TECHNISCHE UNIVERSITÄT MÜNCHEN

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY

Master's Thesis in Robotics, Cognition, Intelligence

Physically Consistent Identification of Inertial Parameters for Robotic Arms Using Gradient-based Optimization

Jesús Arturo Sol Navarro

TECHNISCHE UNIVERSITÄT MÜNCHEN

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY

Master's Thesis in Robotics, Cognition, Intelligence

Physically Consistent Identification of Inertial Parameters for Robotic Arms Using Gradient-based Optimization

Physikalisch konsistente Identifikation von Trägheitsparametern für Roboterarme durch gradientenbasierte Optimierung

Author:	Jesús Arturo Sol Navarro
Examiner:	Prof. Dr.-Ing. Alin Albu-Schäffer
Supervisor:	Dr.-Ing. Johannes Englsberger
Submission Date:	06.07.2026

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, 06.07.2026 Jesús Arturo Sol Navarro

Acknowledgments

First and foremost, I want to thank my parents, Arturo and Laura. Even from the other side of the planet, they have always found a way to support me, encourage me, and remind me that I am not alone. Their love and trust have meant more to me than I can put into words.

I would also like to sincerely thank my thesis supervisor, Johannes Englsberger, for his guidance throughout this project. I am grateful for the opportunity to work on such an interesting and challenging topic.

Finally, I want to thank Lena for being by my side through all the highs and lows of this journey. Your encouragement and support made this path much easier to walk.

Abstract

This thesis presents a physically consistent inertial parameter identification method for robotic arms. The approach introduces an alternative parameterization that guarantees positive masses and density-realizable inertia matrices by construction, instead of enforcing physical consistency through constraints. A reformulation of the inverse dynamics model is used to derive an analytical gradient with respect to the reparameterized inertial parameters, enabling gradient-based optimization. The Franka Research 3 robot is used as the evaluation platform in both simulation and real-robot experiments. In both cases, the experiments are repeated over multiple Monte Carlo trials to evaluate how the proposed method performs under randomized conditions. The simulation experiments analyze convergence behavior, robustness, the influence of the training trajectory and torque-noise sensitivity. The real-robot experiments evaluate whether the proposed algorithm can fine-tune CAD inertial parameters and adapt to changes in the robot dynamics caused by an unknown payload. A comparison with a physically consistent identification method is also presented.

Contents

List of Figures

List of Tables

1. Introduction

Robotic manipulators are increasingly used in applications that require accurate motion and reliable interaction with the environment, especially when operating at high speeds, handling payloads, or working in collaborative settings with humans. In such tasks, the quality of the dynamic model plays an important role. Model-based control, trajectory optimization, collision detection, and simulation all rely on an accurate description of the robot dynamics [1, 2, 3]. However, the inertial parameters provided by manufacturers or extracted from CAD models are often inaccurate [4]. This is especially relevant when the robot includes unmodeled components or effects, such as cables, joint friction, varying payloads, or grippers.

For this reason, inertial parameter identification has become an important topic in robotics. The goal is to estimate the parameters of the robot model directly from measured data, such as joint positions, velocities, accelerations, and torques. In other words, the objective is to determine the parameter set that best explains the measured robot motion and joint torques. Atkeson et al. [5] showed that the inverse dynamics model can be written in a form that is linear with respect to the inertial parameters. This property forms the basis of many classical identification methods, which rewrite the robot dynamics as a linear regression problem. This leads to least-squares formulations that are simple and computationally efficient. Nevertheless, these methods have important limitations. The estimated parameters are not always physically meaningful, since an unconstrained least-squares solution may produce negative masses or inertia tensors that do not correspond to any real rigid body [6]. Such parameters may reproduce the measured torques for a specific trajectory, but they can lead to unstable behavior when used for robot control [7].

This thesis focuses on physically consistent inertial parameter identification for robotic manipulators. The main idea is to avoid physically invalid solutions by changing the parameterization of the inertial parameters. Instead of directly optimizing the standard inertial parameters, an alternative parameter vector is introduced. This parameterization guarantees positive mass and density-realizable inertia matrices by construction, instead of enforcing physical consistency through constraints. As a result, every parameter set identified during the optimization corresponds to a physically valid rigid body.

The proposed method is formulated as a gradient-based optimization problem. The inverse dynamics model is reformulated to make its dependence on the link inertia matrices explicit. Based on this reformulation, an analytical gradient of the torque-prediction loss with respect to the new parameter vector is derived. The method is implemented and evaluated using the Franka Research 3 robot, both in simulation and with real-robot data. Further details on the robot model are provided in Appendix B. First, simulation experiments are conducted to investigate the potential and limitations of the proposed method under controlled conditions. Next, the method is evaluated using data collected from the real robot.

1.1. Problem Statement

The objective of this thesis is to develop and evaluate a physically consistent inertial parameter identification method for robotic manipulators. Given measurements of joint positions, velocities, accelerations, and torques, the goal is to estimate a set of inertial parameters that accurately reproduces the robot dynamics while remaining physically consistent.

The problem can be summarized by the following requirements. First, the identified parameters should lead to low torque-prediction error on trajectories that were not used during training. Second, the estimated parameters should satisfy physical consistency conditions, including positive mass and density-realizable inertia matrices. Third, the method should be robust with respect to different initial parameter guesses and measurement disturbances. Finally, the approach should be computationally practical enough to be used in simulation studies and real-robot identification experiments.

1.2. Contributions and Overview of the Presented Research

The main contributions of this thesis are the following:

- A physically consistent parameterization of the inertial parameters is introduced. The mass is parameterized using an exponential function, the center of mass is optimized directly, the principal-axis orientation is represented through the exponential map on $SO(3)$, and the principal moments of inertia are parameterized such that the triangle inequalities are satisfied by construction.
- The inverse dynamics model is reformulated to make the dependence on the link inertia matrices explicit. This formulation provides the basis for gradient-based inertial parameter identification.
- The proposed identification method is evaluated in simulation through Monte Carlo experiments considering different initialization perturbations, training trajectories, and torque-noise levels. Its performance is assessed using torque-prediction error, physical consistency, the Frobenius norm, the affine-invariant Riemannian metric, convergence behavior, computation time, and a comparison against PC-IDIM-OLS.
- The proposed method is validated on real Franka Research 3 robot data by tuning the CAD inertial parameters and evaluating torque-prediction accuracy, physical consistency, generalization to unseen trajectories, computation time, and performance against PC-IDIM-OLS.

The remainder of this thesis is organized as follows. Chapter 2 introduces the mathematical tools required for the formulation, including notation, rigid-body kinematics, inverse dynamics, and matrix distance measures. Chapter 3 reviews classical inertial parameter identification methods, base parameters, physical consistency, and excitation trajectory design. Chapter 4

presents the proposed gradient-based identification method, including the lambda parameterization, the mapping between standard and physically consistent parameters, the analytical gradient, and the implementation pipeline. Chapter 5 evaluates the method in simulation, compares it with existing approaches, and presents results obtained from real-robot data. Finally, Chapter 6 summarizes the main findings and discusses possible directions for future work.

2. Mathematical Background for Robotics

This chapter introduces the mathematical background required for the formulations used throughout this thesis. It defines the notation, operators, rigid-body kinematics, robot dynamics, and matrix distance measures needed for the following chapters.

2.1. Notation and Operators

2.1.1. Vectors and Matrices

A vector $\boldsymbol{v} \in \mathbb{R}^n$ is used to represent quantities with multiple components. In Euclidean space, vectors commonly represent positions or displacements, but they can also represent other physical quantities. In this work, vectors are denoted by bold lowercase letters. The vector from reference point a to point b, expressed in reference frame i, is denoted by

$$^{i}\boldsymbol{v}_{a,b}. \tag{2.1}$$

Matrices with m rows and n columns are denoted by bold uppercase letters, such as

$$\boldsymbol{A} \in \mathbb{R}^{m \times n}. \tag{2.2}$$

2.1.2. Skew Operator

The skew operator maps a vector $\boldsymbol{v} = [v_1,\ v_2,\ v_3]^\top \in \mathbb{R}^3$ to the skew-symmetric matrix

$$[\boldsymbol{v}\times] = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}. \tag{2.3}$$

A skew-symmetric matrix is a square matrix whose transpose equals its negative. This representation is particularly handy for expressing cross products in matrix form:

$$\boldsymbol{v} \times \boldsymbol{w} = [\boldsymbol{v}\times]\boldsymbol{w}. \tag{2.4}$$

2.1.3. Double-Tilde Operator

For a vector $\boldsymbol{v} \in \mathbb{R}^3$, the double tilde operator is defined as

$$\widetilde{\widetilde{\boldsymbol{v}}} = \begin{bmatrix} v_1 & 0 & 0 & v_2 & v_3 & 0 \\ 0 & v_2 & 0 & v_1 & 0 & v_3 \\ 0 & 0 & v_3 & 0 & v_1 & v_2 \end{bmatrix}. \tag{2.5}$$

This operator will be useful later in the construction of the regressor matrices.

2.2. Exponential and Logarithmic Maps on SO(3)

The exponential map provides a mapping from the Lie algebra $\mathfrak{so}(3)$ to the Lie group SO(3). The elements of SO(3) are rotation matrices, which are introduced in Sec. 2.3.1. The exponential map is defined as:

$$\begin{aligned} \exp : \mathfrak{so}(3) &\to \mathrm{SO}(3) \\ \boldsymbol{\omega} &\mapsto \mathbf{R}_{3\times 3} \end{aligned} \tag{2.6}$$

This mapping admits a closed-form expression given by Rodrigues' formula [8]:

$$\exp(\boldsymbol{\omega}) = \mathbf{R} = \mathbf{I} + \frac{\sin\theta}{\theta}[\boldsymbol{\omega}\times] + \frac{1-\cos\theta}{\theta^2}[\boldsymbol{\omega}\times]^2. \tag{2.7}$$

Here, $\boldsymbol{\omega}$ denotes the rotation vector expressed through its three components, while θ represents its magnitude:

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}, \qquad \theta = \|\boldsymbol{\omega}\|. \tag{2.8}$$

For $\theta = 0$, the exponential map yields the identity rotation.

The inverse of the exponential map is referred to as the logarithmic map. It maps a rotation matrix in SO(3) to its corresponding element in $\mathfrak{so}(3)$:

$$\begin{aligned} \log : \mathrm{SO}(3) &\to \mathfrak{so}(3) \\ \mathbf{R}_{3\times 3} &\mapsto \boldsymbol{\omega}. \end{aligned} \tag{2.9}$$

To compute the logarithmic map, the rotation angle is first obtained from the trace of the rotation matrix. The cosine of the angle is given by

$$\cos\theta = \frac{1}{2}\left(\mathrm{tr}(\mathbf{R}) - 1\right) \tag{2.10}$$

The sine of the angle can then be written as

$$\sin\theta = \left(1-\cos^2\theta\right)^{1/2} = \frac{1}{2}\sqrt{\left(3-\mathrm{tr}(\mathbf{R})\right)\left(1+\mathrm{tr}(\mathbf{R})\right)} \tag{2.11}$$

Finally, the rotation vector is recovered as

$$\log(\mathbf{R}) = \boldsymbol{\omega} = \frac{\theta}{2\sin\theta}\begin{bmatrix} R_{32}-R_{23} \\ R_{13}-R_{31} \\ R_{21}-R_{12} \end{bmatrix} \tag{2.12}$$

According to [9], the exponential map is well-defined and surjective, meaning that every rotation matrix in SO(3) can be represented by at least one rotation vector in $\mathfrak{so}(3)$. However, the exponential map is not globally one-to-one, since different rotation vectors may represent the same rotation. By restricting the rotation angle to the principal interval $0 \leq \theta < \pi$, the logarithmic map provides a locally unique inverse of the exponential map. This local correspondence between $\mathfrak{so}(3)$ and SO(3) will be used in the subsequent chapters of this thesis.

2.3. Robot Kinematics

Robot kinematics studies the motion of rigid bodies without considering the forces that produce it. This section introduces only the kinematic concepts required in this work, including rigid-body transformations, body velocities, adjoint transformations, and Jacobians.

