Stabilization of a Rotary Inverted Pendulum with Neuromorphic Vision Feedback

DIPLOMA THESIS

Conducted in partial fulfillment of the requirements for the degree of a

Diplom-Ingenieur (Dipl.-Ing.)

supervised by

Dipl.-Ing. Gerald Ebmer
Associate Prof. Dipl.-Ing. Dr.-Ing. Wolfgang Kemmetmüller

submitted at the

TU Wien
Faculty of Electrical Engineering and Information Technology
Automation and Control Institute

by
Lukas Pechhacker
Matriculation number 01326655

Wien, October 2024

Complex Dynamical Systems Group
A-1040 Wien, Gußhausstr. 27–29, Internet: https://www.acin.tuwien.ac.at

Preamble

I want to acknowledge Associate Prof. Dipl.-Ing. Dr.-Ing. Wolfgang Kemmetmüller for his lectures on optimal control and estimation, which intensified my interest in this topic. Further, I am thankful to have conducted this thesis under his supervision.

I want to express great gratitude to my supervisor and, most importantly, my friend Dipl.-Ing. Gerald Ebmer. His research insights sparked my fascination in the exciting field of applying event-based cameras to real-time control systems. I greatly appreciate my friend Andreas Hirtenlehner, who supported me during this thesis while running our business with outstanding dedication.

I am deeply grateful to my parents, whose unwavering support made my studies and this path possible. Last, I would like to thank Pia Weinhart for her wholehearted support.

Wien, October 2024

Abstract

The rotary inverted pendulum, known as the *Furuta pendulum*, is a well-studied, non-linear, and underactuated system consisting of a driven arm with a freely rotating pendulum attached. Previous studies focused on controlling the *Furuta pendulum* by directly measuring the pendulum's angle. This thesis aims to stabilize the *Furuta pendulum* using neuromorphic vision feedback from an *event-based camera*.

In this thesis, a mechanical design of the *Furuta pendulum* is introduced. Subsequently, a mathematical model is derived, and the respective parameters are estimated using a non-linear least squares method. The reference trajectories, including the swing-up and the swing-down trajectories, are obtained by solving an optimal control problem. The system states are estimated using an *Extended Kalman Filter* to stabilize the *Furuta pendulum* about the reference trajectories with a time-varying *LQR*. An exponential decay for the accumulator of the traditional *Hough transform* algorithm is introduced as the proposed method to obtain the pendulum's angle from the camera events directly.

The proposed method is validated through simulation and experiments on the physical *Furuta pendulum*. It is shown that the stabilization of the *Furuta pendulum* with event camera feedback about the reference trajectories can be achieved by applying the proposed method.

Kurzzusammenfassung

Das rotierende, inverse Pendel, auch *Furuta-Pendel* genannt, ist ein gut erforschtes, nichtlineares und unteraktuiertes System, welches aus einem aktuierten Arm besteht, an dem ein frei rotierendes Pendel befestigt ist. Frühere Arbeiten konzentrierten sich auf die Regelung des *Furuta-Pendels* durch direkte Messung des Pendelwinkels. Diese Arbeit zielt jedoch darauf ab, das *Furuta-Pendel* durch neuromorphes Vision-Feedback von einer *event-basierten Kamera* zu stabilisieren.

Es wird der für diese Arbeit konstruierte mechanischer Aufbau des *Furuta-Pendels* vorgestellt. Anschließend wird daraus ein mathematisches Modell hergeleitet und die Modellparameter werden mithilfe der *Methode der nichtlinearen kleinsten Quadrate* geschätzt. Die Solltrajektorien, einschließlich der Auf- und Abschwingtrajektorie, werden durch Lösung eines Optimalsteuerungsproblems ermittelt. Der Zustand des Systems wird mit einem *Extended Kalman Filter* geschätzt, um das *Furuta-Pendel* um die Solltrajektorien mit einem zeitvarianten *LQR* zu stabilisieren. Ein adaptierter *Hough-Transformations-Algorithmus* mit exponentiellem Abklingen der Akkumulatoreinträge zur Detektion von Geraden wird als Methode eingeführt, um den Winkel des Pendels direkt aus den Kamera-Events unter Berücksichtigung der projektiven Geometrie zu berechnen.

Die vorgestellte Methode wird erfolgreich in Simulation und am physischen *Furuta-Pendel* validiert. Es wird gezeigt, dass die Stabilisierung des *Furuta-Pendels* mit event-basiertem Kamera-Feedback um die Referenztrajektorien durch Anwendung der vorgestellten Methode erreicht werden kann.

Contents

List of Figures

List of Tables

1 Introduction

Robotic systems often operate in environments specifically designed for robots without perceiving the surrounding scene. However, with adequate perception, these systems would be able to perform increasingly complex tasks – just like humans do. One key modality for visual perception is the camera. More specifically, conventional frame cameras are often used to enable vision-based feedback, which typically provides camera images to the image processing algorithm at equidistant time intervals [1].

While frame-based cameras are still popular choices for vision feedback, they come with various limitations. Full frames are transferred, and at least parts are processed at each time step, leading to high computational effort. Latency induced due to low frame rates or high computation times can be inadequate for specific applications. Moreover, images from frame-based cameras can suffer from motion blur, posing additional challenges for the image processing algorithm to perceive the scene. [1]

Event-based cameras, further referred to as event cameras, are a new type of camera that transmit changes in the brightness of the scene for each individual pixel asynchronously rather than full frames. This approach has several advantages, such as low latency in the µs range, a high dynamic range, and reduced motion blur. Using asynchronous feedback from event cameras can be referred to as neuromorphic vision feedback because of the bio-inspired working principle. [1]

The rotary inverted pendulum, also called the *Furuta pendulum* [2], is an underactuated, non-linear system, as shown in Figure 1.1. It consists of an actuated arm and an attached pendulum that can freely rotate around the arm's longitudinal axis [2]. The *Furuta pendulum* is well-studied (see [2–4]) in control engineering and is known to have an unstable equilibrium point in the upright pendulum position.

Figure 1.2 shows a subset of the captured frames from a conventional frame camera and a subset of the events from the event camera of the *Furuta pendulum* swinging back and forth around the lower equilibrium position. The event camera generates a stream of events rather than full frames, as the frame camera does. Green markings denote an increase in the brightness of the corresponding pixel, while blue markings represent a decrease in brightness. The images at $t = 6.8\,\mathrm{s}$ and $t = 7.2\,\mathrm{s}$ hardly show motion blur because of the low angular velocity. The image shows some noticeable motion blur at $t = 7\,\mathrm{s}$ due to the pendulum's high angular velocity, as depicted in Figure 1.2.

1.1 Aim of this Thesis

To the best of the author's knowledge, the inverted pendulum has only been stabilized around the upper equilibrium using feedback from an event camera, as shown in [5].

This thesis, however, aims to extend prior work and additionally stabilize the rotary inverted pendulum about reference trajectories, including the swing-up, with event-based

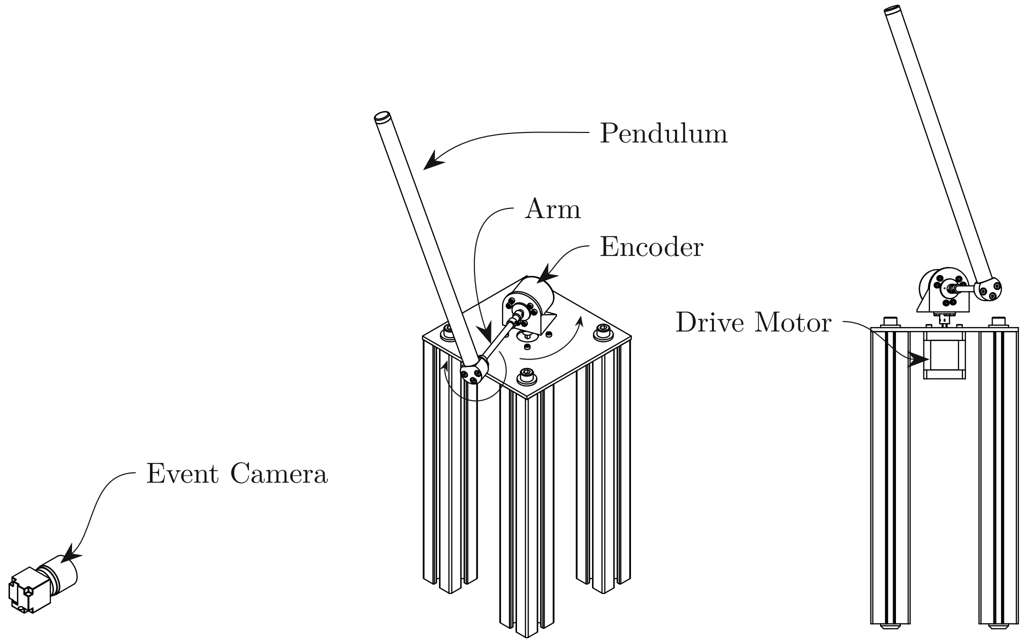

Figure 1.1: An overview showing the main components of the *Furuta pendulum* and the event camera, as utilized in this work.

vision feedback. Moreover, this work investigates how the *Hough transform* can be used with event cameras in low-latency applications to track objects despite additional events generated by the movement of background objects in the environment.

1.2 Literature and Previous Works

Since the inverted pendulum is a common experimental platform for demonstrating non-linear control, it has also been used for experiments with vision-based feedback.

In [6], a pendulum on a cart is stabilized around its upper equilibrium using an infrared LED on the tip. The camera captures the tip of the pendulum from the top. From the cart's position and the position of the pendulum's tip, the pendulum's angle is calculated and used as feedback for the controller [6]. However, this approach requires the infrared LED as an active element. Furthermore, this setup would not allow for the pendulum's swing-up because of the ambiguity of the tip's position information and occlusions. A fisheye camera model is used in [7], where an inverted pendulum on a cart is stabilized by introducing a calibration-free method to obtain the pendulum's angle. However, the vision processing algorithm detects the pendulum using two LEDs, which not only requires active elements on the pendulum but also induces a time delay of 20 ms. The work in [8] uses a smartphone attached to the pendulum to detect a static marker in the background to stabilize the inverted pendulum on a cart around the upper equilibrium point. This

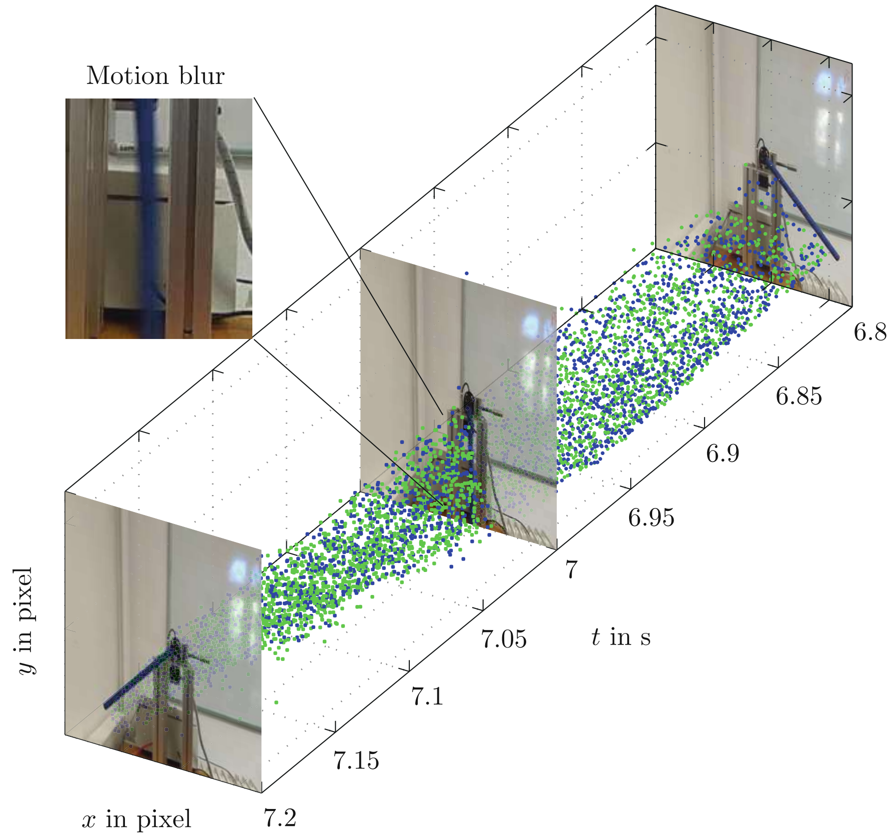

Figure 1.2: A subsampled set of the events generated by the event camera and the images taken from a traditional frame camera.

eye-in-hand method detects the translation and the rotation of the pendulum's state and transmits the measurement data over a wireless network [8], inducing a time delay of approximately 30 ms. Hence, this approach is only applicable to systems with slow dynamics. In [9], the swing-up of the pendulum mounted on an industrial robot using vision feedback is achieved. However, after the swing-up, the feedback signal is switched to a potentiometer angle measurement to stabilize the pendulum initially before switching again to the camera-based angle measurements [9]. In [10], the swing-up of an inverted pendulum on a cart is performed using a conventional frame camera and a cascaded particle filter to estimate the system's state. The camera measurement updates at a rate of 30 Hz are sufficient to stabilize this specific pendulum [10].

In [11], an event camera, compared to a conventional frame camera, achieved significant data reduction and faster object tracking with a two-axis robot, showing the benefits of event cameras. Another comparative analysis is done in [12], where eye-in-hand visual servoing was applied to an industrial robot. The comparison shows that the image-based *Kanade-Lucas-Tomase* tracker with the *Harris* corner detector fails to detect and track the objects under low light conditions or when the tracked object moves too fast [12]. The loss of tracking data is caused by latency and motion blur [12].

A closed-loop control of a dual-copter platform utilizing an event camera, aligning the dual-copter's orientation to a half-black and half-white-colored disc, is shown in [13]. The disc's orientation is determined by means of a sliding-window *Hough transform* and acts as an input to a *Kalman filter* to estimate the system's state [13]. Using this approach, all events in this sliding window have the same weighting factor. Thus, the latest events are not prioritized, which could lead to an error for larger window sizes. In [5], the angle of a pencil is obtained from the event stream without the need for a sliding window. Unlike in [13], the events are directly transformed into the *Hough space*, which is updated on every new event with an additional decay. This approach has the benefit that the latest events have a higher weighting factor in the *Hough space* than older events. Using this approach, a pencil was stabilized in the upright equilibrium position in [5]. Another example of event-camera-based closed-loop tacking control can be found in [14], where the events are directly the inputs of a spiking neural network, whose neurons are in a *Hough space*-like alignment. This method, however, requires a neuromorphic processor to benefit from the spiking neural network architecture.

1.3 Overview of this Thesis

Chapter 2 gives an overview of the mechanical design and the considerations taken regarding subsequent tracking with the event camera. In order to derive the mathematical model of the *Furuta pendulum* using the *Euler-Lagrange* equations, the coordinate frames are introduced. The model parameters are estimated using a non-linear least squares parameter estimation method.

In Chapter 3, an adapted *Hough transform* with exponential decay is introduced. This method obtains the pendulum's angle by transforming the events generated by the event camera into the *Hough space*. An analytical function is derived from the pinhole camera model to compensate for the perspective projection of the pendulum.

In Chapter 4, an optimization problem is formulated to obtain the *Furuta pendulum's* reference swing-up, transfer, and swing-down trajectories. Due to model errors and unmodeled dynamics, a feedback controller is introduced to stabilize the pendulum about the reference trajectories and the equilibrium points. Next, an *Extended Kalman Filter* is derived to estimate the *Furuta pendulum's* state.

The proposed method is validated by simulation and by means of experiments on the physical *Furuta pendulum* in Chapter 5. A comparative analysis between the stabilization about the reference trajectories with encoder feedback and with event camera feedback validates the closed-loop performance.

Chapter 6 summarizes this thesis and concludes with the main findings of stabilizing the *Furuta pendulum* about the reference trajectories using event camera feedback.

2 Design and Mathematical Modeling

This chapter describes the mechanical design of the *Furuta pendulum* and emphasizes the decisions made regarding the subsequent vision processing. A mathematical model is derived using the *Euler-Lagrange* equations to allow for model-based control of the *Furuta pendulum*. This chapter concludes with a non-linear least-squares parameter estimation based on physical measurements from the assembled *Furuta pendulum*.

2.1 Mechanical Design

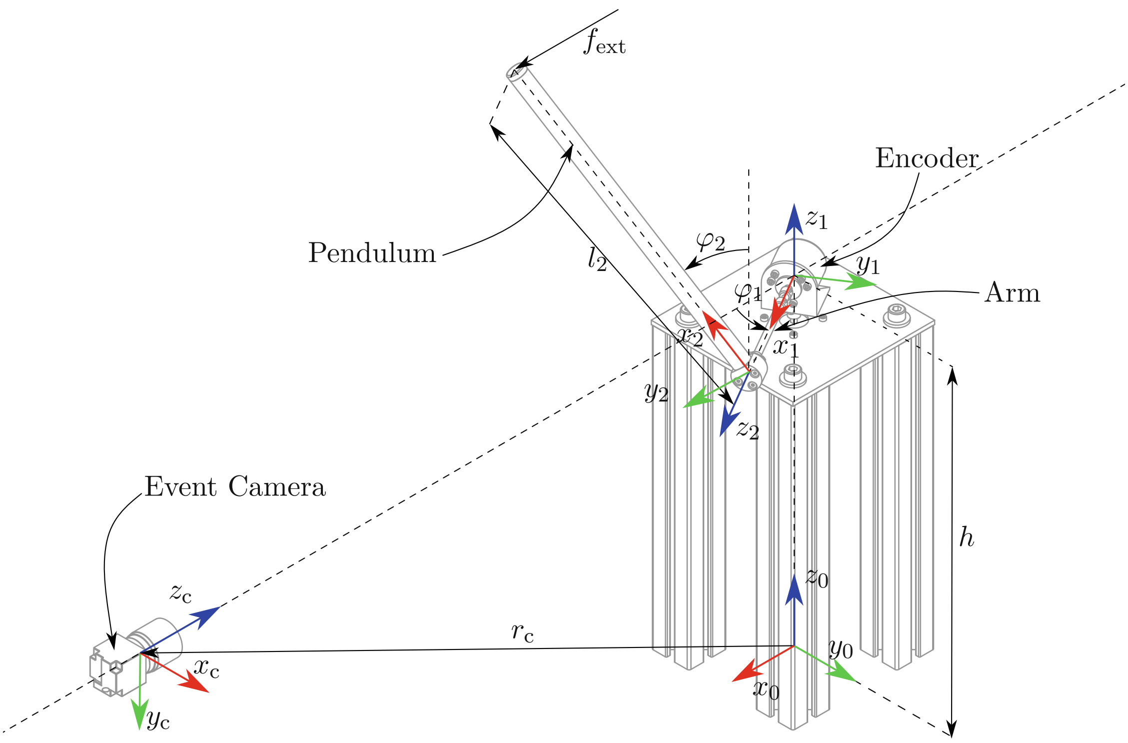

Figure 2.1: The mechanical design of the *Furuta pendulum* with the event camera and the coordinate systems for the mathematical modeling.

Figure 2.1 shows a schematic overview of the mechanical construction of the *Futura pendulum*. It essentially consists of a drive motor, an arm, and the pendulum itself. The drive motor is mounted on a baseplate, bolted to four aluminum profiles, and drives the arm, allowing it to rotate in the horizontal plane, as described by the angle φ_1. The rotary encoder connects the arm's rod with the drive motor to let the pendulum, which has a fixed connection to the rod of the arm, freely rotate around the rotating axis of the encoder. The pendulum's angle φ_2 is measured from the vertical axis to the pendulum, as depicted in Figure 2.1. In order to have a reference measurement of the pendulum's angle φ_2, an encoder is installed. Since this thesis aims to obtain the pendulum's angle by processing the events of the camera, the visual appearance of the pendulum is crucial. Therefore, dark blue paintwork is chosen to ensure that the pendulum stands out against the light-colored background. Reflections on the pendulum are minimized by choosing a matt finish. A cylindrical shape of the pendulum is chosen to ensure that the projection on the camera sensor has clear edges.

In order to simplify the mathematical modeling, several *Cartesian* coordinate systems are introduced, as shown in Figure 2.1. The space-fixed base coordinate system (x_0, y_0, z_0), further denoted as Σ_0, acts as the reference frame. The body-fixed coordinate frame (x_1, y_1, z_1), further denoted as Σ_1, is attached to the encoder at the height h with respect to the z_0-direction. The origin of the coordinate frame (x_2, y_2, z_2), further denoted as Σ_2, is shifted by l_1 in the x_1-direction to the center of the pendulum's rod.

The event camera's optical axis z_c is parallel to the x_0-axis and it is placed at the distance r_c with respect to the base frame Σ_0. The origin of the camera's coordinate system (x_c, y_c, z_c) is at the optical center of the camera.

Figure 2.2: The assembled *Futura pendulum*.

Figure 2.2 shows an image of the assembled *Furuta pendulum*. The baseplate is built from aluminum and bolted to the aluminum profiles. The drive motor is a *NEMA 17* stepper motor with 200 steps per revolution [15] and is linked to the 3D-printed mounting bracket for connecting the encoder with the drive motor adaptor. The 600 pulses per revolution incremental encoder is connected via a rod to the 3D-printed pendulum adaptor.

