa 32U4 enriches hands-on learning, bridging theoretical principles with practical implementation. It allows for the exploration of both model-based and data-driven control strategies, demonstrating the transition from traditional methods to modern techniques in robotics.

For an exhaustive analysis of the Pololu Balboa 32U4, including its mechanical, electrical, actuator, sensor, and motor specifications, please see Appendix A. This chapter will focus solely on aspects pertinent to the mathematical modeling of the robot.

2.2.2 Self-balancing system

The system discussed consists of a rigid pendulum mounted on the axle of a wheel, driven by a torque along a horizontal track (Figure 2.2). The connection between the pendulum and the wheel features a revolute joint modeled as ideal—without friction or clearance. This configuration allows for two degrees of freedom: the wheel's horizontal position x and the pendulum's tilt angle θ relative to the vertical, where $\theta = 0$ denotes an unstable equilibrium. With only the driving torque τ_0 as a control input, the system qualifies as under-actuated.

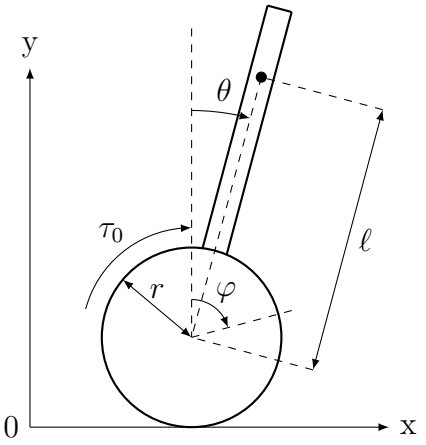

Figure 2.2: Model of the self-balancing system.

In a right-handed coordinate system where positive angles correspond to counter-clockwise rotations when viewed from the positive end of an axis toward the origin, the position of the pendulum's center of mass can be described by the coordinates:

$$x_p = x_w + \ell \sin\theta, \tag{2.1}$$

$$y_p = y_w + \ell \cos\theta. \tag{2.2}$$

Given that the wheel rolls without slipping, the relationship between the wheel's horizontal displacement and its rotation can be mathematically expressed. The translational velocity of the wheel's center, $\dot{x}_w$, is directly proportional to its angular velocity $\dot{\varphi}$, as dictated by the wheel's radius r:

$$\dot{x}_w = r\dot{\varphi}. \tag{2.3}$$

Integrating Equation (2.3) determines the horizontal position x_w of the wheel:

$$x_w = x_0 + r\varphi. \tag{2.4}$$

Here, x_0 represents the initial horizontal position and φ the cumulative rotation angle of the wheel. The vertical position y_w of the wheel's center remains constant, equal to the radius of the wheel, since the wheel maintains contact with the ground:

$$y_w = r. \tag{2.5}$$

2.2.3 List of parameters

The parameters of the robot, including their descriptions and values, are detailed in section 2.3. These parameters are organized in Table 2.1 for geometrical aspects and Table 2.2 for motor-related specifications.

Table 2.1: List of model parameters.

Symbols	Description	Values	Units
m_w	Mass of the wheels	0.0042	kg
m_p	Mass of the pendulum	0.316	kg
r	External radius of the wheels	0.040	m
r_i	Internal radius of the wheels	0.031	m
h	Height of the pendulum	0.109	m
d	Depth of the pendulum	0.022	m
ℓ	Distance between pendulum and wheel centre of mass	0.023	m
I_p	Moment of inertia of the pendulum[1]	444.43×10^{-6}	kg m^2
I_w	Moment of inertia of the two wheels[2]	26.89×10^{-6}	kg m^2

Table 2.2: List of motor parameters.

Symbols	Description	Values	Units
$u_{a,max}$	Battery nominal voltage	7.2	V
R_a	Armature resistance	4	Ω
$k\phi$	Motor constant	0.132	Nm/A
K_ν	Viscous damping coefficient	$1.91 \cdot 10^{-3}$	Nm/(rad/s)
C_r	Resistance constant torque	$14.85 \cdot 10^{-3}$	Nm

[1]The moment of inertia of the pendulum is based on the Huygens-Steiner theorem: $I_p = (h^2 + d^2)m_p/12 + m_r\ell^2$

[2]The moment of inertia of the two wheels is calculated using the formula for a hollow cylinder: $I_w = (r^2 + r_i^2)m_w/2$

2.3 Mathematical model

2.3.1 Mechanical model

Mechanical systems can be modeled using two primary methodologies: Newtonian mechanics, which focuses on forces within the system, and Lagrangian mechanics, which is based on energy principles. The key distinction lies in their approach to constraints: Newtonian mechanics models each component separately and includes explicit forces to maintain constraints, while the Lagrangian method systematically eliminates constraints from the dynamics through an energy-based framework. Although both yield equivalent results, they differ in the insights they provide into the mechanics involved. This manuscript will employ Newtonian mechanics for deriving the mathematical model of the self-balancing system, given its straightforward application to individual body dynamics.

The initial step in analyzing the system involves creating a free-body diagram, as shown in Figure 2.3. In the diagram, F_N denotes the normal force and F_T the tangential force at the point of contact, with F_f representing the friction force. The forces F_x and F_y are the reaction forces at the revolute joint, which cannot transmit torque. Consequently, the driving torque τ_0 is applied exclusively to the wheel.

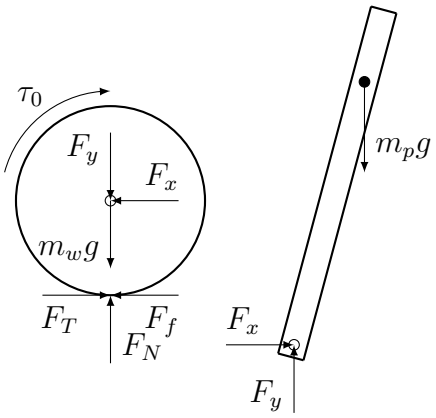

Figure 2.3: Free-body diagram.

The motion equations for the wheel and pendulum are detailed as follows:

$$m_w \ddot{x}_w = F_T - F_x - F_f, \tag{2.6}$$

$$0 = F_N - F_y - m_w g, \tag{2.7}$$

$$I_w \ddot{\varphi} = -rF_T + rF_f + \tau_0. \tag{2.8}$$

For the pendulum, the dynamic equilibrium is governed by:

$$m_p \ddot{x}_p = F_x, \tag{2.9}$$
$$m_p \ddot{y}_p = F_y - m_p g, \tag{2.10}$$
$$I_p \ddot{\theta} = -F_x \ell \cos\theta + F_y \ell \sin\theta. \tag{2.11}$$

Incorporating Equations (2.9) and (2.10) into Equation (2.11), and substituting $\ddot{x}_p$ and $\ddot{y}_p$ using the second derivatives of Equations (2.1) and (2.2), the resulting dynamic equation for the pendulum becomes:

$$I_p \ddot{\theta} + m_p \ell^2 \ddot{\theta} + m_p r \ell \ddot{\varphi} - m_p g \ell \sin\theta = 0. \tag{2.12}$$

Similarly, substituting Equations (2.6) and (2.7) into Equation (2.8) and applying the time derivatives of Equations (2.1), (2.2), and (2.3) gives:

$$I_w \ddot{\varphi} + r^2 (m_w + m_p) \ddot{\varphi} + m_p r \ell \ddot{\theta} \cos\theta - m_p r \ell \dot{\theta}^2 \sin\theta = \tau_0. \tag{2.13}$$

2.3.2 Electrical model

To effectively control the self-balancing robot, the system utilizes two small DC motors. Incorporating these motors into the mathematical model of the system necessitates a detailed description of their characteristics, particularly how the applied voltage translates into driving torque.

A schematic of the DC motor's electrical circuit is depicted in Figure 2.4. In this diagram, L_a and R_a represent the inductance and armature resistance, respectively. The armature current and input voltage are denoted by i_a and u_a, while τ_m represents the motor torque applied to the robot's wheel.

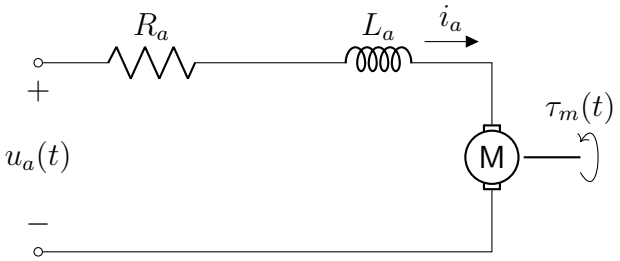

Figure 2.4: Schematic diagram of a DC motor.

Applying Kirchhoff's laws to the DC motor's electrical circuit allows for the derivation of its operational equations:

$$u_a = R_a \cdot i_a + L_a \frac{di_a}{dt} + k\phi \cdot \dot{\varphi}, \tag{2.14}$$

where $k\phi$ is the motor constant, and $\dot{\varphi}$ represents the motor's rotational speed. Given that the electrical time constant is significantly smaller than the mechanical time constant, the inductance is negligible, simplifying the equation to

$$u_a = R_a \cdot i_a + k\phi \cdot \dot{\varphi}. \tag{2.15}$$

The expression for the electromagnetic torque generated by a DC motor, which is directly proportional to the armature current, can be succinctly formulated as:

$$C_{em} = k\phi \cdot i_a. \tag{2.16}$$

The motor torque applied to the wheel isn't solely determined by the electromagnetic torque due to influences such as viscous damping and friction. The formula for motor torque, taking these factors into account, is:

$$\tau_m = C_{em} - K_\nu \cdot \dot{\varphi} - C_r \cdot \operatorname{sgn}(\dot{\varphi}), \tag{2.17}$$

where K_ν represents the viscous damping coefficient, and C_r denotes the constant resistive torque from friction and other mechanical resistances opposing motion.

Finally, by substituting Equations (2.16) and (2.17) into Equation (2.15), and considering that the self-balancing system is operated by two motors (hence, $\tau_0 = 2 \cdot \tau_m$), the input control voltage can be determined as follows:

$$u_a = \frac{R_a}{k\phi}\left(\frac{\tau_0}{2} + K_\nu \cdot \dot{\varphi} + C_r \cdot \operatorname{sgn}(\dot{\varphi})\right) + k\phi \cdot \dot{\varphi}. \tag{2.18}$$

To streamline the formulation, new constants a, b, and c are defined to directly relate the input voltage to the torque and motor speed:

$$u_a = a \cdot \tau_0 + b \cdot \dot{\varphi} + c \cdot \operatorname{sgn}(\dot{\varphi}), \tag{2.19}$$

where $a = \frac{R_a}{2k\phi}$, $b = \frac{R_a K_\nu}{k\phi} + k\phi$, and $c = \frac{R_a C_r}{k\phi}$.

2.3.3 Linearization

Initially, Equation (2.12) and Equation (2.13) are linearized around the operating point ($\theta = 0$), assuming $\cos\theta \approx 1$ and $\sin\theta \approx \theta$ for small angle approximations. These simplifications allow for the transformation of the system dynamics into a second-order linear matrix form:

$$E \begin{bmatrix} \ddot{\varphi}(t) \\ \ddot{\theta}(t) \end{bmatrix} + G\theta(t) = F\tau_0(t), \tag{2.20}$$

where

$$E = \begin{bmatrix} I_w + r^2(m_w + m_p) & m_p r\ell \\ m_p r\ell & I_p + m_p \ell^2 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ -m_p g\ell \end{bmatrix}, \quad F = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

In this system, E represents the inertia matrix, G the gravitational vector and F the control vector of the self-balancing robot.

For the self-balancing robot, the following state vector is employed to stabilize the system, ensuring that $\lim_{t\to\infty} x(t) = 0$:

$$x(t) = \begin{bmatrix} \varphi(t) \\ \theta(t) \\ \dot{\varphi}(t) \\ \dot{\theta}(t) \end{bmatrix}.$$

The state-space equations can be expressed as follows:

$$\dot{x} = Ax + B\tau_0, \tag{2.21}$$
$$y = Cx, \tag{2.22}$$

where

$$A = \left[\begin{array}{c:c:cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hdashline 0 & -E^{-1}F & 0 & 0 \\ 0 & & 0 & 0 \end{array}\right], \quad B = \left[\begin{array}{c} 0 \\ 0 \\ \hdashline E^{-1}G \end{array}\right], \quad C = \begin{bmatrix} 1, 0, 0, 0 \end{bmatrix}.$$

Lastly, the electrical model is included, and a refined state-space representation is provided as follows:

$$\dot{x} = Ax + B' \cdot \left[u_a - b\dot{\varphi} - c \cdot \operatorname{sgn}(\dot{\varphi})\right], \tag{2.23}$$

with constant $B' = \frac{1}{a}B$.
Note, hereafter, the matrix B' will be denoted simply as B.

Parameterization

Utilizing the parameters listed in Table 2.1 and 2.2, the continuous model-based system, as described by Equation (2.23), is characterized with the matrices A and B defined as:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -69.4 & 0 & 0 \\ 0 & 150 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 273 \\ -130 \end{bmatrix}.$$

To align with the sensor's sampling rate of $\Delta t = 10$ ms, the system has been discretized using MATLAB's `c2d(sys, Ts)` function, which transforms continuous systems into their discrete counterparts based on the specified sampling time `Ts`. The resulting discrete state-space representation utilizes matrices labeled with subscript dt.

$$A_{dt} = \begin{bmatrix} 1 & -0.0035 & 0.01 & -1.2e-05 \\ 0 & 1 & 0 & 0.01 \\ 0 & -0.7 & 1 & -0.0035 \\ 0 & 1.5 & 0 & 1 \end{bmatrix}, \quad B_{dt} = \begin{bmatrix} 0.014 \\ -0.0065 \\ 2.7 \\ -1.3 \end{bmatrix}.$$

The stability characteristics of the robot are closely tied to the properties of the system matrix A and its discrete counterpart A_{dt}.

$$\text{eig}(A) = \{0, 0, 12, -12\}, \quad \text{eig}(A_{dt}) = \{1, 1, 1.1, 0.88\}.$$

The presence of zero and positive eigenvalues in matrix A and eigenvalues greater than and equal to one for A_{dt} indicate the system's inherent instability at the equilibrium point. This instability is manifested by the robot's tendency to fall from an upright position in the absence of active control. Therefore, to maintain stability, it is essential to design and implement a control system that adjusts the eigenvalues to reside entirely within the left-half of the complex plane for continuous-time systems, and inside the unit circle for discrete-time systems.

This manuscript will propose and evaluate two approaches to achieve this stabilization: the model-based control approach and the data-driven control approach.

Chapter 3

Model-based control

3.1 Introduction

This chapter explores the implementation of model-based control strategies. It focuses on determining the feedback gain through various established control strategies. The discussion begins with the Linear-Quadratic Regulator (LQR), which utilizes a cost function to optimize the feedback gain. This method is juxtaposed with pole placement methods such as Bessel and ITAE, which control pole locations to specifically influence system dynamics and response characteristics.

Further sections conduct numerical simulations to compare these methods concerning system dynamics and responses. This includes an analysis of performance metrics associated with different feedback gains and the selection of an optimal feedback gain for each control strategy. Additionally, the impact of system noise on control performance is examined.

The chapter concludes with practical experiments conducted on the Balboa 32U4. These experiments serve to validate the model-based control strategies and provide a comparative analysis of the performances metrics compared to the simulation results.

3.2 Controller design

With the inherent instability of the modeled system established, the next phase is to develop a control strategy that not only stabilizes the system but also enables the robot to adhere to a predefined trajectory $r(t)$, which dictates the desired position over time. This strategy employs two feedback gains, K_r and K_x, that continuously adjust the input voltage based on real-time measurements. K_r is designed to ensure

that the robot follows the desired trajectory accurately, whereas K_x stabilizes the system based on the state measurements $x(t)$. Figure 3.1 depicts this process through a block diagram illustrating the implemented feedback mechanisms.

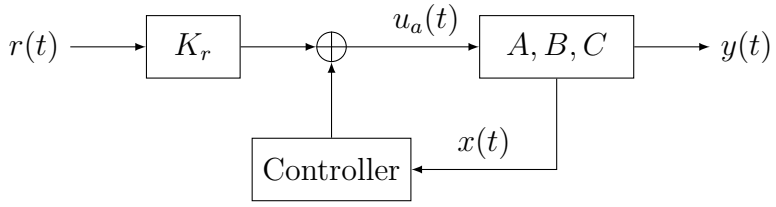

Figure 3.1: Block diagram representing the closed-loop control system.

The configuration of K_r is designed to ensure that the steady-state output corresponds precisely to the reference trajectory ($y_{ss} = r_{ss}$), and to achieve zero rate of change in the state at steady state ($\dot{x}_{ss} = 0$). The steady-state control input u_{ss} and the state x_{ss} are defined by $u_{ss} := N_u r_{ss}$ and $x_{ss} := N_x r_{ss}$, respectively. The feedback gain K_r along with the precompensator gains N_x and N_u are derived from the following equations:

$$K_r = N_u + K_x \cdot N_x,$$
$$\begin{bmatrix} N_x \\ N_u \end{bmatrix} = \begin{bmatrix} A & B \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ I \end{bmatrix}.$$

For the balancing robot's model, these parameters simplify to:

$$N_x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad N_u = 0.$$

The controller operates based on measurements of state variables and consists of three primary components. The main component applies a feedback gain K_x directly to the state vector $x(t)$. The additional components are specifically designed to address and simplify certain motor dynamics as outlined in Equation (2.23). The first of these components mitigates the impact of motor speed on the voltage, represented by the coefficient b. The second component compensates the voltage offset due to friction, which depends on the direction of wheel speed and is characterized by the coefficient c. Consequently, the full control input $u_a(t)$ can be simplified as:

$$u_a(t) = -K_x \cdot \begin{bmatrix} \varphi(t) - r(t) \\ \theta(t) \\ \dot{\varphi}(t) \\ \dot{\theta}(t) \end{bmatrix} + b \cdot \varphi(t) + c \cdot \text{sign}(\dot{\varphi}(t)). \tag{3.1}$$

K_x can be designed using various parameterization methods. The Linear-Quadratic Regulator (LQR) approach utilizes cost matrices Q and R to balance the trade-off between minimizing state variable deviations and controlling input magnitudes. Alternatively, Bessel and ITAE (Integral of Time-weighted Absolute Error) pole placement methods focus on settling time to shape the system's transient response.