2.3.1. Rotation Matrices and Homogeneous Transformations

The rotation matrix ${}^{j}\boldsymbol{R}_{i} \in \mathbb{R}^{3\times3}$ describes the orientation of frame i with respect to frame j. It transforms the coordinates of a point expressed in frame i into coordinates expressed in frame j:

$$ {}^{j}\boldsymbol{v}_{a,b} = {}^{j}\boldsymbol{R}_{i}\,{}^{i}\boldsymbol{v}_{a,b}. \tag{2.13} $$

Valid three-dimensional rotation matrices always preserve Euclidean distance between points. They belong to the special orthogonal group $SO(3)$ because they are orthogonal and have determinant equal to one. Orthogonality implies that the transpose of a rotation matrix is equal to its inverse:

$$ \left({}^{j}\boldsymbol{R}_{i}\right)^{-1} = \left({}^{j}\boldsymbol{R}_{i}\right)^{T} = {}^{i}\boldsymbol{R}_{j}. \tag{2.14} $$

A homogeneous transformation $\boldsymbol{H} \in \mathbb{R}^{4\times4}$ is important in robotics because it provides a compact way to combine rotation and translation, allowing poses and coordinate transformations between reference frames to be described using a single matrix. In contrast, rotation matrices describe only the relative orientation between reference frames. It transforms the coordinates of a point expressed in frame i into coordinates expressed in frame j, with the position vector written in homogeneous coordinates:

$$ \begin{bmatrix} {}^{j}\boldsymbol{v}_{a,b} \\ 1 \end{bmatrix} = {}^{j}\boldsymbol{H}_{i} \begin{bmatrix} {}^{i}\boldsymbol{v}_{a,b} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{j}\boldsymbol{R}_{i} & {}^{j}\boldsymbol{v}_{j,i} \\ \boldsymbol{0}_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} {}^{i}\boldsymbol{v}_{a,b} \\ 1 \end{bmatrix}. \tag{2.15} $$

The inverse of a homogeneous transformation can be computed using the transpose of the rotation matrix and the corresponding transformed translation vector:

$$ \left({}^{j}\boldsymbol{H}_{i}\right)^{-1} = \begin{bmatrix} \left({}^{j}\boldsymbol{R}_{i}\right)^{T} & -\left({}^{j}\boldsymbol{R}_{i}\right)^{T}{}^{j}\boldsymbol{v}_{j,i} \\ \boldsymbol{0}_{1\times3} & 1 \end{bmatrix} = {}^{i}\boldsymbol{H}_{j}. \tag{2.16} $$

2.3.2. Velocities

In the context of robotic manipulators, each link of the robot is modeled as a rigid body. The vector ${}^{k}\boldsymbol{\nu}_{0,k} \in \mathbb{R}^{6}$ represents the velocity of a rigid body k relative to the world frame expressed in frame k. It combines its linear and angular velocity components:

$$ {}^{k}\boldsymbol{\nu}_{0,k} = \begin{bmatrix} {}^{k}\dot{\boldsymbol{x}}_{0,k} \\ {}^{k}\boldsymbol{\omega}_{0,k} \end{bmatrix} \tag{2.17} $$

Here, $\dot{\boldsymbol{x}}$ denotes the linear velocity component, while $\boldsymbol{\omega}$ denotes its angular velocity component.

2.3.3. Jacobian Matrix

In robotics, different types of Jacobians can be defined depending on the frame in which the velocity is expressed. In this thesis, only the body Jacobian is considered.

The body Jacobian ${}^{k}\boldsymbol{J}_{0,k}(\boldsymbol{q})$ maps the generalized velocities $\dot{\boldsymbol{q}}$ to the body velocity of link k relative to the base frame 0, expressed in the body-fixed frame k:

$$ {}^{k}\boldsymbol{\nu}_{0,k} = {}^{k}\boldsymbol{J}_{0,k}(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{2.18} $$

2.3.4. Adjoint Transformations

The adjoint transformation $\boldsymbol{Ad}_{{}^{j}\boldsymbol{H}_{i}} \in \mathbb{R}^{6\times 6}$ maps velocity vectors from frame i to frame j according to

$$ {}^{j}\boldsymbol{\nu}_{k} = \boldsymbol{Ad}_{{}^{j}\boldsymbol{H}_{i}}\,{}^{i}\boldsymbol{\nu}_{k}. \tag{2.19} $$

Using the rotation matrix and translation vector contained in the homogeneous transformation ${}^{j}\boldsymbol{H}_{i}$, the adjoint matrix is given by

$$ \boldsymbol{Ad}_{{}^{j}\boldsymbol{H}_{i}} = \begin{bmatrix} {}^{j}\boldsymbol{R}_{i} & [{}^{j}\boldsymbol{v}_{j,i}\times]\,{}^{j}\boldsymbol{R}_{i} \\ \boldsymbol{0}_{3\times 3} & {}^{j}\boldsymbol{R}_{i} \end{bmatrix}. \tag{2.20} $$

The corresponding inverse adjoint maps velocity vectors in the opposite direction, from frame j back to frame i, and is written as

$$ \boldsymbol{Ad}^{-1}_{{}^{j}\boldsymbol{H}_{i}} = \boldsymbol{Ad}_{{}^{i}\boldsymbol{H}_{j}} = \begin{bmatrix} \left({}^{j}\boldsymbol{R}_{i}\right)^{T} & -\left({}^{j}\boldsymbol{R}_{i}\right)^{T}[{}^{j}\boldsymbol{v}_{j,i}\times] \\ \boldsymbol{0}_{3\times 3} & \left({}^{j}\boldsymbol{R}_{i}\right)^{T} \end{bmatrix}. \tag{2.21} $$

Since wrenches, which are introduced later in Sec. 2.4.3, are dual to velocity vectors, they are transformed using the inverse transpose of the adjoint transformation:

$$ {}^{j}\boldsymbol{w}_{k} = \boldsymbol{Ad}^{-T}_{{}^{j}\boldsymbol{H}_{i}}\,{}^{i}\boldsymbol{w}_{k}. \tag{2.22} $$

2.3.5. Lie Bracket Matrix

The Lie bracket matrix, denoted by $\mathbf{adj}(\boldsymbol{\nu})$, represents the action of a velocity vector $\boldsymbol{\nu}$ on another element of the Lie algebra through the Lie bracket. The Lie bracket matrix is given by

$$ \mathbf{adj}(\boldsymbol{\nu}) = \begin{bmatrix} [\boldsymbol{\omega}\times] & [\dot{\boldsymbol{x}}\times] \\ \boldsymbol{0}_{3\times 3} & [\boldsymbol{\omega}\times] \end{bmatrix}. \tag{2.23} $$

An important property used in the derivations presented in the following chapters is that the Lie bracket of a velocity vector with itself is zero. Applying the Lie bracket matrix to the same velocity vector yields

$$ \mathbf{adj}(\boldsymbol{\nu})\,\boldsymbol{\nu} = \begin{bmatrix} [\boldsymbol{\omega}\times]\dot{\boldsymbol{x}} + [\dot{\boldsymbol{x}}\times]\boldsymbol{\omega} \\ [\boldsymbol{\omega}\times]\boldsymbol{\omega} \end{bmatrix} = \boldsymbol{0}_{6\times 1}. \tag{2.24} $$

The upper component vanishes because the cross product is antisymmetric, while the lower component vanishes because the cross product of a vector with itself is zero.

2.4. Robot Dynamics

Robot dynamics relates the motion of a robotic system to the forces and torques that produce it. This section introduces the inertial description of rigid bodies and the inverse dynamics model used to compute the generalized torques required for a given motion.

2.4.1. Inertia of a Rigid Body

The mass distribution of a rigid body directly determines its motion in response to external forces. For link k, this distribution is described by its mass, the position of its center of mass (CoM), and its moment of inertia. The mass m_k describes the resistance of body k to linear acceleration. The center of mass defines the average location of the body's mass. The CoM position of body k, expressed in frame k, is given by

$$ {}^{k}\boldsymbol{f}_{k,\mathrm{CoM}} = \begin{bmatrix} x & y & z \end{bmatrix}^{T} \tag{2.25} $$

While the mass describes the resistance to translational motion, the moment of inertia describes the resistance to rotational motion. The inertia tensor of body k, expressed at its CoM, is given by

$$ \boldsymbol{I}_{k_{CoM}} = \begin{bmatrix} XX_k & XY_k & XZ_k \\ XY_k & YY_k & YZ_k \\ XZ_k & YZ_k & ZZ_k \end{bmatrix}. \tag{2.26} $$

The diagonal elements XX_k, YY_k, and ZZ_k describe the moments of inertia about the x-, y-, and z-axes of frame k, respectively. They quantify how strongly the body resists angular acceleration about each axis. The off-diagonal elements XY_k, XZ_k, and YZ_k are the products of inertia, which describe the coupling between rotations about different axes. In robotics, it is often more common to express the moment of inertia with respect to the origin of the link-fixed frame, rather than at the CoM. Therefore, the moment of inertia $\boldsymbol{I}_{k_{CoM}}$ can be shifted from the CoM to the origin of frame k using Steiner's theorem (or parallel-axis theorem):

$$ \boldsymbol{I}_k = \boldsymbol{I}_{k_{CoM}} - m_k[{}^{k}\boldsymbol{f}_{k,\mathrm{CoM}}\times][{}^{k}\boldsymbol{f}_{k,\mathrm{CoM}}\times] \tag{2.27} $$

The mass, CoM position, and moment of inertia can then be combined into a 6×6 inertia matrix :

$$ {}^{k}\boldsymbol{M}_k = \begin{bmatrix} m_k\,\boldsymbol{I}_{3\times 3} & -\left[\, m_k\, {}^{k}\boldsymbol{f}_{k,\mathrm{CoM}}\times \right]^{T} \\ \left[\, m_k\, {}^{k}\boldsymbol{f}_{k,\mathrm{CoM}}\times \right] & \boldsymbol{I}_k \end{bmatrix} \tag{2.28} $$

This representation shows that the inertial behavior of a rigid body can be described by a compact set of physical quantities, commonly referred to in the literature as inertial parameters. For multibody systems, such as robotic manipulators, the inertia matrix serves as a fundamental building block of the dynamic model. Once defined for each link, these matrices allow the robotic equations of motion to be derived systematically.

2.4.2. Inertial-Parameter Coefficient Matrix

The matrix $\boldsymbol{A} \in \mathbb{R}^{6\times 10}$, presented in [10], is used to express the momentum ${}^k\boldsymbol{M}_k{}^k\boldsymbol{\nu}_k$ of body k in a form that is linear in the inertial parameters. The matrix $\boldsymbol{A}(\boldsymbol{x})$ collects the coefficients multiplying the inertial parameter vector $\boldsymbol{\pi}_k$:

$$ {}^k\boldsymbol{M}_k{}^k\boldsymbol{\nu}_k = \boldsymbol{A}({}^k\boldsymbol{\nu}_k)\boldsymbol{\pi}_k. \tag{2.29} $$

It is defined as:

$$ \boldsymbol{A}(\boldsymbol{\nu}_k) = \begin{bmatrix} {}^k\dot{\boldsymbol{x}}_k & [{}^k\boldsymbol{\omega}_k\times] & \boldsymbol{0}_{3\times 6} \\ \boldsymbol{0}_{3\times 1} & -[{}^k\dot{\boldsymbol{x}}_k\times] & {}^k\widetilde{\widetilde{\boldsymbol{\omega}}}_k \end{bmatrix} \tag{2.30} $$

This operator will be useful later in the construction of the regressor matrices.