2.2 Pendulum Modeling

In this section, the equations of motion based on the mechanical model of the *Furuta pendulum* are derived using *Lagrangian* mechanics. An overview of the *Furuta Pendulum* with the coordinate systems and respective dimensions is depicted in Figure 2.1. Consider the base coordinate system Σ_0 and the arm-fixed coordinate system Σ_1, whose translation of the origin with respect to Σ_0 is described by the vector

$$\mathbf{d}_0^1 = \begin{bmatrix} 0 \\ 0 \\ h \end{bmatrix} . \tag{2.1}$$

In order to describe the relative rotations of the coordinate systems, the basic rotation matrices about the y-axis and the z-axis are written as [16]

$$\mathbf{R}_y(\varphi) = \begin{bmatrix} \cos(\varphi) & 0 & \sin(\varphi) \\ 0 & 1 & 0 \\ -\sin(\varphi) & 0 & \cos(\varphi) \end{bmatrix} , \tag{2.2a}$$

$$\mathbf{R}_z(\varphi) = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) & 0 \\ \sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{2.2b}$$

Let the rotation matrix $\mathbf{R}_0^1 = \mathbf{R}_z(\varphi_1)$ describe the rotation φ_1 of the coordinate system Σ_1 around the z_0-axis, yielding to the homogeneous transformation matrix [16]

$$\mathbf{T}_0^1 = \begin{bmatrix} \mathbf{R}_0^1 & \mathbf{d}_0^1 \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} . \tag{2.3}$$

By introducing the vector

$$\mathbf{d}_1^2 = \begin{bmatrix} l_1 \\ 0 \\ 0 \end{bmatrix} \tag{2.4}$$

originating from the arm's coordinate system Σ_1 to the origin of the pendulum's coordinate system Σ_2 with the respective rotation matrix $\mathbf{R}_1^2 = \mathbf{R}_y(90°)\mathbf{R}_z(\varphi_2 + 180°)$, the homogeneous transformation matrix

$$\mathbf{T}_1^2 = \begin{bmatrix} \mathbf{R}_1^2 & \mathbf{d}_1^2 \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} \tag{2.5}$$

is obtained. The arm's center of gravity in its coordinate system Σ_1 is

$$\mathbf{p}_{1,1c} = \begin{bmatrix} l_{c1} \\ 0 \\ 0 \end{bmatrix} , \tag{2.6}$$

where l_{c1} is the respective distance to the origin of Σ_1. Similarly, the pendulum's center of gravity in its coordinate system Σ_2 can be written as

$$\mathbf{p}_{2,2c} = \begin{bmatrix} l_{c2} \\ 0 \\ 0 \end{bmatrix} , \tag{2.7}$$

allowing us to express the homogeneous coordinates [16] of the centers of gravity as

$$\mathbf{P}_{i,ic} = \begin{bmatrix} \mathbf{p}_{i,ic} \\ 1 \end{bmatrix} , \quad i = 1, 2 . \tag{2.8}$$

Finally, the centers of gravity, transformed into the base coordinate frame Σ_0, are given by

$$\mathbf{P}_{0,1c} = \mathbf{T}_0^1 \mathbf{P}_{1,1c} \tag{2.9}$$

and

$$\mathbf{P}_{0,2c} = \mathbf{T}_0^1 \mathbf{T}_1^2 \mathbf{P}_{2,2c} = \mathbf{T}_0^2 \mathbf{P}_{2,2c} . \tag{2.10}$$

The velocities of the centers of gravity of the pendulum and arm

$$\mathbf{v}_{0,ic} = \frac{\mathrm{d}}{\mathrm{d}t} \mathbf{p}_{0,ic} , \quad i = 1, 2 \tag{2.11}$$

in the base frame Σ_0 are expressed using the time derivative of the centers of gravity in the respective coordinate systems $\mathbf{p}_{0,ic}$.

In order to determine the rotational kinetic energy, the angular velocity matrix [16] for the arm and the pendulum in the base coordinates Σ_0 is written as

$$\mathbf{S}(\boldsymbol{\omega}_{0,i}) = \left(\frac{\mathrm{d}}{\mathrm{d}t} \mathbf{R}_0^i \right) \left(\mathbf{R}_0^i \right)^{\mathrm{T}} , \quad i = 1, 2 \tag{2.12}$$

and has the form of a skew-symmetric matrix [16]

$$\mathbf{S}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} , \tag{2.13}$$

with the angular velocity vector $\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^{\mathrm{T}}$. This allows for extracting the angular velocity vectors of the arm $\boldsymbol{\omega}_{0,1}$ and the pendulum $\boldsymbol{\omega}_{0,2}$ in the base frame Σ_0.

The arm and pendulum's body-fixed coordinate systems are chosen such that the principal axes of the bodies are aligned with the coordinate systems, leading to diagonal inertia matrices [16]

$$\mathbf{I}_i = \begin{bmatrix} I_{i,xx} & 0 & 0 \\ 0 & I_{i,yy} & 0 \\ 0 & 0 & I_{i,zz} \end{bmatrix} , \quad i = 1, 2 . \tag{2.14}$$

The kinetic energy is the sum of the rotational and translational energies of the pendulum and the arm [16]

$$\mathcal{T} = \sum_{i=1}^{2} \frac{1}{2} (\boldsymbol{\omega}_{0,i})^{\mathrm{T}} \mathbf{R}_0^i \mathbf{I}_i (\mathbf{R}_0^i)^{\mathrm{T}} \boldsymbol{\omega}_{0,i} + \sum_{i=1}^{2} \frac{1}{2} m_i \mathbf{v}_{0,ic}^{\mathrm{T}} \mathbf{v}_{0,ic} \ . \tag{2.15}$$

Similarly, the potential energy

$$\mathcal{V} = -\sum_{i=1}^{2} \mathbf{a}_{\mathrm{g}}^{\mathrm{T}} \mathbf{p}_{0,ic} m_i \tag{2.16}$$

is expressed using the constant gravitational acceleration $\mathbf{a}_{\mathrm{g}} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^{\mathrm{T}}$, the centers of gravity $\mathbf{p}_{0,ic}$, and the respective masses m_i.

The *Lagrangian*, denoted by $\mathcal{L}$, is the difference between the system's kinetic and the potential energy and can be written as [16]

$$\mathcal{L} = \mathcal{T} - \mathcal{V} \ . \tag{2.17}$$

Previous research modeled the pendulum and the arm with viscous damping using the damping coefficients $d_i, i = 1, 2$ [2]. Thus, the dissipative energy can be described using *Rayleigh's dissipation function* [17]

$$\mathcal{R} = \sum_{i=1}^{2} \frac{1}{2} d_i \omega_i^2 \ , \quad i = 1, 2 \tag{2.18}$$

with $\omega_i = \dot{\varphi}_i$ to finally derive the equations of motion [17]

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\partial \omega_1}\right) - \frac{\partial \mathcal{L}}{\partial \varphi_1} + \frac{\partial \mathcal{R}}{\partial \omega_1} = \tau_{\mathrm{m}} \ , \tag{2.19a}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\partial \omega_2}\right) - \frac{\partial \mathcal{L}}{\partial \varphi_2} + \frac{\partial \mathcal{R}}{\partial \omega_2} = \tau_{\mathrm{ext}} \tag{2.19b}$$

from the *Lagrangian* $\mathcal{L}$, *Rayleigh's dissipation function* $\mathcal{R}$, the motor torque τ_{m}, and the external torque $\tau_{\mathrm{ext}} = f_{\mathrm{ext}} l_2$ induced by an external force at the tip of the pendulum that is perpendicular to the x_1-x_2 plane. The stepper motor, that actuates the arm, is controlled by a stepper motor driver. The interaction of the dynamics from the pendulum to the arm is therefore neglected allowing to use the arm's angular acceleration as the system input u resulting in the non-linear model of the pendulum

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u, f_{\mathrm{ext}}) \tag{2.20}$$

with the state vector $\mathbf{x} = \begin{bmatrix} \varphi_1 & \omega_1 & \varphi_2 & \omega_2 \end{bmatrix}^{\mathrm{T}}$ and the non-linear system dynamics

$$\mathbf{f}(\mathbf{x}, u, f_{\mathrm{ext}}) = \begin{bmatrix} \omega_1 \\ u \\ \omega_2 \\ \frac{(-\omega_1^2 (I_{2,xx} - I_{2,yy} - l_{c2}^2 m_2)\sin(\varphi_2) + l_1 l_{c2} m_2 u)\cos(\varphi_2) + g \sin(\varphi_2) l_{c2} m_2 + f_{\mathrm{ext}} l_2 - d_2 \omega_2)}{l_{c2}^2 m_2 + I_{2,zz}} \end{bmatrix} . \tag{2.21}$$

The mathematical model with the system dynamics, as expressed in (2.21), is derived using MAPLE [18]. Further, the equation $\mathbf{f}(\mathbf{x}_\mathrm{e}, u_\mathrm{e}, f_\mathrm{ext,e}) = \mathbf{0}$ for $u = u_\mathrm{e} = 0$ and $f_\mathrm{ext} = f_\mathrm{ext,e} = 0$ is solved to derive the equilibrium points

$$\left\{ \mathbf{x}_\mathrm{e} \,|\, \mathbf{x}_\mathrm{e} = \begin{bmatrix} x_{1,\mathrm{e}} & 0 & k180^\circ & 0 \end{bmatrix}^\mathrm{T} : x_{1,\mathrm{e}} \in \mathbb{R} \wedge k \in \mathbb{Z} \right\} , \tag{2.22}$$

whereas even numbers of k denote the equilibrium points, where the pendulum is upright. Uneven numbers of k are the points of equilibrium where the pendulum is hanging downwards.

2.3 Non-linear Least Squares Parameter Estimation

This chapter describes the estimation of the parameters of the mathematical model from (2.21). The pendulum's mass m_2, the length l_2, and the arm's length l_1 are directly measured. The remaining parameters are estimated using a non-linear least-squares method based on measurements from the physical *Furuta pendulum*.

Considering the geometrical properties of the pendulum and the dynamics given in (2.21), the moment of inertia $I_{2,xx}$ has a minor influence on the model dynamics and is therefore calculated. The pendulum is modeled as a hollow cylinder with the mass m_2, the outer radius r_o and the inner radius r_i, which leads to the following expression for calculating the moment of inertia:

$$I_{2,xx} = \frac{m_2}{2}(r_\mathrm{o}^2 + r_\mathrm{i}^2) \ . \tag{2.23}$$

The remaining model parameters of the *Furuta pendulum* from (2.21), namely the moment of inertia $I_{2,zz}$, the friction coefficient d_2, and the distance to the pendulum's center of mass l_{c2}, are obtained using a non-linear least squares parameter estimation. To do so, the model input u^* of a swing-up trajectory with guessed parameters close to the actual parameters is calculated by solving the optimization problem for the swing-up, as described in Chapter 4. The input u^* is used as the input on the physical *Furuta pendulum* with the initial state $\mathbf{x}_0^\mathrm{T} = \begin{bmatrix} 0 & 0 & 180^\circ & 0 \end{bmatrix}$ at the lower equilibrium point. The measured angle $\varphi_{2,\mathrm{m}}$ in the time interval $t \in [t_0, t_\mathrm{m}]$, taken from the encoder, is used to formulate the non-linear least squares optimization problem

$$\min_{\mathbf{p}} \quad J(\mathbf{p}) = \int_{t_0 = 0\,\mathrm{s}}^{t_\mathrm{m} = 2.5\,\mathrm{s}} (\varphi_{2,\mathrm{m}}(t) - \varphi_2(t))^2 \ \mathrm{d}t \tag{2.24a}$$

$$\text{subject to} \quad \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u^*, 0, \mathbf{p}), \quad \mathbf{x}(t_0) = \mathbf{x}_0 \ , \tag{2.24b}$$

$$\mathbf{p}_\mathrm{min} \le \mathbf{p} \le \mathbf{p}_\mathrm{max} \ , \tag{2.24c}$$

similarly to [19] with the parameter vector

$$\mathbf{p}^\mathrm{T} = \begin{bmatrix} d_{2,\mathrm{est}} & I_{2,zz,\mathrm{est}} & l_{c2,\mathrm{est}} \end{bmatrix} , \tag{2.25}$$

the lower bound $\mathbf{p}_\mathrm{min}$, the upper bound $\mathbf{p}_\mathrm{max}$ with

$$\mathbf{p}_\mathrm{min}^\mathrm{T} = \begin{bmatrix} 10^{-4}\,\mathrm{N\,m\,s} & 0\,\mathrm{kg\,m^2} & 0.1\,\mathrm{m} \end{bmatrix} , \tag{2.26a}$$

$$\mathbf{p}_\mathrm{max}^\mathrm{T} = \begin{bmatrix} 10^{-3}\,\mathrm{N\,m\,s} & \infty\ \mathrm{kg\,m^2} & 0.3\,\mathrm{m} \end{bmatrix} , \tag{2.26b}$$

and the initial guess of the parameter vector

$$\hat{\mathbf{p}}_0^{\mathrm{T}} = \begin{bmatrix} 5.5 \cdot 10^{-4}\,\mathrm{N\,m\,s} & 4 \cdot 10^{-4}\,\mathrm{kg\,m^2} & 0.15\,\mathrm{m} \end{bmatrix} . \tag{2.27}$$

The constrained optimization problem (2.24) was solved using MATLAB's `fmincon` function [20] in combination with the interior-point method, additionally requiring an ordinary differential equation solver for (2.24b), which was solved using MATLAB's `ode45` [20]. The optimization algorithm converged to the minimum

$$\hat{\mathbf{p}}^{\mathrm{T}} = \begin{bmatrix} 5.5429 \cdot 10^{-4}\,\mathrm{N\,m\,s} & 3.5052 \cdot 10^{-4}\,\mathrm{kg\,m^2} & 0.159\,29\,\mathrm{m} \end{bmatrix} . \tag{2.28}$$

Due to the pendulum's symmetry, the moment of inertia about the y_2-axis $I_{2,yy}$ can be assumed as $I_{2,yy} = I_{2,zz}$.

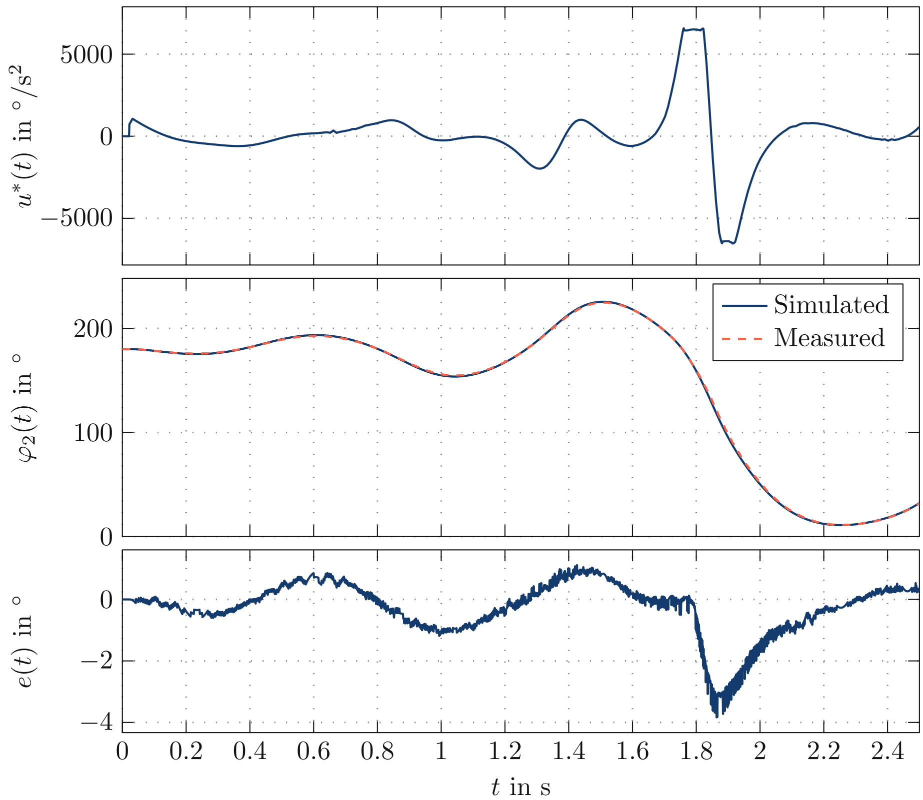

Figure 2.3: Comparison of the measured and the simulated open loop swing-up of the *Furuta pendulum* with the final set of parameters. The plot shows the error $e(t) = \varphi_2(t) - \varphi_{2,\mathrm{m}}(t)$, where $\varphi_2(t)$ denotes the simulated pendulum angle, and $\varphi_{2,\mathrm{m}}(t)$ is the encoder measurement from the physical *Furuta pendulum*.

Figure 2.3 shows a comparative plot of the measured angle $\varphi_{2,\mathrm{m}}(t)$ and the simulated angle $\varphi_2(t)$ of the *Furuta pendulum* model from (2.21) with the final set of parameters as summarized in Table 2.1. The simulated pendulum angle $\varphi_2(t)$ fits the measured pendulum angle $\varphi_{2,\mathrm{m}}(t)$ well over a wide time range, as shown in the error plot with

$e(t) = \varphi_2(t) - \varphi_{2,\mathrm{m}}(t)$. However, during high angular acceleration of the pendulum's arm at $t \in [1.8\,\mathrm{s}, 2.0\,\mathrm{s}]$, the magnitude of the error increases. This behavior could indicate that unmodeled effects, such as the finite stiffness of the 3D-printed mounting bracket, contribute to the error. Furthermore, noise in the error plot $e(t)$ is prominent due to the quantization from the encoder in the measured angle $\varphi_{2,\mathrm{m}}(t)$.

Parameter	**Method**	**Value**
$I_{2,xx}$	calculated	$2.83 \cdot 10^{-6}\,\mathrm{kg\,m^2}$
$I_{2,yy}$	estimated	$3.51 \cdot 10^{-4}\,\mathrm{kg\,m^2}$
$I_{2,zz}$	estimated	$3.51 \cdot 10^{-4}\,\mathrm{kg\,m^2}$
l_2	measured	$0.3\,\mathrm{m}$
$l_{\mathrm{c}2}$	estimated	$0.159\,\mathrm{m}$
m_2	measured	$0.05\,\mathrm{kg}$
d_2	estimated	$5.54 \cdot 10^{-4}\,\mathrm{N\,m\,s}$
g	constant	$9.81\,\mathrm{m/s^2}$
r_{o}	measured	$0.008\,\mathrm{m}$
r_{i}	measured	$0.007\,\mathrm{m}$
l_1	measured	$0.131\,\mathrm{m}$

Table 2.1: The parameters used for the mathematical model of the *Furuta pendulum* and the respective identification method.

3 Vision-Based Pendulum Angle Detection

Event cameras generate asynchronous events based on brightness changes, specifically the photocurrent changes of each individual pixel [1]. In order to obtain the pendulum's angle from these asynchronous events, an event processing algorithm is required. The events generated by the movement of the pendulum in a sufficiently small time window can be approximated to be collinear and, therefore, lie in a straight line. A well-known algorithm for detecting straight lines and their respective angle is the *Hough transform* [21, 22]. However, this algorithm is typically used for detecting lines in full-frame images. The following section describes a method for applying the *Hough transform* on sparse event streams.

Since the aim is to obtain the pendulum's angle from events generated by the stationary camera, the perspective projection must be considered. This chapter derives a mathematical representation of this projection using the pinhole camera model [23], which allows for compensating the non-linear relation between the pendulum's actual angle and the projected angle obtained from the *Hough transform*.

3.1 Event-based Hough Transform with Exponential Decay

This section introduces the method used in this work to obtain the angle of collinear events from the set of events generated by the event camera. Each event

$$e_k = (t_k, p_k, x_k, y_k) \tag{3.1}$$

from an event camera consists of the timestamp t_k, the polarity of the brightness change $p_k \in \{-1, 1\}$, and the coordinates on the camera sensor $P_k = (x_k, y_k)$ [1]. However, the event camera used in this work transmits event packets as a set of events ε_i at the time t_i rather than individual events e_k [24, 25].

The coordinates (x_k, y_k) of one isolated event e_k on the sensor plane can lie on an infinite number of straight candidate lines

$$d_k(\varphi_{2,\mathrm{h}}) = x_k \sin(\varphi_{2,\mathrm{h}}) + y_k \cos(\varphi_{2,\mathrm{h}}) \quad , \quad \varphi_{2,\mathrm{h}} \in [0, \pi[\ , \tag{3.2}$$

parametrized by the candidate line's signed normal distance to the origin d_k and its angle $\varphi_{2,\mathrm{h}}$ with respect to the sensor's y-axis [22].

Figure 3.1 illustrates the set of event coordinates $\mathcal{P}_i$ and $\mathcal{P}_{i-1}$ on the sensor plane from two subsequent event packets, which are transformed into the *Hough space* using (3.2), forming a set of sinusoidal-shaped functions, where each event e_k on the sensor plane refers to one sinusoidal-shaped function $d_k(\varphi_{2,\mathrm{h}})$ in the *Hough space*. Since the events in Figure 3.1 are collinear, the sinusoidal-shaped functions have a unique intersection point $(d_{\mathrm{int}}, \varphi_{2,\mathrm{int}})$ [22], as shown in Figure 3.1. Thus, the problem of detecting collinear events

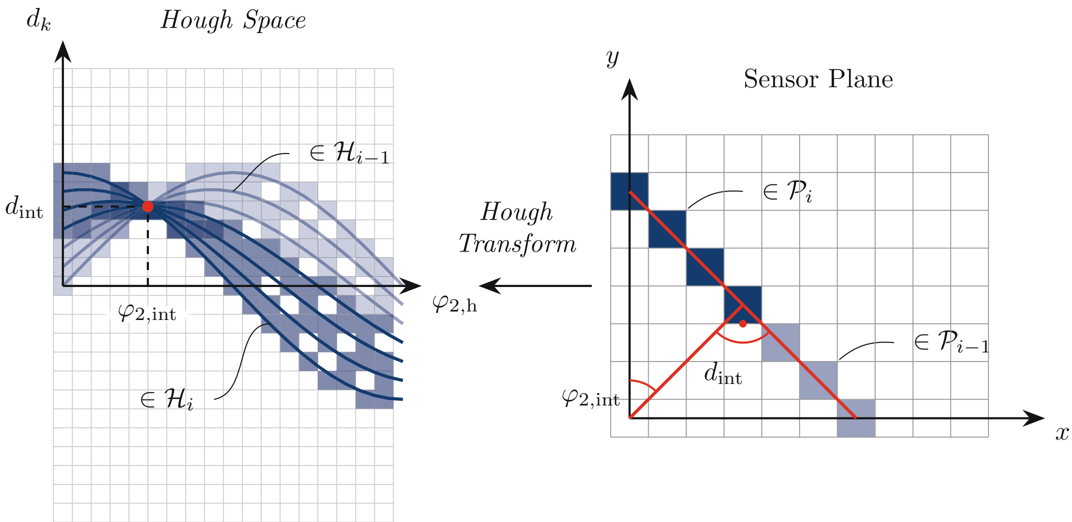

Figure 3.1: Graphical representation of the *Hough transform* with two subsequent event packets of collinear events.

on the sensor plane is transformed into a problem of finding the intersection point in the *Hough space* [22]. In order to find the intersection point, the angle

$$\varphi_{2,\text{h}} = a\Delta\varphi_2 \ , \quad a = 0, 1, \dots, N_\varphi - 1 \ , \tag{3.3a}$$

$$\Delta\varphi_2 = \frac{180°}{N_\varphi} \ , \tag{3.3b}$$

is quantized to $N_\varphi \in \mathbb{N}$ angles. Further, the distance d_k is quantized to

$$\bar{d}_k = \frac{N_d}{2} + \text{round}(x_k \sin(a\Delta\varphi_2) + y_k \cos(a\Delta\varphi_2)) \quad , \quad a = 0, 1, \dots, N_\varphi - 1 \ , \tag{3.4}$$

with the function for rounding to the closest integer value

$$\text{round}(r) = \lfloor r + 0.5 \rfloor \ , \tag{3.5}$$

where the operator $\lfloor \cdot \rfloor$ denotes the floor function. The even parameter N_d must be chosen such that the offset $N_d/2$ is at least the size of the camera's sensor diagonal in pixels. For every tuple $(\bar{d}_k, a)$, the respective entry $h_{\bar{d}_k,a}$ in the $N_d \times N_\varphi$ matrix

$$\mathbf{H} = \begin{bmatrix} h_{0,0} & h_{0,1} & \cdots & h_{0,N_\varphi-1} \\ h_{1,0} & h_{1,1} & \cdots & h_{1,N_\varphi-1} \\ \vdots & \vdots & \cdots & \vdots \\ h_{N_d-1,0} & h_{N_d-1,1} & \cdots & h_{N_d-1,N_\varphi-1} \end{bmatrix} \tag{3.6}$$

is incremented. This allows for the detection of collinear events by finding the maximum in the matrix $\mathbf{H}$. More specifically, for every new set of received event coordinates $\mathcal{P}_i$, the set of hough-transformed events

$$\mathcal{H}_i = \{(\bar{d}_k(x_k, y_k, a), a) \mid a = 0, ..., N_\varphi - 1, \forall (x_k, y_k) \in \mathcal{P}_i\} \tag{3.7}$$

is calculated, and each element $h_{\bar{d}_k,a}$ of the matrix $\mathbf{H}$ is incremented by one if the tuple $(\bar{d}_k, a)$ is in the set $\mathcal{H}_i$. The intersection point in the quantized *Hough space* is the entry in the matrix $\mathbf{H}$ with the highest number of increments.