3.2.1 Linear-Quadratic Regulator (LQR)

The Linear-Quadratic Regulator (LQR) compute an optimal feedback gains K_{LQR} by minimizing a quadratic cost function $J(x, u_a)$ considering the state vector $x(t)$ and the control input $u_a(t)$. This function balances the trade-off between desired system performance, quantified by matrix Q, and the control effort, quantified by matrix R [48]:

$$J(x, u_a) = \int_0^\infty (x^T(t)Qx(t) + u_a^T(t)Ru_a(t))\, dt. \tag{3.2}$$

For a balancing robot, the careful selection of the positive definite matrices Q and R is essential. The predominant variable for control feedback involves the cost term associated with the wheel position and input variable. Altering this ratio results in varied system behavior. Consequently, Q is fixed while R is varied, treating it as a free parameter to be tuned. This configuration is expressed as follows:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad R = \text{Free parameter}.$$

By employing computational tools such as MATLAB's `lqr(A, B, Q, R)` function, the optimal feedback gain K_{LQR} is determined for various cost factor R that will affect control strategy:

Table 3.1: Feedback gains K_{LQR} for varying cost factors R.

R	K_{LQR}
2	[-0.5 -7.1 -0.16 -0.63]
4	[-0.25 -5.3 -0.091 -0.46]
8	[-0.13 -4.2 -0.055 -0.36]
16	[-0.062 -3.5 -0.035 -0.3]

These results illustrate the impact of varying R on the controller's aggressiveness, confirming that lower values lead to a more assertive control strategy.

3.2.2 Pole placement methods

The feedback gain K_x can be computed by strategically fixing the poles at predetermined locations. This section delves into two specialized pole placement methods: Bessel and ITAE (Integral of Time-weighted Absolute Error).

Bessel pole locations

Originating from Bessel filters, Bessel pole placement method is known for its near-linear phase response with minimal group delay, beneficial for systems needing rapid yet smooth stabilization. The method strategically places the poles to minimize overshoot and oscillations while maintaining fast responsiveness [48].

The poles for a system with four state variables and a specified settling time t_s can be calculated using the formula:

$$\mathcal{L}_{Bessel} = \frac{1}{t_s} \cdot [-4.016 \pm 5.072i, -5.528 \pm 1.655i]. \tag{3.3}$$

This formulation derives from the Bessel polynomial, expressed as:

$$y_n(x) = \sum_{k=0}^{n} \frac{(n+k)!}{(n-k)!k!} \left(\frac{x}{2}\right)^k. \tag{3.4}$$

The polynomial coefficients determine the specific locations of the poles to optimize the system's dynamic response.

The poles identified are subsequently utilized to calculate the feedback gain K_{Bessel}. This computation is performed using algorithms such as the Bass-Gura formula or through MATLAB's `place(A, B, p)` function, where `p` denotes the specified poles.

ITAE pole locations

Conversely, the ITAE (Integral of Time-weighted Absolute Error) pole placement method focuses on minimizing the integral of time-weighted absolute error, making it highly suitable for systems where steady-state accuracy and minimal long-term error are priorities. This approach adjusts the pole locations to optimize the time-weighted performance, thereby improving both the transient and steady-state behavior of the system [48].
For a given settling time t_s with four state variables, the ITAE pole locations could be defined as:

$$\mathcal{L}_{ITAE} = \frac{1}{t_s} \cdot [-4.236 \pm 12.617i, -6.254 \pm 4.139i]. \tag{3.5}$$

These poles are derived by matching the n^{th} polynomial that minimizes the integral of time multiplied by the absolute value of the error (ITAE), defined as:

$$J_{ITAE} = \int_0^\infty t\,|e(t)|\,dt, \tag{3.6}$$

where $e(t)$ is the error signal in response to a step function.

Similar to the Bessel method, feedback gains K_{ITAE} are derived using MATLAB's `place(A, B, p)` function to ensure the system meets the timing and performance specifications.

Comparative analysis

Both methods tailor the system's response by specifying how quickly and smoothly the balancing robot should return to equilibrium. To provide a comparative analysis, feedback gains computed for various settling times using both Bessel and ITAE methods are summarized in the Table 3.2.

Table 3.2: Feedback gains K_{Bessel} and K_{ITAE} for varying settling time t_s.

t_s	K_{Bessel}				K_{ITAE}			
0.8 s	[-0.11	-3.4	-0.045	-0.28]	[-0.76	-6.8	-0.17	-0.55]
0.9 s	[-0.067	-2.9	-0.031	-0.23]	[-0.48	-5.4	-0.12	-0.42]
1 s	[-0.044	-2.5	-0.023	-0.2]	[-0.31	-4.4	-0.085	-0.34]
1.1 s	[-0.03	-2.3	-0.017	-0.17]	[-0.21	-3.8	-0.064	-0.28]

These results highlight how each method influences the controller's responsiveness, with the Bessel approach typically producing a less aggressive response compared to ITAE for the same settling time.

3.3 Numerical simulation

To analyze the model-based control, simulations of the system dynamics are performed depending on the input variable. In these simulations a random Gaussian noise $n(t) \in \mathbb{R}^n$ is incorporated to emulate measurement and process noise, reflecting the real-world inaccuracies in sensor readings and system behavior. This noise is defined with a standard deviation σ_n, representing the uncertainty in the measurements. The system dynamics are governed by the following equation:

$$u_a(t) = -K_x \cdot \begin{bmatrix} \varphi(t) - r(t) \\ \theta(t) \\ \dot{\varphi}(t) \\ \dot{\theta}(t) \end{bmatrix},$$

$$x(t + \Delta t) = A_{dt} \cdot x(t) + B_{dt} \cdot u_a(t) + n(t). \tag{3.7}$$

During simulations, the coefficient b and the offset term c from Equation (2.23) are not included because these components are directly compensated within the feedback loop during actual experimental implementations. However, they are still considered when calculating the actual voltage required.

3.3.1 System dynamics

The first testing concerns the system dynamics in different circumstances. In these tests the Bessel and ITAE methods are fixed with a settling time of one second and a costs factor $R = 8$ for LQR. Moreover, no noise is implemented $n(t) = 0$.

The initial response of the robot's control system to an angular displacement of 10° is assessed to understand how different control strategies react. Figure 3.2 illustrates the unique response characteristics of each method. The Bessel method, which prioritizes minimizing overshoot, demonstrates a slower response characteristic of its conservative design. In contrast, the ITAE method exhibits a quicker response but with significant oscillations. The LQR method, tailored through its cost matrices Q and R, aims to achieve a balance between state performance and control effort, resulting in a moderated response. Additionally, the voltage response $u_a(t)$ shows a notable offset of 0.4 V.

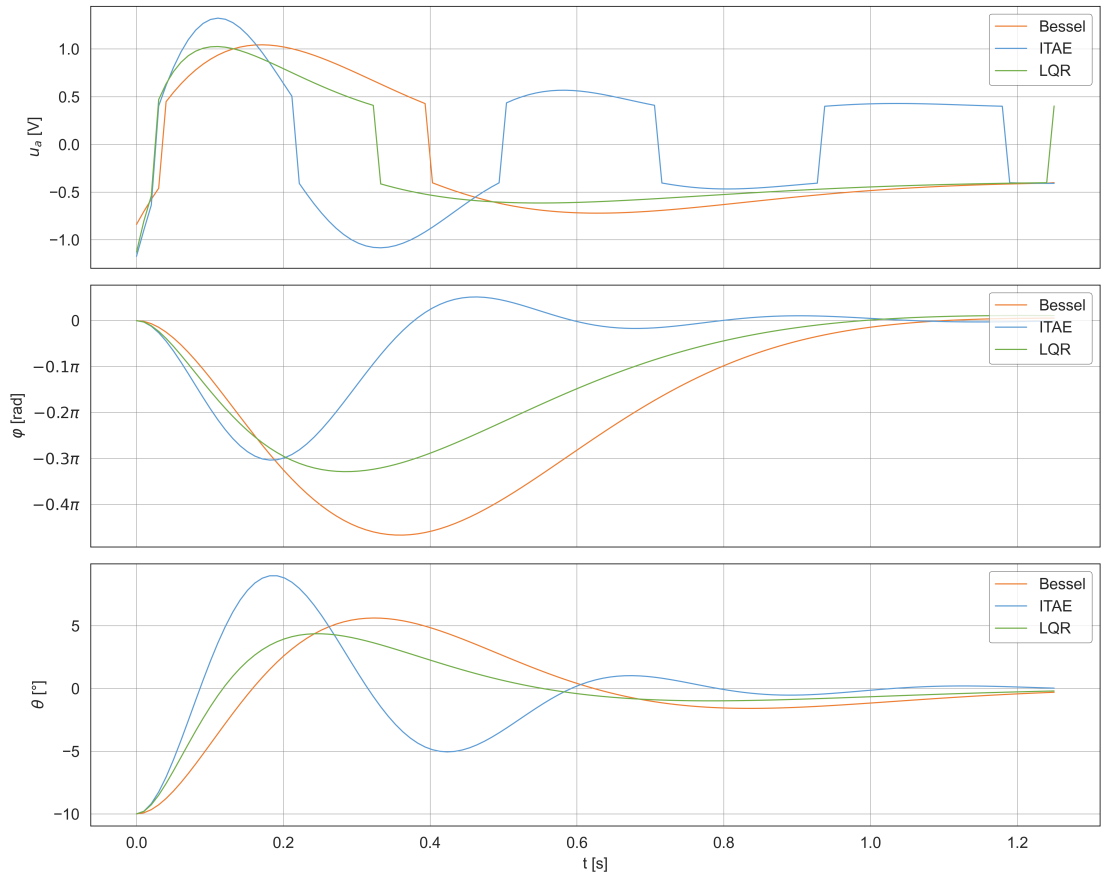

Figure 3.2: Simulation of voltage response $u_a(t)$, wheel rotation angle $\varphi(t)$, and angular position $\theta(t)$ following an initial perturbation where $\theta(0) = 10°$ under Bessel, ITAE and LQR control methods.

The initial response analysis provides insights into the characteristics of each control method. However, reproducing these simulations in experimental settings is challenging due to the difficulty in obtaining measurements with identical initial states. To bridge the gap between theoretical simulations and practical experiments, a structured reference tracking scenario is implemented on the Balboa 32U4 robot. It includes:

- Initial 10-second stabilization,
- 5-second rightward movement followed by 10-second stabilization,
- 5-second leftward movement concluded with 10-second stabilization.

Additionally, to simulate the realistic operational environment, Gaussian noise with a standard deviation of $\sigma_n = 0.5°$ is introduced into the system. The effects of this noise on the control outcomes will be analyzed in subsequent sections. The reference tracking performance, depicted in Figure 3.3, visually compares the different control strategies over the sequence.

Despite the inherent noise introduced to simulate disturbances, each method adheres closely to the intended path, showcasing the effectiveness of the feedback gains. Oscillations are evident due to the noise, yet they are well within controllable limits.

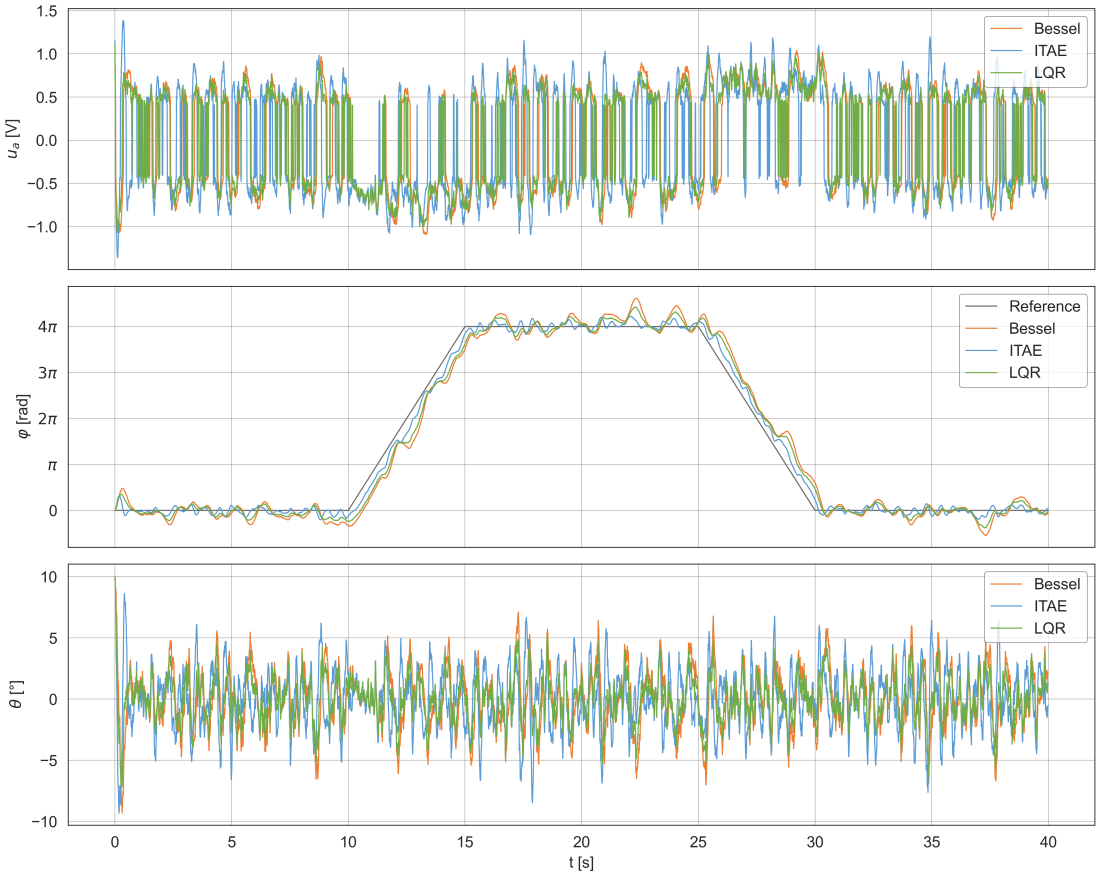

Figure 3.3: Simulation of voltage response $u_a(t)$, wheel rotation angle $\varphi(t)$, and angular position $\theta(t)$ over a 40-second test period under Bessel, ITAE and LQR control methods.

3.3.2 Feedback performance

For each control method, it is preferable to find the optimal feedback to stabilize the balancing robot. To assess the effectiveness of each feedback gain, specific performance metrics are utilized. The primary goal is to minimize these metrics according to the distinctive parameters of each method: settling time for Bessel and ITAE methods, and cost factor R for LQR.

Performance metrics

Performance metrics are introduced to evaluate the robot's operational efficiency in terms of energy consumption, adherence to the reference trajectory, and oscillatory behavior around the equilibrium point over a specified duration t_1. The metrics are detailed in Table 3.3.

Table 3.3: Performance metrics for evaluating feedback control effectiveness.

Symbol	Formula	Parameter description
$\overline{u}_a$	$\frac{1}{t_1}\int_0^{t_1} \lvert u_a(t)\rvert \, dt$	Mean absolute voltage: Energy consumption over time.
ϵ_φ	$\frac{1}{t_1}\int_0^{t_1} \varphi(t) - r(t) \, dt$	Mean wheel position error: Deviation from reference trajectory.
σ_φ	$\sqrt{\frac{1}{t_1}\int_0^{t_1} (\varphi(t) - r(t) - \epsilon_\varphi^2 \, dt}$	Standard deviation of wheel position: Variability around the reference trajectory.
σ_θ	$\sqrt{\frac{1}{t_1}\int_0^{t_1} \theta(t)^2 \, dt}$	Standard deviation of angular position: Oscillations around the equilibrium position.

Parameters analysis

To determine the optimal feedback gains, simulations incorporating Gaussian noise with a standard deviation of $\sigma_n = 0.5°$ were performed. These simulations varied the parameters of each feedback control method and calculated corresponding performance metrics. Specifically, Bessel method's settling time was adjusted between 0.3 and 1.2 seconds, ITAE method's settling time ranged from 0.6 to 1.9 seconds, and LQR method's cost factor R spanned from 0.25 to 15.

Across all three methods presented in Figure 3.4 and Figure 3.5, a consistent trend emerges. Lower control parameters, indicative of more aggressive control actions, necessitate higher energy consumption $\overline{u}_a$ to swiftly correct any deviations. This results in less deviation from the reference signal σ_φ, demonstrating a trade-off between energy efficiency and system stability. Conversely, with softer control settings, σ_φ increases, indicating that less stringent control actions may lead to larger oscillations and potentially higher overall energy consumption as the system requires more corrections to maintain equilibrium.

For the Bessel and ITAE methods, the performance metrics $\overline{u}_a$, ϵ_φ, and σ_θ each reach a minimum at different settling times t_s. Similarly, for the LQR method, optimal performance in terms of $\overline{u}_a$ and ϵ_φ is achieved at distinct cost function values R.