2.4.3. Wrenches

A wrench ${}^k\boldsymbol{\omega}_k \in \mathbb{R}^6$ is a vector that represents the forces and moments acting on the rigid body k expressed in frame k. It combines the linear force component and the torque component into a single vector:

$$ {}^k\boldsymbol{\omega}_k = \begin{bmatrix} {}^k\boldsymbol{f}_k \\ {}^k\boldsymbol{\tau}_k \end{bmatrix}. \tag{2.31} $$

Here, $\boldsymbol{f}$ denotes the force applied to the link, while $\boldsymbol{\tau}$ denotes the torque. A wrench can be interpreted as the force counterpart of a velocity. While a velocity vector describes the instantaneous motion of a rigid body, a wrench describes the generalized force acting on it. A wrench can be mapped into generalized joint torques through the transpose of the corresponding Jacobian:

$$ \boldsymbol{\tau} = {}^k\boldsymbol{J}_{0,k}^T{}^k\boldsymbol{\omega}_k \tag{2.32} $$

2.4.4. Inverse Dynamics Model

For a robotic manipulator with n generalized coordinates, the inverse dynamics model computes the generalized torques $\boldsymbol{\tau} \in \mathbb{R}^n$ required to produce a motion described by the generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$, velocities $\dot{\boldsymbol{q}} \in \mathbb{R}^n$, and accelerations $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$:

$$ \boldsymbol{\tau} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{2.33} $$

The generalized torques are the torques associated with the generalized coordinates and therefore act along the unconstrained degrees of freedom of the system. The derivation of the inverse dynamic model is not repeated in this work. Instead, the formulation presented in [11] is adopted for a fixed-base robotic manipulator. This formulation is based on Newton's second law, which relates the derivative of the generalized momentum to the generalized force. The model can be understood as the result of summing the dynamic contributions of all links $k \in \{1, \ldots, n_{\text{links}}\}$:

$$\underbrace{\sum_k \left({}^k\mathbf{J}_{0,k}^T\, {}^k\mathbf{M}_k\, {}^k\mathbf{J}_{0,k}\right)}_{\mathbf{M}} \ddot{\mathbf{q}} + \underbrace{\sum_k \left({}^k\mathbf{J}_{0,k}^T\, {}^k\mathbf{C}_k\, {}^k\mathbf{J}_{0,k} + {}^k\mathbf{J}_{0,k}^T\, {}^k\mathbf{M}_k\, {}^k\dot{\mathbf{J}}_{0,k}\right)}_{\mathbf{C}} \dot{\mathbf{q}} -$$
$$\underbrace{-\sum_k \left({}^k\mathbf{J}_{0,k}^T\, {}^k\boldsymbol{\omega}_{k,\text{grav}}\right)}_{\boldsymbol{\tau}_{\text{grav}}} = \underbrace{\sum_k \left({}^k_b\mathbf{J}_{0,k}^T\, {}^k\boldsymbol{\omega}_{k,nc}\right)}_{\boldsymbol{\tau}} \tag{2.34}$$

where $\mathbf{M} \in \mathbb{R}^{n\times n}$ denotes the generalized mass matrix, $\mathbf{C} \in \mathbb{R}^{n\times n}$ the Coriolis matrix, ${}^k\mathbf{C}_k \in \mathbb{R}^{6\times 6}$ the body Coriolis matrix and $\tau_{grav} \in \mathbb{R}^n$ the gravitational torque vector. Additionally, the body Jacobian and its time derivative appear in the equation. The gravitational wrench acting on robot link k is defined as

$${}^k\boldsymbol{\omega}_{k,\text{grav}} = {}^k\boldsymbol{M}_k\, \boldsymbol{Ad}^{-1}_{{}^0\boldsymbol{H}_k} \begin{bmatrix} 0 & 0 & -g & 0 & 0 & 0 \end{bmatrix}^T, \tag{2.35}$$

where g denotes the gravitational acceleration. In this work, it is set to $g = 9.81\,\text{m/s}^2$. Finally ${}^k\boldsymbol{\omega}_{k,nc}$ is the wrench acting on body k along the non-constrained directions.

Eq. (2.34) highlights the central role of the inertia matrix in the dynamic model. Through the body Jacobians, the inertia matrix of each link is projected into the generalized coordinates, contributing to the generalized mass matrix $\boldsymbol{M}$. By expanding the body Coriolis matrix ${}^k\boldsymbol{C}_k$, its dependence on the corresponding link inertia matrix ${}^k\boldsymbol{M}_k$ becomes explicit:

$${}^k\mathbf{C}_k = {}^k\mathbf{M}_k\, \mathbf{adj}\left({}^k\boldsymbol{\nu}_{0,k}\right) - \mathbf{adj}^T\left({}^k\boldsymbol{\nu}_{0,k}\right) {}^k\mathbf{M}_k. \tag{2.36}$$

Therefore, the inertial parameters influence not only the generalized mass matrix, but also the Coriolis and centrifugal terms of the inverse dynamics model. Therefore, the importance of the inertial parameters becomes evident: they define the inertia matrices of the links, which in turn determine the dynamics equations of the robot.

2.5. Matrix Distance Measures

2.5.1. Frobenius Norm

The Frobenius norm is mainly used to quantify the element-wise difference between two matrices. For two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m\times n}$, this difference is defined as

$$\|\boldsymbol{A} - \boldsymbol{B}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} \left(a_{ij} - b_{ij}\right)^2}. \tag{2.37}$$

The Frobenius norm is always non-negative and satisfies

$$\|\boldsymbol{A} - \boldsymbol{B}\|_F = 0 \quad \text{if and only if} \quad \boldsymbol{A} = \boldsymbol{B}. \tag{2.38}$$

2.5.2. Affine-Invariant Riemannian Metric

Given two symmetric positive definite (SPD) matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, the affine-invariant Riemannian metric (AIRM) distance is defined as

$$d_{\mathrm{AIRM}}(\boldsymbol{A},\boldsymbol{B}) = \left\| \mathrm{Log}\left(\boldsymbol{A}^{-1/2}\boldsymbol{B}\boldsymbol{A}^{-1/2}\right) \right\|_F = \sqrt{\sum_i (\log \lambda_i)^2}, \tag{2.39}$$

where λ_i are the eigenvalues of $\boldsymbol{A}^{-1/2}\boldsymbol{B}\boldsymbol{A}^{-1/2}$. The AIRM distance is always non-negative and satisfies

$$d_{\mathrm{AIRM}}(\boldsymbol{A},\boldsymbol{B}) = 0 \quad \text{if and only if} \quad \boldsymbol{A} = \boldsymbol{B}. \tag{2.40}$$

This distance is especially relevant in applications where SPD matrices must be compared independently of the coordinate system used to represent them.

3. Related Work

3.1. Inertial Parameter Identification

Accurate dynamic models are essential for model-based control, simulation, trajectory optimization, and torque prediction of robotic manipulators. However, the inertial parameters provided by CAD models or manufacturers are often insufficiently accurate, especially when the robot contains cables, grippers, or different payloads. For this reason, inertial-parameter identification has been widely studied as a way to estimate the dynamic parameters directly from experimental data.

A key observation in robot dynamics is that, although the equations of motion are nonlinear in the joint positions and velocities, the inverse dynamics can be written as a linear function of the inertial parameters. Atkeson et al. [5] showed that the Newton–Euler equations can be rearranged such that measured joint torques are related linearly to the inertial parameters, allowing the parameters to be estimated using least-squares techniques. This linear-in-the-parameters formulation became the basis for many classical robot dynamic identification methods:

$$\boldsymbol{\tau} = \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{\pi} \tag{3.1}$$

with

$$\boldsymbol{\pi} = [\boldsymbol{\pi}_1^T \ \boldsymbol{\pi}_2^T \ \dots \ \boldsymbol{\pi}_n^T]^T \tag{3.2}$$

For each link i, the standard inertial parameter vector is defined as

$$\boldsymbol{\pi}_i = [m_i, mX_i, mY_i, mZ_i, XX_i, YY_i, ZZ_i, XY_i, XZ_i, YZ_i]^T \in \mathbb{R}^{10} \tag{3.3}$$

These parameters correspond to the link mass, the first moment of mass, and the six independent components of the symmetric inertia tensor expressed in the corresponding link frame.

When measurements are collected over N samples, Eq. (3.2) can be stacked into the observation equation

$$\boldsymbol{T} = \boldsymbol{W}\boldsymbol{\pi} + \boldsymbol{\rho}, \tag{3.4}$$

where $\boldsymbol{T}$ contains the measured joint torques over all samples, $\boldsymbol{W}$ is the stacked observation matrix, and $\boldsymbol{\rho}$ represents measurement noise and unmodelled dynamics. Most classical identification methods are based on this equation. They mainly differ in how the parameters are estimated, how measurement noise is treated, how identifiability is handled, and whether physical consistency is enforced.

3.1.1. Overview of Inertial-Parameter Identification Methods

Several approaches have been proposed for inertial-parameter identification of robotic manipulators. Classical methods estimate the inertial parameters by solving a least-squares problem based on the inverse dynamic model. Later methods improve robustness by using weighted least squares, total least squares, instrumental variables, or output-error formulations. Other approaches focus on physical consistency, either by adding constraints to the optimization problem or by using a parameterization that only allows physically valid inertial parameters.

The methods differ mainly in how they treat measurement noise, nonlinearities, and physical feasibility. Least-squares methods are computationally efficient and remain widely used, but they are sensitive to noisy measurements and do not guarantee physically meaningful parameters. Output-error and Kalman-filter methods can improve robustness or allow recursive estimation, but they usually require more careful tuning or higher computational effort. Instrumental-variable methods reduce the bias caused by noisy regressors, while neural-network-based methods can compensate for nonlinear effects that are difficult to model explicitly. Physically consistent methods address a different limitation by enforcing feasibility of the identified inertial parameters.

The overview in Table 3.1 shows that least-squares-based methods are computationally attractive but do not guarantee physically meaningful parameters. Constrained and semidefinite-programming-based methods address this limitation by explicitly enforcing physical feasibility. However, these methods introduce additional constraints and may increase computational complexity. Reparameterized methods provide an alternative by guaranteeing physical consistency by construction. This thesis follows this direction and investigates whether such a physically consistent parameterization can be combined with gradient-based optimization to identify inertial parameters while preserving physical validity at every iteration.

3.1.2. Finding the Regressor Matrix

In this work, the regressor matrix is calculated as presented in [10]

$$\mathbf{Y} = \sum_k \mathbf{J}_k^T \left(\mathbf{A}(\ddot{\mathbf{x}}_k) - \mathrm{adj}_{0,k}^T \, \mathbf{A}(\mathbf{J}_k \dot{\mathbf{q}}) \right) \tag{3.5}$$

with

$$\ddot{\mathbf{x}}_k = {}^k_b\mathbf{J}_{0,k} \, \ddot{\mathbf{q}} + {}^k_b\dot{\mathbf{J}}_{0,k} \, \dot{\mathbf{q}} + \begin{bmatrix} {}^0\mathbf{R}_k{}^T [0,0,g]^T \\ \mathbf{0}_{3\times 1} \end{bmatrix} \tag{3.6}$$

where $\boldsymbol{A}$ (see Eq 2.30) is the Inertial-Parameter Coefficient Matrix and adj is the Lie Bracket Matrix (see Eq 2.23)

In the following, the constructed regressor matrix $\mathbf{Y}$ is validated by comparing the joint torques obtained from $\mathbf{Y}\boldsymbol{\pi}$ with those computed from the IDM. Equation 3.5 was implemented in MATLAB for this purpose. As shown in Figure 3.1, the torque for all joints obtained from the regressor matches the dynamics-based torque with an error on the order of 10^{-12}. The comparison is performed along the optimal trajectory introduced in Section 3.4.

Table 3.1.: Overview of representative inertial-parameter identification methods.

Approach	**Example method**	**Key characteristic**
Least-squares	IDIM-OLS [12]	Estimates the inertial parameters directly from the linear-in-the-parameters inverse dynamics model using ordinary least squares. It is simple and computationally efficient, but does not enforce physical consistency.
Output-error	DIDIM [13]	Identifies the parameters by comparing measured and simulated robot behavior. This makes it suitable for closed-loop identification, but leads to a nonlinear optimization problem.
Kalman-filter	EKF [14]	Updates the parameter estimate recursively using a state-estimation framework. This allows online identification, but the result depends strongly on noise assumptions and filter tuning.
Instrumental-variable	IDIM-IV [15]	Uses instrumental variables to reduce the bias introduced by noisy regressors. This improves robustness compared with ordinary least squares, but requires suitable instruments.
Neural-network	NN-aided ID [16]	Uses neural networks to learn nonlinear effects or compensate for unmodeled dynamics. This increases flexibility, but reduces physical interpretability and usually requires more data.
Physically consistent	PC-IDIM-OLS [17]	Extends least-squares identification by enforcing physical-consistency constraints on the inertial parameters. This ensures physically meaningful solutions, but requires constrained optimization.