An example of a *Hough space*, represented by the matrix $\mathbf{H}$, generated from event packets with collinear events is shown in Figure 3.1, where the entries of the matrix $\mathbf{H}$ are illustrated as rectangular cells. Darker-colored cells refer to larger matrix entries. However, to consider older events while also taking the latest events with a higher weighting factor into account, a discrete-time exponential decay

$$\mathbf{H}(t)|_{t=j\Delta t^+} = \tau \mathbf{H}(t)|_{t=j\Delta t^-} \quad , \quad \mathbf{H}(0) = \mathbf{0} \; , \tag{3.8}$$

with the design parameter $\tau \in [0, 1[$ and a cycle time of Δt, is introduced[1]. The index $j \in \mathbb{N}_0$ denotes the j-th computation time step. Figure 3.1 shows that the events from $\mathcal{P}_{i-1}$ are considered to have already decayed and are thus shown with a lighter color tone. Figure 3.2 visualizes the state of the matrix $\mathbf{H}$ from the example in Figure 3.1 as a 3D bar plot.

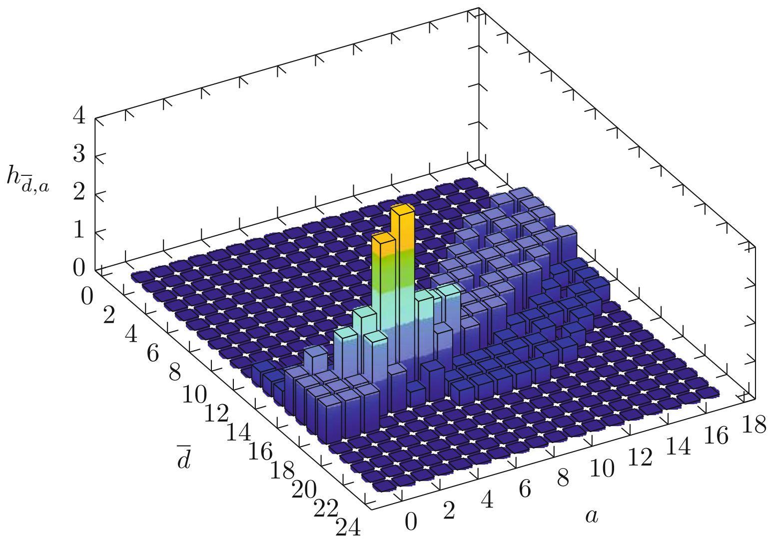

Figure 3.2: A 3D bar plot of the elements $h_{\bar{d},a}$ of the matrix $\mathbf{H}$ with events from two subsequent event coordinate packets containing collinear events.

In order to find the indices $(\bar{d}_{i,\mathrm{int}}, a_{i,\mathrm{int}})$ of the matrix $\mathbf{H}$ with the largest entry at the computation step $i \in \mathbb{N}_0$, a maximum search in the region $\mathcal{S}_i$ around the previous maximum $(\bar{d}_{i-1,\mathrm{int}}, a_{i-1,\mathrm{int}})$ is used, as shown in Figure 3.3. The maximum search starts in the region $\mathcal{S}_0$, which covers the pendulum's known initial position with $\varphi_1 = 0°$ and $\varphi_2 = 180°$. The search region $\widetilde{\mathcal{S}}_i$ that does not extend beyond the boundaries of the *Hough*

[1]The time $t = j\Delta t^-$ denotes the moment of the function call in the program that computes the exponential decay. Similarly, $t = j\Delta t^+$ refers to the completion time of the respective function call.

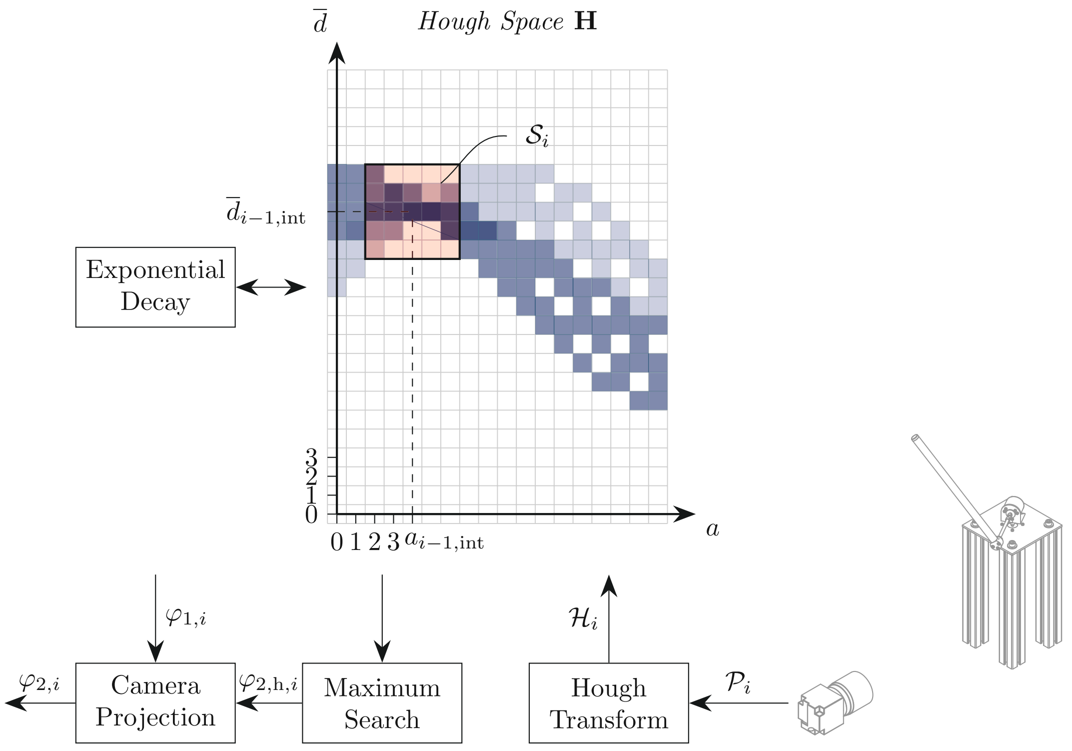

Figure 3.3: An overview of the processing pipeline for the event packets generated by the event camera.

space can be expressed as

$$\mathcal{D}_i = \left\{\bar{d} \in \mathbb{Z} \mid \bar{d} \in \left[\bar{d}_{i-1,\text{int}} - \Delta\bar{d}, \bar{d}_{i-1,\text{int}} - \Delta\bar{d} + 1, \dots, \bar{d}_{i-1,\text{int}} + \Delta\bar{d}\right]\right\}, \tag{3.9a}$$

$$\mathcal{A}_i = \{a \in \mathbb{Z} \mid a \in [a_{i-1,\text{int}} - \Delta a, a_{i-1,\text{int}} - \Delta a + 1, \dots, a_{i-1,\text{int}} + \Delta a]\}, \tag{3.9b}$$

$$\widetilde{\mathcal{S}}_i = \left\{(\bar{d}, a) \mid (\bar{d}, a) \in \mathcal{D}_i \times \mathcal{A}_i : a \geq 0 \wedge a < N_\varphi\right\}, \tag{3.9c}$$

with the design parameters $\Delta\bar{d}$ and Δa that define the search region's size. If the maximum is located at the border of the *Hough space*, $\mathcal{A}_i$ may contain negative values or values larger than $N_\varphi - 1$. If this occurs, the search region must be wrapped accordingly. Imagine a line with an event at (x_0, y_0) in a continuous *Hough space* with an angle of $\varphi_{2,\text{h}} = 0°$ at the distance d_0. Equation (3.2) must be satisfied for the event (x_0, y_0). However, the same line can be represented in the *Hough space* with an angle of $\widetilde{\varphi}_{2,\text{h}} = 180°$ at a signed distance $\widetilde{d}_0 = -d_0$. This fact is taken into account with the term $N_d - \bar{d}$ in the extended search region

$$\mathcal{S}_i = \left\{(N_d - \bar{d}, a \bmod N_\varphi) \mid (\bar{d}, a) \in \mathcal{D}_i \times \mathcal{A}_i : a \geq N_\varphi \vee a < 0\right\} \cup \widetilde{\mathcal{S}}_i, \tag{3.10}$$

as it extends over the boundaries of the *Hough space*. The expression $(a \bmod N_\varphi)$ is the modulo operation used to wrap the indices of a to its respective range from $0, \dots, N_\varphi - 1$.

Considering the *Furuta pendulum's* system dynamics, it is important to design the search region sufficiently large to ensure that the maximum in the matrix $\mathbf{H}$ is always in the search region $\mathcal{S}_i$. However, a larger search region leads to a higher computation time.

The first element $(\bar{d}_{i,\mathrm{int}}, a_{i,\mathrm{int}})$ found in the set

$$\left\{(\bar{d}, a) \mid h_{\bar{d},a} \geq h_{n,m} \forall (n,m) \in \mathcal{S}_i \wedge h_{\bar{d},a} > h_\mathrm{T}\right\} \tag{3.11}$$

of indices, with the threshold $h_\mathrm{T} > 0$, is taken to obtain the angle of collinear events

$$\varphi_{2,\mathrm{h},i} = a_{i,\mathrm{int}} \Delta\varphi_2 - 90^\circ + n_i 180^\circ \ , \tag{3.12}$$

where $n_i \in \mathbb{Z}$ shifts the angle obtained from the *Hough transform* such that it is not restricted to the quantization of the *Hough space*, as expressed in (3.3a).

3.2 Pinhole Camera Projection of the Detected Line

The previously described *Hough transform* algorithm for event data outputs the angle of collinear events from the pendulum's movement. However, the projective geometry must be considered in general to calculate the pendulum's angle from the angle of the collinear events. The projective geometry is derived from a simplified model of the *Furuta pendulum*, where the diameter of the pendulum is neglected, which can be done for sufficiently small pendulum diameters. The projected angle on the sensor plane $\varphi_{2,\mathrm{h}}$ coincides with the pendulum's angle φ_2 if the arm is aligned with the camera's optical axis. In the general case, however, the perspective projection must be taken into account. This is done by deriving an analytical function $\phi_{2,\mathrm{p}}(\varphi_1, \varphi_{2,\mathrm{h}})$, which expresses the pendulum angle φ_2 from the projected angle obtained by the *Hough transform* $\varphi_{2,\mathrm{h}}$ and the arm's angle φ_1 using the pinhole camera model [23].

Figure 3.4 depicts the projective geometry of the *Furuta pendulum* using the pinhole camera model. The camera's optical center is at the origin of the coordinate system $(x_\mathrm{c}, y_\mathrm{c}, z_\mathrm{c})$. The principal axis originates from the optical center and passes through the arm's center of rotation. The principal axis crosses the sensor plane at the point O_i, the origin of the sensor coordinate system $(x_\mathrm{i}, y_\mathrm{i})$. The position of the sensor coordinate system O_i along the principal axis is irrelevant for obtaining the pendulum's angle because the projected angle $\varphi_{2,\mathrm{h}}$ is translational invariant with respect to translations along the camera's principal axis in the following considerations. The sensor plane is defined to be orthogonal to the principal axis. In order to get the pendulum's representation on the sensor plane, the intersection points $\mathbf{I}_1$ and $\mathbf{I}_2$ of the respective projection lines are marked. The vector $\mathbf{d}_\mathrm{p}$ between these two intersection points represents the projection of the pendulum. The angle between the pendulum's projection vector $\mathbf{d}_\mathrm{p}$ and the vertical unit vector $-\mathbf{e}_y$ is denoted as the projected angle $\varphi_{2,\mathrm{h}}$.

In order to obtain the projected angle $\varphi_{2,\mathrm{h}}$, the pendulum's position is expressed with respect to the camera's frame $(x_\mathrm{c}, y_\mathrm{c}, z_\mathrm{c})$, as described by the matrix [23]

$$\mathbf{C}_\mathrm{E} = \begin{bmatrix} \mathbf{R}_\mathrm{c}^0 & \mathbf{d}_\mathrm{c}^0 \\ \mathbf{0}^\mathrm{T} & 1 \end{bmatrix} , \tag{3.13}$$

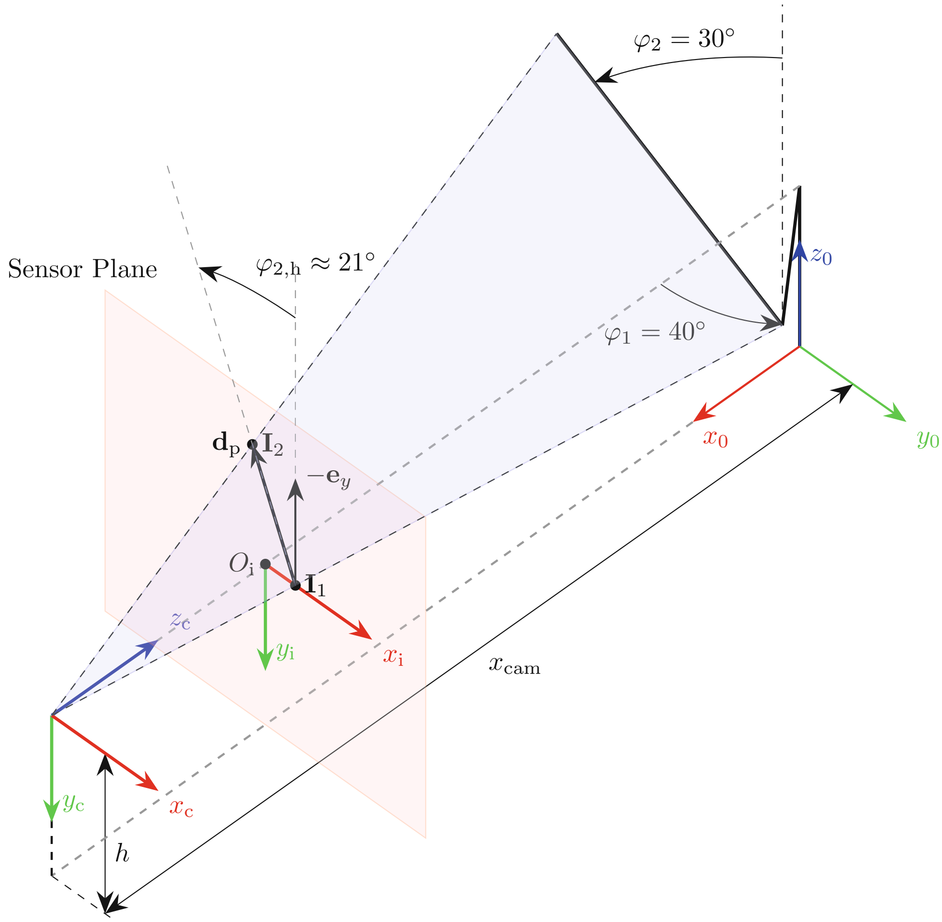

Figure 3.4: A schematic overview of the simplified *Furuta pendulum*, the event camera's sensor plane, and the respective coordinate systems.

with the rotation matrix

$$\mathbf{R}_c^0 = \mathbf{R}_x(90°)\mathbf{R}_z(-90°) \tag{3.14}$$

and the vector

$$\mathbf{d}_c^0 = \begin{bmatrix} 0 \\ h \\ x_{cam} \end{bmatrix} . \tag{3.15}$$

Considering square pixels on the sensor plane and a coinciding origin of the sensor's coordinate system (x_i, y_i) with the principal point $\mathcal{O}_i$, the internal camera matrix with the focal length from the event camera's datasheet f_c [25] is written as [23]

$$\mathbf{C}_I = \begin{bmatrix} f_c & 0 & 0 & 0 \\ 0 & f_c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} . \tag{3.16}$$

Further, the lens distortion is neglected. Next, the points in the two-dimensional sensor coordinate system are represented by the homogeneous coordinates $\widetilde{\mathbf{M}}_{c,i} \in \mathbb{R}^3$. The projection, described by the pinhole camera model, is written using homogeneous coordinates as

$$\widetilde{\mathbf{M}}_{\mathrm{c},i} = \mathbf{C}_{\mathrm{I}}\mathbf{C}_{\mathrm{E}}\mathbf{P}_{0,i} \ , \quad i = 1, 2 \ , \tag{3.17}$$

with $\mathbf{P}_{0,i} \in \mathbb{R}^4$. The homogeneous coordinates of the pendulum's tip $\mathbf{P}_{0,2}$ and the arm's tip $\mathbf{P}_{0,1}$ are expressed as

$$\mathbf{P}_{0,i} = \begin{bmatrix} \mathbf{p}_{0,i} \\ 1 \end{bmatrix}, \quad i = 1, 2 \ . \tag{3.18}$$

Transforming the homogeneous coordinates $\widetilde{\mathbf{M}}_{\mathrm{c},i}$ back to *Euclidean* coordinates $\widetilde{\mathbf{m}}_{\mathrm{c},i} \in \mathbb{R}^2$ with $i = 0, 1$, the vector of the projected pendulum in the sensor frame can be written as

$$\mathbf{d}_{\mathrm{p}}(\varphi_1, \varphi_2) = \widetilde{\mathbf{m}}_{\mathrm{c},2}(\varphi_1, \varphi_2) - \widetilde{\mathbf{m}}_{\mathrm{c},1}(\varphi_1) \ , \tag{3.19}$$

with the position vector $\widetilde{\mathbf{m}}_{\mathrm{c},2}(\varphi_1, \varphi_2)$ of point $\mathbf{I}_2$ and the position vector $\widetilde{\mathbf{m}}_{\mathrm{c},1}(\varphi_1)$ of point $\mathbf{I}_1$, as shown in Figure 3.4. The relation to obtain the angle between the unit vector $-\mathbf{e}_y$ and $\mathbf{d}_{\mathrm{p}}$ is given by

$$-\mathbf{d}_{\mathrm{p}}^{\mathrm{T}}\mathbf{e}_y = \cos(\varphi_{2,\mathrm{h}})\|\mathbf{d}_{\mathrm{p}}\|_2 \ . \tag{3.20}$$

Solving the equation for φ_2 using MAPLE [18] gives the non-linear function $\phi_{2,\mathrm{p}}(\varphi_1, \varphi_{2,\mathrm{h}})$, as shown in Figure 3.5. It is evident that the pendulum's angle coincides with the angle $\varphi_{2,\mathrm{h}}$ for arm angles of $\varphi_1 = l180°, l \in \mathbb{Z}$. However, as the arm angle φ_1 increases, the function $\phi_{2,\mathrm{p}}(\varphi_1, \varphi_{2,\mathrm{h}})$ becomes increasingly non-linear.

Finally, the pendulum's angle at the computation time step i

$$\varphi_{2,i} = \phi_{2,\mathrm{p}}(\varphi_{1,i}, \varphi_{2,\mathrm{h},i}) \tag{3.21}$$

is calculated from the arm's angle $\varphi_{1,i}$ and the projected angle obtained by the *Hough transform* $\varphi_{2,\mathrm{h},i}$.

At a certain critical angle $\varphi_{1,\mathrm{crit}}$, when the angle between the arm and its projection line is 90°, any rotation of the pendulum would not lead to a change in the angle of the vector $\mathbf{d}_{\mathrm{p}}$. Deriving the critical angle $\varphi_{1,\mathrm{crit}}$ and evaluating it for $x_{\mathrm{cam}} = 0.52\,\mathrm{m}$ gives

$$\varphi_{1,\mathrm{crit}} = \arccos\left(\frac{l_1}{x_{\mathrm{cam}}}\right) \approx 75.4° \ . \tag{3.22}$$

Therefore, the absolute arm angle should remain smaller than the critical angle $\varphi_{1,\mathrm{crit}}$. It is also important to be aware that the angular resolution of the pendulum angle measurement $\varphi_{2,i}$ can significantly decrease towards larger arm angles due to the quantization in the *Hough transform* as expressed in (3.3a) and the non-linearity of the function $\phi_{2,\mathrm{p}}(\varphi_{1,i}, \varphi_{2,\mathrm{h},i})$.