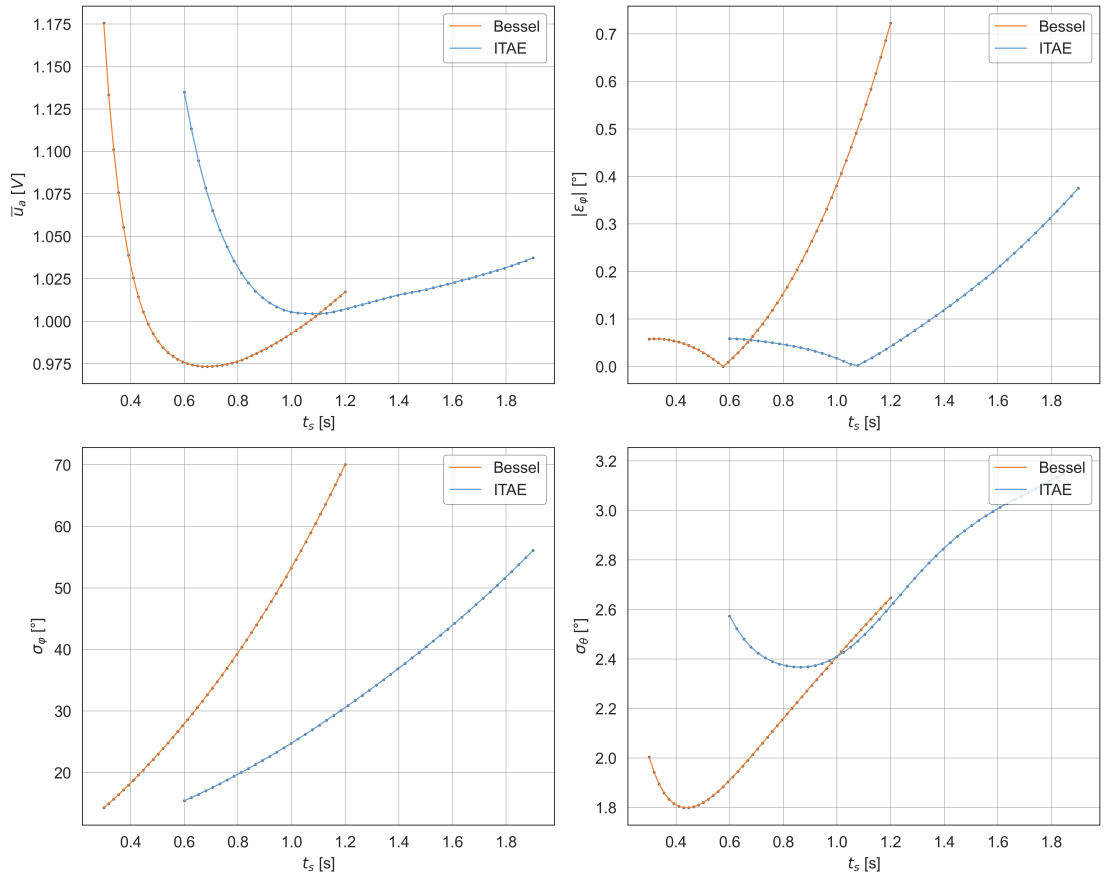

Figure 3.4: Simulation of the performance metrics $\overline{u}_a$, $|\epsilon_\varphi|$, σ_φ, and σ_θ as functions of the settling time t_s under Bessel and ITAE methods.

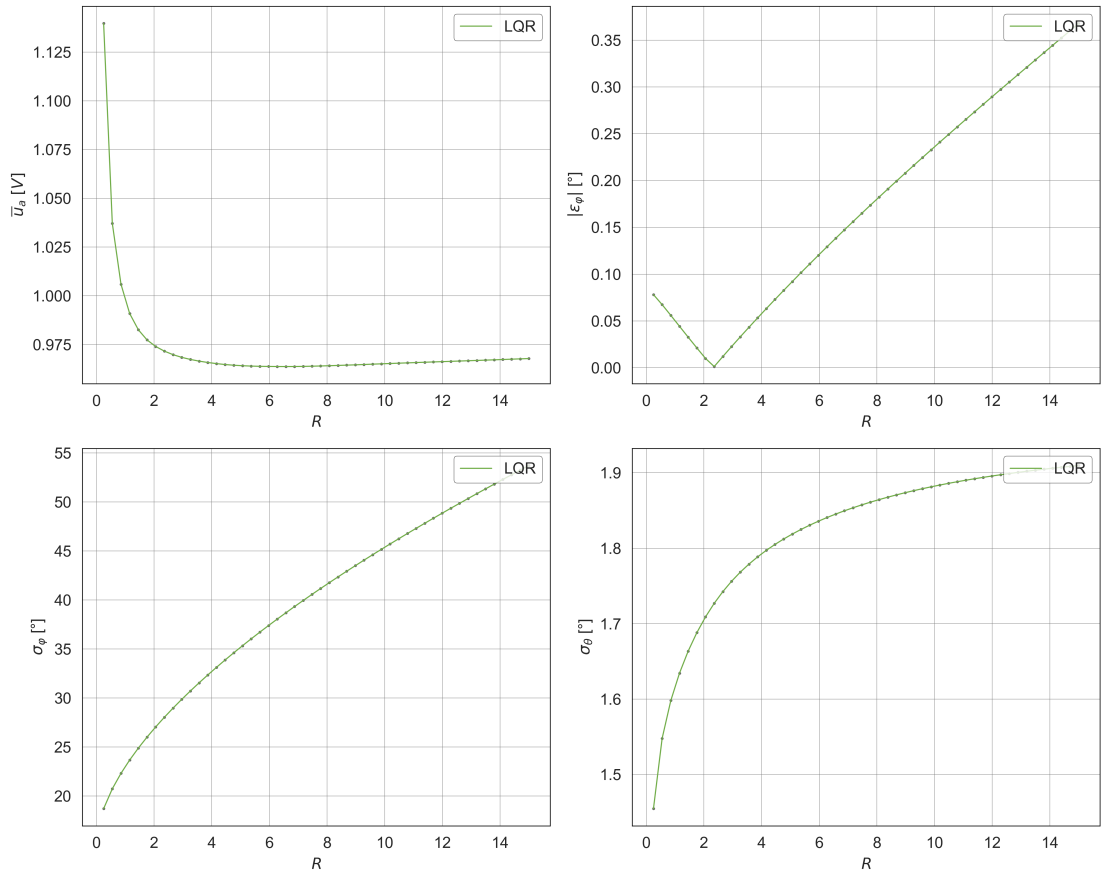

Figure 3.5: Simulation of the performance metrics $\overline{u}_a$, $|\epsilon_\varphi|$, σ_φ, and σ_θ as functions of the cost factor R under LQR control method.

To determine the optimal feedback gains, performance metrics such as ϵ_φ, σ_φ, and σ_θ are plotted against the mean absolute control effort $\overline{u}_a$. Figure 3.6 illustrates the relationship between these metrics and how they evolve as control parameters like settling time and cost factor are adjusted. The directional arrows on the graphs indicate how increasing the control parameters impacts the performance metrics. The objective is to minimize these metrics, identifying the optimal feedback gains by selecting configurations that position the plotted points closest to the origin.

Although all three methods yield similar graphical representations, the LQR method shows the lowest energy consumption, followed by the Bessel method, with ITAE consuming the most. Conversely, in terms of minimizing σ_φ, Bessel and ITAE methods perform better at equivalent energy levels. However, when considering σ_θ, LQR shows the least oscillation, followed by Bessel and then ITAE.

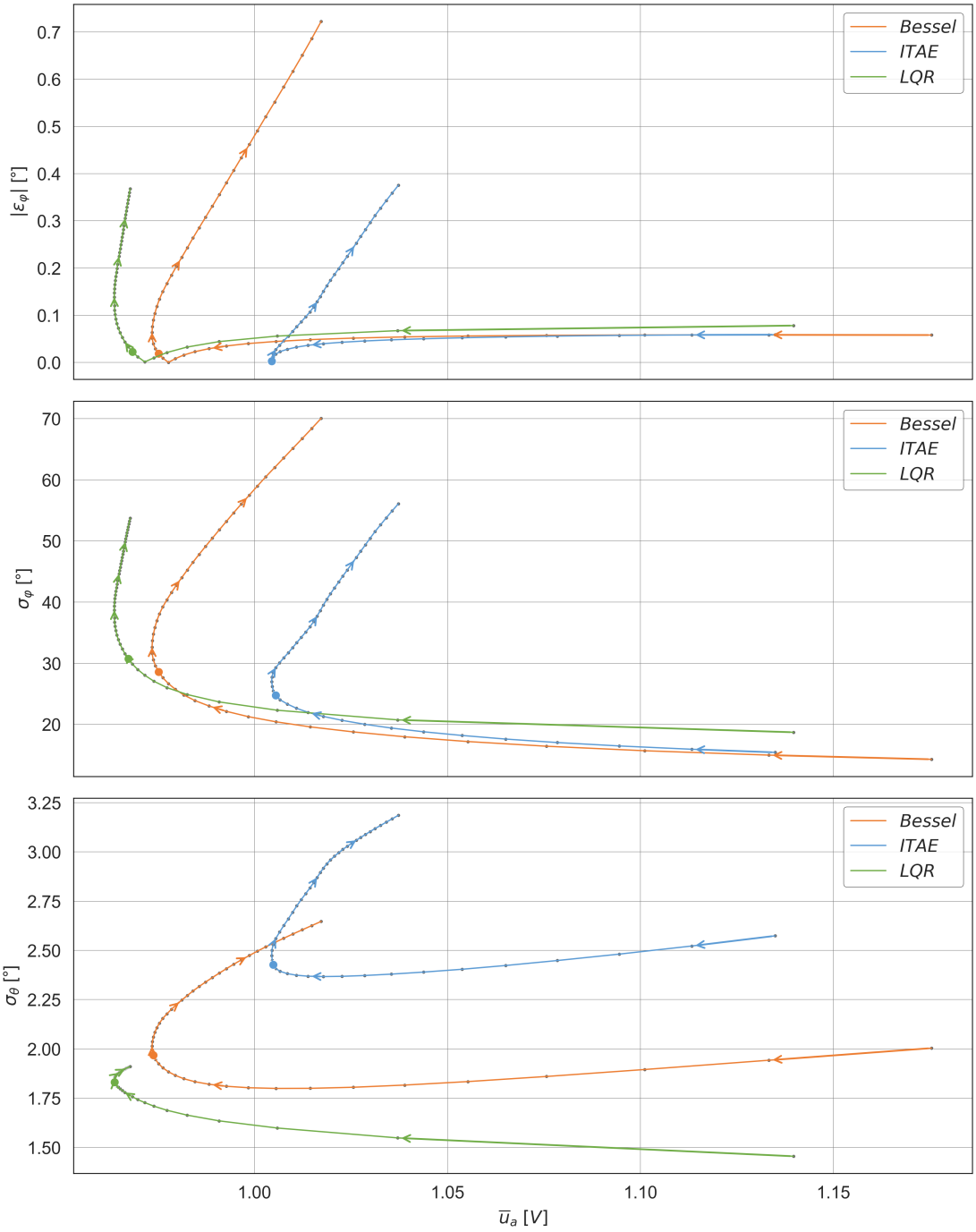

Figure 3.6: Simulation of the performance metrics $|\epsilon_\varphi|$, σ_φ, and σ_θ as functions of the control effort $\overline{u}_a$ under Bessel, ITAE, and LQR control methods.

Table 3.4 lists the optimal control parameters, marked by larger dots of corresponding colors in Figure 3.6. The optimal configurations are a settling time of 0.6 seconds for the Bessel method, 1 second for the ITAE method, and a cost factor of 3 for the LQR method.

Table 3.4: Comparison of optimal control parameters for Bessel, ITAE, and LQR methods based on different performance metrics trade-off.

Trade-off	Bessel	ITAE	LQR
$\overline{u}_a - \lvert\epsilon_\varphi\rvert$	0.61 s	1.1 s	Cost factor 3
$\overline{u}_a - \sigma_\varphi$	0.61 s	1 s	Cost factor 3.3
$\overline{u}_a - \sigma_\theta$	0.65 s	1 s	Cost factor 5.7

3.3.3 System noise

To compute the system dynamics, noise is integrated into the simulation as defined in Equation (3.7). This noise simulates real-world inaccuracies in sensor readings and system behavior, affecting performance metrics. In previous simulations, this was fixed with a standard deviation of $\sigma_n = 0.5°$. The next set of simulations varies this noise level and assesses the performance metrics using the optimal feedback gains previously determined.

Figure 3.7 illustrates that performance metrics deteriorate as noise levels increase. Notably, the ITAE method exhibits the smallest increase in wheel position error, suggesting its robustness against noise, though it requires a higher voltage input to maintain control. In contrast, the LQR method shows the least increase in angular position oscillation, indicating efficient energy usage and reduced tendency for overcorrection. However, ITAE displays the greatest increase, likely due to its predisposition to overcompensate in response to disturbances.

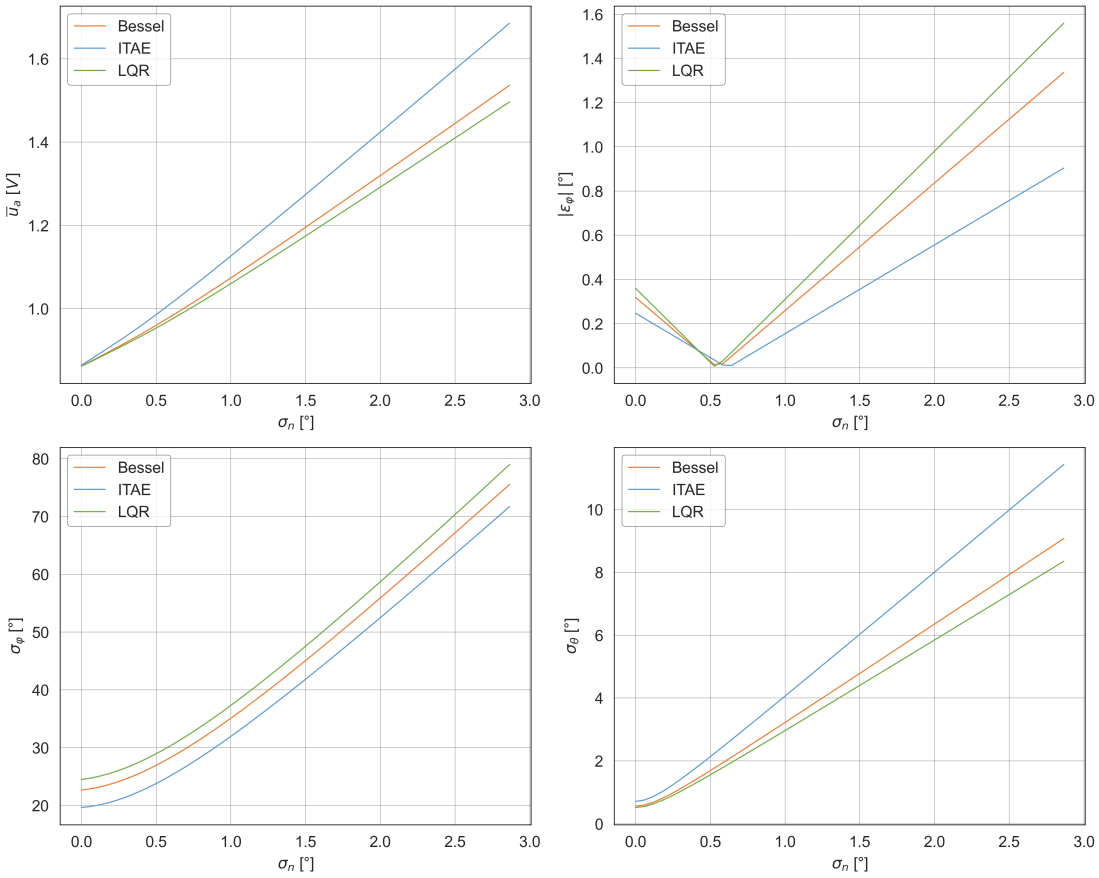

Figure 3.7: Simulation of increasing noise levels on the performance metrics $\overline{u}_a$, $|\epsilon_\varphi|$, σ_φ, and σ_θ for optimal control parameters under Bessel, ITAE, and LQR control methods.

3.4 Experimentation

This section details the application of control strategies tested in simulations to the Balboa 32U4 robot. Each experiment starts with the robot in a vertical position to maintain consistency across tests and allow direct comparison of the feedback gain on the performance metrics.

3.4.1 System dynamics

The first experiment implements the optimal control parameters derived from simulations: a settling time of 1 second for ITAE, 0.6 seconds for Bessel, and a cost function value of 3 for LQR, across similar 40-second reference tracking scenario.

Figure 3.8 demonstrates that all three control methods effectively stabilize the system and accurately follow the reference trajectory, validating the effectiveness of the control strategies and achieving the primary goal of robot stabilization. However, a consistent deviation in the wheel rotation angle $\varphi(t)$ from the reference $r(t)$ is observed. This deviation is attributed to errors in the angle measurement $\theta(t)$, which does not precisely center around 0° and exhibits a drift over time.