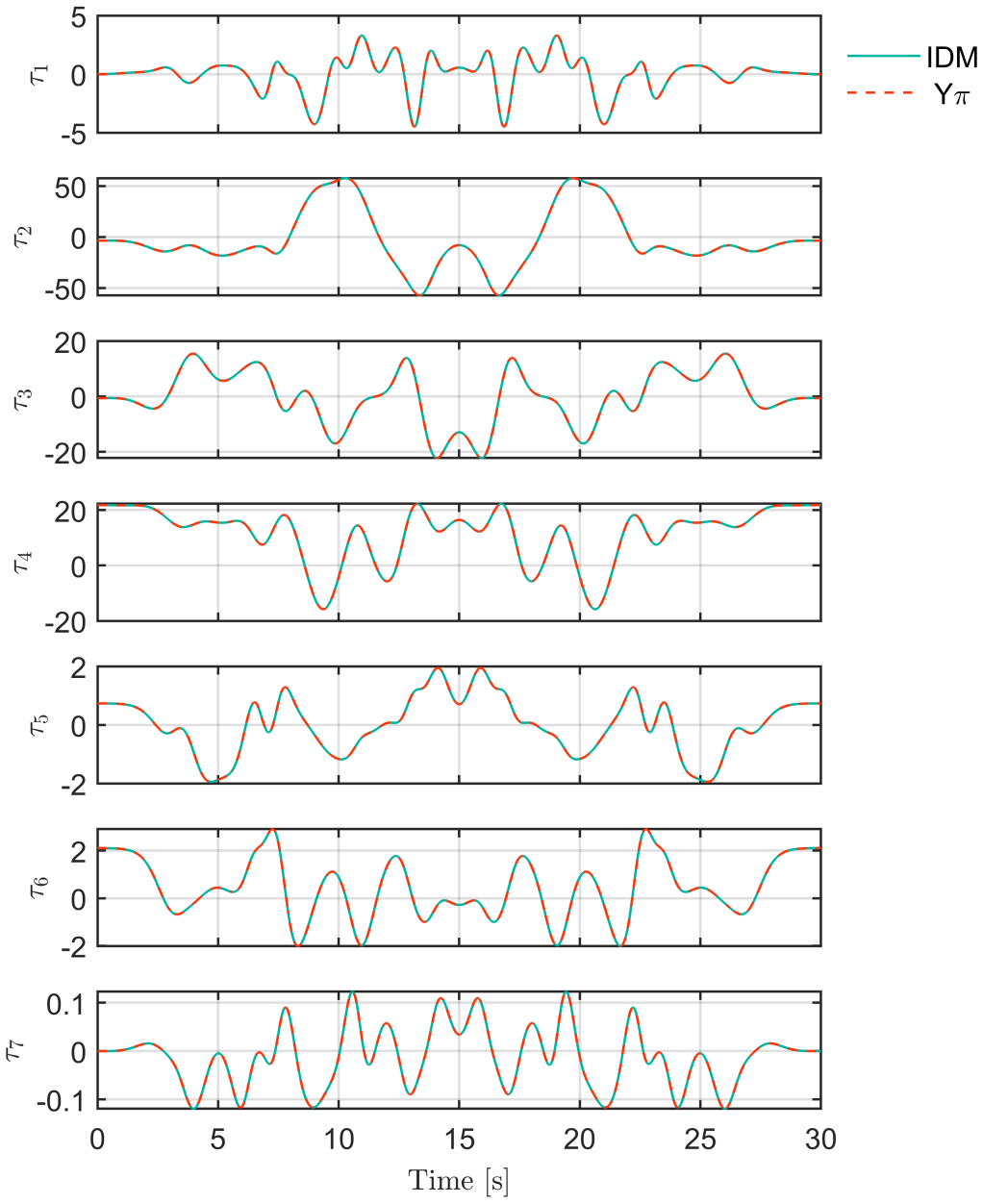

Figure 3.1.: Comparison of joint torques, in Nm, computed using the inverse-dynamics model and the regressor formulation for all seven joints. The curves overlap, confirming their numerical equivalence.

3.1.3. IDIM-OLS

Introduced by Gautier [12], the Inverse Dynamic Identification Model with Ordinary Least Squares (IDIM-OLS) has become one of the classical approaches for inertial parameter

identification. The ordinary least-squares problem is written as

$$\boldsymbol{\pi}^{\star} = \arg\min_{\boldsymbol{\pi}} \|\mathbf{W}\boldsymbol{\pi} - \mathbf{T}\|_2 . \tag{3.7}$$

One of the main characteristics of the method is its simplicity and ease of implementation; however, its performance strongly depends on the quality of the excitation trajectory used to build the observation matrix [15]. It is also important to note that if the observation matrix is not full rank, the identification problem is ill-posed. Therefore, the rank deficiency of the observation matrix must first be addressed by identifying a set of base parameters, as described in Sec. 3.2.

3.1.4. PC-IDIM-OLS

Although IDIM-OLS provides a simple and efficient estimate of the inertial parameters, the unconstrained least-squares solution is not guaranteed to be physically consistent (see Sec. 3.3). PC-IDIM-OLS addresses this limitation by extending the classical IDIM-OLS formulation with physical-consistency constraints on the inertial parameters. Following the pseudo-inertia matrix formulation of Wensing et al. [17], these constraints can be expressed as linear matrix inequalities (LMIs):

$$\begin{aligned} \boldsymbol{\pi}^{\star} &= \arg\min_{\boldsymbol{\pi}} \|\mathbf{W}\boldsymbol{\pi} - \mathbf{T}\|_2 \\ \text{s.t.} \quad \mathbf{P}_k(\boldsymbol{\pi}_k) &\succeq 0, \quad k = 1, \ldots, n_{links}. \end{aligned} \tag{3.8}$$

Here, $\mathbf{P}_k(\boldsymbol{\pi}_k)$ is the pseudo-inertia matrix of link k, which compactly combines the mass, first moment, and rotational inertia into a single matrix:

$$\mathbf{P}_k(\boldsymbol{\pi}_k) = \begin{bmatrix} \boldsymbol{\Sigma}_k & \begin{bmatrix} mX_k & mY_k & mZ_k \end{bmatrix} \\ \begin{bmatrix} mX_k & mY_k & mZ_k \end{bmatrix}^T & m_k \end{bmatrix} \in \mathbb{R}^{4\times 4} \tag{3.9}$$

The matrix $\boldsymbol{\Sigma}_k$ is given by

$$\boldsymbol{\Sigma}_k = \frac{1}{2}\,\mathrm{tr}(\mathbf{I}_k)\mathbf{I}_3 - \mathbf{I}_k, \tag{3.10}$$

These formulations preserve the convexity of the optimization problem, meaning that the obtained optimum is globally optimal. Wensing et al. show that enforcing positive definiteness of the pseudo-inertia matrix is sufficient to guarantee physical consistency of the inertial parameters.

3.2. Base Parameters and Identifiability

Finding a minimum set of non-redundant and identifiable inertial parameters is a fundamental step in the identification process, since it improves the robustness of the result [18]. In the standard inertial parameter vector $\boldsymbol{\pi}$, some parameters do not affect the robot dynamics at all. In addition, some parameters cannot be identified individually, but only through linear combinations with parameters of the adjacent links. This redundancy is a consequence of the

serial kinematic structure of the manipulator and of the motion constraints imposed by the joints [19].

Definition (Base Parameters). The base parameters $\boldsymbol{b}$ are the minimum set of identifiable inertial parameters that fully characterize the robot dynamics. They are sufficient to reproduce the same joint torques as the standard parameter vector. Therefore, the inverse model can also be written as

$$\boldsymbol{\tau} = \mathbf{Y}\boldsymbol{\pi} = \mathbf{Y}_b\boldsymbol{b}, \tag{3.11}$$

where $\mathbf{Y}_b$ is the reduced regressor matrix, with one column for each base parameter in $\boldsymbol{b}$.

Equivalently, the base parameter set can be interpreted from the structure of the regressor matrix. As described in [20], it can be obtained by applying column rearrangements to the regressor matrix $\boldsymbol{Y}$ in order to handle its rank deficiency. Let the regressor be partitioned as

$$\mathbf{Y} = \begin{bmatrix}\mathbf{Y}_1 & \mathbf{Y}_2\end{bmatrix}, \qquad \boldsymbol{\pi} = \begin{bmatrix}\boldsymbol{\pi}_1 \\ \boldsymbol{\pi}_2\end{bmatrix}, \tag{3.12}$$

where $\mathbf{Y}_1$ contains the independent columns. If the dependent columns satisfy $\mathbf{Y}_2 = \mathbf{Y}_1\mathbf{R}$, then

$$\boldsymbol{\tau} = \mathbf{Y}\boldsymbol{\pi} = \mathbf{Y}_1\left(\boldsymbol{\pi}_1 + \mathbf{R}\boldsymbol{\pi}_2\right). \tag{3.13}$$

Thus,

$$\mathbf{Y}_b = \mathbf{Y}_1, \qquad \boldsymbol{b} = \boldsymbol{\pi}_1 + \mathbf{R}\boldsymbol{\pi}_2. \tag{3.14}$$

This shows that base parameters are generally linear combinations of standard inertial parameters rather than individual physical quantities.

It is important to note that the base parameter set is not unique. Therefore, different regrouping strategies may produce different, but dynamically equivalent, reduced parameter vectors that satisfy Eq. (3.11). An upper bound on the number of base parameters can be obtained according to [18], which provides a useful estimate of the size of the identifiable parameter set. The number of base parameters is bounded by

$$n_b \leq 7n_r + 4n_p - 3\sigma, \tag{3.15}$$

where n_r is the number of revolute joints, n_p is the number of prismatic joints, and σ is a boolean variable equal to one if the first joint is revolute and zero otherwise.

Finally, base parameters should be distinguished from further reduced parameter sets such as essential parameters. Essential parameters are obtained afterwards by removing base parameters that have an insignificant contribution to the joint torques or that are poorly estimated in the presence of noise [21]. In this work, essential parameters are not considered.

3.2.1. Finding the Base Parameters

The base inertial parameters can be derived from the standard inertial parameters by eliminating those that do not affect the system dynamics and by regrouping parameters that appear only through linear combinations. Two main approaches are commonly used in the literature.

The first one is symbolic and relies on the analytical inspection of the dynamic equations. The second approach is numerical and identifies the independent columns of the regressor matrix using rank-revealing techniques. In [22], Gautier presented a QR-decomposition-based method to determine the base parameters. Numerical methods must be used carefully, since they may underestimate the number of base parameters if the trajectory used is not sufficiently exciting [23]. In the following section, one symbolic method is presented.

Symbolic Approach

Gautier and Khalil [24] proposed one of the earliest and most widely adopted strategies for determining the set of base parameters. In this method, parameters with no effect on the dynamics are first eliminated, after which a recursive regrouping algorithm is applied to address the rank deficiency of the regressor matrix.

1) Parameters with no effect on the dynamic model

The approach is based on the energy formulation of the robot dynamics. Since the joint torques are obtained from the kinetic and potential energies through the Euler–Lagrange equations, an inertial parameter that does not contribute to these energy terms cannot affect the resulting joint torques. These inertial parameters belong typically to the links closer to the robot base [24]. The kinetic and potential energies can be expressed as linear functions of the inertial parameters:

$$E = \sum_{i=1}^{10n} \frac{\partial E}{\partial \pi_i} \pi_i = \sum_{i=1}^{10n} D_{E_i} \pi_i, \tag{3.16}$$

$$U = \sum_{i=1}^{10n} \frac{\partial U}{\partial \pi_i} \pi_i = \sum_{i=1}^{10n} D_{U_i} \pi_i. \tag{3.17}$$

Here, D_{E_i} and D_{U_i} are the kinetic-energy and potential-energy coefficients associated with the inertial parameter π_i, respectively.

Based on the linear representation of the kinetic and potential energies in Eq. (3.16) and Eq. (3.17), the contribution of each inertial parameter to the joint torques can be analyzed through the Euler–Lagrange equations:

$$\boldsymbol{\tau} = \frac{d}{dt}\left(\frac{\partial E}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial E}{\partial \mathbf{q}} + \frac{\partial U}{\partial \mathbf{q}}. \tag{3.18}$$

Observation. If, for a given inertial parameter π_i, $D_{E_i} = 0$ and D_{U_i} is constant, then this parameter does not affect the joint torques and can be eliminated from the standard parameter set. Equivalently, the column of the regressor matrix corresponding to π_i is a zero vector.

A set of practical rules is also presented in [18] to facilitate the identification of parameters with no effect on the joint torques. Here, only the rules relevant to this thesis are reported:

Rule 1: If joint 1 is revolute, then the parameters XX_1, XY_1, XZ_1, YY_1, YZ_1, mZ_1, and m_1 have no effect on the dynamic model.

Rule 2: If the rotation axis of revolute joint 1 is parallel to the direction of gravitational acceleration, then mX_1 and mY_1 have no effect on the dynamic model.

2) Recursive regrouping algorithm

The authors propose recursive regrouping relations that depend on the type of joint. In particular, different regrouping strategies are defined for revolute and prismatic joints. Since this thesis only considers robots with revolute joints, the corresponding regrouping relations are reported in Table 3.2.

It is worth noting that the relations in Table 3.2 are valid for the modified Denavit–Hartenberg convention. The recursive procedure is applied sequentially, starting from the most distal link and proceeding toward the base.

Table 3.2.: General regrouping relations for revolute joints, where $c_i = \cos(\alpha_i)$ and $s_i = \sin(\alpha_i)$.