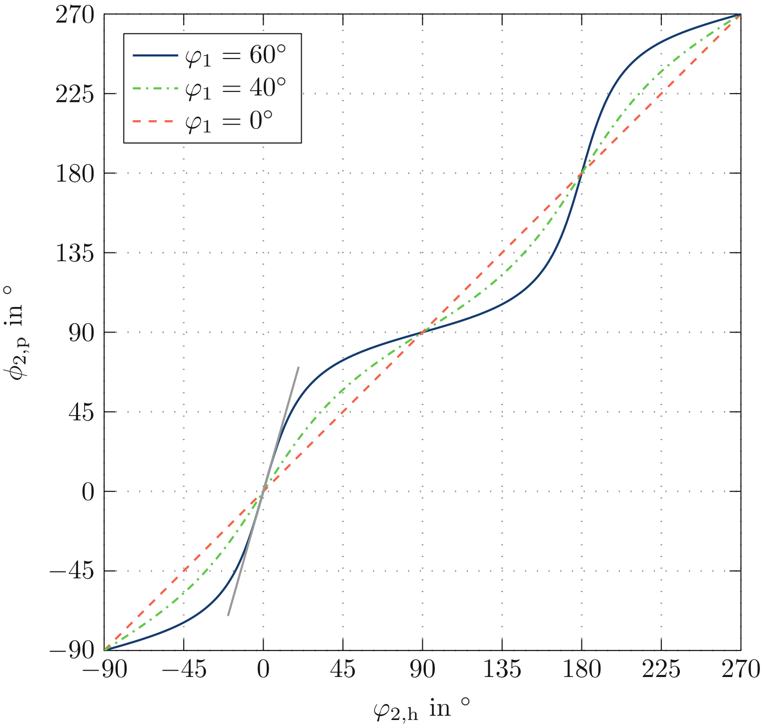

Figure 3.5: The function for obtaining the pendulum angle $\phi_{2,\mathrm{p}}(\varphi_1, \varphi_{2,\mathrm{h}})$ from the projected angle $\varphi_{2,\mathrm{h}}$ for a variation of arm angles φ_1.

4 Trajectory Generation, State Estimation, and Control of the Furuta Pendulum

This chapter describes the reference trajectory generation for the *Furuta pendulum's* swing-up and swing-down trajectories. Further, the generation of a trajectory for balancing the pendulum in the upright position from an arm angle of $\varphi_1 = 0°$ to $\varphi_1 = -30°$, denoted as transfer trajectory, is described. An optimal control problem is formulated and solved using an interior-point method to obtain the reference trajectories. In order to stabilize the *Furuta Pendulum* about the reference trajectories and the equilibrium points, a discrete, time-varying *Linear Quadratic Regulator* (*LQR*) is designed. This, however, requires the knowledge of the entire system state. Therefore, a discrete-time *Extended Kalman Filter* is designed to estimate the *Furuta pendulum's* system state.

4.1 Optimization Problem for the Trajectory Generation

This work aims to stabilize the *Furuta pendulum* about reference trajectories using vision feedback. The optimization problem to obtain these reference trajectories is described in this section. First, starting from the *Furuta pendulum's* initial position, hanging downwards with $\varphi_1 = 0°$ and $\varphi_2 = 180°$, the reference *swing-up* trajectory aims to guide the *Furuta pendulum* to the upright position with $\varphi_1 = \varphi_2 = 0°$. Starting from the upright position, the *Furuta pendulum* follows the *transfer* trajectory to the terminal arm angle at $\varphi_1 = -30°$ while balancing the pendulum. After reaching the arm's terminal angle, the *Furuta pendulum* follows a swing-down trajectory that ends at the lower equilibrium point with $\varphi_1 = 0°$ and $\varphi_2 = 180°$.

The three reference trajectories are obtained by solving an optimal control problem with the angular acceleration u as input to get a smooth trajectory for the velocity input of the drive motor. In order to formulate the discrete-time optimal control problem, an equidistant time grid

$$t_i = t_0 + i\Delta t, \quad i = 0, ..., N-1 \ , \tag{4.1a}$$

$$\Delta t = \frac{t_N - t_0}{N-1} \ , \tag{4.1b}$$

with the start time t_0 of the respective trajectory, the end time t_N, the step size Δt, and the total number of grid points N. Without limiting the generality, the start time is defined to be $t_0 = 0\,\mathrm{s}$, $t_N = 2.5\,\mathrm{s}$, and $N = 2501$. Both the system state $\mathbf{x}(t_i) = \mathbf{x}_i \in \mathbb{R}^4$ and the system input $u(t_i) = u_i \in \mathbb{R}$ for $i = 1, 2, \ldots, N$ are discretized to write the vector of system states and inputs

$$\mathbf{y} = \begin{bmatrix} \mathbf{x}_1^{\mathrm{T}} & u_1 & \mathbf{x}_2^{\mathrm{T}} & u_2 & \ldots & \mathbf{x}_N^{\mathrm{T}} & u_N \end{bmatrix}^{\mathrm{T}} . \tag{4.2}$$

The optimal control problem is formulated as

$$\begin{aligned}
\min_{\mathbf{y}} \quad & J(\mathbf{y}) = \sum_{i=1}^{N} l(\mathbf{x}_i, u_i) \ , && (4.3a)\\
\text{subject to} \quad & \mathbf{g}(\mathbf{y}) = \mathbf{0} \ , && (4.3b)\\
& -\mathbf{b}_x \leq \mathbf{x}_i \leq \mathbf{b}_x \ , && (4.3c)\\
& -b_u \leq u_i \leq b_u, \quad i = 1, 2, \ldots, N \ , && (4.3d)
\end{aligned}$$

with the vector of equality constraints $\mathbf{g}(\mathbf{y}) \in \mathbb{R}^{4(N+1)}$, the bound for the system state $\mathbf{b}_x^{\mathrm{T}} = \begin{bmatrix} 45° & \infty & \infty & \infty \end{bmatrix}$, the bound for the system input $b_u = 120\,\mathrm{rad/s^2} \approx 6875.5\,°/\mathrm{s}^2$, the terminal state $\mathbf{x}_{\mathrm{T}}$, and the cost function

$$l(\mathbf{x}_i, u_i) = 1 + \frac{1}{2}\Big((\mathbf{x}_i - \mathbf{x}_{\mathrm{T}})^{\mathrm{T}}\mathbf{Q}(\mathbf{x}_i - \mathbf{x}_{\mathrm{T}}) + Ru_i^2\Big) \ , \tag{4.4}$$

with $\mathbf{Q} > \mathbf{0} \in \mathbb{R}^{4\times4}$ and $R > 0 \in \mathbb{R}$. Further, the *Furuta pendulum's* system dynamic is discretized using a *Runge-Kutta* 4th-order method [26], formulated as equality constraints

$$\mathbf{g}_i = \mathbf{x}_{i+1} - \mathbf{x}_i - \frac{1}{6}(\boldsymbol{\xi}_{1,i} + 2\boldsymbol{\xi}_{2,i} + 2\boldsymbol{\xi}_{3,i} + \boldsymbol{\xi}_{4,i}), \quad i = 1, 2, \ldots, N-1 \ , \tag{4.5}$$

with

$$\begin{aligned}
\boldsymbol{\xi}_{1,i} &= \Delta t \mathbf{f}(\mathbf{x}_i, u_i, 0) \ , && (4.6a)\\
\boldsymbol{\xi}_{2,i} &= \Delta t \mathbf{f}(\mathbf{x}_i + \frac{1}{2}\boldsymbol{\xi}_{1,i}, u_i, 0) \ , && (4.6b)\\
\boldsymbol{\xi}_{3,i} &= \Delta t \mathbf{f}(\mathbf{x}_i + \frac{1}{2}\boldsymbol{\xi}_{2,i}, u_i, 0) \ , && (4.6c)\\
\boldsymbol{\xi}_{4,i} &= \Delta t \mathbf{f}(\mathbf{x}_i + \boldsymbol{\xi}_{3,i}, u_{i+1}, 0) \ . && (4.6d)
\end{aligned}$$

Together with the equality constraints for $\mathbf{x}_1$ and $\mathbf{x}_N$,

$$\begin{aligned}
\mathbf{g}_0 &= \mathbf{x}_1 - \mathbf{x}_{\mathrm{I}} \ , && (4.7a)\\
\mathbf{g}_N &= \mathbf{x}_N - \mathbf{x}_{\mathrm{T}} \ , && (4.7b)
\end{aligned}$$

the vector of equality constraints is written as

$$\mathbf{g}(\mathbf{y}) = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \vdots \\ \mathbf{g}_N \end{bmatrix} . \tag{4.8}$$

The optimization problem was solved using MATLAB's non-linear programming solver `fmincon` [27] for constrained optimization problems. It is configured to use the interior-point algorithm because it is the recommendation for large-scale optimization problems [28].

Figure 4.1 shows the calculated reference swing-up trajectory, which starts at $\mathbf{x}_{\mathrm{I,sup}}^{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 180^\circ & 0 \end{bmatrix}$ and ends at $t_N = 2.5\,\mathrm{s}$ with $\mathbf{x}_{T,\mathrm{sup}}^{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$. The deflection of the arm's angle φ_1 stays well within the box constraints. Low deflections of the arm's angle φ_1 are important for detecting the pendulum's angle with the introduced *Hough transform* algorithm because the angular resolution of the detected angle $\varphi_{2,i}$ can decrease with larger arm deflections due to the projection, as shown in Figure 3.5. Therefore, the arm's angle φ_1 is box-constrained to $\pm 45^\circ$ as expressed in (4.3c). Additionally, the respective element in the matrix $\mathbf{Q}$ is chosen to be considerably higher than all other matrix elements in $\mathbf{Q}$. The box constraints for the input u limit the angular acceleration of the stepper motor, as shown in Figure 4.1. This is important to avoid losing steps when performing experiments on the physical *Furuta pendulum*. Therefore, the stepper motor's holding torque $\tau_{\mathrm{hold}} = 0.45\,\mathrm{N\,m}$ [15] is compared to the nominal motor torque, which is derived from the *Euler-Lagrange* equations (2.19). Neglecting viscous damping for the arm by setting the damping coefficient $d_1 = 0\,\mathrm{N\,m\,s}$, measuring the arm's mass m_1, calculating the arm's moment of inertia $\mathbf{I}_1$, and assuming that the arm's center of gravity is at $l_{\mathrm{c}1} = l_1/2$, the motor torque τ_{m} for each trajectory is calculated and shown in Figures 4.1, 4.2, and 4.3. The plots show that the nominal motor torque has a margin with respect to the holding torque τ_{hold}.

The obtained transfer trajectory starts at $\mathbf{x}_{\mathrm{I,tf}} = \mathbf{x}_{T,\mathrm{sup}}$ and moves the arm's angle φ_1 to the terminal angle -30° while balancing the pendulum, as shown in Figure 4.2. The transfer trajectory ends at $t_N = 2.5\,\mathrm{s}$ at the terminal state $\mathbf{x}_{T,\mathrm{tf}}^{\mathrm{T}} = \begin{bmatrix} -30^\circ & 0 & 0 & 0 \end{bmatrix}$.

The initial state for the swing-down at $t_0 = 0\,\mathrm{s}$ is $\mathbf{x}_{\mathrm{I,sdo}} = \mathbf{x}_{T,\mathrm{tf}}$, as shown in Figure 4.3. The swing-down trajectory ends at $t_N = 2.5\,\mathrm{s}$ at the terminal state $\mathbf{x}_{T,\mathrm{sdo}}^{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 180^\circ & 0 \end{bmatrix}$. Unlike for the swing-up, the input u stays well within the box constraints.

The chosen parameters for obtaining the reference trajectories by solving the optimization problem (4.3) are summarized in Table 4.1.

Trajectory	$\mathbf{x}_{\mathrm{I}}$	$\mathbf{x}_{\mathrm{T}}$	$\mathbf{Q}$	R
Swing-up	$\begin{bmatrix} 0 \\ 0 \\ 180^\circ \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathrm{diag}(400, 1, 1, 1)$	0.01
Transfer	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} -30^\circ \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathrm{diag}(1, 1, 1, 1)$	0.1
Swing-down	$\begin{bmatrix} -30^\circ \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 180^\circ \\ 0 \end{bmatrix}$	$\mathrm{diag}(1, 1, 1, 1)$	0.01

Table 4.1: The parameters used to solve the optimal control problems for the *Fututa pendulum*.

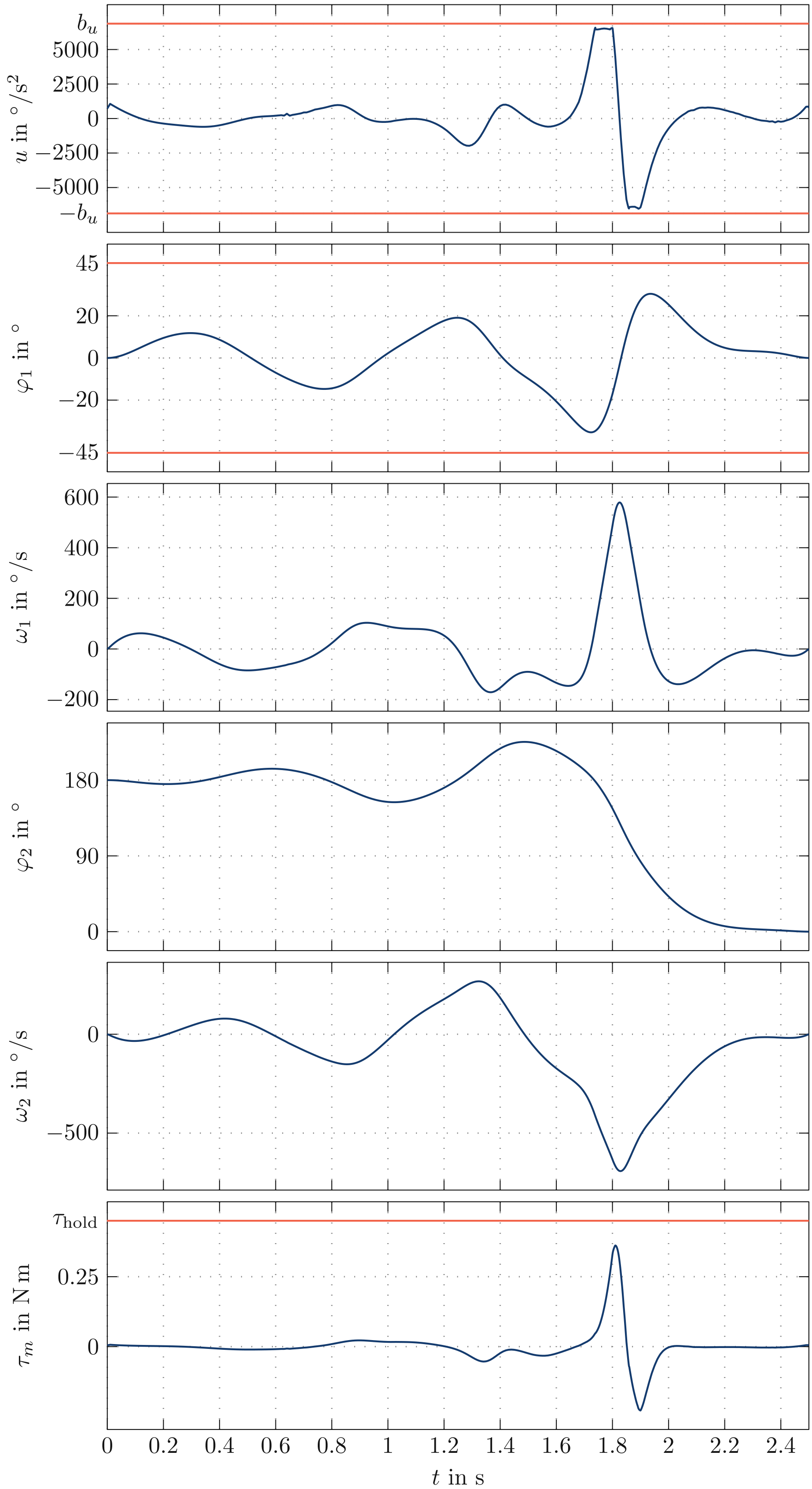

Figure 4.1: The reference swing-up trajectory for the *Furuta pendulum*.

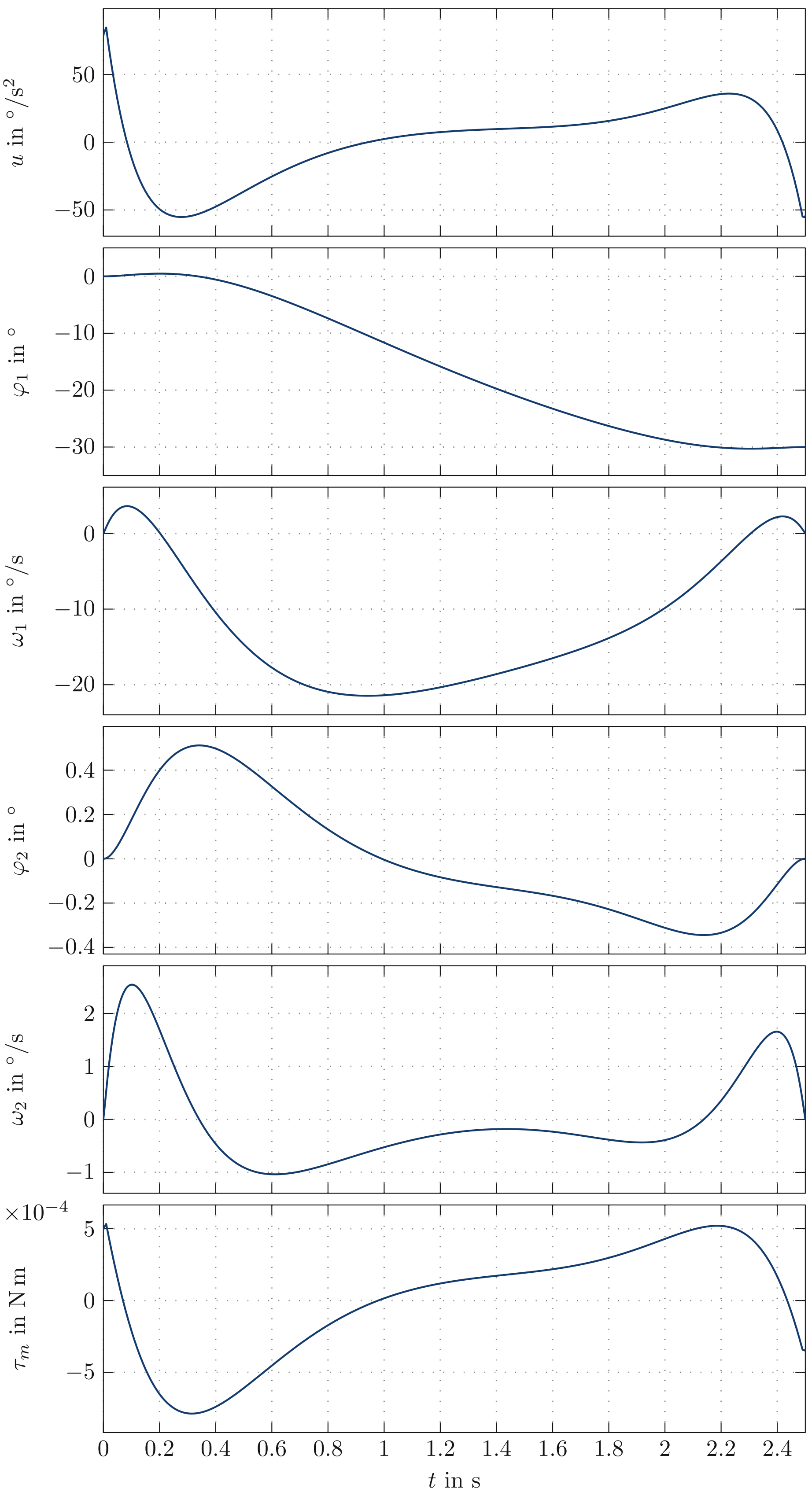

Figure 4.2: The reference transfer trajectory for the *Furuta pendulum*.

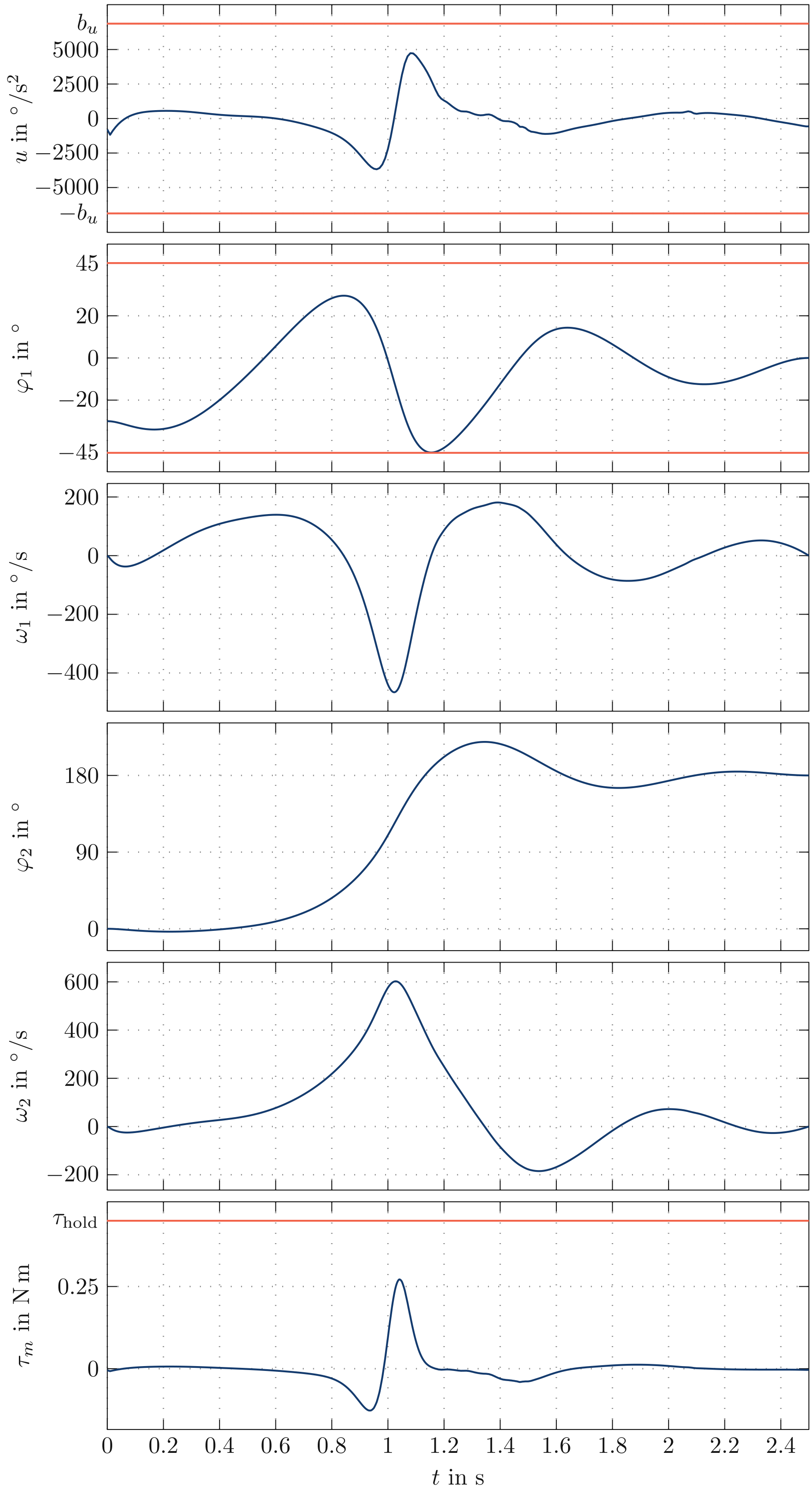

Figure 4.3: The reference swing-down trajectory for the *Furuta pendulum*.