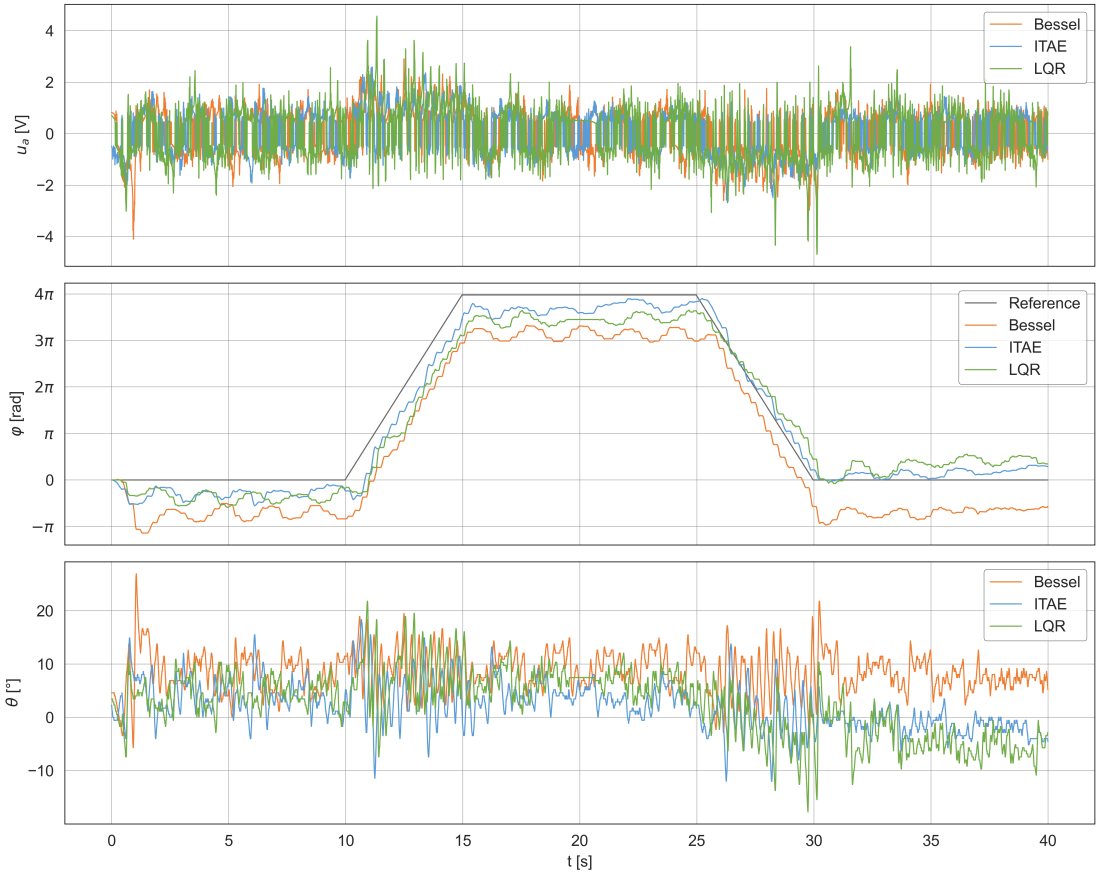

Figure 3.8: Experimentation of voltage response $u_a(t)$, wheel rotation angle $\varphi(t)$, and angular position $\theta(t)$ over a 40-second test period under Bessel, ITAE and LQR control methods.

A 40-second duration ensures adequate data collection for significant performance metrics, presented in Table 3.5. These metrics are compared to those from Figure 3.7 to estimate the system's noise standard deviation σ_n, approximated at $\sigma_n = 0.9°$.

Table 3.5: Performance metrics for optimal control parameters under Bessel, ITAE and LQR control methods to quantify noise standard deviation.

Control method	$\overline{u}_a$	$\lvert\epsilon_\varphi\rvert$	σ_φ	σ_θ
Bessel	0.82 V	133°	37°	9.2°
ITAE	0.78 V	28°	48°	4.8°
LQR	0.92 V	37°	77°	6.5°

3.4.2 Feedback performance

Upon approximating the system noise, this analysis compares the performance metrics from both experimentation and simulations, assuming a noise standard deviation $\sigma_n = 0.9°$. Refer to Figure 3.9 and Figure 3.10 for visual comparisons.

Initially, across all control methods, significant deviations in mean wheel position error ϵ_φ from the simulated values are evident. This discrepancy largely arises from measurement drifts and inaccuracies in angle detection previously discussed. A potential solution could involve modifying the reference tracking controller, which currently only incorporates a proportional gain of the reference signal. Introducing an additional integral control component might address this steady-state error effectively.

The ITAE and Bessel methods show similar patterns in mean absolute voltage $\overline{u}_a$ and the standard deviation of wheel position σ_φ when compared to their simulated counterparts, affirming the simulation model's reliability as a predictive tool. However, some observed differences are likely due to the system's inherent non-linearity and the simplifications employed in the simulations.

Regarding the standard deviation of angular position σ_θ, perfectly matching experimental outcomes with simulations is challenging, particularly at extreme control settings. Lower settings lead to over-oscillations due to excessive responsiveness, while higher settings cause sluggish responses and increased angular deviations. Such disparities may also be linked to accelerometer limitations in accurately measuring rapid angular changes and the physical imbalances in the Balboa robot's setup.

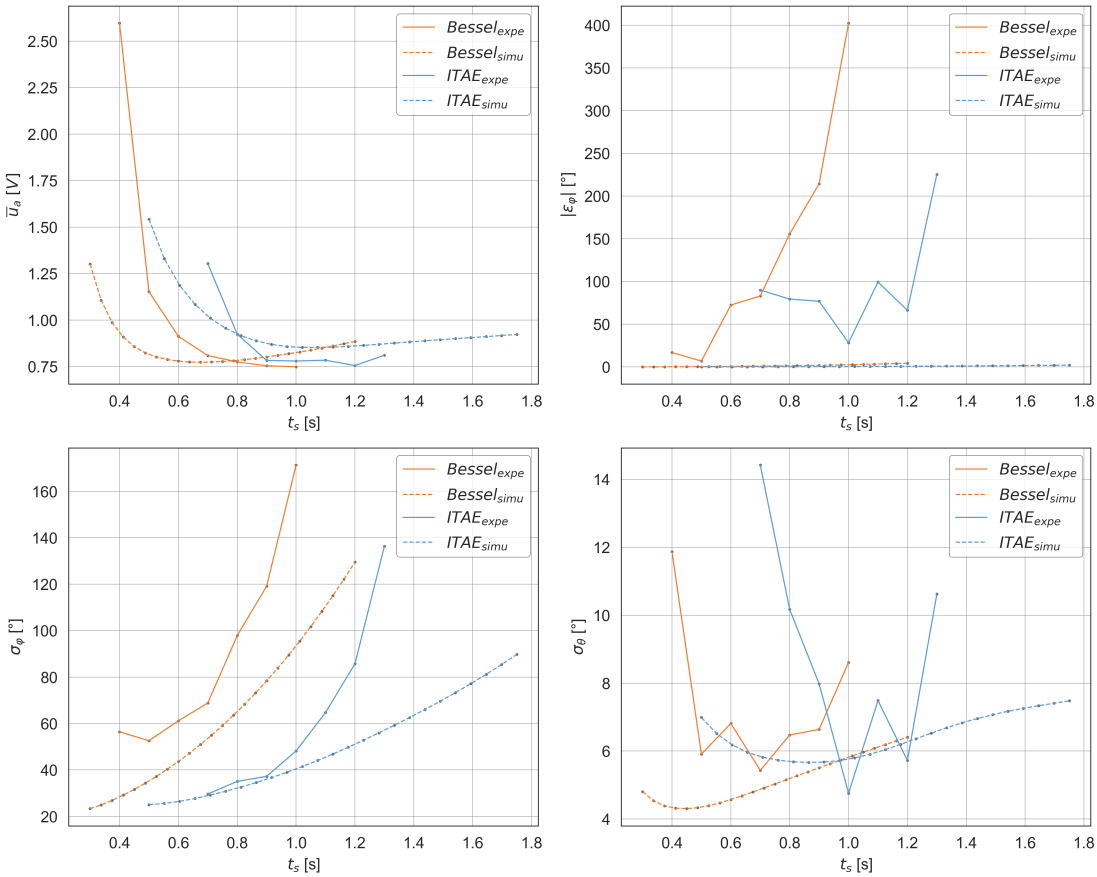

Figure 3.9: Comparison of experimentation and simulation of the performance metrics $\overline{u}_a$, $|\epsilon_\varphi|$, σ_φ, and σ_θ as functions of the settling time t_s under Bessel and ITAE control methods.

The LQR method in Figure 3.10 shows a marked discrepancy between simulation and experimental results, yet maintaining the expected behavior where increasing the cost function reduces energy consumption but raises the deviation from the reference path σ_φ.

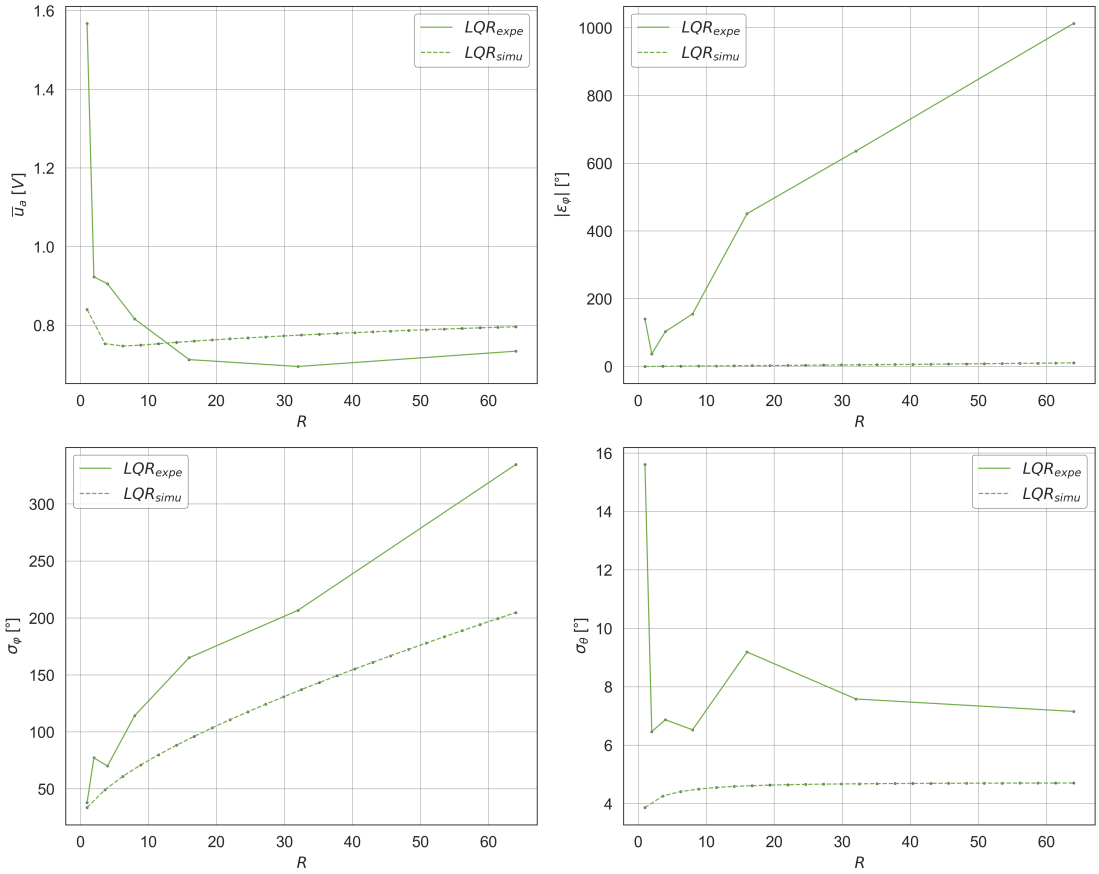

Figure 3.10: Comparison of experimentation and simulation of the performance metrics $\overline{u}_a$, $|\epsilon_\varphi|$, σ_φ, and σ_θ as functions of the cost factor R under LQR control method.

In Figure 3.11, the graph compares the mean absolute voltage $\overline{u}_a$ and the standard deviation of wheel position $\sigma\varphi$ across different control strategies both in experimentation and simulation. Those metrics highlight the sensitivity of the controller parameters. The overlapping lines indicate a close agreement between experimental and simulated results, underscoring the effectiveness of the control adjustments.

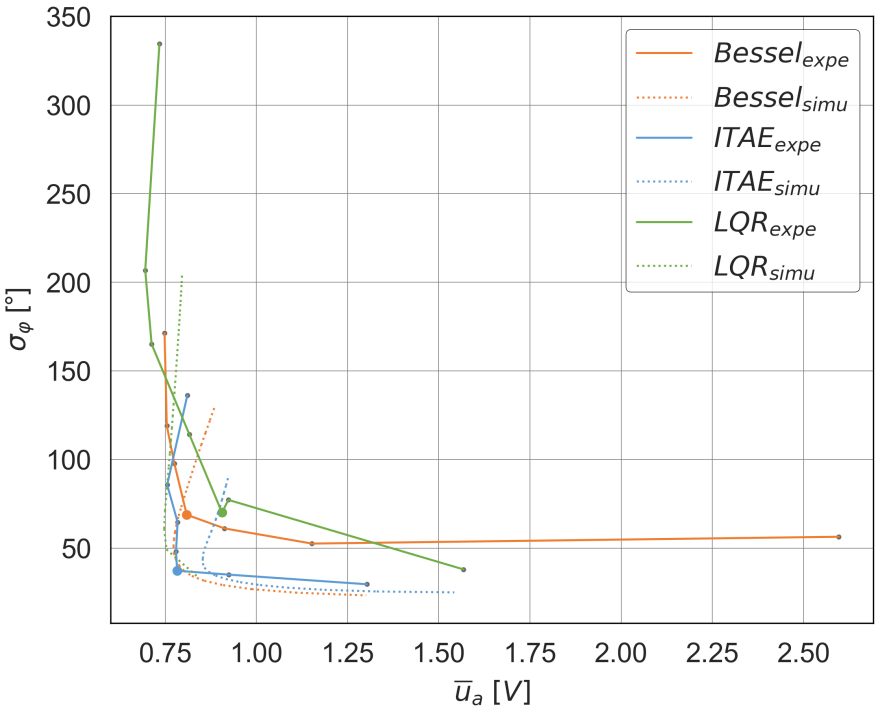

Figure 3.11: Comparaison between experimental and simulated performance metrics σ_φ as a function of control effort $\overline{u}_a$ under Bessel, ITAE, and LQR control methods.

Table 3.6 presents a comparison of optimal settings derived from both simulations and practical experiments. This juxtaposition highlights a general alignment between the simulation and experimentation, although simulations often suggest more aggressive control strategies. This variance is likely due to simulations not fully capturing mechanical imperfections and real-world disturbances that can induce over-oscillations. Consequently, while simulations advocate for tighter controls to optimize theoretical performance, practical adjustments are made to accommodate real-world variability and enhance operational robustness.

Table 3.6: Optimal control parameters from simulation and experimentation under Bessel, ITAE, and LQR control methods.

Control method	In simulation	In experiment
Bessel	0.6 s	0.7 s
ITAE	1 s	1 s
LQR	Cost factor 3	Cost factor 4

3.5 Summary

Initially, it was essential to characterize the system's model and determine motor characteristics. Subsequently, three control methods—LQR, Bessel pole placement, and ITAE pole placement—were applied to stabilize the balancing robot. The model-based approach demonstrated its potential as both simulations and experimental tests successfully stabilized the robot and followed a predefined reference track. However, a noticeable steady-state error was observed, indicating a need for improvements in the reference tracking method, especially if navigation is the primary goal of future research.

Various performance metrics were introduced to quantify controller performance. By comparing these metrics from simulations and experiments, the system noise was quantified with a standard deviation of 0.9°. Furthermore, these metrics helped identify the optimal parameters for each control method. The optimal settling time for Bessel was 0.6 seconds in simulations and 0.7 seconds in experiments; for ITAE, both suggested 1 second; and for LQR, simulations recommended a cost factor of 3, while experiments indicated 4. The close alignment of parameters derived from simulations and experiments validated the model-based approach.

Chapter 4

Data-driven control

4.1 Introduction

This chapter explores the determination of feedback gain through data-driven techniques, focusing on systems without a predefined model. This approach is especially useful for systems with complex or partially known dynamics. The chapter begins by outlining the mathematical formulations used to compute feedback gains through pole placement and system identification methods.

Subsequent sections will present numerical simulations to assess and compare these methods. This includes introducing a new performance metric to compare data-driven feedback with model-based feedback. Initial simulations will be conducted without external inputs to evaluate the impact of the data acquisition period and chosen pole locations. Subsequent simulations will introduce external inputs, examining the effects of its standard deviation, noise standard deviation, and similar variables as before.

The chapter concludes with practical experiments performed on the Balboa 32U4 robot, aiming to validate the proposed data-driven control strategies. Results from using a fast-response controller both with and without external inputs will be analyzed first. Following this, the performance of a controller with a slower response rate and without external inputs will be examined. The chapter will also touch upon the effects of applying different filters to improve experimental outcomes.

4.2 Controller design

The objective of designing a data-driven controller is to establish a discrete feedback gain K_{DD} that positions the system's poles within the unit circle to stabilize the system. This process involves collecting state and input variable measurements at various time steps. Since the system is inherently unstable, a preliminary controller with feedback gain K_x is employed to manage the input variable $u_a(t)$, thereby stabilizing the system sufficiently to acquire significant data. The framework for this data-driven approach is depicted in Figure 4.1, where an unknown model is initially controlled, followed by data-driven computation based on measurements of states $x(t)$ and control input $u_a(t)$ to derive K_{DD}. An external variable $v(t)$ can be introduced to the input, influencing system dynamics.