Parameter	General regrouping relation
XX'_i	$XX_i - YY_i$
XX'_{i-1}	$XX_{i-1} + YY_i + 2d_i mZ_i + d_i^2 m_i$
YY'_{i-1}	$YY_{i-1} + c_i^2 YY_i + 2d_i c_i^2 mZ_i + \left(a_i^2 + d_i^2 c_i^2\right) m_i$
ZZ'_{i-1}	$ZZ_{i-1} + s_i^2 YY_i + 2d_i s_i^2 mZ_i + \left(a_i^2 + d_i^2 s_i^2\right) m_i$
XY'_{i-1}	$XY_{i-1} + a_i s_i mZ_i + a_i d_i s_i m_i$
XZ'_{i-1}	$XZ_{i-1} - a_i c_i mZ_i - a_i d_i c_i m_i$
YZ'_{i-1}	$YZ_{i-1} + c_i s_i YY_i + 2d_i c_i s_i mZ_i + d_i^2 c_i s_i m_i$
mX'_{i-1}	$mX_{i-1} + a_i m_i$
mY'_{i-1}	$mY_{i-1} - s_i mZ_i - d_i s_i m_i$
mZ'_{i-1}	$mZ_{i-1} + c_i mZ_i + d_i c_i m_i$
m'_{i-1}	$m_{i-1} + m_i$

3.2.2. Base Parameters of the Franka Research 3

In the following section, the methods introduced above are applied to determine the base parameters of the Franka Research 3.

First, following Rule 1 and Rule 2, the parameters with no effect on the dynamic model are identified. As a result, nine inertial parameters from joint 1 are directly removed. Second, the regrouping relations in Table 3.2 are applied to links 2 through 7, allowing three parameters to be regrouped for each link. The recursive regrouping algorithm was implemented in MATLAB using the modified D–H parameters reported in Appendix B.

This procedure results in a total of 43 base parameters, which satisfies the upper bound of 46 obtained from Eq. (3.15). The resulting base parameter set is provided in Appendix B. Table 3.3 provides a visual summary of the results obtained in this section. The union of the parameters marked in green and purple forms a set of 43 base inertial parameters for the FR3.

In the same way, the elimination and regrouping procedure is applied to the columns of the full regressor matrix. After removing the dependent columns, the reduced regressor $\mathbf{Y}_b$ is obtained with $\text{rank}(\mathbf{Y}_b) = n_b$ and can be used together with the base parameter vector.

Table 3.3.: Classification of the standard inertial parameters according to their role in the parameter reduction.

m_1	mX_1	mY_1	mZ_1	XX_1	YY_1	ZZ_1	XY_1	XZ_1	YZ_1
m_2	mX_2	mY_2	mZ_2	XX_2	YY_2	ZZ_2	XY_2	XZ_2	YZ_2
m_3	mX_3	mY_3	mZ_3	XX_3	YY_3	ZZ_3	XY_3	XZ_3	YZ_3
m_4	mX_4	mY_4	mZ_4	XX_4	YY_4	ZZ_4	XY_4	XZ_4	YZ_4
m_5	mX_5	mY_5	mZ_5	XX_5	YY_5	ZZ_5	XY_5	XZ_5	YZ_5
m_6	mX_6	mY_6	mZ_6	XX_6	YY_6	ZZ_6	XY_6	XZ_6	YZ_6
m_7	mX_7	mY_7	mZ_7	XX_7	YY_7	ZZ_7	XY_7	XZ_7	YZ_7

Regrouped parameters — No-effect parameters

Combined parameters — Independent parameters

To verify that the regrouping and elimination strategy was correctly implemented, the joint torques obtained from the full model, $\mathbf{Y}\boldsymbol{\pi}$, are compared with those from the reduced model, $\mathbf{Y}_b\boldsymbol{b}$. The comparison is performed along the optimal trajectory introduced in Section 3.4. Only the result for joint 5 is shown in Figure 3.2 and the maximum errors for all joints remain on the order of 10^{-13} Nm.

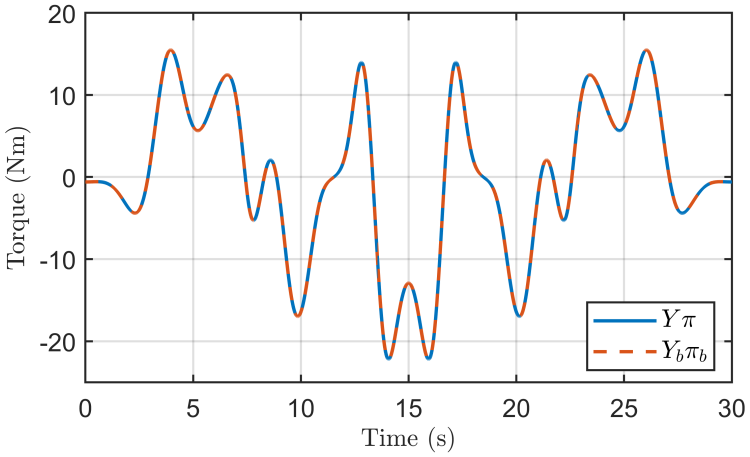

Figure 3.2.: Comparison of joint 5 torques computed with the full and reduced regressors.

3.3. Physical Consistency of Inertial Parameters

Early inertial parameter identification methods often did not explicitly account for the fact that not every identified parameter vector corresponds to a physically valid rigid body. The requirement of positive mass is straightforward, but it is only one part of the problem. The three-dimensional rotational inertia must also be physically meaningful, since arbitrary values may not correspond to any real mass distribution. Using such parameters in a robot model can result in unstable control performance and unrealistic simulations [7].

This motivated several approaches for incorporating physical feasibility into the identification process. Previous works addressed this in different ways, for example by using CAD inertial parameters as prior information or by enforcing constraints within semidefinite programming formulations. In many of these approaches, positive definiteness of the inertia matrix was considered sufficient to guarantee physical consistency [7, 25, 26, 27, 28].

Traversaro et al. showed that positive definiteness is necessary, but not sufficient, for full physical consistency [6]. In addition, the principal moments of inertia must satisfy the triangle inequalities:

$$J_1 \leq J_2 + J_3, \qquad J_2 \leq J_1 + J_3, \qquad J_3 \leq J_1 + J_2. \tag{3.19}$$

This distinction allows identification methods to be compared in terms of semi-consistency, where only positive definiteness is enforced, and full physical consistency, where the triangle inequalities are also satisfied.

Based on this distinction, different methods have been proposed to enforce full physical consistency during identification. Traversaro et al. formulated the identification problem as an optimization problem on manifolds. A key advantage of this approach is that the inertial parameters remain physically valid at every iteration, unlike standard constrained optimization methods, where feasibility is typically guaranteed only at convergence.

A different approach was proposed by Wensing et al. [17], who incorporated physical

consistency directly into the identification problem using linear matrix inequality constraints. These convex constraints can accelerate convergence and improve robustness to measurement noise. The concept of density realizability with respect to the frame origin is introduced in this work. It is useful because it provides a criterion for verifying whether the identified inertial parameters correspond to a physically valid rigid body. Let the rotational inertia tensor of link k be expressed through its eigendecomposition as

$$\mathbf{I}_k = \mathbf{R}\mathbf{J}\mathbf{R}^T, \qquad \mathbf{R} \in \mathrm{SO}(3), \qquad \mathbf{J} = \mathrm{diag}(J_1, J_2, J_3), \quad J_i > 0. \tag{3.20}$$

Then, $\mathbf{I}_k$ is density-realizable if and only if the principal moments of inertia J_1, J_2, and J_3 satisfy the triangle inequalities in (3.19). The condition $\mathbf{R} \in \mathrm{SO}(3)$ ensures that the eigenvectors define a valid rotational frame.

Definition (Full Physical Consistency). A set of standard inertial parameters is said to be fully physically consistent if and only if, for each link k,

$$m_k > 0 \quad \text{and} \quad \mathbf{I}_k \text{ is density realizable.} \tag{3.21}$$

In summary, physical consistency can be enforced either by adding constraints to the identification problem or by using a parametrization that only permits physically valid parameters. The latter usually improves physical interpretability, but often increases computational complexity.

3.4. Exciting Trajectories

The trajectory followed by the robotic manipulator during the identification experiment is essential for improving parameter estimation accuracy [29]. A sufficiently exciting trajectory reveals the influence of each base parameter on the robot dynamics and improves the conditioning of the identification problem. Proper excitation can also improve the convergence speed of the estimation process and increase its robustness against measurement noise [30].

Since the 2000s, the generation of optimal excitation trajectories has been a recurring research topic in the robot identification community [20]. The design of excitation trajectories generally involves two main choices: the parameterization used to describe the joint trajectories and the loss function of the optimization problem.

A common strategy is to parameterize the joint trajectories using finite Fourier series, which provide smooth periodic motions and allow analytical computation of joint velocities and joint accelerations [30, 31, 32, 33]. Due to their periodicity, Fourier-based trajectories can be repeated over several cycles, allowing redundant measurements to be collected and used to improve the signal-to-noise ratio. Other works formulate the problem directly in terms of polynomial or spline trajectories, which can incorporate boundary conditions and joint constraints [34, 5].

Accurate inertial parameter identification requires trajectories that make the robot dynamics sufficiently informative while respecting the physical limits of the manipulator. For this reason,

trajectory design is typically formulated as a constrained optimization problem. A generic trajectory optimization problem can be formulated as

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & J(\mathbf{W}(\boldsymbol{\theta})) \\ \text{s.t.} \quad & \mathbf{q}_{\min} \leq \mathbf{q}(t, \boldsymbol{\theta}) \leq \mathbf{q}_{\max}, \\ & \dot{\mathbf{q}}_{\min} \leq \dot{\mathbf{q}}(t, \boldsymbol{\theta}) \leq \dot{\mathbf{q}}_{\max}, \\ & \ddot{\mathbf{q}}_{\min} \leq \ddot{\mathbf{q}}(t, \boldsymbol{\theta}) \leq \ddot{\mathbf{q}}_{\max}, \\ & \boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau}(t, \boldsymbol{\theta}) \leq \boldsymbol{\tau}_{\max}, \qquad \forall t \in [0, T], \end{aligned} \tag{3.22}$$

where $\boldsymbol{\theta}$ denotes the trajectory parameters, $\mathbf{W}$ is the observation matrix, and $J(\cdot)$ is the chosen loss function.

The loss function is typically defined using numerical properties of the observation matrix.. An optimality criterion commonly used is the condition number of the observation matrix [35, 33, 36], defined as

$$\text{cond}(\mathbf{W}) = \frac{\sigma_{\max}(\mathbf{W})}{\sigma_{\min}(\mathbf{W})}, \tag{3.23}$$

where $\sigma_{\max}(\mathbf{W})$ and $\sigma_{\min}(\mathbf{W})$ are the largest and smallest singular values of $\mathbf{W}$, respectively. A smaller condition number indicates a better-conditioned observation matrix and therefore a more robust estimation problem. Other commonly used criteria include maximizing the smallest singular value, minimizing the Frobenius condition number, and maximizing determinant-based measures of the information matrix $\boldsymbol{W}^T\boldsymbol{W}$ [35].

Regardless of the chosen criterion, the resulting optimization problem is generally non-convex. Therefore, the global optimum is difficult to guarantee, and different initial conditions may lead to different locally optimal excitation trajectories.

3.4.1. Creating Optimal Trajectories

The trajectory design approach used in this work adopts the Fourier-based parameterization proposed by Park [31]. The desired joint trajectory is expressed as the sum of a polynomial component and a Fourier component. The implementation in this work follows this dual parameterization, but uses a different optimization objective.