4.2 Design of the Extended Kalman Filter

Since the *Furuta pendulum's* system states φ_1 and ω_1 from (2.21) are derived by integrating the input u, a reduced-order system is introduced for the state estimation with an *Extended Kalman Filter* [29, 30]. Reducing the state vector to $\mathbf{x}_\mathrm{r}^\mathrm{T} = \begin{bmatrix}\varphi_2 & \omega_2 & f_\mathrm{ext}\end{bmatrix}$ allows for writing the reduced-order system

$$\dot{\mathbf{x}}_\mathrm{r} = \mathbf{f}_\mathrm{r}(\mathbf{x}_\mathrm{r}, \mathbf{u}_\mathrm{r}) \ , \quad \mathbf{x}_\mathrm{r}(t_0) = \mathbf{x}_{\mathrm{r},0} \ , \tag{4.9a}$$

$$y_\mathrm{r} = h(\mathbf{x}_\mathrm{r}) = \varphi_2 \ , \tag{4.9b}$$

with the new input $\mathbf{u}_\mathrm{r}^\mathrm{T} = \begin{bmatrix}u & u_\omega\end{bmatrix}$, and the reduced-order system dynamics

$$\mathbf{f}_\mathrm{r}(\mathbf{x}_\mathrm{r}, \mathbf{u}_\mathrm{r}) = \begin{bmatrix} \omega_2 \\ \frac{(-u_\omega^2(I_{2,xx}-I_{2,yy}-l_{c2}^2 m_2)\sin(\varphi_2)+l_1 l_{c2} m_2 u)\cos(\varphi_2)+g\sin(\varphi_2) l_{c2} m_2 + f_\mathrm{ext} l_2 - d_2\omega_2)}{l_{c2}^2 m_2 + I_{2,zz}} \\ 0 \end{bmatrix} . \tag{4.10}$$

Next, the reduced-order system is discretized with the new input variable $\mathbf{u}_\mathrm{r}^\mathrm{T}(kT_s) = \mathbf{u}_{\mathrm{r},k}^\mathrm{T} = \begin{bmatrix}u_k & u_{\omega,k}\end{bmatrix}$, which is constant over the sampling interval T_s, and the system state $\mathbf{x}_\mathrm{r}(kT_s) = \mathbf{x}_{\mathrm{r},k}$, with $k \in \mathbb{N}_0$. The angular velocity input $u_{\omega,k}$ is derived by integrating u_k using the *Euler method*, which is also used for discretizing the reduced-order system

$$\mathbf{x}_{\mathrm{r},k+1} = \mathbf{F}_{\mathrm{r},k}(\mathbf{x}_{\mathrm{r},k}, \mathbf{u}_{\mathrm{r},k}, \mathbf{w}_k) = \mathbf{x}_{\mathrm{r},k} + T_\mathrm{s}\mathbf{f}_\mathrm{r}(\mathbf{x}_{\mathrm{r},k}, \mathbf{u}_{\mathrm{r},k}) + \mathbf{w}_k \ , \tag{4.11a}$$

$$y_{\mathrm{r},k} = h(\mathbf{x}_{\mathrm{r},k}) + v_k \ , \tag{4.11b}$$

where $\mathbf{w}_k \in \mathbb{R}^3$ is the process noise, and $v_k \in \mathbb{R}$ is measurement noise. Both $\mathbf{w}_k$ and v_k are assumed to be zero-mean *Gaussian* noises. The estimated state $\hat{\mathbf{x}}_{\mathrm{r},k} = \begin{bmatrix}\hat{\varphi}_{2,k} & \hat{\omega}_{2,k} & \hat{f}_{\mathrm{ext},k}\end{bmatrix}^\mathrm{T}$ is denoted as $\hat{\mathbf{x}}_{\mathrm{r},k}^+$ for the posteriori estimate at the computation step k with measurements up to the computation step k. Similarly, $\hat{\mathbf{x}}_{\mathrm{r},k}^-$ denotes the a-priori estimate considering measurements up to the computation step $k-1$. Together with the output matrix

$$\mathbf{C} = \frac{\partial h(\hat{\mathbf{x}}_{\mathrm{r},k}^-)}{\partial \mathbf{x}_{\mathrm{r},k}} = \begin{bmatrix}1 & 0 & 0\end{bmatrix} , \tag{4.12}$$

the update step from the measurement y_k is given as [29, 30]

$$\mathbf{l}_k = \mathbf{P}_k^-\mathbf{C}^\mathrm{T}(\mathbf{C}\mathbf{P}_k^-\mathbf{C}^\mathrm{T} + R)^{-1} \ , \tag{4.13a}$$

$$\hat{\mathbf{x}}_{\mathrm{r},k}^+ = \hat{\mathbf{x}}_{\mathrm{r},k}^- + \mathbf{l}_k(y_k - h(\hat{\mathbf{x}}_{\mathrm{r},k}^-)) \ , \qquad \hat{\mathbf{x}}_{\mathrm{r},0}^- = \mathbf{x}_{\mathrm{r},0} \ , \tag{4.13b}$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{l}_k\mathbf{C})\mathbf{P}_k^- \ , \qquad \mathbf{P}_0^- = 0.1\mathbf{I} \ , \tag{4.13c}$$

with the gain $\mathbf{l}_k \in \mathbb{R}^3$, the covariance matrix of the estimation error $\mathbf{P}_k \in \mathbb{R}^{3\times3}$, and the parameter $R \in \mathbb{R}$. The covariance matrix $\mathbf{P}_k$ is initialized to $\mathbf{P}_0^- = 0.1\mathbf{I}$ because the initial state $\mathbf{x}_{\mathrm{r},0}^\mathrm{T} = \begin{bmatrix}180° & 0 & 0\end{bmatrix}$ of the *Furuta pendulum* is well-known. Let

$$\mathbf{\Phi}_{\mathrm{r},k} = \frac{\partial \mathbf{F}_{\mathrm{r},k}(\hat{\mathbf{x}}_{\mathrm{r},k}^+, \mathbf{u}_{\mathrm{r},k}, \mathbf{0})}{\partial \mathbf{x}_{\mathrm{r},k}} \tag{4.14}$$

be the time-varying dynamic matrix of the reduced-order system; then, the propagation step [29, 30] is given as

$$\hat{\mathbf{x}}^{-}_{\mathrm{r},k+1} = \mathbf{F}_{\mathrm{r},k}(\hat{\mathbf{x}}^{+}_{\mathrm{r},k}, \mathbf{u}_{\mathrm{r},k}, \mathbf{0}) \ , \tag{4.15a}$$

$$\mathbf{P}^{-}_{k+1} = \mathbf{\Phi}_{\mathrm{r},k}\mathbf{P}^{+}_{k}\mathbf{\Phi}^{\mathrm{T}}_{\mathrm{r},k} + \mathbf{Q} \ , \tag{4.15b}$$

with the parameter $\mathbf{Q} \in \mathbb{R}^{3\times3}$. Experiments on the physical *Furuta pendulum* have shown that choosing the parameters $\mathbf{Q} = \mathrm{diag}(0.01, 0.01, 0.005)$ and $R = 1$ results in sufficient estimation characteristics.

The arm's estimated angular velocity $\hat{\omega}_{1,k}$ at the computation step k is integrated using the *Euler method* and can be expressed as $\hat{\omega}_{1,k+1} = \hat{\omega}_{1,k} + T_s u_k$. Further, the arm's estimated angle $\hat{\varphi}_{1,k}$ is obtained from the stepper motor controller. Finally, the *Furuta pendulum's* estimated state

$$\hat{\mathbf{x}}_k = \begin{bmatrix} \hat{\varphi}_{1,k} \\ \hat{\omega}_{1,k} \\ \hat{\varphi}_{2,k} \\ \hat{\omega}_{2,k} \end{bmatrix} \tag{4.16}$$

is formed from the arm's estimated angle $\hat{\varphi}_{1,k}$, the arm's estimated angular velocity $\hat{\omega}_{1,k}$, and the estimated state variables $\hat{\varphi}_{2,k}$ and $\hat{\omega}_{2,k}$. The estimation of the external force $\hat{f}_{\mathrm{ext},k}$ has the benefit that external forces at the tip of the pendulum are estimated without leading to a large estimation error of $\hat{\varphi}_{2,k}$ and $\hat{\omega}_{2,k}$.

4.3 Design of the Time-Varying LQR

In order to control the *Futura pendulum* about the previously described reference trajectories for the swing-up, the transfer, and the swing-down, a time-varying *Linear Quadratic Regulator* (*LQR*) [29] is designed. Further, the *Furuta pendulum* is stabilized in the upper and the lower equilibrium points after the swing-up and the swing-down, respectively.

Let $\Delta\mathbf{x}(t)$ and $\Delta u(t)$ be sufficiently small deviations from the reference trajectory with $\mathbf{x}^*(t)$ and $u^*(t)$. Then, the actual system state $\mathbf{x}(t)$ and the actual input $u(t)$ from the *Furuta pendulum* system, as expressed in (2.21), are written as

$$\mathbf{x}(t) = \mathbf{x}^*(t) + \Delta\mathbf{x}(t) \ , \tag{4.17a}$$

$$u(t) = u^*(t) + \Delta u(t) \ . \tag{4.17b}$$

The state space model for small deviations from the reference trajectory is given as

$$\Delta\dot{\mathbf{x}}(t) = \mathbf{A}(t)\Delta\mathbf{x}(t) + \mathbf{b}_u(t)\Delta u(t) \ , \quad \Delta\mathbf{x}(t_0) = \Delta\mathbf{x}_0 \ , \tag{4.18}$$

with the system matrix $\mathbf{A}(t)$ and the input vector $\mathbf{b}_u(t)$, as expressed in

$$\mathbf{A}(t) = \frac{\partial\mathbf{f}(\mathbf{x}^*(t), u^*(t), 0)}{\partial\mathbf{x}} \ , \quad \mathbf{b}_u(t) = \frac{\partial\mathbf{f}(\mathbf{x}^*(t), u^*(t), 0)}{\partial u} \ . \tag{4.19}$$

Further, the system is approximated using the zero-order hold method [31] with MATLAB's `c2d` function [20] and is written as the discretized system

$$\Delta\mathbf{x}_{k+1} = \mathbf{\Phi}_k\Delta\mathbf{x}_k + \mathbf{\Gamma}_k\Delta u_k, \quad \Delta\mathbf{x}(t_0) = \Delta\mathbf{x}_0 \ , \tag{4.20}$$

with $\Delta\mathbf{x}_k = \Delta\mathbf{x}(kT_\mathrm{s})$, $\Delta u_k = \Delta u(kT_\mathrm{s})$, and $k \in 0, 1, \ldots, N$. The discretized system is extended by an integrator

$$\Delta x_{\mathrm{I},k+1} = \Delta x_{\mathrm{I},k} + \mathbf{a}_\mathrm{I}^\mathrm{T} \Delta\mathbf{x}_k, \quad \Delta x_\mathrm{I}(t_0) = 0 \;, \tag{4.21}$$

with the vector

$$\mathbf{a}_\mathrm{I}^\mathrm{T} = \begin{bmatrix} -T_\mathrm{s} & 0 & 0 & 0 \end{bmatrix} \tag{4.22}$$

to compensate for constant disturbances. Finally, the extended system

$$\underbrace{\begin{bmatrix} \Delta\mathbf{x}_{k+1} \\ \Delta x_{\mathrm{I},k+1} \end{bmatrix}}_{\Delta\mathbf{x}_{\mathrm{e},k+1}} = \underbrace{\begin{bmatrix} \mathbf{\Phi}_k & \mathbf{0} \\ \mathbf{a}_\mathrm{I}^\mathrm{T} & 1 \end{bmatrix}}_{\mathbf{\Phi}_{\mathrm{e},k}} \underbrace{\begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta x_{\mathrm{I},k} \end{bmatrix}}_{\Delta\mathbf{x}_{\mathrm{e},k}} + \underbrace{\begin{bmatrix} \mathbf{\Gamma}_k \\ 0 \end{bmatrix}}_{\mathbf{\Gamma}_{\mathrm{e},k}} \Delta u_k, \quad \Delta\mathbf{x}_\mathrm{e}(t_0) = \begin{bmatrix} \Delta\mathbf{x}_0 \\ 0 \end{bmatrix} , \tag{4.23}$$

is written with the extended system matrix $\mathbf{\Phi}_{\mathrm{e},k}$, the extended state vector $\Delta\mathbf{x}_{\mathrm{e},k}$, and the extended input vector $\mathbf{\Gamma}_{\mathrm{e},k}$. This allows for designing the time-varying *LQR* by solving the discrete *Riccati equation* [29]

$$\mathbf{k}_k^\mathrm{T} = (\mathbf{\Gamma}_{\mathrm{e},k}^\mathrm{T} \mathbf{P}_{k+1} \mathbf{\Gamma}_{\mathrm{e},k} + R)^{-1} \mathbf{\Gamma}_{\mathrm{e},k}^\mathrm{T} \mathbf{P}_{k+1} \mathbf{\Phi}_{\mathrm{e},k} \;, \tag{4.24a}$$

$$\mathbf{P}_k = \mathbf{Q} + \mathbf{\Phi}_{\mathrm{e},k}^\mathrm{T} \mathbf{P}_{k+1} \mathbf{\Phi}_{\mathrm{e},k} - \mathbf{\Phi}_{\mathrm{e},k}^\mathrm{T} \mathbf{P}_{k+1} \mathbf{\Gamma}_{\mathrm{e},k} \mathbf{k}_k^\mathrm{T} \;, \tag{4.24b}$$

with the positive definite matrix $\mathbf{Q} \in \mathbb{R}^{5\times5}$, the positive design parameter $R \in \mathbb{R}$, the gain vector $\mathbf{k}_k \in \mathbb{R}^5$, and the matrix $\mathbf{P}_k \in \mathbb{R}^{5\times5}$ backward in time starting from the solution of the algebraic *Riccati equation* where $\mathbf{P}_{N+1} = \mathbf{P}_N = \mathbf{P}_\infty$ [29]. The control law of the time-varying *LQR* is given by

$$\Delta u_k = -\mathbf{k}_k^\mathrm{T} \Delta\mathbf{x}_{\mathrm{e},k} \;. \tag{4.25}$$

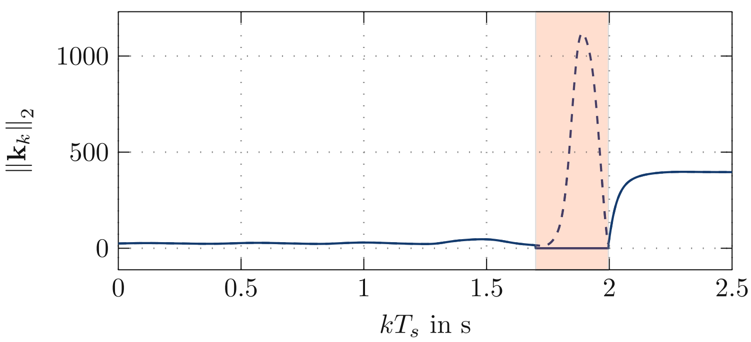

Figure 4.4: The *Euclidean* norm of the *LQR* gain vector during the swing-up. The peak of the norm of the gain vector around the uncontrollable state is dashed.

However, the controllability matrix $\mathcal{C} = \begin{bmatrix} \mathbf{\Gamma}_{\mathrm{e},k} \;, & \mathbf{\Phi}_{\mathrm{e},k}\mathbf{\Gamma}_{\mathrm{e},k} \;, & \ldots \;, & \mathbf{\Phi}_{\mathrm{e},k}^4 \mathbf{\Gamma}_{\mathrm{e},k} \end{bmatrix}$ loses its full rank when the pendulum angle is $\varphi_2 = 90° + i180°, i \in \mathbb{Z}$. Therefore, the *LQR* from (4.25) is switched off when following the reference swing-up trajectory when the pendulum is close to the uncontrollable state. However, the *Euclidean* norm of the gain vector

$\|\mathbf{k}_k\|_2$ peaks around the uncontrollable state, as shown with the dashed line in Figure 4.4. Therefore, the *LQR* is switched off when $kT_s \in [1.7\,\mathrm{s}, 1.996\,\mathrm{s}]$ in order to avoid switching the *LQR* at high magnitudes of $\mathbf{k}_k$. The continuous line in Figure 4.4 shows the *Euclidean* norm of the gain vector $\|\mathbf{k}_k\|_2$ considering that $\mathbf{k}_k = \mathbf{0}$ when $kT_s \in [1.7\,\mathrm{s}, 1.996\,\mathrm{s}]$. Further, the *LQR* is switched off around the uncontrollable state when following the swing-down trajectory at $kT_s \in [0.9\,\mathrm{s}, 1.0\,\mathrm{s}]$. Additionally, the integrator state $\Delta x_{\mathrm{I},k}$ is reset to zero when the controller is switched off and during the swing-up.

The *Furuta pendulum* is stabilized at each equilibrium point using the gain vector

$$\mathbf{k}_\infty^{\mathrm{T}} = (\mathbf{\Gamma}_{\mathrm{e},N}^{\mathrm{T}} \mathbf{P}_\infty \mathbf{\Gamma}_{\mathrm{e},N} + R)^{-1} \mathbf{\Gamma}_{\mathrm{e},N}^{\mathrm{T}} \mathbf{P}_\infty \mathbf{\Phi}_{\mathrm{e},N} \ , \tag{4.26}$$

where $\mathbf{P}_\infty$ is the solution of the algebraic *Riccati equation*, which was obtained using MATLAB's `idare` function [20].

Experiments on the physical *Furuta pendulum* have shown that choosing the design parameters as $\mathbf{Q} = \mathrm{diag}(5 \cdot 10^3, 10^2, 10^{-1}, 10^{-1}, 10^2)$ and $R = 10$ achieves sufficient control characteristics for low arm deflections from the reference trajectories for the swing-up and transfer trajectory control. Similarly, $\mathbf{Q} = \mathrm{diag}(10^2, 1, 10^2, 1, 1)$ and $R = 0.1$ are chosen for the swing-down trajectory control.

5 Validation

This chapter discusses the results from the validation of the proposed method introduced in the previous chapters to stabilize the *Furuta pendulum* about the reference trajectories by means of event camera feedback. Firstly, the closed-loop system, consisting of the *Furuta pendulum's* mathematical model, the *Extended Kalman Filter*, the reference trajectory generator, the simulated event camera feedback, and the time-varying *LQR*, is validated by simulation. Secondly, experiments on the physical *Furuta pendulum* with event-based camera feedback are conducted to validate the functionality and compare the closed-loop system performance using encoder feedback. Thirdly, an experiment with an external force applied to the tip of the pendulum is performed to assess the robustness to external disturbances.

5.1 Validation by Simulation

In this section, the simulated *Furuta pendulum* is stabilized about the reference trajectories in MATLAB/SIMULINK [20] in order to validate the proposed system design and architecture.

The camera feedback is simulated by first projecting the simulated pendulum angle φ_2 onto the camera sensor to obtain the projected angle of the pendulum $\varphi_{2,\mathrm{h},i}$ by solving (3.20). In order to simulate the output of the *Hough transform* algorithm, the projected angle $\varphi_{2,\mathrm{h},i}$ is quantized to $0.1°$ to match the *Hough space* quantization. Next, the angle $\varphi_{2,i}$ is calculated from the quantized projected angle $\varphi_{2,\mathrm{h},i}$ using (3.21) to obtain the pendulum's angle from the simulated camera feedback.

The trajectory generator is implemented using a finite-state machine and performs the swing-up, followed by a 10 s stabilization phase in the upper equilibrium point. After the stabilization phase, the transfer trajectory moves the arm's angle to a terminal angle of $-30°$ while balancing the pendulum, followed by another 10 s stabilization phase. Then, the simulated *Furuta pendulum* follows the reference swing-down trajectory starting from the arm's angle of $-30°$ to reach the final state at the lower equilibrium point. The *Extended Kalman Filter* and the time-varying *LQR* are implemented with the parameters described in Sections 4.2 and 4.3 with a cycle time of 1 ms.

Figure 5.1 shows the results of the *Furuta pendulum's* simulated swing-up. The area highlighted in red shows where the controller is switched off. The proposed control system architecture successfully swings up the *Furuta pendulum* in simulation. However, between $t = 2\,\mathrm{s}$ and $t = 2.5\,\mathrm{s}$ the simulated trajectory deviates from the reference trajectory. The reason for this could be the discretization of the system dynamics in the formulation of the optimization problem and the numerical inaccuracies induced by the simulation solver.