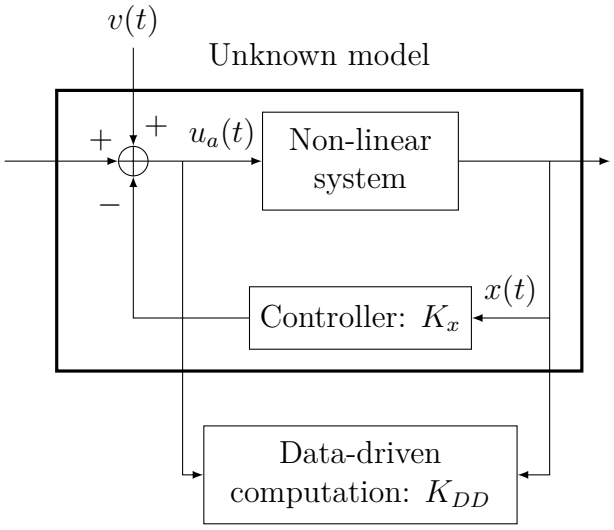

Figure 4.1: Data-driven control system.

Data are captured through sensors at a specified sampling rate Δt over a period $T \times \Delta t$ seconds. These measurements are organized into vectors as follows:

$$U_0 := \begin{bmatrix} u_a(0) & u_a(\Delta t) & \dots & u_a((T-1)\Delta t) \end{bmatrix} \in \mathbb{R}^{1\times(T-1)}, \tag{4.1}$$

$$X_0 := \begin{bmatrix} x(0) & x(\Delta t) & \dots & x((T-1)\Delta t) \end{bmatrix} \in \mathbb{R}^{n\times(T-1)}, \tag{4.2}$$

$$X_1 := \begin{bmatrix} x(\Delta t) & x(2\Delta t) & \dots & x(T\Delta t) \end{bmatrix} \in \mathbb{R}^{n\times(T-1)}, \tag{4.3}$$

where $n = 4$ represents the number of state variables. U_0, X_0, and X_1 encapsulate control inputs and state vectors at current and subsequent time steps, respectively.

To calculate the data-driven feedback gain K_{DD}, confirming the controllability of the system's unstable states is crucial. When the system model is unknown, it becomes impractical to directly assess the rank condition of the controllability matrix. Nevertheless, controllability can be inferred through physical reasoning. For instance, in a balancing robot, the applied torque to the wheel impacts both the pendulum angle and the wheel's orientation, suggesting that the state variables φ, θ, and their derivatives are likely controllable.

This section explores two data-driven methods to compute K_{DD}, each tailored to desired pole locations $\mathcal{L}_{desired}$. The first method computes based on a transformation matrix aligned with the desired poles, while the second identifies the system first and then calculates the feedback gain K_{DD}.

4.2.1 Pole placement method

The pole placement method, as detailed in recent research [46], demonstrates the feasibility of setting closed-loop eigenvalues precisely at predetermined locations directly from data. This method has been evaluated through numerical simulations, showcasing its advantages over traditional model-based approaches.

This approach utilizes a transformation matrix M that is designed to align the system's behavior with specified eigenvalues, representing the desired locations for the system poles. The mathematical formulation of this alignment is expressed as:

$$0 = (X_1 - X_0\Lambda)M_i, \quad \forall i \in \{1, \ldots, n\}, \tag{4.4}$$

where Λ is a diagonal matrix containing the eigenvalues $\mathcal{L}_{desired}$, which are the desired locations of the system poles.

Once M is determined, the data-driven feedback gain K_{DD} is calculated using:

$$K_{DD} = -U_0 M (X_0 M)^{\dagger}, \tag{4.5}$$

where $(X_0 M)^{\dagger}$ represents the Moore-Penrose pseudoinverse of $X_0 M$.

This computation ensures that the system's characteristic equation meets the condition $\det(A_{dt} - B_{dt}K_{DD} - \lambda I) = 0$ for each λ in $\mathcal{L}_{desired}$, thus aligning the system's dynamics with the desired stability characteristics, as proofed in Appendix B.

4.2.2 System identification method

Following the work of [43], another method worth discussing is the system identification approach. This method involves estimating the discrete system matrices A_{dt}

and B_{dt} , which articulate the dynamics of the system. The dynamic relationship is captured by the equation:

$$X_1 = \left[\begin{array}{c:c} A_{dt} & B_{dt} \end{array} \right] \left[\begin{array}{c} X_0 \\ \hdashline U_0 \end{array} \right]. \tag{4.6}$$

From this relationship, the system matrices are computed as:

$$\left[\begin{array}{c:c} A_{dt} & B_{dt} \end{array} \right] = X_1 \left[\begin{array}{c} X_0 \\ \hdashline U_0 \end{array} \right]^{\dagger}, . \tag{4.7}$$

With these identified matrices A_{dt} and B_{dt}, the feedback gain K_{DD} can then be calculated using a pole placement algorithm, which ensures the poles of the closed-loop system are placed at the desired locations $\mathcal{L}_{desired}$. This can be efficiently performed using MATLAB's `place` function:

$$K_{DD} = \texttt{place}(A_{dt},\ B_{dt},\ \mathcal{L}_{desired}).$$

4.3 Numerical simulation

To assess the performance of the data-driven control methods, numerical simulations are carried out. These simulations incorporate system dynamics similar to those used in model-based control, employing an input variable $u_a(t)$ and introducing random Gaussian noise $n(t) \in \mathbb{R}^n$ to simulate real-world disturbances. The input voltage is adjusted using a feedback gain K_x, where its pole locations are denoted as $\mathcal{L}_x$, augmented by an external input $v(t)$. The system's evolution is described by:

$$u_a(t) = -K_x \cdot x(t) + v(t), \tag{4.8}$$

$$x(t + \Delta t) = A_{dt} \cdot x(t) + B_{dt} \cdot u_a(t) + n(t). \tag{4.9}$$

After simulating the system's states and input variables for a specified period, the matrices U_0, X_0, and X_1 are constructed as defined in Equations (4.1), (4.2), and (4.3), respectively.
For the same set of measurements, both the pole placement and system identification methods are applied using identical pole locations. Since the system operates in discrete time, it is essential to convert the continuous-time poles $\mathcal{L}$ to discrete-time poles $\mathcal{L}_{dt}$. This conversion is achieved through the following relationship:

$$\mathcal{L}_{dt} = \exp\left(\Delta t \cdot \mathcal{L}\right). \tag{4.10}$$

To evaluate the efficacy of these methods, the Mean Relative Error (MRE) is introduced as a performance metric to minimize:

$$\text{MRE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{K_{DD,i} - K_{MB,i}}{K_{MB,i}}\right|, \tag{4.11}$$

where K_{DD} represents the feedback gain derived from the data-driven method as calculated using Equation (4.5) or (4.7), depending on the desired poles $\mathcal{L}_{desired}$. This feedback gain K_{DD}will give poles $\mathcal{L}_{DD}$. The K_{MB} is the desired feedback gain computed using a model-based approach for the same poles $\mathcal{L}_{desired}$, as detailed in chapter 3.

Each simulation evaluates the Mean Relative Error (MRE), calculated as outlined in Equation (4.11). Additionally, the evolution of the poles, denoted by $\mathcal{L}_{DD}$, is analyzed through the eigenvalues of the expression $\text{eig}(A_{dt} - B_{dt} \cdot K_{DD})$, where A and B are the system matrices determined by the model-based approach.

Two simulation approaches are assessed, focusing on the MRE and pole locations. The first approach evaluates the system without any external input ($v(t) = 0$), while the second approach explores the effects of introducing an external input ($v(t)$).

4.3.1 Without external input $v(t) = 0$

In this section, the system operates without external input ($v(t) = 0$), and measurements of the states X_1 and X_2 along with the input variable U_0 are obtained using the feedback gain $K_x = K_{Bessel}(t_s = 0.6\text{s})$. This setup simulates the system's response as if controlled by an optimal feedback gain derived from Bessel poles with a settling time of 0.6 seconds. The system noise $n(t)$ is set to a standard deviation of 0.05°, and its impact on the system dynamics will be discussed subsequently.

Influence of data acquisition period

This simulation investigates the impact of the data acquisition period. The desired poles are set as defined in subsection 3.2.2, with the number of samples markedly influencing the resultant data-driven feedback gains K_{DD}, computed via Equations (4.5) and (4.7). The MRE and pole dynamics are analyzed based on the measurement period utilized on the same computational setup.

Figure 4.2 on the left contrasts the evolution of MRE between the pole placement method and the system identification method. It is observed that the pole placement method converges with just five samples, while the system identification method

requires more samples to produce definitive results, accompanied by noticeable oscillations. Both methods exhibit an MRE baseline of about 3.

The right panel of Figure 4.2 displays pole positions on the real-imaginary axis, where darker colors represent a higher number of samples. Each color gradient is associated with a specific data acquisition period, allowing for the identification of the measurement duration for each pole by its color. The green crosses mark the target poles $\mathcal{L}_{desired}$, showing the intended locations.

This visualization confirms that the pole placement method swiftly stabilizes to consistent pole values, with most hidden behind the prominent dark orange point. These poles do not exactly align with $\mathcal{L}_{desired}$, leading to a consistent MRE offset. They correspond to the poles of the feedback gain K_x employed by the unknown controller, indicating that it is feasible to discern the poles of a system under an unidentified controller. The poles identified by the system identification method appear more dispersed, correlating with the MRE variability and tending to converge close to those determined by the pole placement method. However, some poles identified through this method lie outside the unit circle, suggesting they fall outside the stability zone and indicating that the corresponding feedback gain may not stabilize the system effectively.

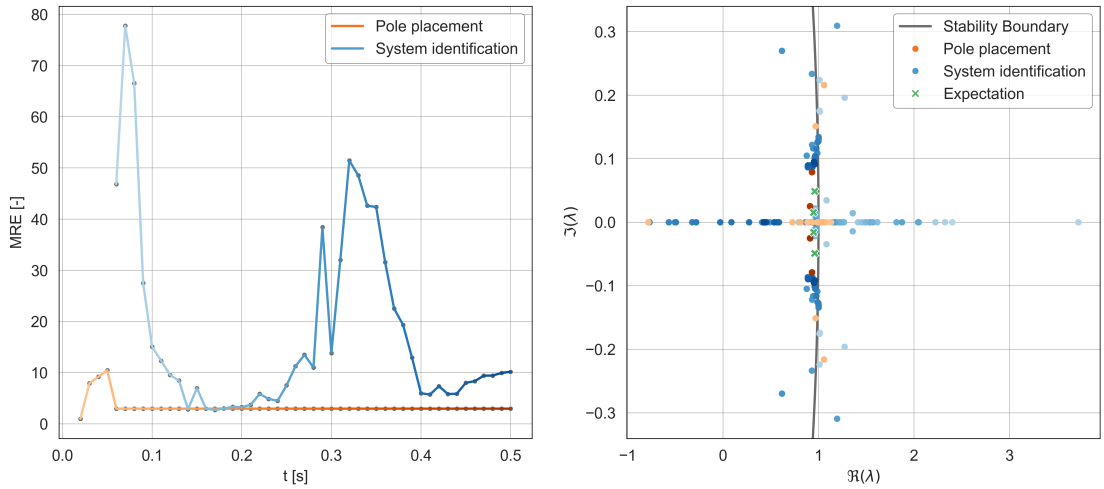

Figure 4.2: Simulation results depicting the Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods without external input.

Influence of pole location

This analysis explores the effect of changing the pole locations $\mathcal{L}_{desired}$. A fixed data acquisition period of 1 second is used, and the pole locations are varied as follows: $\mathcal{L}_{desired} = \mathcal{L}_{Bessel}(t_s = 0.1 - 1.2\text{s})$.

Figure 4.3 examines the evolution of the MRE in relation to the settling time that influences Bessel pole locations. Both the pole placement and system identification methods show an increasing MRE as the settling time deviates from 0.6 seconds, which is precisely the settling time used to calculate the internal controller of the system, denoted as $K_x = K_{Bessel}(t_s = 0.6\text{s})$.

The graph on the right on Figure 4.3 compares the poles identified by both methods by varying the desired poles, represented by green crosses. The color coding corresponds to specific settling times, allowing for identification of the respective settling time for each desired pole. It is observed that the poles identified using the pole placement method remain largely unchanged despite variations in settling time. Conversely, the system identification method exhibits only minor variations. However, for some settings, it identifies poles outside the stability boundary, which could potentially destabilize the system.

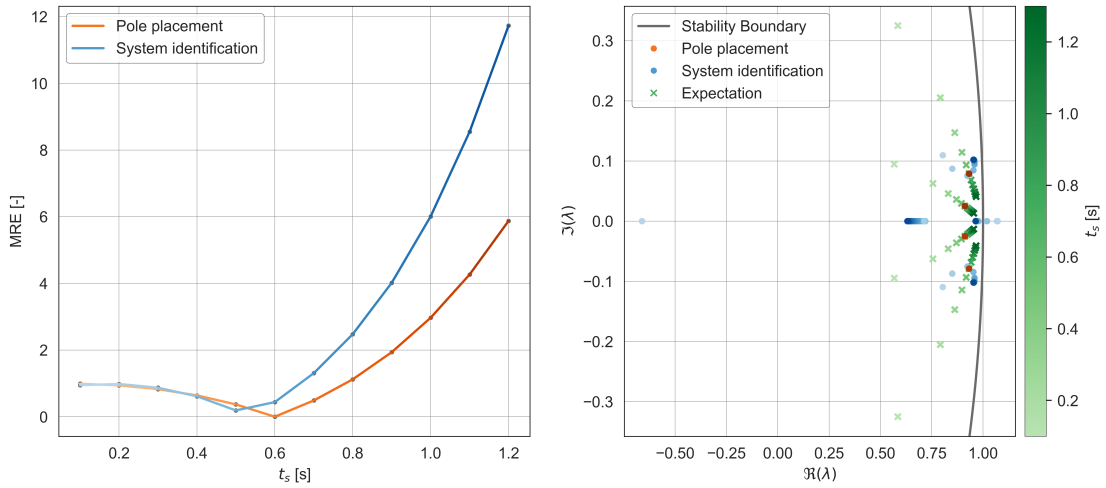

Figure 4.3: Simulation results depicting the Mean Relative Error (MRE) and pole positions under varying pole locations, analyzed using pole placement and system identification methods without external input.

In the absence of external input $v(t) = 0$, the feedback gains K_{DD} derived from Equations (4.5) and (4.7) approximate the actual feedback gain K_x. A sample size of 100 is generally sufficient for ensuring convergence. The choice of desired poles appears to minimally impact these results. The pole placement method tends to converge faster and is less sensitive to the selection of poles compared to the system identification method.

4.3.2 With external input $v(t) \neq 0$

This section analyzes the effect of an external input $v(t)$ added to the input variable $u_a(t)$, quantified by its standard deviation σ_v. Measurements of the states X_1 and X_2 and the input variable U_0 are obtained using the feedback gain $K_x = K_{Bessel}(t_s = 0.6s)$. The desired poles are set to $\mathcal{L}_{desired} = \mathcal{L}_{Bessel}(t_s = 1s)$, using the corresponding model-based feedback gain $K_{MB} = K_{Bessel}(t_s = 1s)$, and the system noise $n(t)$ is fixed at a standard deviation of 0.05° with 100 samples corresponding to a data acquisition period of 1 second.

Influence of standard deviation σ_v

The influence of σ_v is explored to determine its impact on the system's response. The external input aims to ensure sufficient frequency content in the input signal to excite all system modes, referred to as persistence of excitation.

Figure 4.4 shows the impact of varying σ_v on system dynamics. Insufficient excitation fails to adequately perturb the system, leading to inaccurate feedback gain characterization, while excessive excitation may lead to saturation and potentially destabilize the system. The pole placement method demonstrates higher sensitivity to variations in excitation compared to the system identification method, which exhibits greater stability. A standard deviation of σ_v between 0.5 V and 1 V is considered optimal. For subsequent tests, the lower bound of $\sigma_v = 0.5V$ is selected to minimize risk of instability.

The right graph in Figure 4.4 illustrates that σ_v significantly influences pole positions, with very high or low values displacing poles away from their expected locations, sometimes even outside the stability boundary. The poles obtained via the pole placement method show greater dispersion compared to those from the system identification method.

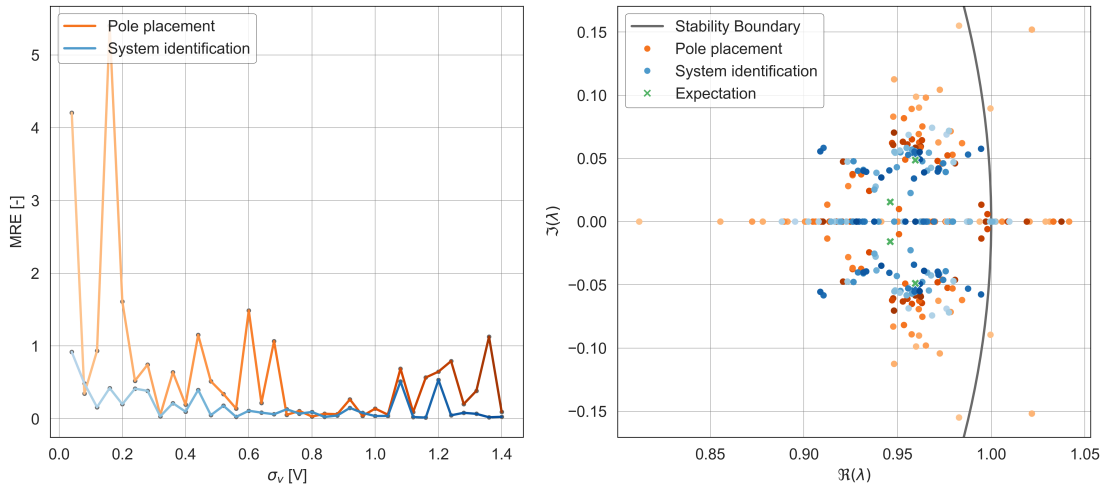

Figure 4.4: Simulation results depicting the Mean Relative Error (MRE) and pole positions under varying standard deviations σ_v, analyzed using pole placement and system identification methods with external input.