This choice of parameterization is motivated by two main advantages. First, the periodicity of the trajectories can improve the signal-to-noise ratio, since repeated cycles provide redundant information. Second, the use of trigonometric functions gives smooth analytical expressions for the position, velocity, and acceleration. For joint i, the joint position is

$$q_i(t) = \lambda_i(t) + \delta_i(t), \tag{3.24}$$

where the polynomial term $\lambda_i(t)$ is used to satisfy the boundary conditions, while the Fourier term $\delta_i(t)$ provides periodic excitation. These two components are defined as

$$\lambda_i(t) = \lambda_{i,0} + \lambda_{i,1}t + \lambda_{i,2}t^2 + \lambda_{i,3}t^3 + \lambda_{i,4}t^4 + \lambda_{i,5}t^5, \tag{3.25}$$

and

$$\delta_i(t) = \sum_{m=1}^{M} a_{im} \cos\left(m\omega_f t\right). \tag{3.26}$$

Here, $\lambda_{i,j}$ are the polynomial coefficients, a_{im} are the Fourier coefficients, M is the number of harmonics, $\omega_f = 2\pi/t_f$ is the fundamental pulsation, and t_f is the trajectory duration. The boundary conditions for position, velocity, and acceleration are written as

$$\begin{aligned} q_i(0) &= \lambda_i(0) + \delta_i(0), \quad q_i(t_f) = \lambda_i(t_f) + \delta_i(t_f), \\ \dot{q}_i(0) &= \dot{\lambda}_i(0) + \dot{\delta}_i(0), \quad \dot{q}_i(t_f) = \dot{\lambda}_i(t_f) + \dot{\delta}_i(t_f), \\ \ddot{q}_i(0) &= \ddot{\lambda}_i(0) + \ddot{\delta}_i(0), \quad \ddot{q}_i(t_f) = \ddot{\lambda}_i(t_f) + \ddot{\delta}_i(t_f). \end{aligned} \tag{3.27}$$

The boundary conditions provide six equations, which are sufficient to determine the six polynomial coefficients analytically. Therefore, only the Fourier coefficients remain as independent optimization variables. They are collected in $\boldsymbol{\theta} \in \mathbb{R}^{nM}$:

$$\boldsymbol{\theta} = \mathrm{vec}(\mathbf{A}), \qquad \mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1M} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nM} \end{bmatrix}. \tag{3.28}$$

Using this parameterization, the trajectory optimization problem is formulated as

$$\begin{aligned} \boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} \quad & \log(\sigma_{\max}(\bar{\mathbf{W}}(\boldsymbol{\theta}))) - \log(\sigma_{\min}(\bar{\mathbf{W}}(\boldsymbol{\theta}))) \\ \text{s.t.} \quad & \mathbf{q}_{\min} \le \mathbf{q}(t, \boldsymbol{\theta}) \le \mathbf{q}_{\max}, \\ & \dot{\mathbf{q}}_{\min} \le \dot{\mathbf{q}}(t, \boldsymbol{\theta}) \le \dot{\mathbf{q}}_{\max}, \\ & \ddot{\mathbf{q}}_{\min} \le \ddot{\mathbf{q}}(t, \boldsymbol{\theta}) \le \ddot{\mathbf{q}}_{\max}, \qquad \forall t \in [0, t_f], \end{aligned} \tag{3.29}$$

where $\bar{\mathbf{W}}$ is the column-normalized observation matrix, obtained by scaling each column of the observation matrix $\mathbf{W}$ by its Euclidean norm. The quantities $\sigma_{\max}(\bar{\mathbf{W}})$ and $\sigma_{\min}(\bar{\mathbf{W}})$ denote the largest and smallest singular values of $\bar{\mathbf{W}}$, respectively. Thus, the objective corresponds to minimizing the logarithm of the condition number of the normalized observation matrix.

3.4.2. Trajectories for the Franka Research 3

This section presents the results obtained using the trajectory optimization method described previously.

Because the optimization problem is non-convex, the optimization was repeated five times with different initial guesses. The trajectory with the lowest final loss was selected.

The optimization problem was implemented in MATLAB and solved using `fmincon` . Each run was computationally expensive: on a laptop with an Intel Core i7 processor and 16 GB of RAM, 50 iterations took on average more than nine hours. Evaluating the objective function at every trajectory sample would have made the computation even more expensive. To reduce this cost, the objective function was evaluated at 80 equally spaced samples between 0 and t_f. The full trajectory was discretized with a sampling time of 0.1 s. Position, velocity, and

acceleration constraints were enforced at all trajectory samples, even though only the 80 selected samples were used in the objective function. Once the optimal Fourier coefficients $\boldsymbol{\theta}^{\star}$ are found, the trajectory can be resampled at a higher resolution.

The initial and final joint positions, $q_i(0)$ and $q_i(t_f)$, were set to the default joint configuration of the Franka R3, listed in Appendix B. Five harmonics were used in the Fourier parameterization. The velocity and acceleration limits were taken from the Franka R3 specifications, also reported in Appendix B.

As shown in Fig. 3.3, the optimized end-effector path exhibits an out-and-back motion: the arm moves away from the default joint configuration, reaches an intermediate position around $t_f/2$, and then returns along the same path. The corresponding joint-position trajectories are shown in Fig. A.2. All imposed constraints were satisfied.

The cost function decreased from 7.4006 to 1.6897. Since the objective minimizes the logarithm of the condition number, these values correspond to condition numbers of approximately 1636.9663 and 5.4178, respectively. This indicates a substantial improvement in the conditioning of the normalized observation matrix.

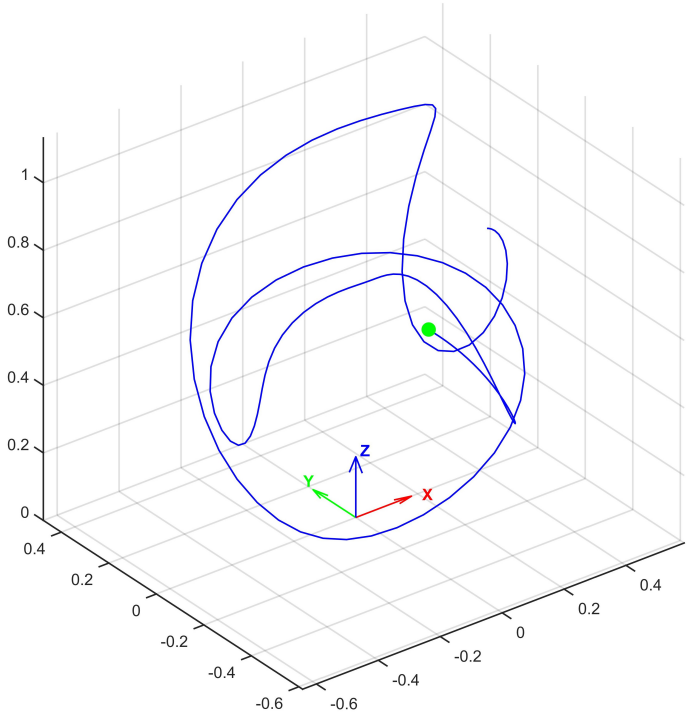

Figure 3.3.: End-effector path generated by the Fourier-based optimal trajectory. The green marker denotes the coincident start and end position. Axes are in meters.

4. Gradient-based Inertial Parameter Identification

4.1. Reformulation of the Inverse Dynamics Model

Given a dataset containing N measurements, the generalized joint torques can be predicted using the inverse dynamics model introduced in Eq. (2.34). For a candidate inertial parameter vector $\boldsymbol{\pi}$, the estimated torque at sample i is obtained from the inverse dynamics model:

$$\boldsymbol{\tau}_{est}^{(i)} = \mathrm{IDM}\left(\boldsymbol{q}^{(i)}, \dot{\boldsymbol{q}}^{(i)}, \ddot{\boldsymbol{q}}^{(i)}, \boldsymbol{\pi}\right), \qquad i = 1, \ldots, N, \tag{4.1}$$

Expanding the inverse dynamics expression in terms of the contributions of the individual robot links yields the expression below.

$$\begin{aligned}\boldsymbol{\tau}_{\mathrm{est}}^{(i)} = \sum_{k=1}^{n_{\mathrm{links}}} \Big(& {}^{k}\mathbf{J}_{0,k}^{T(i)}\, {}^{k}\mathbf{M}_k\, {}^{k}\mathbf{J}_{0,k}^{(i)}\, \ddot{\boldsymbol{q}}^{(i)} + {}^{k}\mathbf{J}_{0,k}^{T(i)}\, {}^{k}\mathbf{M}_k\, {}^{k}\dot{\mathbf{J}}_{0,k}^{(i)}\, \dot{\boldsymbol{q}}^{(i)} \\ & + {}^{k}\mathbf{J}_{0,k}^{T(i)}\, {}^{k}\mathbf{C}_k^{(i)}\, {}^{k}\mathbf{J}_{0,k}^{(i)}\, \dot{\boldsymbol{q}}^{(i)} - {}^{k}\mathbf{J}_{0,k}^{T(i)}\, {}^{k}\boldsymbol{\omega}_{k,\mathrm{grav}} \Big).\end{aligned} \tag{4.2}$$

The inertia matrix ${}^{k}\mathbf{M}_k$ depends directly on the inertial parameters of link k, as described in Section 2.4.1. The body Coriolis matrix ${}^{k}\mathbf{C}_k{}^{(i)}$ and the gravitational wrench $\boldsymbol{\omega}_{k,grav}$ can then be substituted. In addition, the kinematic relationship ${}^{k}\boldsymbol{\nu}_{0,k} = {}^{k}\mathbf{J}_{0,k}\,\dot{\mathbf{q}}$ is applied, yielding the following equation:

$$\begin{aligned}\boldsymbol{\tau}_{\mathrm{est}}^{(i)} = \sum_{k=1}^{n_{\mathrm{links}}} \Bigg[& {}^{k}\mathbf{J}_{0,k}^{T(i)} {}^{k}\mathbf{M}_k\, {}^{k}\mathbf{J}_{0,k}^{(i)} \ddot{\boldsymbol{q}}^{(i)} + {}^{k}\mathbf{J}_{0,k}^{T(i)} {}^{k}\mathbf{M}_k\, {}^{k}\dot{\mathbf{J}}_{0,k}^{(i)} \dot{\boldsymbol{q}}^{(i)} \\ & + {}^{k}\mathbf{J}_{0,k}^{T(i)} \underbrace{\left({}^{k}\mathbf{M}_k \cancel{\mathbf{adj}\left({}^{k}\boldsymbol{\nu}_{0,k}^{(i)}\right)} - \mathbf{adj}^{T}\left({}^{k}\boldsymbol{\nu}_{0,k}^{(i)}\right) {}^{k}\mathbf{M}_k \right)}_{{}^{k}\mathbf{C}_k{}^{(i)}} {}^{k}\boldsymbol{\nu}_{0,k} \\ & - {}^{k}\mathbf{J}_{0,k}^{T(i)} \underbrace{\left({}^{k}\mathbf{M}_k \mathbf{Ad}_{{}^{0}\mathbf{H}_k}^{-1} [0,0,-g,0,0,0]^{T} \right)}_{\boldsymbol{\omega}_{k,grav}} \Bigg]\end{aligned}$$

The first term of the body Coriolis matrix vanishes because the Lie bracket operator is applied to the same velocity vector that defines it, as shown in Eq. (2.24). After removing this term, the common factor ${}^{k}\mathbf{J}_{0,k}^{T}\, {}^{k}\mathbf{M}_k$ can be extracted. In addition, the term involving the inverse

adjoint transformation can be simplified.

$$\boldsymbol{\tau}_{\text{est}}^{(i)} = \sum_{k=1}^{n_{\text{links}}} \left[{}^{k}\mathbf{J}_{0,k}^{T(i)}\, {}^{k}\mathbf{M}_{k}\, {}^{k}\ddot{\mathbf{x}}_{0,k}^{(i)} - {}^{k}\mathbf{J}_{0,k}^{T(i)}\, \mathbf{adj}^{T}\!\left({}^{k}\boldsymbol{\nu}_{0,k}^{(i)}\right) {}^{k}\mathbf{M}_{k}\, {}^{k}\boldsymbol{\nu}_{0,k} \right] \tag{4.3}$$

Here, ${}^{k}\ddot{\mathbf{x}}_{0,k}$ is the body acceleration vector of link k, which includes the translational, rotational, and gravitational contributions. It is defined as:

$${}^{k}\ddot{\mathbf{x}}_{0,k}^{(i)} = {}^{k}\mathbf{J}_{0,k}^{(i)}\ddot{\boldsymbol{q}}^{(i)} + {}^{k}\dot{\mathbf{J}}_{0,k}^{(i)}\dot{\boldsymbol{q}}^{(i)} + \begin{bmatrix} {}^{0}\mathbf{R}_{k}{}^{T}[0,0,g]^{T} \\ \mathbf{0}_{3\times 1} \end{bmatrix} \tag{4.4}$$

The compact form of the inverse dynamics model in Eq. (4.3) makes its dependence on the inertia matrices, and therefore on the inertial parameters, explicit.

4.2. Optimization Problem

The reformulated inverse dynamics model expresses the predicted joint torques explicitly as a function of the inertial parameters. The identification problem can therefore be formulated by minimizing the torque-prediction error over all N samples:

$$\boldsymbol{\pi}^{\star} = \arg\min_{\boldsymbol{\pi}} \mathcal{L}(\boldsymbol{\pi}), \qquad \mathcal{L}(\boldsymbol{\pi}) = \frac{1}{N}\sum_{i=1}^{N} G_i. \tag{4.5}$$

For sample i, the weighted torque prediction error is defined as :

$$G_i = \frac{1}{2}\left(\boldsymbol{\tau}_{\text{est}}^{(i)} - \boldsymbol{\tau}_{\text{msrd}}^{(i)}\right)^{T} \mathbf{W} \left(\boldsymbol{\tau}_{\text{est}}^{(i)} - \boldsymbol{\tau}_{\text{msrd}}^{(i)}\right), \tag{4.6}$$

where $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a diagonal positive-definite weighting matrix defined as

$$\mathbf{W} = \operatorname{diag}(w_1, \ldots, w_n). \tag{4.7}$$

The weights are determined automatically from the measured torques in the training set:

$$w_k = \frac{\tilde{w}_k}{\frac{1}{n}\sum_{\ell=1}^{n}\tilde{w}_\ell}, \qquad \tilde{w}_k = \frac{1}{\tau_{\text{rms},k}^2 + \sigma^2}, \tag{4.8}$$

where $\tau_{\text{rms},k}$ denotes the RMS measured torque of joint k, and $\sigma = 0.5$ is a regularization parameter. This choice normalizes the torque errors with respect to their joint magnitudes, preventing high-torque joints from dominating the cost while ensuring that the mean diagonal weight is equal to one.