The experimentally chosen external force of 5 mN is applied at the tip of the pendulum at $t = 5.5\,\mathrm{s}$ to induce an arm deflection, similar to the discussed experiment on the physical

Furuta pendulum. However, it is important to have low arm deflections to limit the loss of angular resolution with event camera feedback. Figure 5.2 shows the stabilization phase with the external force at the tip of the pendulum. The time-varying *LQR* stabilizes the *Furuta pendulum* in simulation, but the arm's angle deflects to approximately $\varphi_1 \approx -20°$. In this phase, the integrator state $\Delta x_{\mathrm{I},k}$ is increasing. Thus, the arm's angle is approaching the reference value. After releasing the external force at $t = 7.5\,\mathrm{s}$, the arm's angle returns to the reference trajectory, but it shows an offset of approximately 5°. This behavior can be observed because the error of the arm's angle to the reference trajectory is integrated with the integrator state $\Delta x_{\mathrm{I},k}$ while the external force is present. As the integrator state $\Delta x_{\mathrm{I},k}$ decreases, the arm's angle φ_1 returns to the reference trajectory. The quantization effect is clearly visible in the estimated system states and the estimated external force, as shown in Figure 5.2. Further, oscillations of the arm's angle φ_1 about the reference trajectory are observed. Disabling the quantization of the projected angle $\varphi_{2,\mathrm{h},i}$ eliminates these noticeable oscillations. Thus, the angular resolution should be as high as possible to reduce these oscillations. Compared to Figure 5.1, the quantization noise seems more prominent because of the axis' scaling.

After the transfer, the *Furuta pendulum* follows the swing-down trajectory at $t = 25\,\mathrm{s}$, as shown in Figure 5.3, to be finally stabilized about the lower equilibrium point. Similarly to the swing-up, the controller is switched off in the area highlighted in red. The observed trajectory of the simulated *Furuta pendulum* matches the reference trajectory well.

This section shows, that the system architecture with its chosen design parameters is able to stabilize the *Furuta pendulum* about the reference trajectories in simulation. However, the *Hough transform* algorithm is not implemented in simulation. The following section describes the implementation and the experimental setup on the physical Furuta pendulum with event camera feedback. Further, the results from experiments on the physical *Furuta pendulum* are discussed.

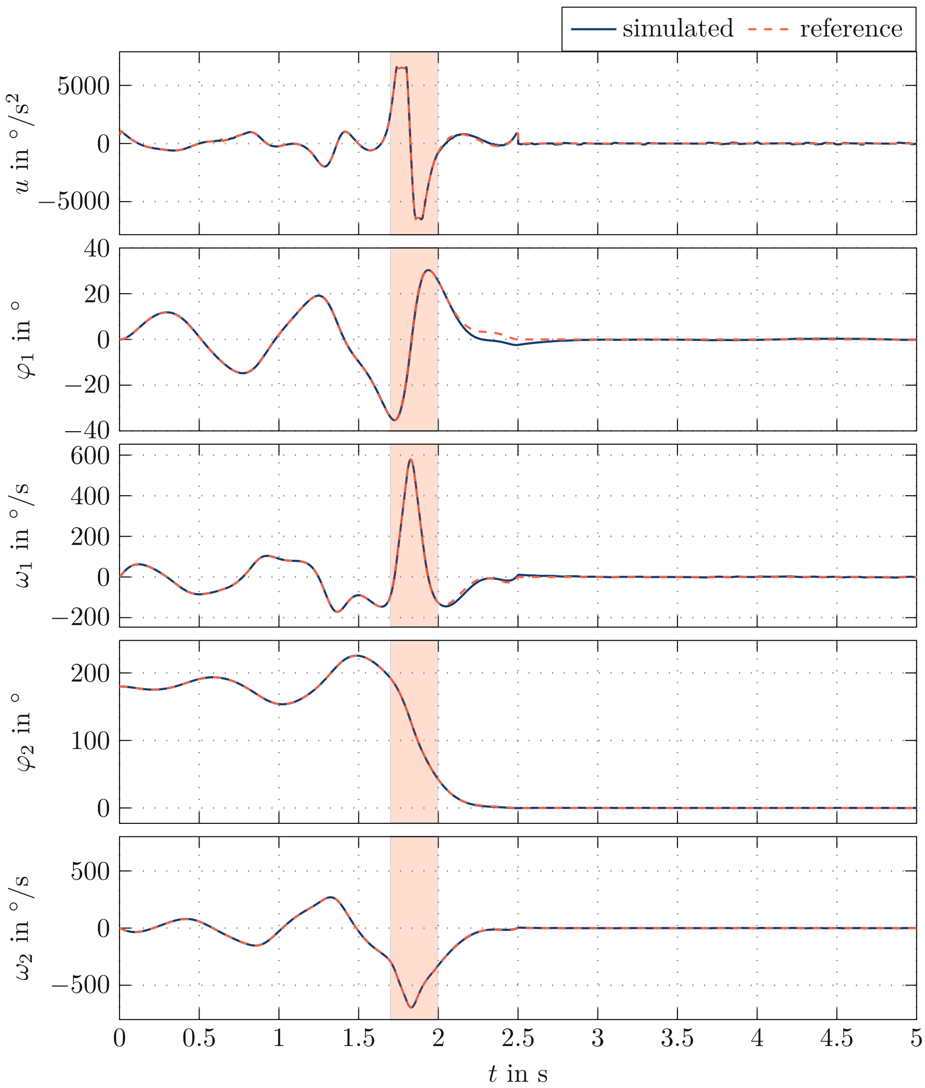

Figure 5.1: The simulation of the swing-up with the estimated system states $\hat{\varphi}_1$, $\hat{\omega}_1$, $\hat{\varphi}_2$, $\hat{\omega}_2$, the simulated input u, and the reference trajectory.

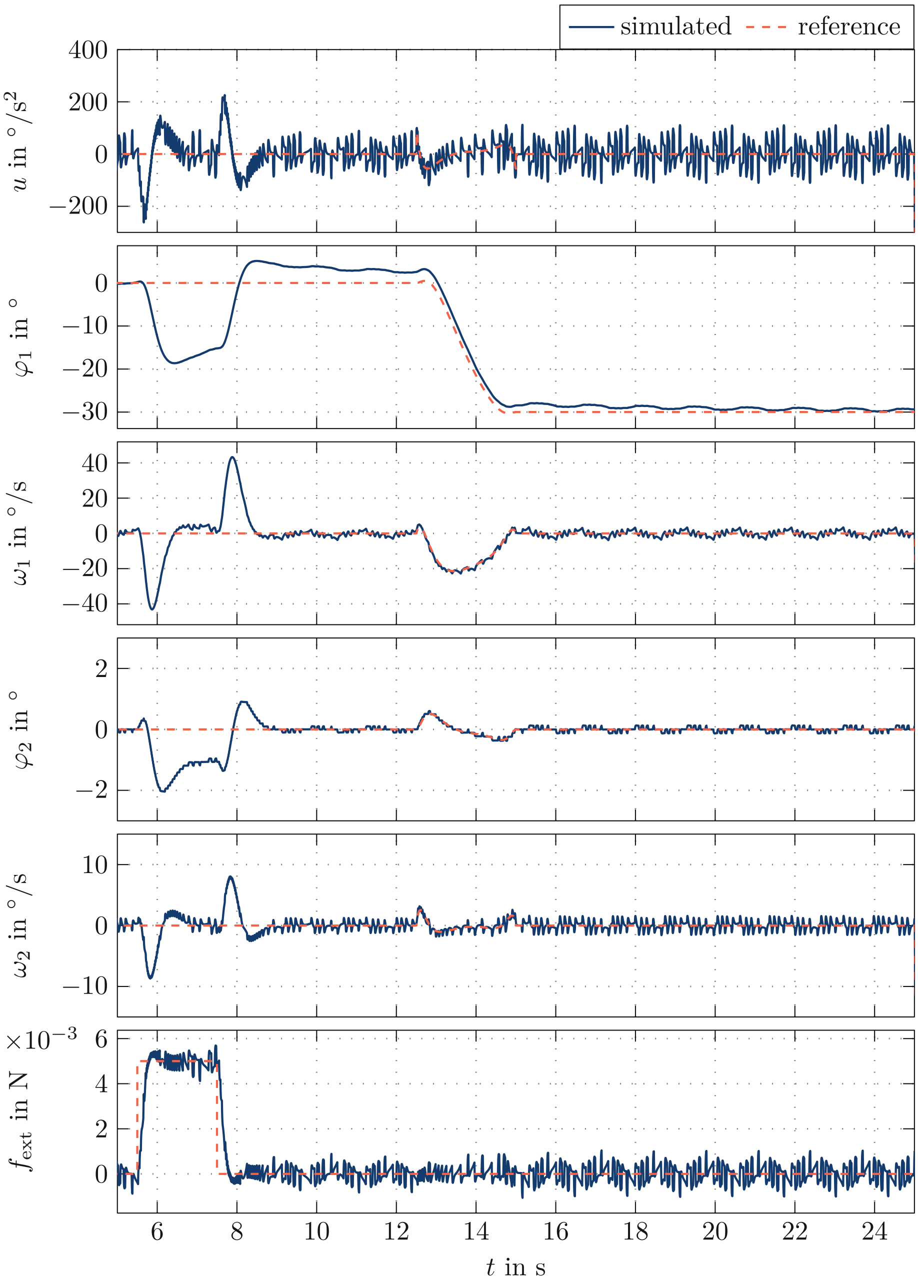

Figure 5.2: The simulation of the transfer with the estimated system states $\hat{\varphi}_1$, $\hat{\omega}_1$, $\hat{\varphi}_2$, $\hat{\omega}_2$, the simulated input u, and the reference trajectory. An external force f_{ext} is applied at $t = 5.5\,\mathrm{s}$ at the tip of the pendulum.

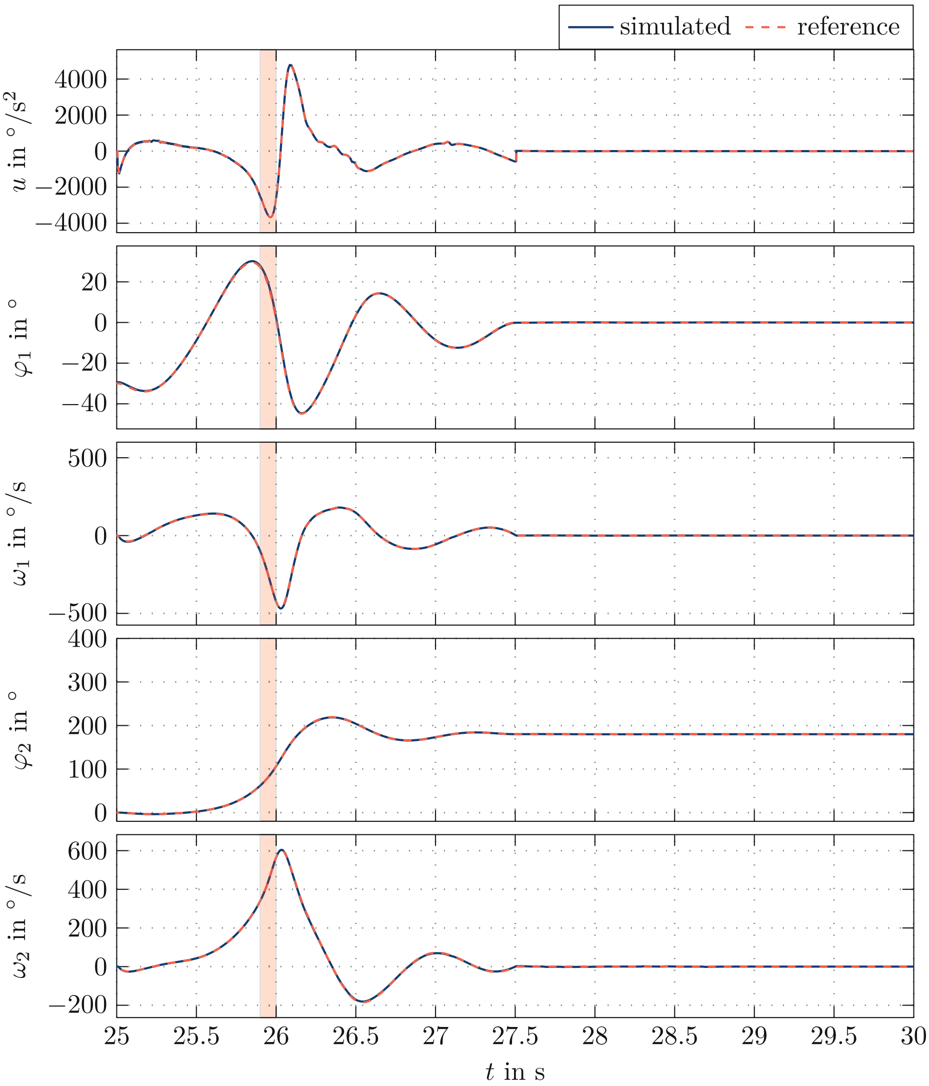

Figure 5.3: The simulation of the swing-down with the estimated system states $\hat{\varphi}_1$, $\hat{\omega}_1$, $\hat{\varphi}_2$, $\hat{\omega}_2$, the simulated input u, and the reference trajectory.

5.2 Validation by Experiments

This section starts with a description of the experimental setup and discusses the results of the conducted experiments with the physical *Furuta pendulum*. Experiments with encoder feedback and with event camera feedback are performed to do a comparative analysis. Similarly to the experiment in simulation, an external force is applied at the tip of the pendulum while balancing with event camera feedback in order to assess the robustness against external forces. Figure 5.4 shows an overview of the experimental setup and its

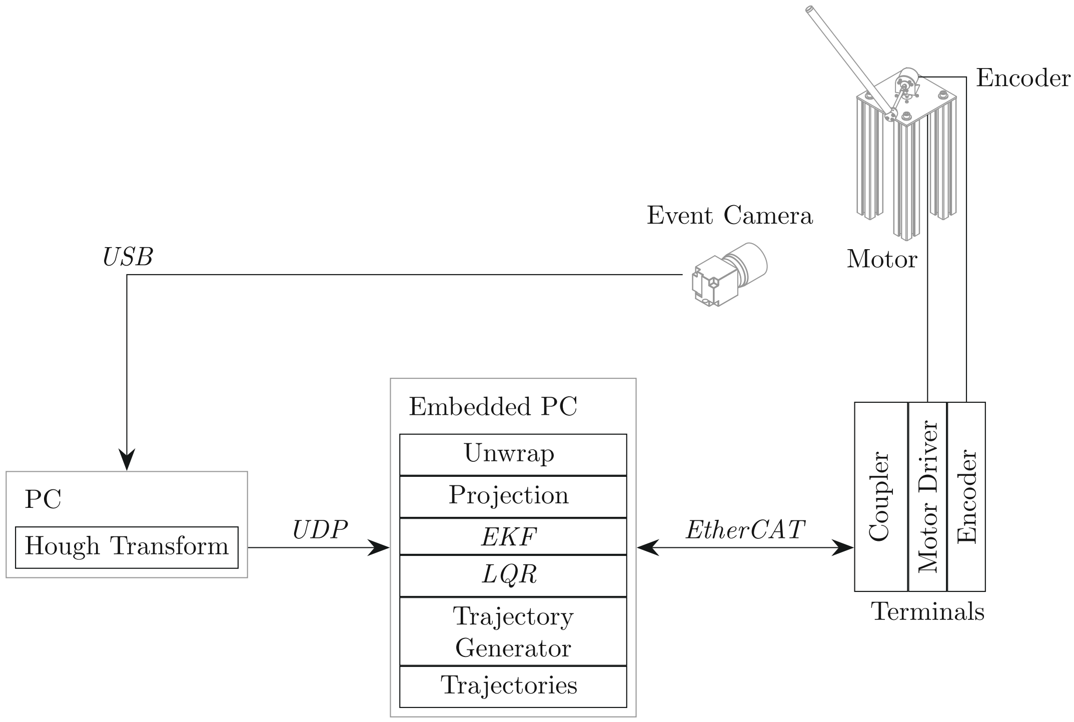

Figure 5.4: An overview of the setup for the experiments showing the hardware components and the respective software modules.

components with their respective software modules. The event camera is connected via *USB* to a *Windows 10* PC, where the *Hough transform* algorithm with exponential decay, as described in Chapter 3, is implemented. The *Metavision SDK* [32] is used to interface the event camera and to implement the *Hough transform* with exponential decay using *Microsoft's Visual C++* [33]. An *Ethernet* connection to the *Beckhoff CX5130* Embedded PC is used to establish a *UDP* connection for the transfer of the angle obtained from the *Hough transform* algorithm $\varphi_{2,\mathrm{h},i}$. Further, the calculated event rate over a time window of 30 ms is transferred for evaluation purposes. The Embedded PC receives the *UDP* frames with the angle obtained from the *Hough transform* algorithm to perform an unwrapping of the angle, as described in (3.12). Subsequently, the projection, as described in (3.21), is calculated. The *Extended Kalman Filter* (*EKF*) estimates the *Furuta pendulum's* system state to stabilize the *Furuta pendulum* about the reference trajectories from Chapter 4 using the time-varying *LQR*. The trajectory generator module outputs the offline calculated reference swing-up, the transfer, and the swing-down trajectories. Communication with the *Beckhoff* terminals to drive the stepper motor using the *EL7041* terminal with 1/64 micro-stepping and to interface the encoder with the *EL5101* terminal is achieved utilizing the *EtherCAT* protocol. The *Beckhoff EK1100* is the *EtherCAT* bus coupler for the terminals.

The software modules on the Embedded PC, as depicted in Figure 5.4, are implemented using *Beckhoff's TC1200 PLC* [34], *TF6311* [35] for the *UDP* communication, and *TF2000* [36] for interacting with the *PLC* using a web-based user interface.

The *SilkyEvCam EvC3A* event camera offers a resolution of 640 × 480 pixels and is equipped with an 8 mm focal length lens [25]. The event camera is mounted at the distance $x_{\mathrm{cam}} = 0.52\,\mathrm{m}$. The aperture is set to 16 to have an extensive depth of field. The lens was focused using a smartphone with an OLED display with a blinking pattern, allowing the event camera to be focused like a frame camera. The contrast sensitivity bias settings `bias_diff_on` and `bias_diff_off` of the event camera are set to have a good trade-off between noise and sensitivity to the movement of the pendulum [24]. The dead time bias `bias_refr` is increased to reduce the number of generated events from the same pixel by a significant brightness change [24]. All other bias settings are left to their default values. Table 5.1 summarizes the chosen bias settings.

Bias Setting	**Value**
`bias_diff`	299
`bias_diff_off`	171
`bias_diff_on`	445
`bias_fo`	1477
`bias_hpf`	1448
`bias_pr`	1250
`bias_refr`	1652

Table 5.1: The bias settings of the event camera used in the experimental setup.

Parameter	**Value**
N_φ	1800
N_d	1600 pixel
$\Delta\overline{d}$	60
Δa	50
τ	0.3
h_{T}	10 events

Table 5.2: The parameters for the vision processing algorithm.

The *Hough transform* algorithm was implemented using *Microsoft Visual C++* [33], as described in Chapter 3. The *Hough space* is quantized to 0.1° and 1600 pixel, respectively.

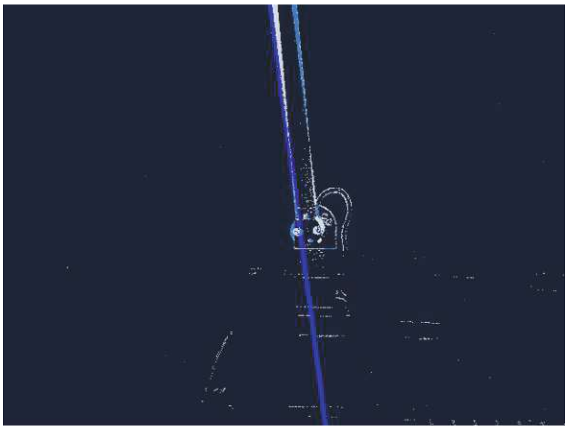

Figure 5.5: An event frame during the end of the swing-up with an accumulation time of 100 ms and with the line detected by the *Hough transform* algorithm.

The maximum search is done in the region ± 60 pixel and $\pm 5°$ around the previously found maximum, which is a good trade-off between processing time and detection reliability. Further, the threshold h_{T} for updating the last found maximum in the *Hough space* from (3.11) is chosen to be 10 events to detect the pendulum reliably but still be robust against noise. The decay parameter from (3.8) is chosen to be $\tau = 0.3$ with a cycle time of 2 ms, which allows for detecting the pendulum when having fast movements during the swing-up and the swing-down, but also to reliably detect the pendulum when there are few events while balancing the pendulum in the upper or lower equilibrium position. The cycle time for the computation of the decay is chosen to be 2 ms because it is a computation-intensive calculation. The parameters for the vision processing are summarized in Table 5.2. In order to increase the maximum event rate that is processable by the algorithm, the compiler optimization was set to optimize for speed. Moreover, optimizations in the code, such as parallelizing the exponential decay calculation using *Microsoft's Parallel Patterns Library* [33], are implemented. The *UDP* communication to send the detected angle in the *Hough space* every 1 ms is done using *Microsoft's winsock2* library [33]. *Prophesee's* `metavision_hal_viewer` [32] example is extended to additionally visualize the detected line from the *Hough transform* using *OpenCV* [37]. Figure 5.5 shows an event frame with an accumulation time of 100 ms of the detected line during the swing-up.

5.2.1 Stabilization about the Reference Trajectories

Figures 5.6 and 5.7 compare the *Furuta pendulum's* swing-up using event-based vision and encoder feedback. The proposed setup stabilizes the *Furuta pendulum* about the reference swing-up trajectory with event camera feedback. The controller is turned off in the red highlighted area, leading to errors relative to the reference trajectory for the pendulum's angular velocity $\Delta\omega_2 = \omega_2^* - \hat{\omega}_2$ and the pendulum's angle $\Delta\varphi_2 = \varphi_2^* - \hat{\varphi}_2$, which both peak in this phase for the swing-up with encoder and event camera feedback. However, $\Delta\omega_2$ and $\Delta\varphi_2$ reach sufficiently small values before the controller is switched back on at $t = 1.997$ s. The arm's angle φ_1 deflects almost 30° from the reference trajectory, as shown in Figure 5.7 with $\Delta\varphi_1 = \varphi_1^* - \hat{\varphi}_1$ and $\Delta\omega_1 = \omega_1^* - \hat{\omega}_1$. This effect could result from unmodeled effects, such as the finite stiffness of the 3D-printed parts. The encoder measures the pendulum's angle $\varphi_{2,\text{enc}}$ to compare the accuracy of the event-based angle measurement. The error of the camera measurement $\Delta\varphi_{2,\text{cam}} = \varphi_{2,\text{enc}} - \varphi_{2,\text{cam}}$ correlates with the estimated angular velocity of the pendulum $\hat{\omega}_2$, indicating that the error results from the latency of the camera measurement $\varphi_{2,\text{cam}}$. The event rate R_{ev} reaches its maximum of approximately $1.5 \cdot 10^6$ events per second during the fast movement of the pendulum and the arm in the event camera's field of view. This shows that the implemented *Hough transform* with exponential decay detects the pendulum even with extensive movement in the scene. However, a considerable amount of generated events does not originate from the pendulum's movement but from the movement of the arm or the encoder.