Influence of noise standard deviation σ_n

Previously, the noise standard deviation σ_n was set to 0.05°. This section examines its impact on the MRE and the stability of pole locations. In Figure 4.5, the variation of MRE in response to different levels of noise is assessed. Two notable trends are observed:

Firstly, an increase in noise leads to higher MRE values, indicating degraded performance; lower noise levels are associated with improved MRE values. Secondly, the MRE for the system identification method remains generally lower and more stable across varying noise levels, unlike the pole placement method, which shows significant spikes in MRE at higher noise levels.

The analysis of pole locations further illustrates that while the poles calculated via system identification cluster closely around the expected values, those from the pole placement method are more dispersed and may drift outside the stability boundary at higher σ_n levels. This dispersion explains the heightened sensitivity of the pole placement method to noise.

A noise standard deviation σ_n of less than 0.2° is recommended to ensure reliable convergence and minimize the risk of instability.

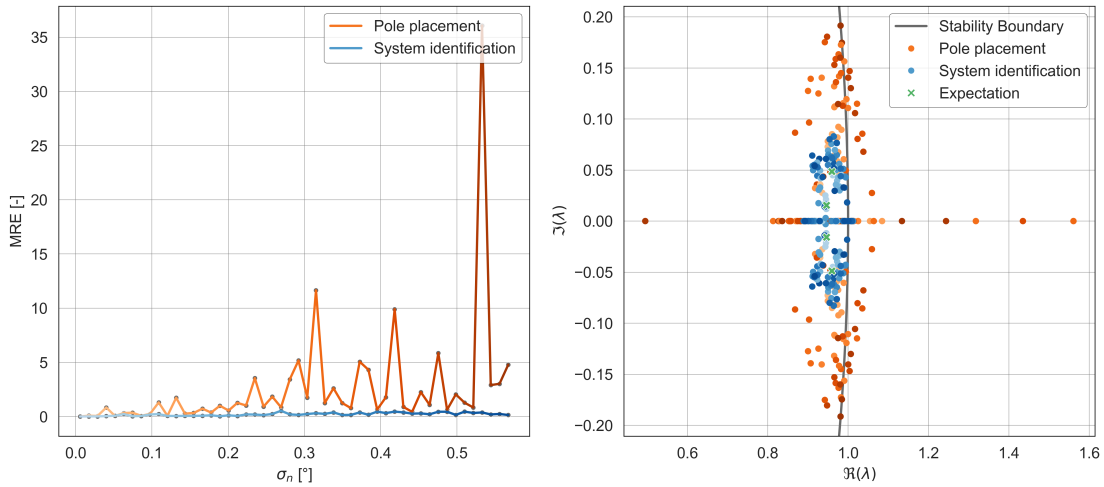

Figure 4.5: Simulation results depicting the Mean Relative Error (MRE) and pole positions under varying noise standard deviations σ_n, analyzed using pole placement and system identification methods with external input.

Influence of data acquisition period

With established reasonable noise standard deviation and external input parameters, the next aspect to explore is the data acquisition period's influence on convergence. Figure 4.6 presents a comparison of the convergence times for the two methodologies. The system identification method converges faster, typically within approximately 0.4 seconds, compared to about 0.5 seconds for the pole placement method. Examination of the pole colors reveals an actual convergence of both methods towards the expected poles.

Table 4.1 presents a summary of the feedback gains achieving the lowest MRE for each method, alongside the expected gains calculated via the model-based approach. The pole placement method estimates obtain a MRE of 6%, while the system identification method achieves a closer estimation of only 4%.

Table 4.1: Simulation comparison of feedback gains derived from pole placement and system identification methods against model-based predictions.

Feedback method	Feedback gain				MRE
Model-based	[-0.04	-2.41	-0.021	-0.19]	/
Pole placement	[-0.04	-2.37	-0.024	-0.19]	0.06
System identification	[-0.04	-2.39	-0.023	-0.19]	0.04

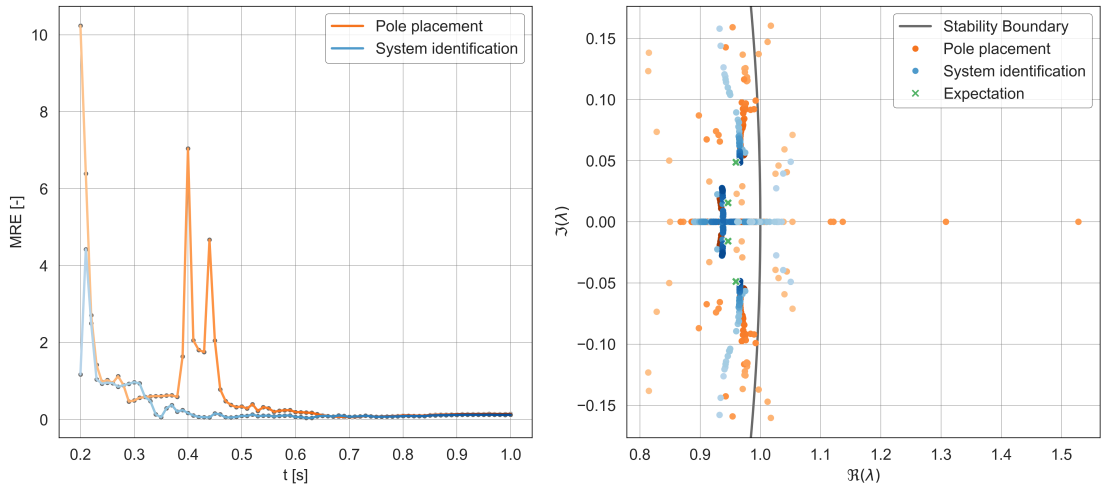

Figure 4.6: Simulation results depicting the Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods with external input.

Influence of pole location

This final simulation compares both data-driven methods by varying the desired pole locations. Similar to the scenario with no external input, the Bessel poles are adjusted based on their settling time, with each time represented by a unique color tint.

Figure 4.7 illustrates the adaptability of both methods to changes in pole locations, unlike in the previous simulation without external input, where the poles' positions are influenced by the chosen poles.

By examining the MRE, it is observed that the values remain within reasonable limits, and the corresponding poles stay within the stability boundary. This indicates that for an unknown system, where only the states and input variable are measured and the internal controller's feedback is unknown, it is feasible to determine appropriate feedback gains by merely specifying desired pole locations without explicitly defining the system model.

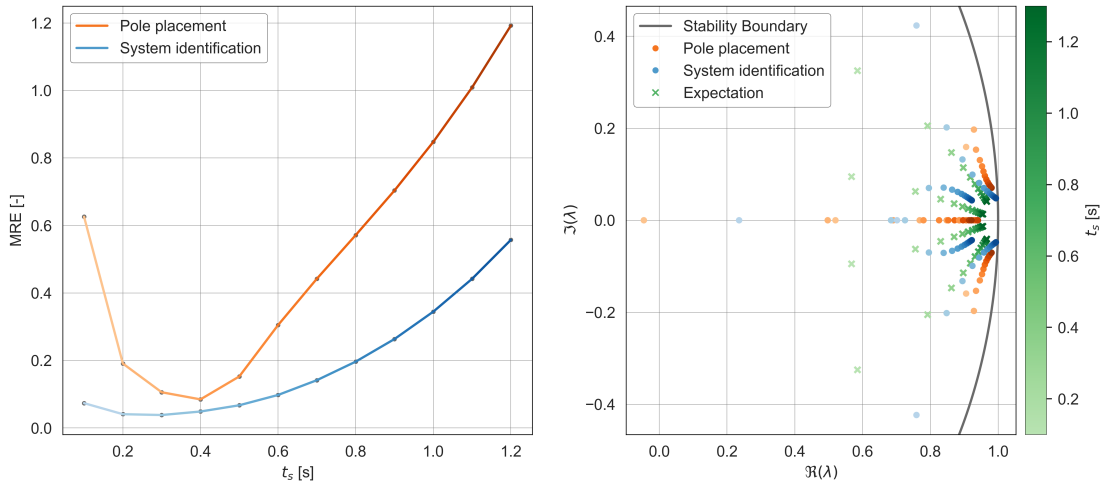

Figure 4.7: Simulation results depicting the Mean Relative Error (MRE) and pole positions under varying pole locations, analyzed using pole placement and system identification methods with external input.

4.4 Experimentation

Contrasting with numerical simulations that utilize system dynamics simulated via matrices derived from a model-based approach, this section employs experimental measurements from the Balboa 32U4 robot. To ensure sufficient data capture, an internal controller is implemented, represented by a feedback gain K_x, with its pole locations denoted as $\mathcal{L}_x$. This controller may vary depending on the specific tests conducted.

After recording the system's states and input variables over a designated period, the matrices U_0, X_0, and X_1 are constructed in accordance with Equations (4.1), (4.2), and (4.3). Subsequently, using some predetermined desired poles, a data-driven feedback gain K_{DD} is calculated using either Equation (4.5) or (4.7). The analysis proceeds by monitoring the MRE and tracking the evolution of the poles, referred to as $\mathcal{L}_{DD}$, via the eigenvalues of $\text{eig}(A_{dt} - B_{dt} \cdot K_{DD})$, where A and B are the system matrices defined by the model-based approach.

The initial experimentation will employ a controller with a rapid response, specifically a feedback gain $K_x = K_{Bessel}(t_s = 0.6s)$. This setup will facilitate the examination of the influence of an external input, mirroring the methodology used in the simulation phase.

Subsequent experiments will involve a controller characterized by a slower response, utilizing a feedback gain $K_x = K_{Bessel}(t_s = 0.9s)$ to assess different system dynamics under varied control conditions.

4.4.1 Fast response controller

A feedback gain $K_x = K_{Bessel}(t_s = 0.6s)$ will be initially implemented in the controller. This configuration, as indicated by the model-based approach, provides a fast response, minimizing oscillations and enabling rapid system stabilization in response to perturbations, including those from an external input $v(t)$. For these tests, the desired poles used in the data-driven equations are fixed at $\mathcal{L}_{desired} = \mathcal{L}_{Bessel}(t_s = 1s)$.

The experiments will first be conducted without any external input ($v(t) = 0$) and subsequently with an external input ($v(t) \neq 0$).

Without external input $v(t) = 0$

In the absence of external input, measurements with a fast controller show convergence, as illustrated in Figure 4.8. Both the pole placement and system identification methods achieve a MRE that converges around 5, with the pole placement method exhibiting a more gradual convergence compared to the system identification method. However, the poles from both methods do not align with expected locations, with system identification poles occasionally falling outside the stability boundary, indicating non-ideal convergence.

To enhance the accuracy of the RME and pole locations, a prefilter to adjust for consistent voltage offsets c due to friction is applied to the input variable.

Following the application of this prefilter, recalibrated MRE and pole locations are presented in Figure 4.9, demonstrating significantly improved results. The poles are closer to the expected ones and mostly within the stability zone, showing a substantial improvement. However, differences in convergence relative to the expected poles are still evident. Although the data-driven poles $\mathcal{L}_{DD}$ do not align perfectly with the desired poles $\mathcal{L}_{desired}$ or the implemented feedback poles $\mathcal{L}_x$, they are predominantly influenced by $\mathcal{L}_x$, affirming the simulation predictions that data-driven poles and feedback gains are slightly influenced by the desired poles $\mathcal{L}_{desired}$.

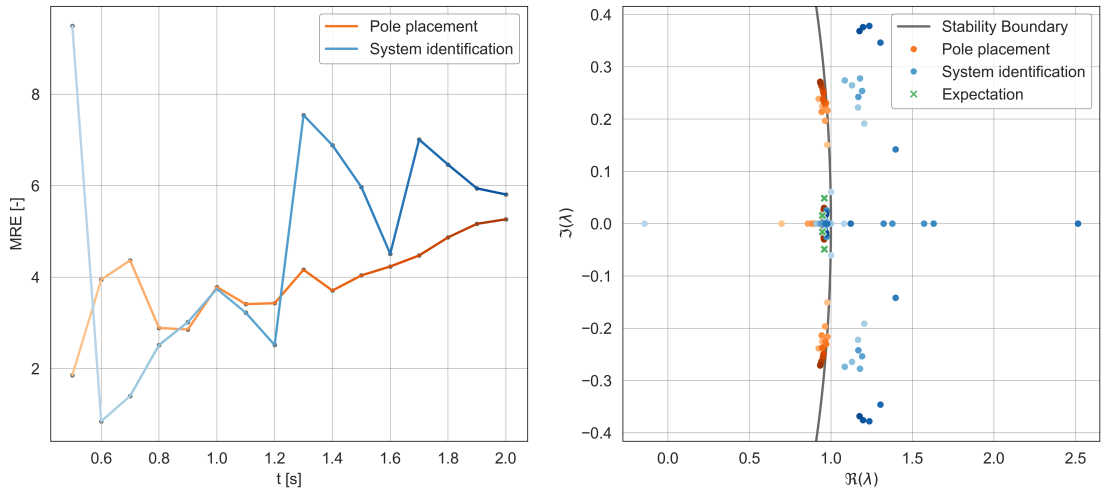

Figure 4.8: Experimental results depicting Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods. This setup involves a fast-response controller without external input.

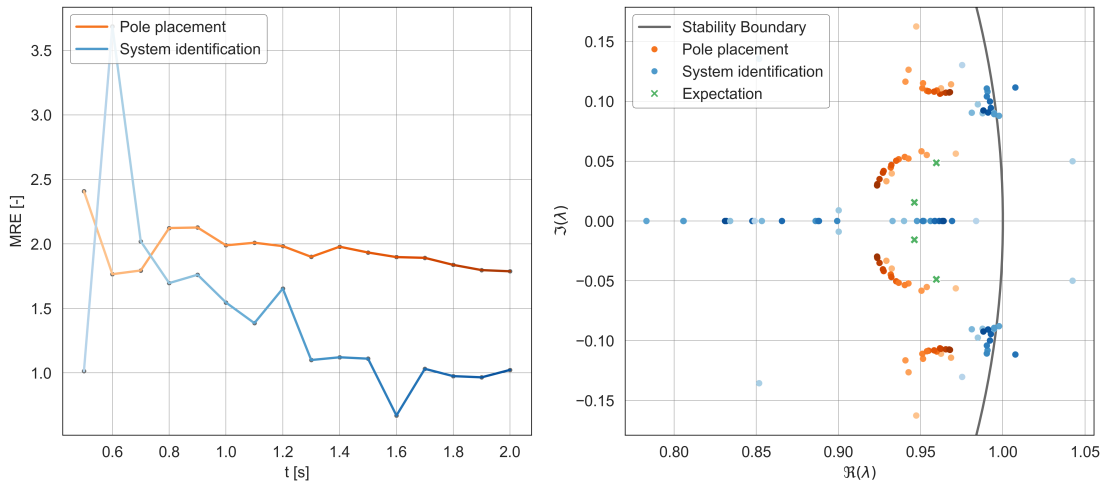

Figure 4.9: Experimental results showing Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods. This setup involves a fast-response controller, including prefiltering, without external input.

With external input $v(t) \neq 0$

In subsequent tests, measurements are captured while applying an external input $v(t)$ to the input variable. This external input is a Gaussian noise with a fixed standard deviation of $\sigma_v = 0.5$V, as suggested by the simulation.

Figure 4.10 illustrates the challenges encountered under this setup. The system identification method fails to converge, exhibiting unpredictable spikes in MRE and resulting in some poles being positioned outside the stability zone. Although the pole placement method achieves MRE convergence and its poles are relatively close to the desired locations, there are still instances where the poles fall outside the stability boundary. Under these conditions, prefiltering the offset does not ameliorate the outcomes.

As previously analyzed in the model-based approach, the system is subject to significant noise, denoted by a high noise standard deviation σ_n, due to inaccurate angle measurements that exhibit drift over time. Simulations have shown that this high noise level can lead to divergent results when applying data-driven feedback. Introducing an external input under these conditions add system noise due to the fast response, rendering the data non-representative. Consequently, applying data-driven methods to such data often results in the computation of unstable feedback gains.

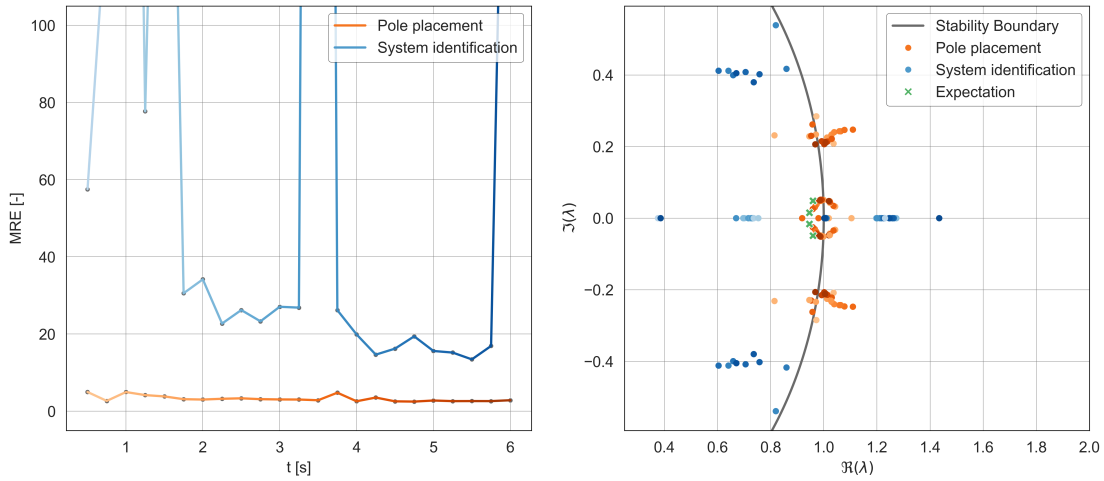

Figure 4.10: Experimental results showing Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods. This setup involves a fast-response controller with external input.