Directly solving this optimization problem with respect to $\boldsymbol{\pi}$ does not guarantee that the resulting parameters satisfy the physical-consistency conditions introduced in Section 3.3. To avoid such solutions, an alternative parameterization of the standard inertial parameters is introduced.

4.3. Lambda Parametrization

The purpose of the alternative parameterization is to guarantee physical consistency by construction, rather than by imposing constraints on the optimization problem. For each link k, the parameterization guarantees positive mass and density realizability, as defined in Section 3.3.

(Positive Mass) Since the exponential function is strictly positive, the mass of link k is parameterized as

$$m_k = e^{\mu_k}. \tag{4.9}$$

(CoM position) In contrast to the standard inertial parameter vector, the CoM position is optimized directly:

$${}^k\mathbf{f}_{k,\mathrm{CoM}} = \begin{bmatrix} x_k & y_k & z_k \end{bmatrix}^T. \tag{4.10}$$

(Valid Rotation) The orientation of the principal axes of inertia is parameterized by the rotation vector $\boldsymbol{\omega}_k$. As mentioned in Section 2.2, the exponential map on $\mathrm{SO}(3)$ takes a three-dimensional vector and outputs a valid rotation matrix:

$$\exp(\boldsymbol{\omega}) = \mathbf{R}_k \qquad with \qquad \boldsymbol{\omega}_k = \begin{bmatrix} \omega_{1,k} \\ \omega_{2,k} \\ \omega_{3,k} \end{bmatrix}. \tag{4.11}$$

(Triangle Inequalities) To satisfy the triangle inequalities in Eq. (3.19), the principal moments of inertia are parameterized as

$$J_1 = {b_k}^2 + {c_k}^2, \quad J_2 = {a_k}^2 + {c_k}^2, \quad J_3 = {a_k}^2 + {b_k}^2. \tag{4.12}$$

The identification problem can therefore be reformulated as an unconstrained optimization while ensuring that every solution corresponds to a physically consistent inertia. Collecting the newly introduced optimization variables gives the parameter vector of link k,

$$\boldsymbol{\lambda}_k = [\, \mu_k,\ x_k,\ y_k,\ z_k,\ \omega_{1k},\ \omega_{2k},\ \omega_{3k},\ a_k,\ b_k,\ c_k \,]^T \in \mathbb{R}^{10} \tag{4.13}$$

In this way, the inertia matrix of link k can be expressed equivalently as a function of the lambda parameters:

$${}^k\mathbf{M}_k(\boldsymbol{\lambda}_k) = \begin{bmatrix} e^{\mu_k}\mathbf{I}_{3\times 3} & \left[\, e^{\mu_k}\mathbf{f}_{k,\mathrm{CoM}}\times \,\right]^T \\ \left[\, e^{\mu_k}\mathbf{f}_{k,\mathrm{CoM}}\times \,\right] & \mathbf{I}_{k_{\mathrm{CoM}}} - e^{\mu_k}\left[\, {}^k\mathbf{f}_{k,\mathrm{CoM}}\times \,\right]\left[\, {}^k\mathbf{f}_{k,\mathrm{CoM}}\times \,\right]^T \end{bmatrix} \tag{4.14}$$

Using the principal moments defined in Eq. 4.12 and the rotation matrix introduced in Eq. 2.7, the inertia tensor of link k, expressed at its CoM, is reconstructed as

$$\mathbf{I}_{k_{\mathrm{CoM}}}(\boldsymbol{\lambda}_k) = \boldsymbol{R}_k \,\mathrm{diag}(J_1, J_2, J_3)\boldsymbol{R}_k^T \tag{4.15}$$

Finally, the n_{links} parameter vectors are collected into the complete parameter vector

$$\boldsymbol{\lambda} = \begin{bmatrix}\boldsymbol{\lambda}_1 & \cdots & \boldsymbol{\lambda}_{n_{links}} \in \mathbb{R}^{10\,n_{links}}\end{bmatrix}. \tag{4.16}$$

4.3.1. Mapping

In some cases, attaching a gripper or other accessories to the manipulator may alter its inertial parameters in ways that are difficult to predict. Therefore, a reasonable initialization for the optimization algorithm is the set of standard parameters currently assumed to represent the system. On the other hand, after completing the optimization process, it may be necessary to recover the physically interpretable standard parameters. Both procedures can be performed through a consistent forward and inverse mapping, once an eigenvalue ordering and rotation-vector convention are fixed.

Forward Map

For link k, the standard inertial parameter vector is transformed into the physically consistent parameterization through

$$\mathcal{M}(\boldsymbol{\pi}_k) = \boldsymbol{\lambda}_k \tag{4.17}$$

The steps required to perform this mapping are described below. First, the logarithm of the mass is computed as

$$\mu_k = \log(m_k), \tag{4.18}$$

and the CoM coordinates are recovered from the first moments of mass:

$${}^k\mathbf{f}_{k,\mathrm{CoM}} = \begin{bmatrix}x_k\\ y_k\\ z_k\end{bmatrix} = \frac{1}{m_k}\begin{bmatrix}mX_k\\ mY_k\\ mZ_k\end{bmatrix}. \tag{4.19}$$

The inertia tensor is then transferred from the link-frame origin to the CoM using the parallel-axis theorem:

$$\boldsymbol{I}_{k_{CoM}} = \boldsymbol{I}_k + m_k[{}^k\boldsymbol{f}_{k,\mathrm{CoM}}\times][{}^k\boldsymbol{f}_{k,\mathrm{CoM}}\times] \tag{4.20}$$

An eigenvalue decomposition is applied to obtain the principal moments of inertia and the orientation of the principal axes:

$$\boldsymbol{I}_{k_{CoM}} = \boldsymbol{R}_k \,\mathrm{diag}\,(J_1, J_2, J_3)\,\boldsymbol{R}_k^{\top}. \tag{4.21}$$

The rotation vector $\boldsymbol{\omega}_k$ is obtained from the rotation matrix $\boldsymbol{R}_k$ using the logarithmic map (see Section 2.2):

$$\boldsymbol{\omega}_k = \begin{bmatrix}\omega_{1,k}\\ \omega_{2,k}\\ \omega_{3,k}\end{bmatrix} = \log(\boldsymbol{R}_k). \tag{4.22}$$

Finally, the parameters a_k, b_k, and c_k are obtained from the principal moments:

$$a_k = \sqrt{\frac{J_2 + J_3 - J_1}{2}},$$
$$b_k = \sqrt{\frac{J_1 + J_3 - J_2}{2}},$$
$$c_k = \sqrt{\frac{J_1 + J_2 - J_3}{2}}.$$

The resulting parameters are then collected into the vector:

$$\boldsymbol{\lambda}_k = [\, \mu_k,\ x_k,\ y_k,\ z_k,\ \omega_{1k},\ \omega_{2k},\ \omega_{3k},\ a_k,\ b_k,\ c_k \,]^T \tag{4.23}$$

Inverse Map

For link k, the physically consistent parameter vector can be transformed back into the standard inertial parameter vector through

$$\mathcal{M}^{-1}(\boldsymbol{\lambda}_k) = \boldsymbol{\pi}_k. \tag{4.24}$$

The mass is first recovered from μ_k as

$$m_k = e^{\mu_k}. \tag{4.25}$$

The first moments of mass are then obtained from the CoM coordinates:

$$\begin{bmatrix} mX_k \\ mY_k \\ mZ_k \end{bmatrix} = m_k\, {}^k\boldsymbol{f}_{k,\mathrm{CoM}} \tag{4.26}$$

The orientation of the principal axes is recovered from the rotation vector using the exponential map (see Section 2.2):

$$\boldsymbol{R}_k = \exp(\boldsymbol{\omega}_k)\,. \tag{4.27}$$

The principal moments of inertia are computed as

$$\begin{aligned} J_1 &= b_k^2 + c_k^2, \\ J_2 &= c_k^2 + a_k^2, \\ J_3 &= a_k^2 + b_k^2. \end{aligned} \tag{4.28}$$

Using the rotation matrix from Eq. 4.27 and the principal moments from Eq. 4.28, the inertia tensor at the CoM is reconstructed as

$$\boldsymbol{I}_{k\mathrm{CoM}} = \boldsymbol{R}_k\ \mathrm{diag}(J_{1,k}, J_{2,k}, J_{3,k})\ \boldsymbol{R}_k^\top. \tag{4.29}$$

The inertia tensor is then transferred from the CoM to the link-frame origin using the parallel-axis theorem:

$$\boldsymbol{I}_k = \boldsymbol{I}_{k_{CoM}} - m_k[{}^k\boldsymbol{f}_{k,\mathrm{CoM}}\times][{}^k\boldsymbol{f}_{k,\mathrm{CoM}}\times] \tag{4.30}$$

Finally, the inertia components are extracted from the symmetric matrix $\boldsymbol{I}_k$, and the standard inertial parameter vector is assembled as

$$\boldsymbol{\pi}_k = [\, m_k,\ mX_k,\ mY_k,\ mZ_k,\ XX_k,\ YY_k,\ ZZ_k,\ XY_k,\ XZ_k,\ YZ_k \,]^T \tag{4.31}$$

4.4. Gradient Computation

The optimization problem requires the gradient of the loss function with respect to the parameter vector $\boldsymbol{\lambda}$. This gradient is available in closed form. Using the chain rule, the contribution of sample i is

$$\nabla_{\boldsymbol{\lambda}} G_i = \left(\frac{\partial \boldsymbol{\tau}_{\text{est}}^{(i)}}{\partial \boldsymbol{\lambda}} \right)^T \frac{\partial G_i}{\partial \boldsymbol{\tau}_{\text{est}}^{(i)}} \in \mathbb{R}^{10\, n_{\text{links}}}. \tag{4.32}$$

Since $\mathbf{W}$ is symmetric, the derivative of the sample-wise loss with respect to the estimated torque is

$$\frac{\partial G_i}{\partial \boldsymbol{\tau}_{\text{est}}^{(i)}} = \mathbf{W} \left(\boldsymbol{\tau}_{\text{est}}^{(i)} - \boldsymbol{\tau}_{\text{msrd}}^{(i)} \right) \in \mathbb{R}^n. \tag{4.33}$$

The torque Jacobian with respect to the complete parameter vector will be denoted by

$$\frac{\partial\, \boldsymbol{\tau}_{\text{est}}^{(i)}}{\partial\, \boldsymbol{\lambda}} = \begin{bmatrix} \boldsymbol{D}_1^{(i)} & \cdots & \boldsymbol{D}_{n_{\text{links}}}^{(i)} \end{bmatrix} \in \mathbb{R}^{n \times 10\, n_{\text{links}}}, \tag{4.34}$$

where $\mathbf{D}_k^{(i)}$ is the block associated with link k:

$$\boldsymbol{D}_k^{(i)} = \frac{\partial \boldsymbol{\tau}_{\text{est}}^{(i)}}{\partial \boldsymbol{\lambda}_k} = \begin{bmatrix} \boldsymbol{d}_{k,1}^{(i)} & \cdots & \boldsymbol{d}_{k,10}^{(i)} \end{bmatrix} \in \mathbb{R}^{n \times 10}. \tag{4.35}$$

Each column gives the derivative of the estimated joint-torque vector with respect to one element of $\boldsymbol{\lambda}_k$. For the p-th parameter of link k, the corresponding column is

$$\begin{aligned} \boldsymbol{d}_{k,p}^{(i)} &= \frac{\partial \boldsymbol{\tau}_{\text{est}}^{(i)}}{\partial \lambda_{k,p}} \\ &= {}^k\mathbf{J}_{0,k}^{T(i)} \left[\frac{\partial\, {}^k\mathbf{M}_k}{\partial \lambda_{k,p}}\, {}^k\ddot{\mathbf{x}}_{0,k}^{(i)} - \mathbf{adj}^T \left({}^k\boldsymbol{\nu}_{0,k}^{(i)} \right) \frac{\partial\, {}^k\mathbf{M}_k}{\partial \lambda_{k,p}}\, {}^k\boldsymbol{\nu}_{0,k}^{(i)} \right] \in \mathbb{R}^n, \end{aligned} \tag{4.36}$$

for $p \in \{1, \ldots, 10\}$.

Because of the serial kinematic structure, the inertial parameters of link k contribute only to the generalized torques of joint k and its parent joints. Therefore, rows $k+1, \ldots, n$ of $\mathbf{D}_k^{(i)}$ are zero.