After $t = 2.5\,\text{s}$, in the stabilization phase, the arm's angle φ_1 approaches the reference angle as the integrator starts to integrate the error $\Delta\varphi_1$ between the arm's angle and the reference trajectory. However, oscillations in the arm's angle φ_1 and the pendulum's angle φ_2 are observed, as shown in Figures 5.8 and 5.9. The oscillations in the stabilization with encoder feedback could result from the low angular resolution of the encoder measurement of 0.15°. Similarly, oscillations in the stabilization with event camera feedback are observed, but with a higher magnitude and a higher frequency in φ_2 and ω_2. These oscillations could originate from the angular quantization in the *Hough space* of 0.1° in combination with the measurement noise and the latency in the camera measurement $\varphi_{2,\text{cam}}$. The error of the camera-based measurement $\Delta\varphi_{2,\text{cam}}$ shows some outliers of over 2° in magnitude, but although the event rate R_{ev} is relatively low, the *Hough transform* with exponential decay successfully detects the pendulum even in conditions with low event rates. The transfer starts after the stabilization phase at $t = 12.5\,\text{s}$ to move the arm's angle φ_1 to its terminal angle of $-30°$ while balancing the pendulum. Both the control with event camera feedback and with the encoder feedback achieve the transfer of the *Furuta pendulum*'s arm to the terminal angle and stabilize it before the *Furuta pendulum* follows the swing-down trajectory.

The swing-down trajectory starts at $t = 25\,\text{s}$, as depicted in Figure 5.10, and ends at the lower equilibrium point. The controller is turned off at $t \in [25.9\,\text{s}, 26.0\,\text{s}]$, as shown in the plot Δu in Figure 5.11. The errors with respect to the pendulum's reference trajectory $\Delta\varphi_2$ and $\Delta\omega_2$ for the swing-down with encoder and with camera feedback show a similar behavior, indicating that model errors or unmodeled effects could be the reason for this. Increased noise is visible for the swing-down by means of event camera feedback. The camera measurement error $\Delta\varphi_{2,\text{cam}}$ correlates with the estimated pendulum's angular velocity $\hat{\omega}_2$. This correlation could be prominent due to the latency of the camera measurements. As depicted in Figure 5.10, the event rate R_{ev} peaks at $t \approx 25.95\,\text{s}$ to its maximum of over $1.5 \cdot 10^6$ events per second, which is similar to the event rate during the swing-up.

The swing-down trajectory ends at $t = 27.5\,\text{s}$ to start the stabilization at the lower equilibrium point with hardly any movement of the pendulum, as shown in the plot of the pendulum's angular velocity ω_2 in Figure 5.10. Although the event rate decays to approximately 200 events per second, the *Hough transform* with exponential decay is still able to detect the pendulum reliably.

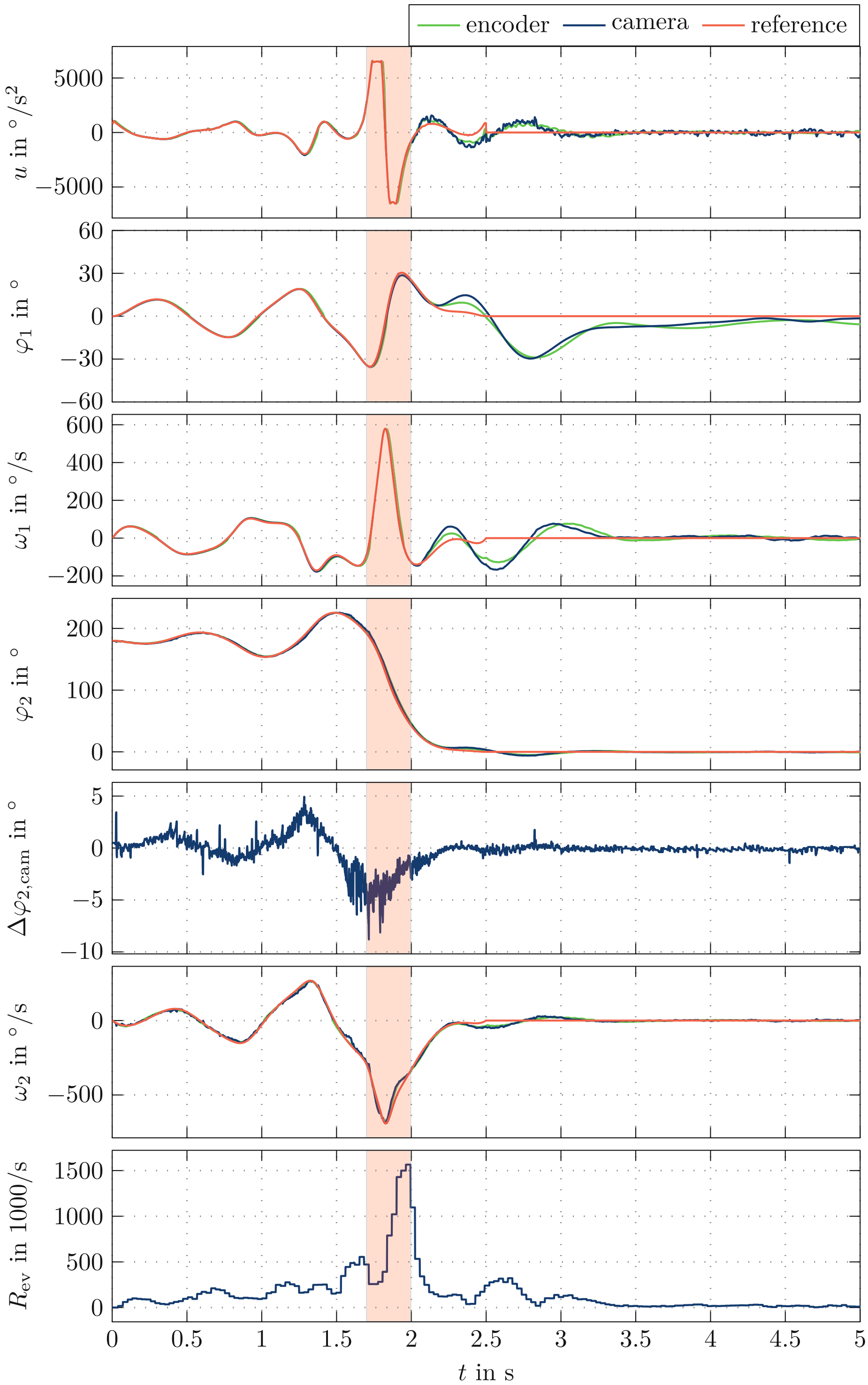

Figure 5.6: The experimental swing-up trajectory with encoder and event-based vision feedback. The reference values refer to u^*, φ_1^*, ω_1^*, φ_2^*, and ω_2^*, respectively. The state variables denote the estimated values of $\hat{\varphi}_1$, $\hat{\omega}_1$, $\hat{\varphi}_2$, and $\hat{\omega}_2$ based on encoder or camera feedback respectively. The event rate R_ev is calculated over a time window of 30 ms.

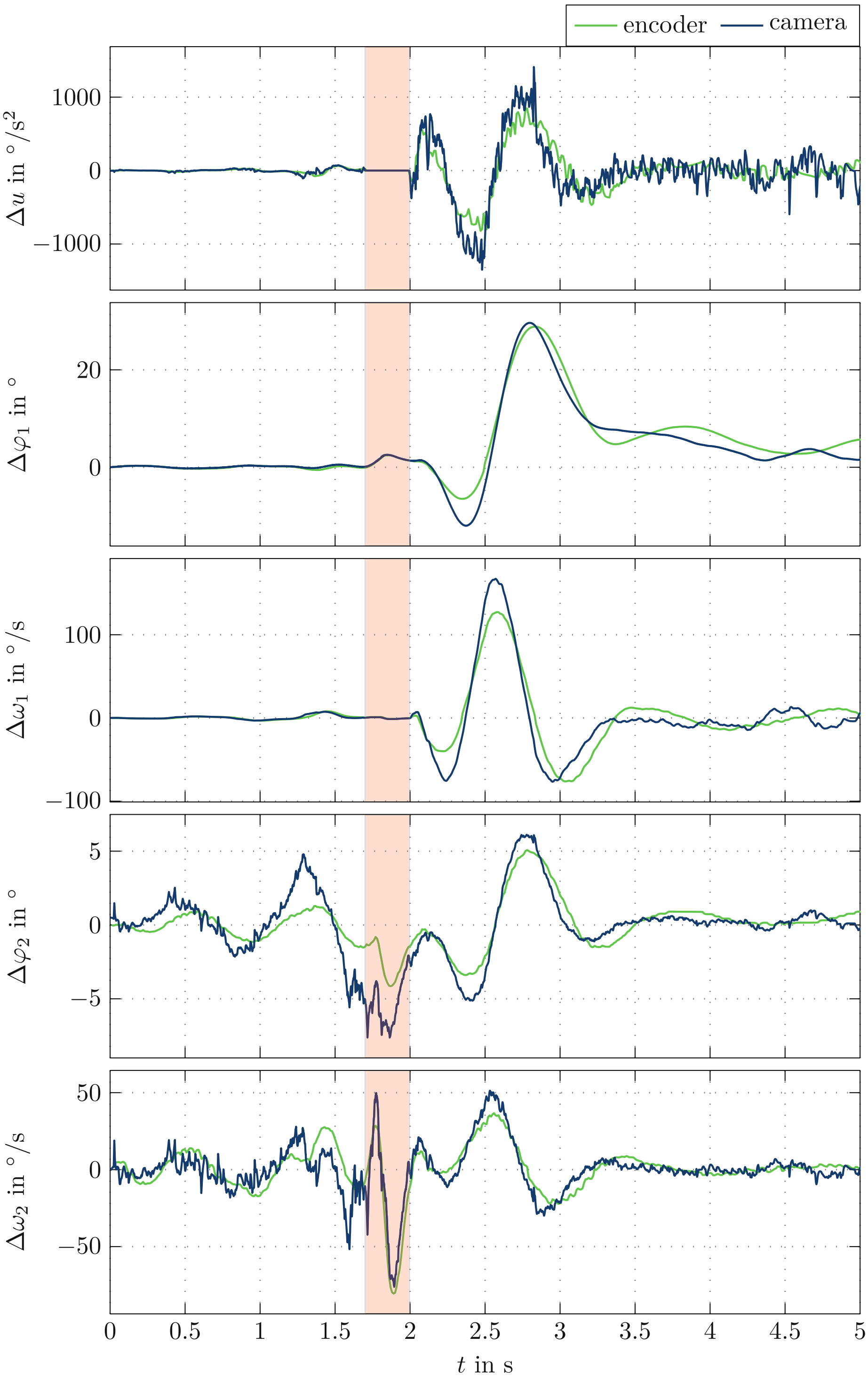

Figure 5.7: The error of the estimated system state with respect to the reference swing-up trajectory with encoder and event-based vision feedback.

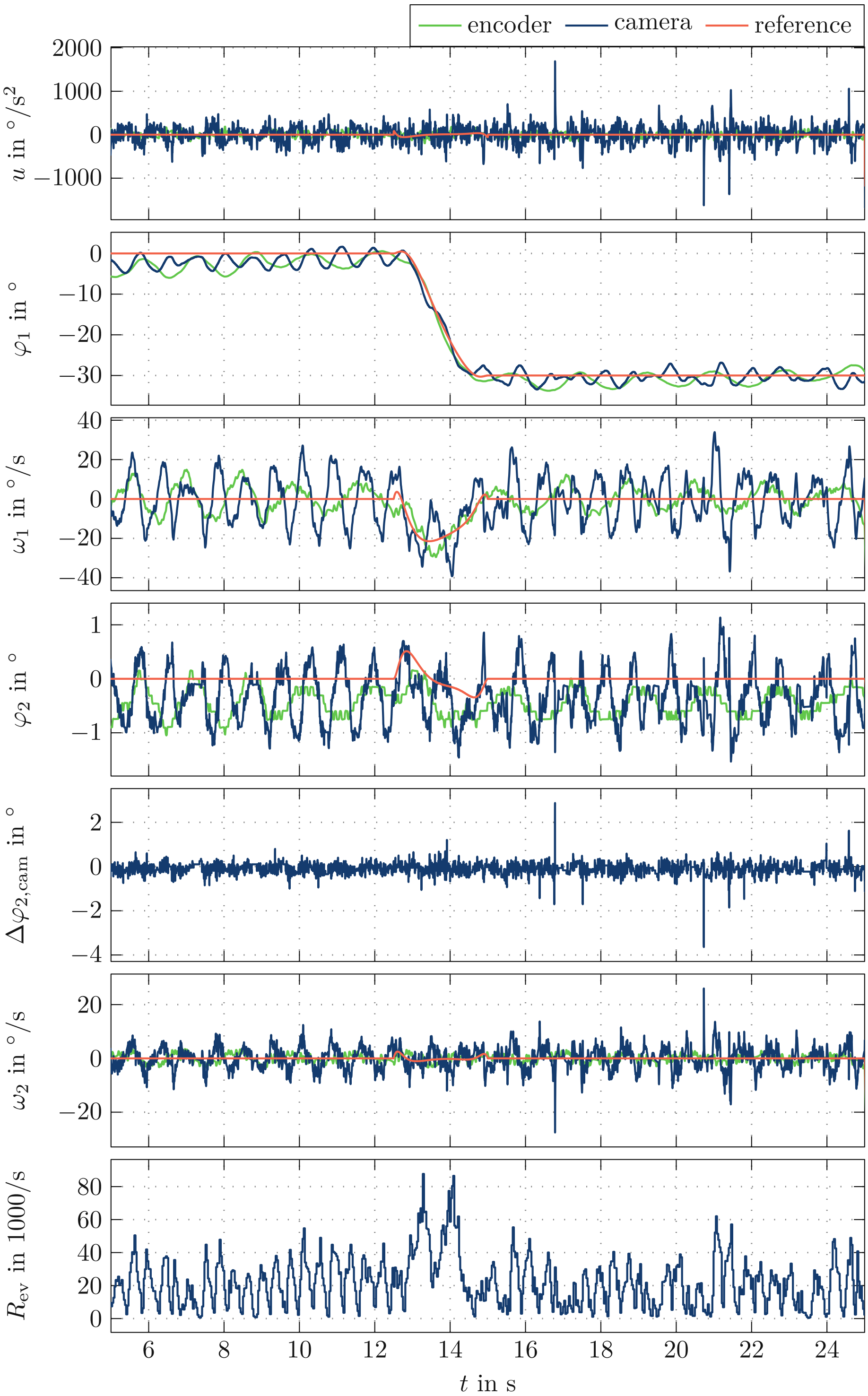

Figure 5.8: The experimental transfer trajectory with encoder and event-based vision feedback. The reference values refer to u^*, φ_1^*, ω_1^*, φ_2^*, and ω_2^*, respectively. The state variables denote the estimated values of $\hat{\varphi}_1$, $\hat{\omega}_1$, $\hat{\varphi}_2$, and $\hat{\omega}_2$ based on encoder or camera feedback respectively. The event rate R_{ev} is calculated over a time window of 30 ms.

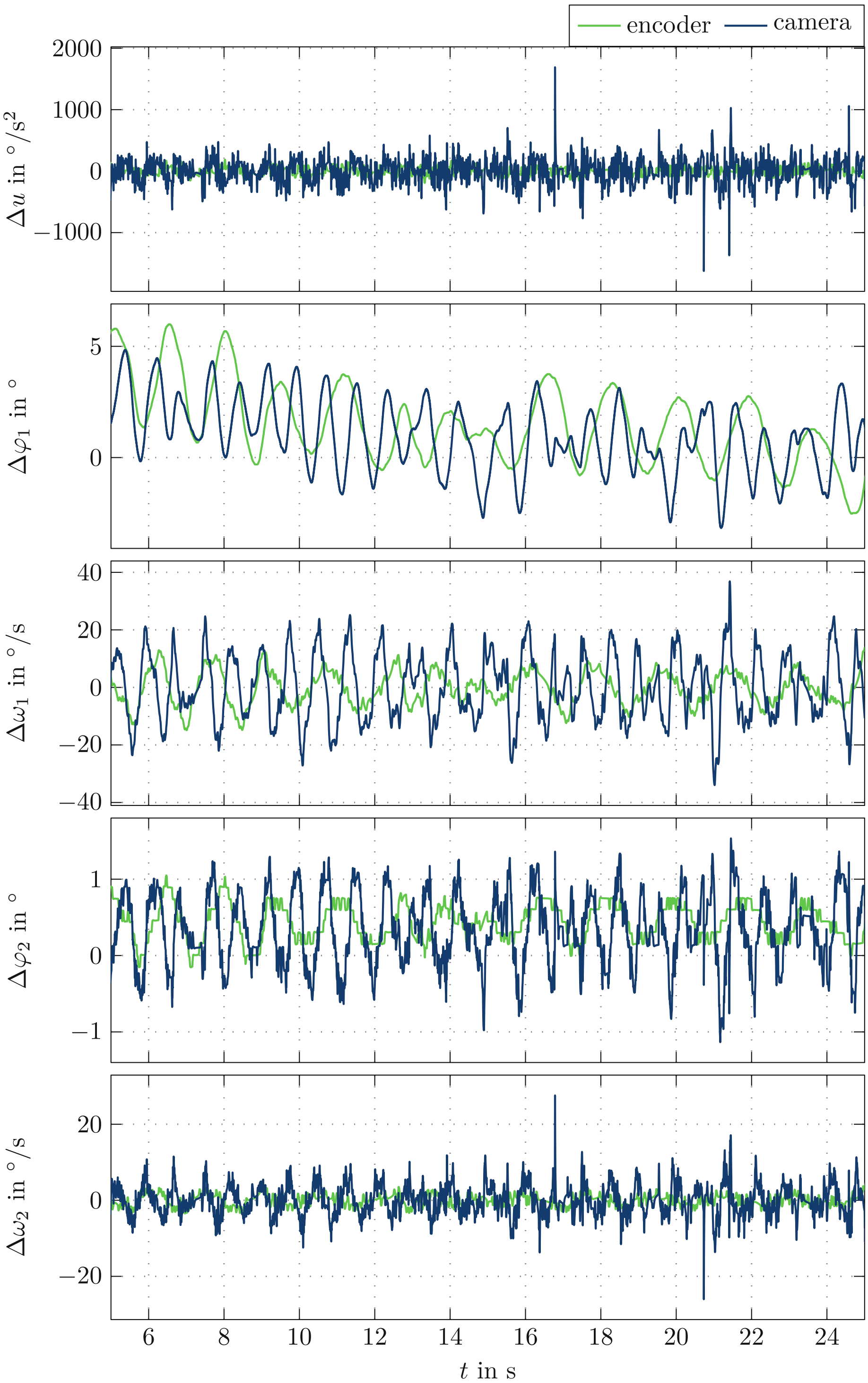

Figure 5.9: The error of the estimated system state with respect to the reference transfer trajectory with encoder and event-based vision feedback.

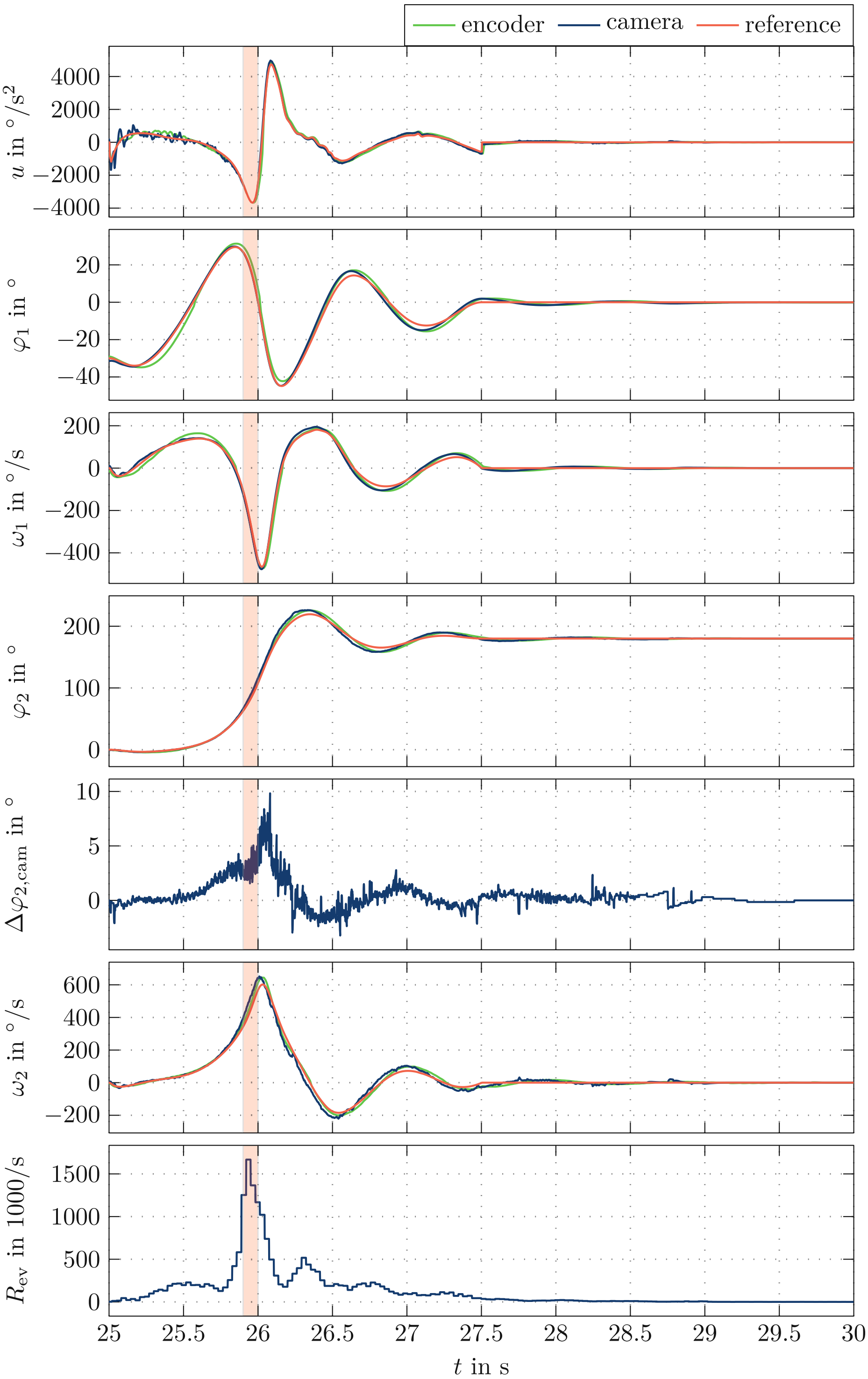

Figure 5.10: The experimental swing-down trajectory with encoder and event-based vision feedback. The reference values refer to u^*, φ_1^*, ω_1^*, φ_2^*, and ω_2^*, respectively. The state variables denote the estimated values of $\hat{\varphi}_1$, $\hat{\omega}_1$, $\hat{\varphi}_2$, and $\hat{\omega}_2$ based on encoder or camera feedback respectively. The event rate R_{ev} is calculated over a time window of 30 ms.