4.4.2 Slow response controller

Given the limitations in exciting all system modes using an external input, a controller with a slower response is employed. This approach allows more accurate angle measurements with sufficient time for data acquisition, but also leads to an increase in oscillations due to natural disturbances. To effectively manage these dynamics, a feedback gain $K_x = K_{Bessel}(t_s = 1\text{s})$ is implemented in the controller.

Influence of data acquisition period

The influence of the data acquisition period on system behavior is first analyzed by setting the desired poles for the data-driven equations to $\mathcal{L}_{desired} = \mathcal{L}_{Bessel}(t_s = 0.6\text{s})$.

Figure 4.11 shows rapid convergence for the pole placement method, whereas the system identification method exhibits divergence. Analysis of the poles reveals that some are outside the stability zone for both methods, indicating that the stability of the system using these data-driven methods cannot be guaranteed.

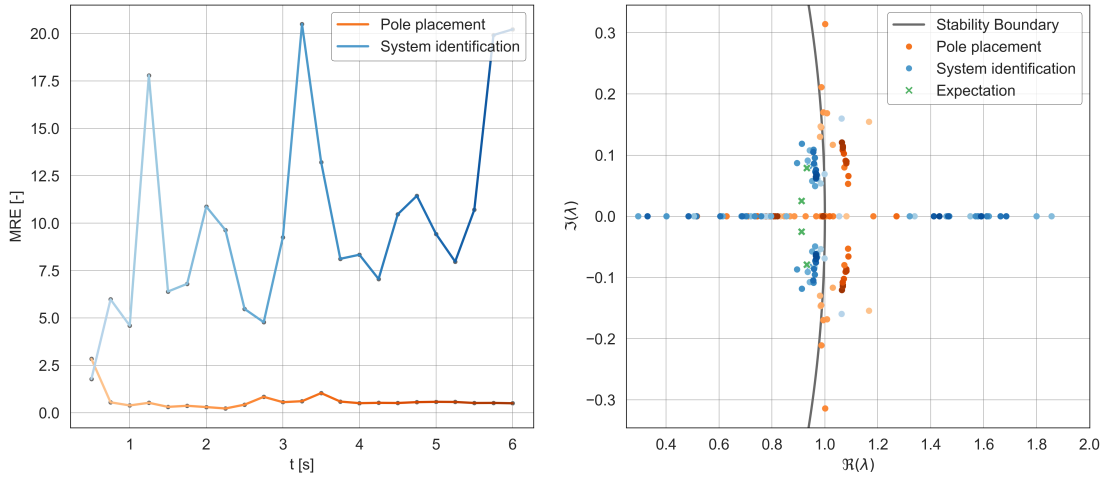

Figure 4.11: Experimental results showing Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods. This setup involves a slow-response controller without external input.

To enhance these results, the same prefilter used in previous tests is applied to correct the offset gain. Figure 4.12 demonstrates notable improvements in MRE for both methods; the system identification method occasionally matches the pole

placement method's performance, although the latter is more consistent. These results contradict the simulations which suggested opposite behaviour for each method.

Comparison of the poles depicted in Figure 4.12 with earlier findings demonstrates that the majority of poles now reside within the stability zone. Table 4.2 aggregates the optimal feedback gains derived from each data-driven method and contrasts them with the expected model-based gains. Despite certain discrepancies, these feedback gains sufficiently stabilize the robot. Notably, in this test, the system identification method achieves a slightly better MRE of 27% compared to 42% with the pole placement method.

Table 4.2: Experimental comparison of feedback gains derived from pole placement and system identification methods against model-based predictions.

Feedback method	Feedback gain				MRE
Model-based	[-0.29	-4.86	-0.28	-0.42]	/
Pole placement	[-0.08	-3.70	-0.22	-0.22]	0.42
System identification	[-0.44	-6.46	-0.30	-0.35]	0.27

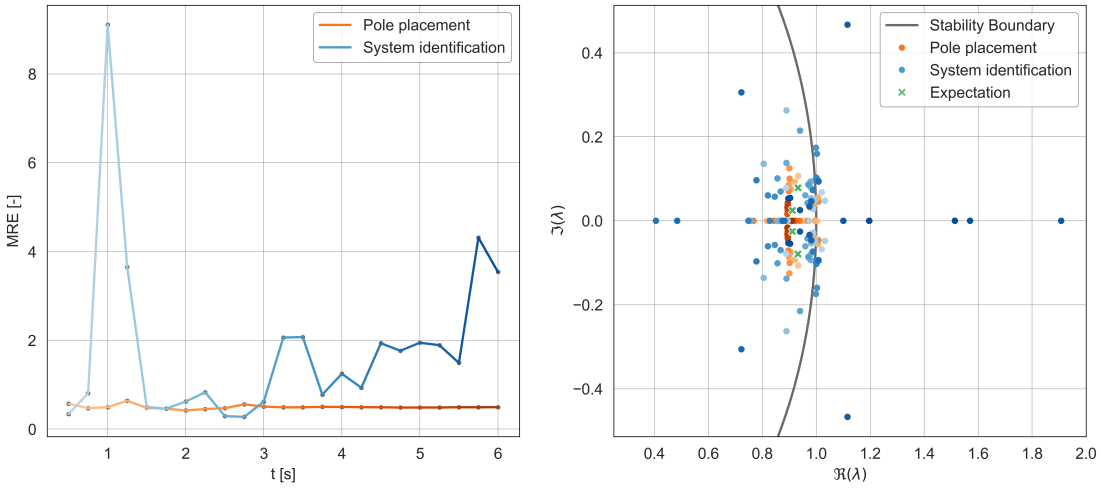

Figure 4.12: Experimental results showing Mean Relative Error (MRE) and pole positions under varying data acquisition periods, analyzed using pole placement and system identification methods. This setup involves a slow-response controller, including prefiltering, without external input.

The implementation of the offset prefilter significantly enhances system performance for both fast and low-response controllers. It demonstrates that for data-driven methods, accurately defining each state variable of the model is crucial for obtaining conclusive results. An alternative method involves incorporating an additional state variable to represent the sign of the wheel speed. This modification necessitates the identification of five new poles. These can also be determined using either the Bessel or ITAE methods, specifically with a five-pole configurations.

Influence of pole location

The influence of the pole location $\mathcal{L}_{desired}$ is analyzed using a controller with a low response rate where a prefilter is applied to mitigate voltage offset. A data acquisition period of three seconds is used, as this period has previously produced convincing results.

Figure 4.13 demonstrates that the MRE for various pole locations remains within appropriate limits, verifying the effectiveness of the method in deriving a feedback gain appropriate to the specified poles. Some poles fall outside the stability zone, indicating divergences in calculations or discrepancies in how the system matrices A_{dt} and B_{dt} represent the actual model. Consequently, poles derived from these matrices using $\text{eig}(A_{dt} - B_{dt} \cdot K_{DD})$ might inaccurately project the actual pole location.

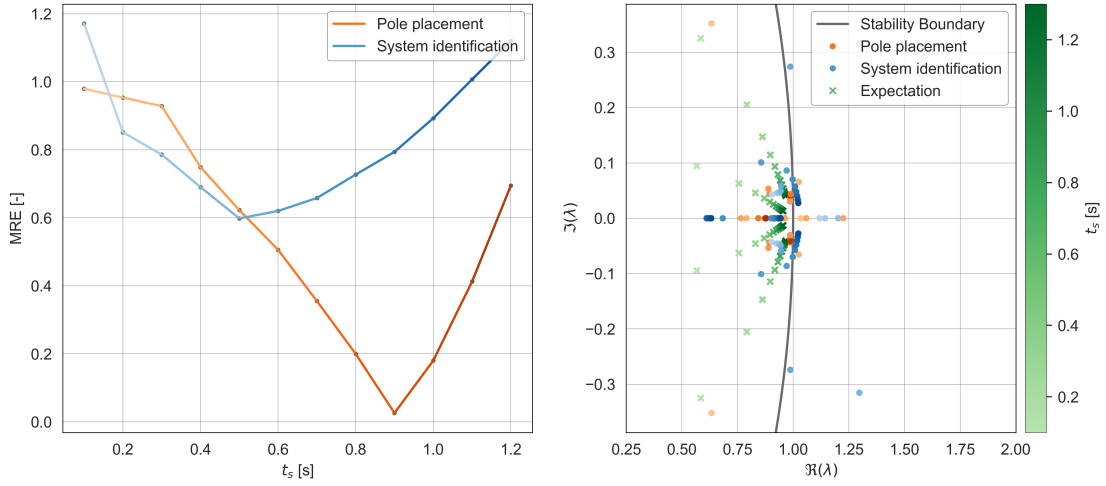

Figure 4.13: Experimental results showing Mean Relative Error (MRE) and pole positions under varying pole locations, analyzed using pole placement and system identification methods. This setup involves a slow-response controller, including prefiltering, without external input.

4.5 Summary

After validating the model-based approach, data-driven methods, specifically pole placement and system identification, were implemented and compared. In the various simulations and experiments, performance metrics and pole locations were assessed which require the model-based results.

Simulations implementing data-driven methods with external input provided conclusive results. However, experimental trials using the same conditions did not yield convincing outcomes due to the high noise levels in the system.

Consequently, data-driven methods were applied to a system free from external noise. In this scenario, two tendencies were observed for fast and slow controllers.

- For fast controllers, the data-driven methods converged to the poles implemented in the controller, independent of the chosen poles. This corresponds to simulations with no external input
- For slow controllers, the data-driven methods converged to the desired poles, mirroring the tendency seen in simulations with external input.

In experimental settings, the pole placement method converges more rapidly than the system identification method, contrasting with the results observed in simulations.

It was noted that data prefiltering could significantly enhance results from data-driven methods. In the case of the balancing robot, voltage offsets caused by friction were filtered out, an adjustment that could alternatively be managed by integrating an additional state. This underscores the importance of precisely defining all state and input variables to achieve significant results with data-driven methods.

Chapter 5

Conclusions and perspectives

This thesis highlights critical aspects of transitioning from classical model-based control to data-driven control methods in self-balancing robots, using the Pololu Balboa 32U4 as a test platform. The aim is to bridge the gap between theoretical mathematics and practical experimentation in active control systems, which are crucial for managing the unstable dynamics of such robots.

Traditional control techniques such as LQR, Bessel, and ITAE pole placement have proven effective in ensuring stability in simulation and experimentation, particularly in tasks like following predefined tracks. However, the presence of steady-state errors during these tests indicates a need for improvement, particularly for applications requiring precise navigation.

In contrast, data-driven control offers a robust alternative by deriving control strategies directly from historical data, eliminating the need for predefined system models. These methods have produced promising results in simulations. In experimental contexts, they are effective in scenarios where data is gathered using slower controllers to handle natural disturbances. However, they are less effective with faster controllers. This limitation is largely due to inaccuracies in sensor data measurement and prevalent system noise, which compromise the integrity of the data-driven results.

For future research, it is crucial to enhance measurement accuracy, potentially through sensor fusion, to improve both system stability and the quality of data-driven results. Although implementing Kalman filters could further enhance these outcomes, this method does require the use of a model. Additionally, investigating non-model-based filtering techniques could broaden the scope and potential of data-driven approaches.

In summary, this research advances the understanding of control strategies for unstable systems, shedding light on the comparative limitations of model-based and data-driven approaches and setting the stage for future enhancements that could facilitate broader real-world applications.

Appendices

Appendix A

Detailed description of the robot

A more detailed description of the Pololu Balboa 32U4 robot is presented here, focusing on its mechanical structure, electrical system, control mechanisms and sensor integration. The mechanical system features an integrated frame and motor drivers, while the electrical system supports power distribution and processing essential for operation.

A.1 Mechanical components

The mechanical system of the Pololu Balboa 32U4 focuses on a highly integrated design that serves dual purposes as both the robot's frame and its control board. This system includes two motor drivers, which are essential for driving the geared motors. These motors are equipped with magnetic encoders that provide feedback on wheel movement, critical for balance and control.

The robot utilizes a gearmotor assembly that features a miniature high-power carbon brushed (HPCB) direct-current (DC) motor. The motor is paired with a 51.45:1 metal gearbox to enhance precision and durability. Furthermore, the magnetic wheel of the encoder is directly mounted on the motor's rotor. The output shaft of the motor includes a 25-tooth gear wheel that meshes with a 41-tooth gear wheel connected to the robot's wheel. This setup ensures effective power transmission from the motor to the robot's wheels, which is vital for maintaining stability and navigating various terrains.

A.2 Electrical components

The electrical system of the Balboa 32U4 is crucial for powering and controlling the robot's functionalities. Central to this system is the control board, which is

seamlessly integrated with the mechanical components of the robot. This control board includes battery terminal connections that source power from a compartment housing six AA batteries within the Balboa chassis. During experimental runs, these batteries provide a nominal voltage of $u_{a,max} = 7.2V$, with each cell delivering 1.2 V.

The control board efficiently manages and distributes power to various subsystems, including the microprocessors and motor drivers. The control of the Balboa 32U4's motors is executed through an Arduino microcontroller, programmed using Arduino IDE Version 2.3.1. This environment allows for the compilation and uploading of control algorithms to the microcontroller via a serial port connection. For real-time data acquisition during experiments, the robot remains tethered to a computer, facilitating direct data communication. Data logging and real-time plotting are managed using CoolTerm, a software tool capable of interfacing directly with the serial port to capture and visualize experimental results.

A.3 Actuator

The actuators in this robot are micro metal gearmotors HPCB, controlled using pulse width modulation (PWM) to adjust their speed and direction. The Arduino code manages this process with an output compare register (OCR), which set the PWM duty cycles for the motors. The input captur register (ICR) is configured at a value of 400, establishing the base frequency for the PWM signals.

The duty cycle for each motor is calculated as the ratio of OCR to ICR. This ratio determines the percentage of the maximum voltage applied $u_{a,max}$ to the motors during each PWM cycle. Using this ratio, it is possible to calculate the voltage u_a:

$$u_a = \frac{OCR}{ICR} u_{a,max}.$$

A.4 Sensors

The Pololu Balboa 32U4 employs various sensors critical for the robot's stabilization and dynamic navigation capabilities. These sensors collect critical data which is essential for real-time decision-making and control algorithm execution. It allows precise measurements of the wheel angle and angular speed via magnetic encoders, as well as the pendulum angle and angular rate through an inertial measurement unit (IMU).

Magnetic encoder disc

Each drive motor is equipped with a magnetic encoder disc. This setup consists of a magnetic disc attached to the extended motor shaft and a pair of Hall effect sensors mounted on the control board. These encoders provide a resolution of 12 counts per revolution of the motor shaft, capturing both rising and falling edges across two channels. For the Balboa 32U4, this results in 1012 encoder counts per complete wheel rotation, calculated as $12 \times 51.45 \times \frac{41}{25}$, equating to 161 ticks per radian.

The Arduino code utilizes functions `encoders.getCountsLeft()` and `encoders.getCountsRight()` to gather the measurements from the encoder for the left and right wheels, respectively. The wheel angle φ at each sampling moment t is determined by averaging the counts from both wheels:

$$\varphi(t) = \frac{\texttt{encoders.getCountsLeft()} + \texttt{encoders.getCountsRight()}}{2 \cdot 161}.$$

Measurements are conducted with a sampling period (Δt) of 10 ms, enabling precise calculations of wheel speed $\dot{\varphi}$ based on encoder feedback. The rate of change of the wheel angle, or angular velocity, is then computed as:

$$\dot{\varphi}(t) = \frac{\varphi(t) - \varphi(t - \Delta t)}{\Delta t}.$$

Inertial Measurement Unit (IMU)

The Balboa 32U4 is equipped with an integrated Inertial Measurement Unit (IMU), essential for determining the robot's orientation and balance. The IMU utilizes the ST LSM6DS33, which combines a 3-axis accelerometer and a 3-axis gyroscope into a single module. The accelerometer is primarily used to determine the pendulum angle θ in a static position, calculated in the Arduino code with the formula:

$$\theta(t) = \arctan\left(\frac{\texttt{imu.a.z}}{\texttt{imu.a.x}}\right),$$

where `imu.a.x` and `imu.a.z` are the accelerometer readings along the horizontal (x-axis) and vertical (z-axis) alignments, respectively. The result from the arctan function is expressed in radians, which is ideal for further computations.