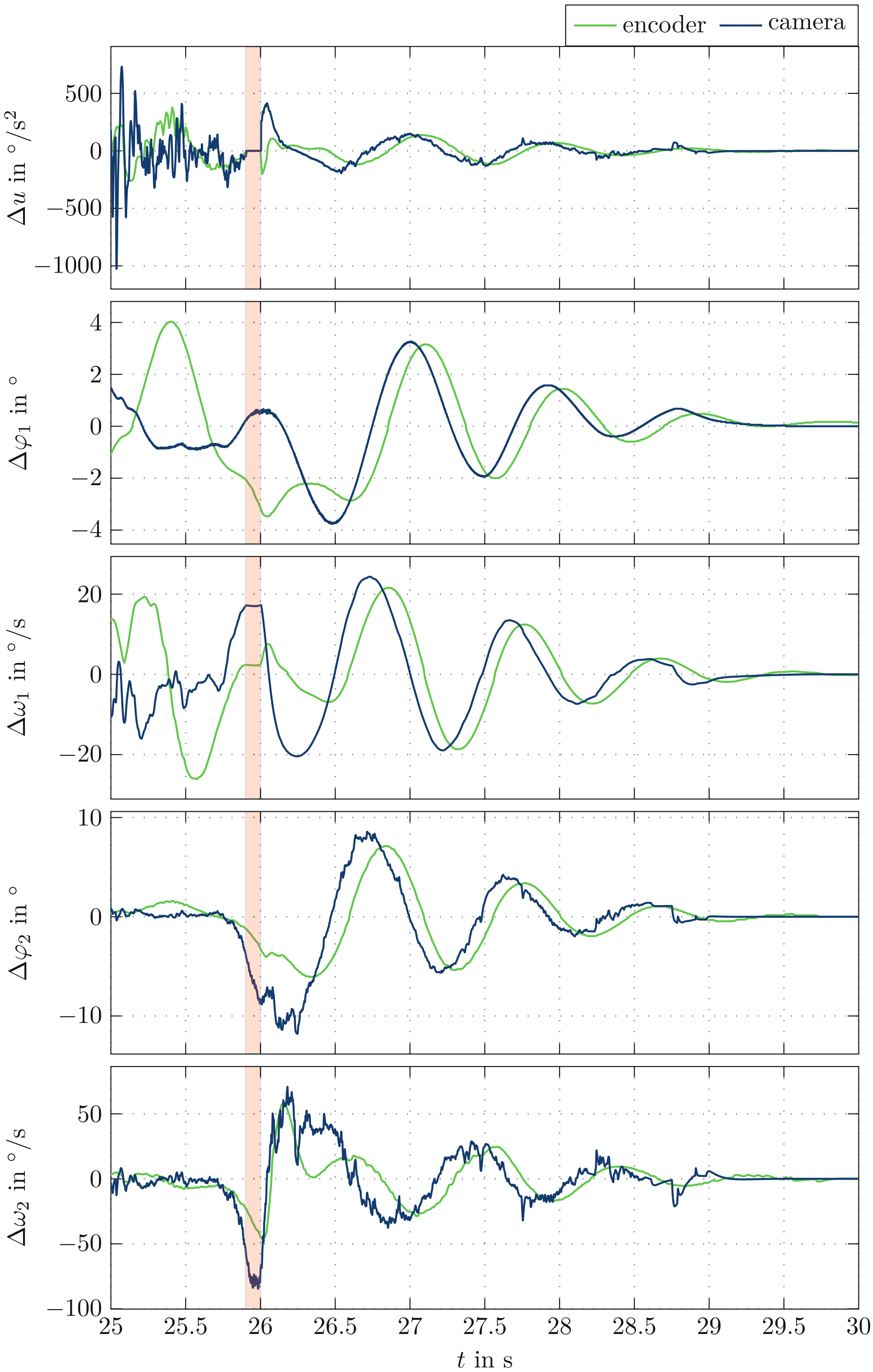

Figure 5.11: The error of the estimated system state with respect to the reference swing-down trajectory with encoder and event-based vision feedback.

5.2.2 Stabilization with an External Force

The conducted experiments on the physical *Furuta pendulum* for the stabilization about the reference trajectories did not consider an external force acting at the tip of the pendulum. Figure 5.12 depicts the experimental results, where an external force is manually applied at $t \approx 3\,\mathrm{s}$ at the tip of the pendulum while balancing the *Furuta pendulum* using event-based vision feedback about the upper equilibrium point. The *Extended Kalman Filter* estimated the force applied to be approximately $\hat{f}_{\mathrm{ext}} \approx 15\,\mathrm{mN}$. This external force leads to an arm deflection of about $-30°$ with a pendulum's angle of approximately $-4°$. As soon as the external force is released at $t \approx 4.5\,\mathrm{s}$, the *Furuta pendulum's* arm returns back to the reference value. However, prominent noise can be observed in the estimation of the external force. This may be explained by the quantization and the noise of the measurement of the pendulum's angle. Furthermore, unmodeled effects such as friction in the encoder bearings could contribute to estimation errors of the external force even in the absence of an external force.

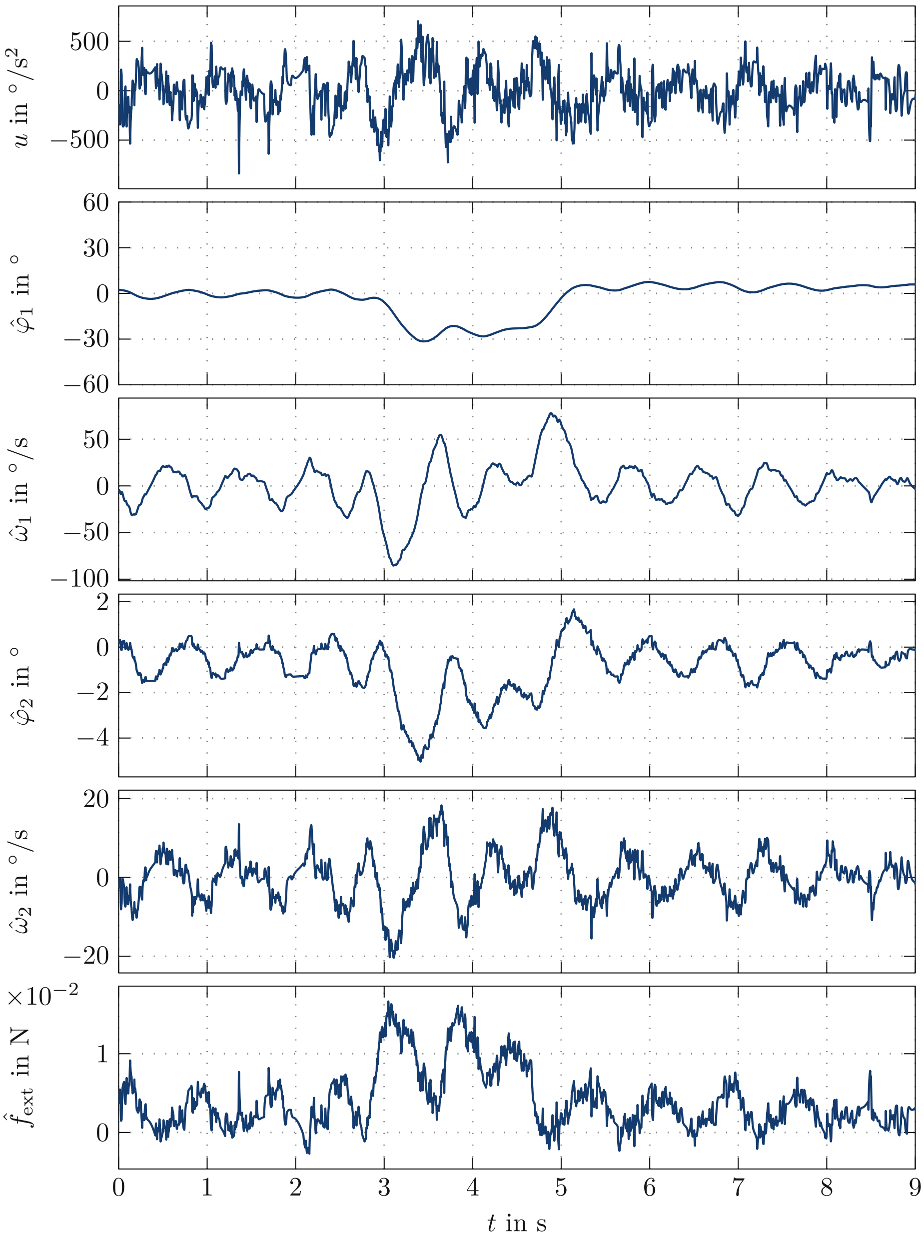

Figure 5.12: The experimental results when applying an external force manually at the tip of the pendulum while balancing the *Furuta pendulum* with event-based vision feedback.

6 Summary and Conclusions

This chapter summarizes this thesis and concludes with the findings on stabilizing the *Furuta pendulum* about the reference trajectories using event camera feedback.

A *Furuta pendulum* was built, considering its visual appearance for detecting and stabilizing the pendulum with event camera feedback. A mathematical model of the *Furuta pendulum* was derived in order to formulate the optimal control problem, estimate the state with an *Extended Kalman Filter*, and stabilize the *Furuta pendulum* with a time-varying *LQR*. A non-linear least squares parameter estimation was implemented to estimate the parameters that were not directly measurable.

Then, the event-based *Hough transform* with exponential decay, an adaption of the classical *Hough transform* algorithm for lines, was introduced to apply it to event packets from the event camera without the need for event frames. This event-processing algorithm was used to detect the pendulum's angle in dynamic scenarios, such as the swing-up. Next, the pinhole camera model considered the projection of the pendulum onto the event camera's sensor. An analytical function was derived to calculate the pendulum's angle from the output of the *Hough transform* algorithm and the arm's angle.

An optimal control problem was formulated and solved to obtain the reference swing-up, the transfer, and the swing-down trajectories. The respective box constraints took into account that the pendulum's arm should not deflect too much in order to limit the loss of resolution resulting from the camera's projection. The estimation of the state and the external force at the tip of the pendulum was done using an *Extended Kalman Filter*. In order to stabilize the *Furuta pendulum* about the reference trajectories, a time-varying *LQR* with an integrator was implemented. Further, the feedback controller was switched off based on the norm of the feedback gain vector and the loss of controllability when the pendulum is horizontal.

The proposed method was successfully applied in the simulation using MATLAB/SIMULINK, demonstrating its ability to swing-up, transfer, and swing-down the simulated *Furuta pendulum*. However, the quantization of the pendulum's angle measurements was clearly visible in the estimated state vector and in the estimation of the external force.

After the validation by simulation, experiments on the physical *Furuta pendulum* were conducted. The physical *Furuta pendulum* followed the reference trajectories while being controlled with the time-varying *LQR* and event camera feedback. The encoder was used to compare the accuracy of the vision-based measurement of the pendulum's angle and to perform a comparative analysis of the swing-up, the transfer, and the swing-down of the *Furuta pendulum*. During the swing-up and swing-down, the error between the camera measurement and the encoder measurement correlated with the pendulum's velocity. This indicates that the latency of the vision pipeline could be the reason for this behavior. Further, oscillations in the stabilization about the upper equilibrium position were observed. These oscillations may be affected by the quantization of the measured pendulum angle, as similar oscillations were observed in the simulation. It was found that the stabilization in simulation without quantization led to an absence of these prominent oscillations. The bias settings of the event camera were chosen to have a good trade-off between the sensitivity to the pendulum's movement and the noise while also reducing the event rate. The proposed event-processing algorithm detects the pendulum in dynamic scenarios, such as the swing-up with over $1.5 \cdot 10^6$ events per second or in the stabilization about the lower equilibrium point with approximately 200 events per second.

The stabilization with camera feedback was compared to the stabilization with encoder feedback. The analysis showed a comparative stabilization of the swing-up and swing-down trajectory, although the error of the state variables with event camera feedback tended to be noisier compared to the stabilization with encoder feedback.

The presence of an external force at the tip of the pendulum led to a deflection of the arm's angle while it was controlled. This deflection could have been reduced by penalizing the respective state in the time-varying *LQR* design, but the closed-loop system became unstable with event camera feedback at a particular increase of the respective design parameter. The proposed design of the time-varying *LQR* successfully controlled the *Furuta pendulum* while being sufficiently robust against the noise in the camera measurements.

The proposed method was able to stabilize the *Furuta pendulum* about the reference trajectories using event camera feedback with a maximum arm deflection of approximately 45° for the swing-down. The *Hough transform* with exponential decay detected the pendulum in dynamic scenarios with event rates ranging from 200 events per second up to over $1.5 \cdot 10^6$ events per second. It was found that the standard deviation between the encoder and the event camera measurement was approximately 0.77° during the stabilization about the reference trajectories with event camera feedback.

Future work could contribute by improving the introduced *Hough transform* with exponential decay. This could be achieved by implementing a non-uniform *Hough space* quantization, which could improve the latency of the vision processing algorithm. Further, a non-blocking implementation of the event processing algorithm could potentially reduce the latency and, thus, the error in the pendulum angle measurement.

Bibliography

[1] G. Gallego *et al.*, "Event-Based Vision: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, Jan. 2022. DOI: `10.1109/TPAMI.2020.3008413`.

[2] K. Furuta, M. Yamakita, S. Kobayashi, and M. Nishimura, "A New Inverted Pendulum Apparatus for Education," *IFAC Proceedings Volumes*, vol. 25, no. 12, pp. 133–138, 1991. DOI: `10.1016/S1474-6670(17)50102-0`.

[3] M. Yamakita, K. Nonaka, Y. Sugahara, and K. Furuta, "Robust State Transfer Control of Double Pendulum," *IFAC Proceedings Volumes*, vol. 27, no. 9, pp. 205–208, Aug. 1994. DOI: `10.1016/S1474-6670(17)45931-3`.

[4] M. Wiklund, A. Kristenson, and K. Aström, "A New Strategy for Swinging Up an Inverted Pendulum," *IFAC Proceedings Volumes*, vol. 26, no. 2, pp. 757–760, Jul. 1993. DOI: `10.1016/S1474-6670(17)48372-8`.

[5] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. Douglas, and T. Delbruck, "A pencil balancing robot using a pair of AER dynamic vision sensors," in *2009 IEEE International Symposium on Circuits and Systems*, Taipei, Taiwan: IEEE, May 2009, pp. 781–784. DOI: `10.1109/ISCAS.2009.5117867`.

[6] H. Wang, A. Chamroo, C. Vasseur, and V. Koncar, "Stabilization of a 2-DOF inverted pendulum by a low cost visual feedback," in *2008 American Control Conference*, Seattle, WA: IEEE, Jun. 2008, pp. 3851–3856. DOI: `10.1109/ACC.2008.4587094`.

[7] K. Hatada, M. Sato, K. Hirata, and Y. Masui, "Synthesis of a Calibration-Free Visual Feedback Controller for an Inverted Pendulum Using a Fisheye Lens," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 12, pp. 13 348–13 358, Dec. 2022. DOI: `10.1109/TIE.2021.3127034`.

[8] A. Brill, J. A. Frank, and V. Kapila, "Visual servoing of an inverted pendulum on cart using a mounted smartphone," in *2016 American Control Conference (ACC)*, Boston, MA, USA: IEEE, Jul. 2016, pp. 1323–1328. DOI: `10.1109/ACC.2016.7525101`.

[9] A. Kolker, A. Winkler, M. Bdiwi, and J. Suchy, "Robot visual servoing using the example of the inverted pendulum," in *10th International Multi-Conferences on Systems, Signals & Devices 2013 (SSD13)*, Hammamet, Tunisia: IEEE, Mar. 2013, pp. 1–6. DOI: `10.1109/SSD.2013.6564139`.

[10] M. Stuflesser and M. Brandner, "Vision-Based Control of an Inverted Pendulum using Cascaded Particle Filters," in *2008 IEEE Instrumentation and Measurement Technology Conference*, Victoria, BC, Canada: IEEE, May 2008, pp. 2097–2102. DOI: `10.1109/IMTC.2008.4547394`.

[11] J. Barrios-Avilés, T. Iakymchuk, J. Samaniego, L. Medus, and A. Rosado-Muñoz, "Movement Detection with Event-Based Cameras: Comparison with Frame-Based Cameras in Robot Object Tracking Using Powerlink Communication," *Electronics*, vol. 7, no. 11, p. 304, Nov. 2018. DOI: `10.3390/electronics7110304`.

[12] R. Muthusamy *et al.*, "Neuromorphic Eye-in-Hand Visual Servoing," *IEEE Access*, vol. 9, pp. 55 853–55 870, 2021. DOI: `10.1109/ACCESS.2021.3071261`.

[13] R. S. Dimitrova, M. Gehrig, D. Brescianini, and D. Scaramuzza, "Towards Low-Latency High-Bandwidth Control of Quadrotors using Event Cameras," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France: IEEE, May 2020, pp. 4294–4300. DOI: `10.1109/ICRA40945.2020.9197530`.

[14] A. Vitale, A. Renner, C. Nauer, D. Scaramuzza, and Y. Sandamirskaya, "Event-driven Vision and Control for UAVs on a Neuromorphic Chip," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China: IEEE, May 2021, pp. 103–109. DOI: `10.1109/ICRA48506.2021.9560881`.

[15] WANTAI MOTOR, *Stepper Motor 42BYGHW811 Datasheet*, 2011. [Online]. Available: `https://www.roboter-bausatz.de/media/pdf/44/21/70/WT42BYGHW811.pdf` (visited on 08/06/2024).

[16] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ: Wiley, 2006.

[17] F. C. Moon, *Applied Dynamics: With Applications to Multibody and Mechatronic Systems*. New York: Wiley, 1998.

[18] Waterloo Maple Inc., *Maple 2019.2*, Waterloo, Kanada, 2019.

[19] T. Glück, A. Eder, and A. Kugi, "Swing-up control of a triple pendulum on a cart with experimental validation," *Automatica*, vol. 49, no. 3, pp. 801–808, Mar. 2013. DOI: `10.1016/j.automatica.2012.12.006`.

[20] The MathWorks Inc., *MATLAB 9.9.0.2037887 (R2020b) Update 8*, Natick, Massachusetts, United States, 2020.

[21] P. V. C. Hough, "Method and means for recognizing complex patterns," US Patent US3069654A, Dec. 1962. [Online]. Available: `https://patents.google.com/patent/US3069654/en` (visited on 02/19/2024).

[22] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972. DOI: `10.1145/361237.361242`.

[23] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2. ed., 7. print. Cambridge, UK: Cambridge University Press, 2010.

[24] Prophesee, *Metavision Intelligence Docs 3.1.2 documentation*, Dec. 2022. [Online]. Available: `https://docs.prophesee.ai/3.1.2/index.html` (visited on 08/06/2024).

[25] CenturyArks Co., Ltd., *SilkyEvCam Event Based Camera Specification*, 2020. [Online]. Available: `https://www.centuryarks.com/images/product/sensor/silkyevcam/SilkyEvCam-USB_Spec_Rev102.pdf` (visited on 08/06/2024).

[26] R. L. Burden and J. D. Faires, *Numerical Analysis*, 7. ed. Pacific Grove, Calif: Brooks/Cole Thomson Learning, 2001.

[27] The MathWorks Inc., *Matlab Optimization Toolbox 9.0 (R2020b)*, Natick, Massachusetts, United States, 2020.

[28] The MathWorks Inc., *Matlab Documentation*, 2020. [Online]. Available: `https://de.mathworks.com/help/matlab/index.html?s_tid=hc_panel` (visited on 08/06/2024).

[29] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems* (Chapman & Hall/CRC Applied Mathematics and Nonlinear Science Series), 2nd ed. Boca Raton: Chapman & Hall/CRC, 2012.

[30] S. S. Haykin, Ed., *Kalman Filtering and Neural Networks*. New York: Wiley, 2001.

[31] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*, 3. ed. Menlo Park, Calif. [u.a.]: Addison-Wesley, 1998.

[32] Prophesee, *Metavision Intelligence 3.1.2 SDK*, 2022.

[33] Microsoft, *Microsoft Visual C++*, 2022.

[34] Beckhoff Automation GmbH & Co. KG, *TC1200 | TwinCAT 3 PLC*, 2022.

[35] Beckhoff Automation GmbH & Co. KG, *TF6311| TwinCAT 3 TCP/UDP Realtime*, 2022.

[36] Beckhoff Automation GmbH & Co. KG, *TF2000 | TwinCAT 3 HMI Server*, 2023.

[37] Bradski, G., *OpenCV*, Dec. 2021.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit gemäß dem Code of Conduct – Regeln zur Sicherung guter wissenschaftlicher Praxis (in der aktuellen Fassung des jeweiligen Mitteilungsblattes der TU Wien), insbesondere ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel, angefertigt wurde. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In– noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Wien, Oktober 2024

Lukas Pechhacker