The gyroscope is tasked with calculating the angular position and speed during dynamic behavior. With a range of ±1000 degrees per second and a 16-bit output, a full-scale reading of 1000 translates to a digital value of 32768. The angular

velocity $\dot{\theta}(t)$ is therefore computed as:

$$\dot{\theta}(t) = \frac{\texttt{imu.g.y} \times 32768}{1000}.$$

Measurements are conducted at a sampling period of Δt, which facilitates timely updates of the robot's angular position:

$$\theta(t + \Delta t) = \theta(t) + \dot{\theta}(t) \cdot \Delta t$$

A.5 Motor parameters

To accurately model the system, defining the operational parameters of the motor is crucial. According to Figure A.1 for the 50:1 gear ratio motor, at a standard voltage of $u_a = 6V$, the motor achieves a no-load speed of $\dot{\varphi}_{no-load} = 650RPM$ and draws a current of $i_{a,no-load} = 150mA$. Under stall conditions, the torque peaks at 7.4 kg-cm (0.724 Nm) with a current draw of $i_{a,stall} = 1.5A$. These characteristics are pivotal for establishing the torque-to-voltage relationship through parameters a, b, and c in Equation (A.1).

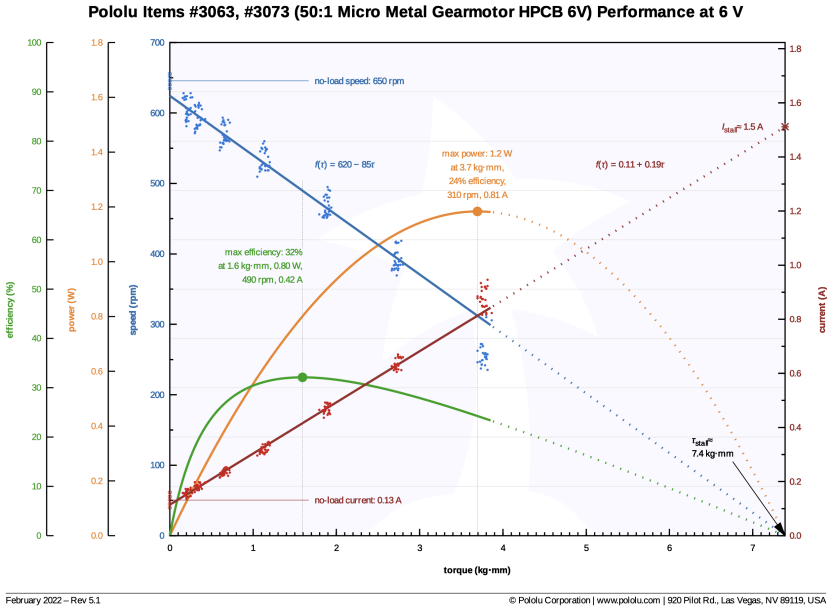

Figure A.1: Datasheet of 50:1 micro metal gearmotor HPCB 6V [49]

Using the stall condition data and Equation (2.15), the motor's resistance R_a is calculated as:

$$R_a = \frac{u_a}{i_{a,stall}} = 4\Omega.$$

From the no-load condition, employing the same Equation (2.15), the electromotive force constant $k\phi$ is derived. It is important to adjust for the additional motor's gear ratio in the Balboa ($GR = 41/25$):

$$k\phi = \frac{u_a - R_a i_{a,no-load}}{\dot{\varphi}_{no-load}} GR = 0.132 \text{ Nm/A}.$$

The parameter a is calculated as 15.15 V/Nm. Given the challenges in estimating the parameters b and c directly from theoretical models, these parameters are proposed to be determined experimentally. The relationship between the motor speed $\dot{\varphi}$ and the control input u_a is captured through experimental data, illustrated in Figure A.2. Motivated by the experimental observations, a linear model is proposed for simplifying the motor equation:

$$u_a(t) = b \cdot \dot{\varphi}(t) + c. \tag{A.1}$$

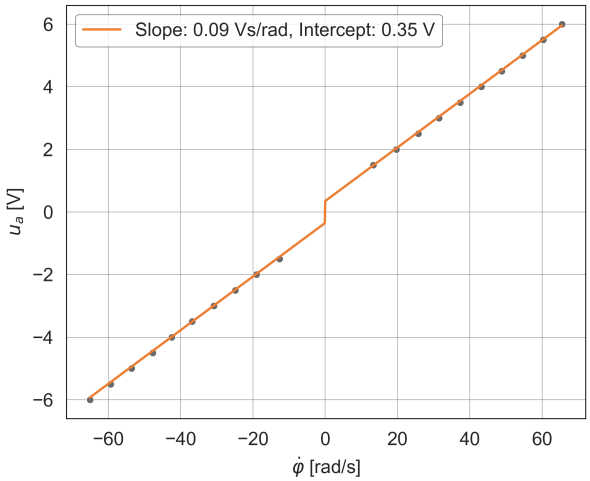

Figure A.2: Motor speed measurement with different voltage.

Based on the linear regression analysis of the experimental measurements, initial values for the parameters were obtained: $b = 0.09\,\text{V}/(\text{rad/s})$ and $c = 0.35\,\text{V}$. Through minor adjustments to these values, optimized parameters were found that reduced oscillations in the Balboa 32U4 system, enhancing control over motor speeds. These adjusted parameters are retained for future tests, as they yield improved results:

$$a = 7.5\,\text{V/Nm}, \quad b = 0.19\,\text{V}/(\text{rad/s}), \quad c = 0.4\,\text{V}$$

From these refined values, the friction factor K_ν and the resistive torque C_r were estimated as follows:

$$K_\nu = 1.91 \cdot 10^{-3} \text{ Nm}/(\text{rad/s}), \quad C_r = 14.85 \cdot 10^{-3} \text{ Nm}.$$

Appendix B

Data-driven pole placement feedback proof

Consider the transformation matrix $M = [m_1, \ldots, m_n]$ with $\text{rank}(M) = n$. The relationship

$$(X_1 - \lambda_i X_0)m_i = [A - \lambda_i I, B] \begin{bmatrix} X_0 \\ U_0 \end{bmatrix} m_i$$

implies that the system (A, B) is controllable, hence $\text{rank}[A - \lambda_i I, B] = n$ and $[A - \lambda_i I, B]$ has a nontrivial m-dimensional right null space. Therefore, it is feasible to select the columns of M such that:

$$\begin{bmatrix} X_0 \\ U_0 \end{bmatrix} m_i \in \mathcal{N}\{[A - \lambda_i I, B]\}.$$

Given $u_{0,T-1}$ is persistently exciting of order $n+1$, it ensures $\text{rank}[X_0, U_0]^T = n+m$, which permits the choice of m_i to fulfill the above condition, thereby proving the existence of M.

To demonstrate that $\text{rank}(M) = n$, observe:

$$\dim\mathcal{N}\left\{\begin{bmatrix} X_0 \\ U_0 \end{bmatrix}\right\} \geq mn,$$

indicating the existence of n linearly independent vectors m_i satisfying the given criteria.

To establish the second part of the claim, consider:

$$0 = (X_1 - \lambda_i X_0)m_i = (AX_0 + BU_0 - \lambda_i X_0)m_i = (A - \lambda_i)X_0 m_i + BU_0 m_i,$$

where the last identity stems from $X_1 = AX_0 + BU_0$, which holds because X_0, X_1, U_0 are generated by system measurements. Using the relation $-U_0 m_i = KX_0 m_i$, and since $\text{rank}(M) = n$ and $\text{rank}(X_0 M) = n$, $(X_0 M)^\dagger$ acts as a right inverse of $X_0 M$. Substituting this identity yields:

$$(A - BK - \lambda_i) X_0 m_i = 0,$$

thus substantiating the claim.

Resources for reproducibility

The simulations and data analyses reported in this paper were performed using MATLAB, python and Arduino IDE. The source code is available to ensure reproducibility of the results and to facilitate further research. The code repository to replicate the results of this study can be accessed at:

`https://github.com/AurelienSoenen/Data-Driven_Balboa32U4`.

Additionally, the code used for the experimental tests on the Balboa 32U4 robot is also available at:

`https://github.com/AurelienSoenen/Balboa32U4`.

Declaration of generative AI tools

During the preparation of this work the author used ChatGPT 4 for drafting, editing, and code generation purposes. After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the content of the publication.

Bibliography

[1] Y. Ha and S. Yuta. Trajectory tracking control for navigation of self-contained mobile inverse pendulum. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, volume 3, pages 1875–1882. IEEE.

[2] Segway personal technology.

[3] F. Grasser, A. D'Arrigo, S. Colombi, and A.C. Rufer. JOE: a mobile, inverted pendulum. 49(1):107–114.

[4] Ronald Ping Man Chan, Karl A. Stol, and C. Roger Halkyard. Review of modelling and control of two-wheeled robots. 37(1):89–103.

[5] Johan Akesson, Anders Blomdell, and Rolf Braun. Design and control of YAIP - an inverted pendulum on two wheels robot. In *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, pages 2178–2183. IEEE.

[6] John Lien, Linda Tu, William Ross, and Colin Burvill. Implementation issues for an inexpensive inverted-pendulum mobile robot. In *2006 International Conference on Information and Automation*, pages 372–377. IEEE.

[7] Han Jian-hai, Zhao Shu-shang, Li Ji-shun, and Li Hang. Research on developed parallel two-wheeled robot and its control system. In *2008 IEEE International Conference on Automation and Logistics*, pages 2471–2475. IEEE.

[8] Rich Ooi. Balancing a two-wheeled autonomous robot.

[9] Jingtao Li, Xueshan Gao, Qiang Huang, Qinjun Du, and Xingguang Duan. Mechanical design and dynamic modeling of a two-wheeled inverted pendulum mobile robot. In *2007 IEEE International Conference on Automation and Logistics*, pages 1614–1619. IEEE.

[10] T. Takei, R. Imamura, and S. Yuta. Baggage transportation and navigation by a wheeled inverted pendulum mobile robot. 56(10):3985–3994.

[11] Mi-Ching Tsai and Jia-Sheng Hu. Pilot control of an auto-balancing two-wheeled cart. 21(7):817–827.

[12] Jia-Sheng Hu and Mi-Ching Tsai. Design of robust stabilization and fault diagnosis for an auto-balancing two-wheeled cart. 22(2):319–338.

[13] K. Pathak, J. Franch, and S.K. Agrawal. Velocity and position control of a wheeled inverted pendulum by partial feedback linearization. 21(3):505–513.

[14] Yeonhoon Kim, Soo Hyun Kim, and Yoon Keun Kwak. Dynamic analysis of a nonholonomic two-wheeled inverted pendulum robot. 44(1):25–46.

[15] S. W. Nawawi, M. N. Ahmad, and J. H. S. Osman. Development of a two-wheeled inverted pendulum mobile robot. In *2007 5th Student Conference on Research and Development*, pages 1–5. IEEE.

[16] M. Muhammad, S. Buyamin, M.N. Ahmad, and S.W. Nawawi. Dynamic modeling and analysis of a two-wheeled inverted pendulum robot. In *2011 Third International Conference on Computational Intelligence, Modelling & Simulation*, pages 159–164. IEEE.

[17] Mishari Alarfaj and George Kantor. Centrifugal force compensation of a two-wheeled balancing robot. In *2010 11th International Conference on Control Automation Robotics & Vision*, pages 2333–2338. IEEE.

[18] Junainah Jahaya, S.W. Nawawi, and Zuwairie Ibrahim. Multi input single output closed loop identification of two wheel inverted pendulum mobile robot. In *2011 IEEE Student Conference on Research and Development*, pages 138–143. IEEE.

[19] Qin Yong, Liu Yanlong, Zang Xizhe, and Liu Ji. Balance control of two-wheeled self-balancing mobile robot based on TS fuzzy model. In *Proceedings of 2011 6th International Forum on Strategic Technology*, pages 406–409. IEEE.

[20] Junfeng Wu, Yuxin Liang, and Zhe Wang. A robust control method of two-wheeled self-balancing robot. In *Proceedings of 2011 6th International Forum on Strategic Technology*, pages 1031–1035. IEEE.

[21] Jingtao Li, Xueshan Gao, Qiang Huang, and Osamu Matsumoto. Controller design of a two-wheeled inverted pendulum mobile robot. In *2008 IEEE International Conference on Mechatronics and Automation*, pages 7–12. IEEE.

[22] Tao Feng, Tao Liu, Xu Wang, Zhao Xu, Meng Zhang, and Sheng-chao Han. Modeling and implementation of two-wheel self-balancing robot equipped with supporting arms. In *2011 6th IEEE Conference on Industrial Electronics and Applications*, pages 713–718. IEEE.

[23] N. M. Abdul Ghani, D. Ju, H. Z Othman, and M. A. Ahmad. Two wheels mobile robot using optimal regulator control. In *2010 10th International Conference on Intelligent Systems Design and Applications*, pages 1066–1070. IEEE.

[24] Wu Junfeng and Zhang Wanying. Research on control method of two-wheeled self-balancing robot. In *2011 Fourth International Conference on Intelligent Computation Technology and Automation*, pages 476–479. IEEE.

[25] Luis F Lupian and Rodrigo Avila. Stabilization of a wheeled inverted pendulum by a continuous-time infinite-horizon LQG optimal controller. In *2008 IEEE Latin American Robotic Symposium*, pages 65–70. IEEE.

[26] Xiaogang Ruan and Jing Chen. H∞ robust control of self-balancing two-wheeled robot. In *2010 8th World Congress on Intelligent Control and Automation*, pages 6524–6527. IEEE.

[27] Tomoya Kanada, Yusuke Watanabe, and Gan Chen. Robust h2 control for two-wheeled inverted pendulum using LEGO mindstorms. In *2011 Australian Control Conference*, pages 136–141.

[28] Y. Takahashi, N. Ishikawa, and T. Hagiwara. Inverse pendulum controlled two wheel drive system. In *SICE 2001. Proceedings of the 40th SICE Annual Conference. International Session Papers (IEEE Cat. No.01TH8603)*, pages 112–115. Soc. Instrum. & Control Eng.

[29] Ahmad Nor Kasruddin Nasir, Mohd. Ashraf Ahmad, Riduwan Ghazali, and Nasrul Salim Pakheri. Performance comparison between fuzzy logic controller (FLC) and PID controller for a highly nonlinear two-wheels balancing robot. In *2011 First International Conference on Informatics and Computational Intelligence*, pages 176–181. IEEE.

[30] Chih-Hui Chiu and Ya-Fu Peng. Design and implement of the self-dynamic controller for two-wheel transporter. In *2006 IEEE International Conference on Fuzzy Systems*, pages 480–483. IEEE.

[31] S.W Nawawi, M.N Ahmad, J.H.S Osman, A.R Husain, and M.F Abdollah. Controller design for two-wheels inverted pendulum mobile robot using PISMC.

In *2006 4th Student Conference on Research and Development*, pages 194–199. IEEE.

[32] Nguyen Gia Minh Thao, Duong Hoai Nghia, and Nguyen Huu Phuc. A PID backstepping controller for two-wheeled self-balancing robot. In *International Forum on Strategic Technology 2010*, pages 76–81. IEEE.

[33] Wu Wei, Ma Xiaoning, and Wang Jijun. Intelligent control in two-wheel self-balanced robot. In *2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering*, page 5610286. IEEE.

[34] Zareena Kausar, Karl Stol, and Nitish Patel. Stability region estimation of statically unstable two wheeled mobile robots. In *2011 IEEE International Conference on Robotics and Biomimetics*, pages 1379–1384. IEEE.

[35] Sun Liang and Feimei Gan. Balance control of two-wheeled robot based on reinforcement learning. In *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, pages 3254–3257. IEEE.

[36] Y Tanaka, Y Ohata, T Kawamoto, and Y Hirata. Adaptive control of 2-wheeled balancing robot by cerebellar neuronal network model. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 1589–1592. IEEE.

[37] Xiaogang Ruan and Jing Chen. On-line NNAC for two-wheeled self-balancing robot based on feedback-error-learning. In *2010 2nd International Workshop on Intelligent Systems and Applications*, pages 1–4. IEEE.

[38] Chaoquan Li, Fangxing Li, Shusan Wang, Fuquan Dai, Yang Bai, Xueshan Gao, and Kejie Li. Dynamic adaptive equilibrium control for a self-stabilizing robot. In *2010 IEEE International Conference on Robotics and Biomimetics*, pages 609–614. IEEE.

[39] M. M. Azimi and H. R. Koofigar. Model predictive control for a two wheeled self balancing robot. In *2013 First RSI/ISM International Conference on Robotics and Mechatronics (ICRoM)*, pages 152–157. IEEE.

[40] Vishaal Krishnan and Fabio Pasqualetti. On direct vs indirect data-driven predictive control. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 736–741. IEEE.

[41] Florian Dörfler, Jeremy Coulson, and Ivan Markovsky. Bridging direct and indirect data-driven control formulations via regularizations and relaxations. 68(2):883–897. Conference Name: IEEE Transactions on Automatic Control.

[42] T M Maupong. Data-driven control: A behavioral approach.

[43] Claudio De Persis and Pietro Tesi. Formulas for data-driven control: Stabilization, optimality, and robustness. 65(3):909–924. Conference Name: IEEE Transactions on Automatic Control.

[44] Jeremy Coulson, John Lygeros, and Florian Dorfler. Data-enabled predictive control: In the shallows of the DeePC. In *2019 18th European Control Conference (ECC)*, pages 307–312. IEEE.

[45] Giacomo Baggio, Vaibhav Katewa, and Fabio Pasqualetti. Data-driven minimum-energy controls for linear systems. 3(3).

[46] Gianluca Bianchin. Data-driven exact pole placement for linear systems. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 685–690. IEEE.

[47] Sayak Mukherjee and Ramij R. Hossain. Data-driven pole placement in LMI regions with robustness guarantees. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 4010–4015. ISSN: 2576-2370.

[48] Jonathan How. Topic 15: Feedback control.

[49] Pololu - 50:1 micro metal gearmotor HPCB 6v with extended motor shaft.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**