The partial derivatives of the inertia matrix with respect to selected elements of the lambda vector are reported in Appendix A. The derivatives with respect to the remaining parameters lead to very large symbolic expressions and are not written explicitly. These expressions are computed symbolically in MATLAB and used directly in the implementation. Their evaluation represents one of the main computational bottlenecks.

Finally, the gradient of the complete loss function is obtained by averaging the contributions of all samples:

$$\nabla_{\boldsymbol{\lambda}} \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\lambda}} G_i. \tag{4.37}$$

4.5. Gradient Verification

To verify the correctness of the analytical gradient used in the optimization procedure, a numerical consistency check based on the first-order Taylor expansion of the loss function is performed.

Let $\mathcal{L} : \mathbb{R}^{10n_{\text{links}}} \to \mathbb{R}$ denote the loss function, and let $\nabla_\lambda \mathcal{L}$ denote its implemented analytical gradient. For a reference parameter vector $\bar{\boldsymbol{\lambda}} \in \mathbb{R}^{10n_{\text{links}}}$ and a perturbation direction $\boldsymbol{\Delta} \in \mathbb{R}^{10n_{\text{links}}}$, the first-order Taylor approximation is

$$\mathcal{L}(\bar{\boldsymbol{\lambda}} + \epsilon\boldsymbol{\Delta}) = \mathcal{L}(\bar{\boldsymbol{\lambda}}) + \epsilon\, \nabla_\lambda \mathcal{L}(\bar{\boldsymbol{\lambda}})^T \boldsymbol{\Delta} + \mathcal{O}(\epsilon^2), \tag{4.38}$$

for sufficiently small $\epsilon > 0$. The corresponding Taylor remainder is defined as

$$R(\epsilon) = \mathcal{L}(\bar{\boldsymbol{\lambda}} + \epsilon\boldsymbol{\Delta}) - \mathcal{L}(\bar{\boldsymbol{\lambda}}) - \epsilon\, \nabla_\lambda \mathcal{L}(\bar{\boldsymbol{\lambda}})^T \boldsymbol{\Delta}. \tag{4.39}$$

If the implemented gradient is correct, the first-order term cancels and the remainder satisfies

$$R(\epsilon) = \mathcal{O}(\epsilon^2). \tag{4.40}$$

Test. The remainder is evaluated for decreasing step sizes ϵ along a randomly sampled direction $\boldsymbol{\Delta}$. As shown in Figure 4.1, $|R(\epsilon)|$ decreases quadratically with ϵ over more than five orders of magnitude. In a log–log plot, this quadratic relation appears as a straight line with slope two:

$$\log |R(\epsilon)| = 2 \log \epsilon + \text{const.} \tag{4.41}$$

The observed quadratic scaling validates the analytical gradient implementation.

4.6. Optimizer

Any suitable first-order optimizer can be used to update the physically consistent parameter vector $\boldsymbol{\lambda}$. In this work, the Adaptive Moment Estimation optimizer, commonly known as Adam, is used because it is computationally efficient and widely adopted. Adam combines momentum-like updates with adaptive learning rates based on estimates of the first and second moments of the gradient [37]. Let $\mathbf{g}_h = \nabla_\lambda \mathcal{L}$ denote the gradient of the loss function at iteration h. Adam computes exponentially weighted estimates of the first and second moments of the gradient:

$$\boldsymbol{m}_h = \beta_1 \boldsymbol{m}_{h-1} + (1 - \beta_1)\boldsymbol{g}_h, \tag{4.42}$$

$$\boldsymbol{v}_h = \beta_2 \boldsymbol{v}_{h-1} + (1 - \beta_2)(\mathbf{g}_h \odot \mathbf{g}_h), \tag{4.43}$$

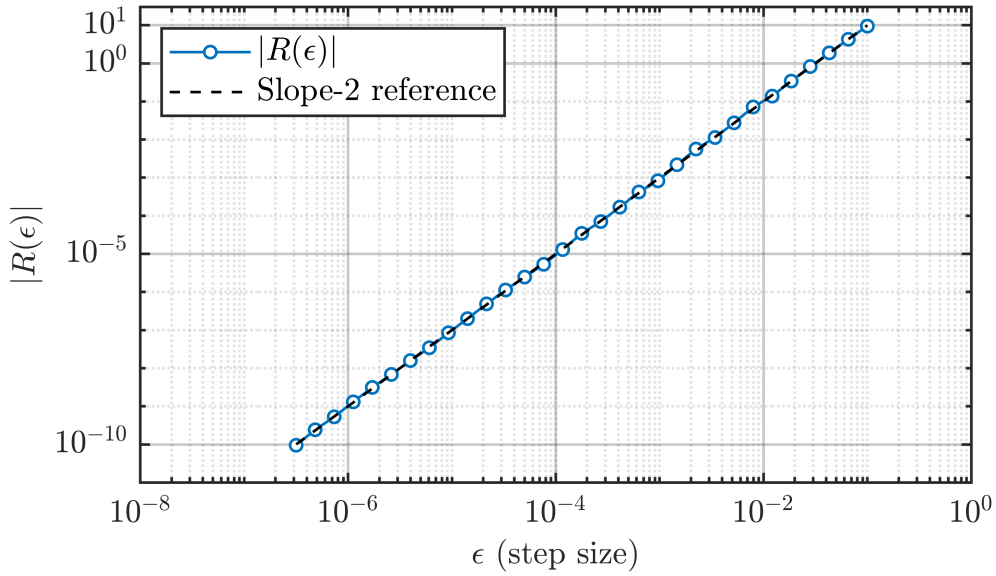

Figure 4.1.: Log–log plot of the Taylor remainder $|R(\epsilon)|$ and the corresponding slope-two reference.

where $\boldsymbol{m}_h$ and $\boldsymbol{v}_h$ are the first- and second-moment estimates, respectively, and $\odot$ denotes element-wise multiplication. The bias-corrected moment estimates are given by

$$\hat{\boldsymbol{m}}_h = \frac{\boldsymbol{m}_h}{1-\beta_1^h}, \tag{4.44}$$

$$\hat{\boldsymbol{v}}_h = \frac{\boldsymbol{v}_h}{1-\beta_2^h}. \tag{4.45}$$

The parameter update is then computed as

$$\boldsymbol{\lambda}_{h+1} = \boldsymbol{\lambda}_h - \eta \frac{\hat{\boldsymbol{m}}_h}{\sqrt{\hat{\boldsymbol{v}}_h} + \epsilon}, \tag{4.46}$$

where η is the learning rate and ϵ is a small constant used for numerical stability.

4.7. Implementation Pipeline

This section provides an overview of the pipeline used to solve the inertial-parameter identification problem. It connects the formulation developed in the previous sections with its implementation. Figure 4.2 summarizes this workflow.

The required signals are the joint positions, joint velocities, joint accelerations, and joint torques. Depending on the robot, some of these quantities may not be directly available. If joint velocities or accelerations are not provided, they must be obtained through numerical differentiation of the measured position or velocity signals. If joint torque sensors are not

available, the joint torques may need to be estimated from the motor currents. After data collection, the signals should be preprocessed to reduce the influence of measurement noise. In this work, window averaging is used for this purpose. Next, the processed dataset is divided into training, validation, and test sets.

The CAD inertial parameters are mapped from the standard parameter vector π to the physically consistent parameter vector λ. This vector provides a good initial guess to start the optimization.

For every sample, the kinematic quantities required by the inverse dynamics model, including the body Jacobians, their time derivatives, body velocities, and body accelerations, are computed once and stored. These quantities depend only on the measured robot motion and are therefore precomputed before the optimization.

It is worth noting that the optimization is not performed in a classical epoch-based manner over all training samples. Instead, each iteration uses a randomly selected mini-batch from the dataset. The estimated torques, the loss function, and the analytical gradient with respect to λ are then evaluated for this batch. This was found to be sufficient while significantly reducing the computational cost, since the derivatives with respect to the inertial parameters are evaluated only for a subset of samples.

The parameter vector is updated using a gradient-based optimization algorithm, such as Adam; however, other suitable first-order optimization methods can also be used. The validation loss is evaluated periodically. The parameter vector yielding the lowest validation loss is retained. The optimization terminates when the maximum number of iterations is reached or when the validation loss does not improve for a specified patience period.

The best parameter vector can be mapped back to the standard inertial parameter vector π. Finally, the generalization capability of the identified model is evaluated on an independent test dataset by comparing the predicted joint torques with the measured joint torques.

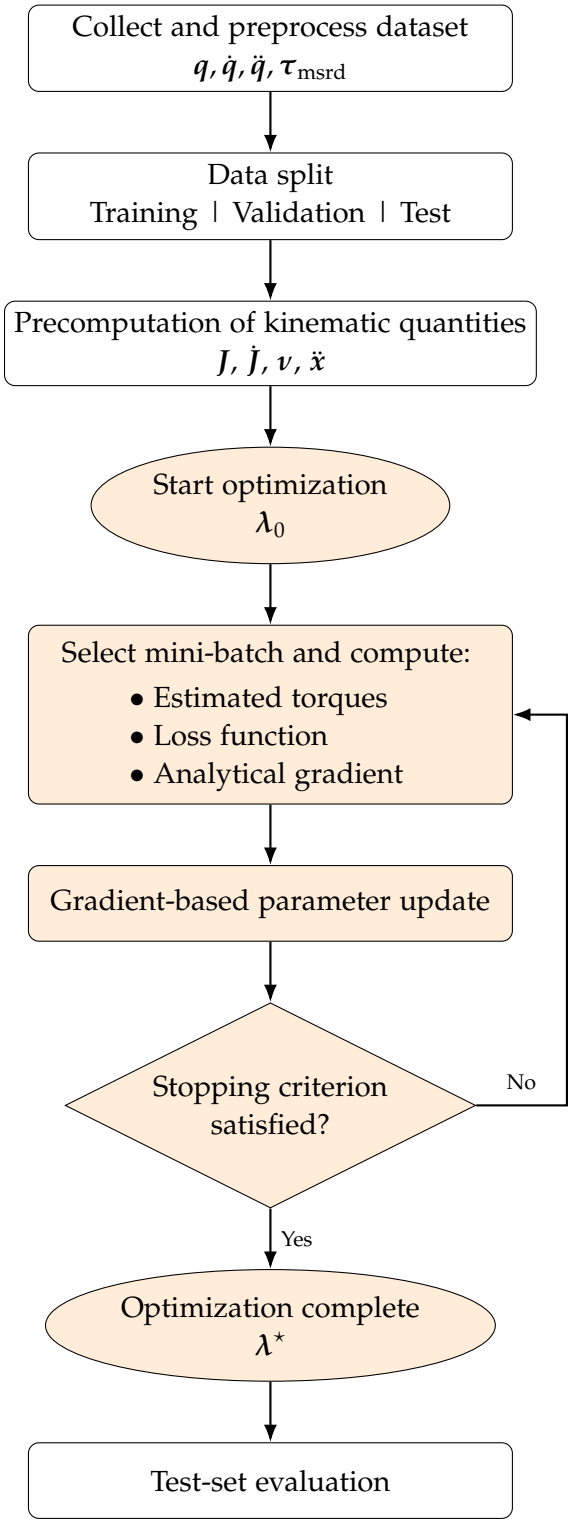

Figure 4.2.: Gradient-based inertial parameter identification workflow. Shaded elements represent the iterative optimization algorithm.

5. Experimental Evaluation and Results

This chapter evaluates the potential and limitations of the proposed gradient-based inertial-parameter identification algorithm in both simulation and real-robot experiments. The simulation studies provide a controlled environment, allowing convergence behavior, robustness, and physical consistency of the proposed method to be analyzed. Different experimental conditions are considered, including ideal noise-free data, different training trajectories, additive torque noise, and a comparison against the PC-IDIM-OLS method.

After the simulation analysis, the proposed method is evaluated using measured data from the real FR3 robot. In this case, the true inertial parameters are not available, and the evaluation therefore focuses on the ability of the identified model to improve torque prediction on unseen trajectories. The real-robot experiments evaluate whether the proposed algorithm can fine-tune CAD inertial parameters and adapt to changes in the robot dynamics caused by an unknown payload, while preserving physical consistency and generalizing beyond the trajectory used for training.

To obtain more statistically meaningful results, the experiments are repeated over multiple Monte Carlo trials. The main evaluation criteria considered in these experiments are:

- **Physical Consistency:** The percentage of trials in which the identified parameters correspond to a fully physically consistent set of inertial parameters ,according to the conditions described in Section 3.3.
- **Number of iterations:** The number of iterations required by the algorithm to reach convergence is recorded. Convergence is determined according to two stopping criteria. First, early stopping is applied when the validation error does not improve for a prescribed number of consecutive iterations, referred to as the patience parameter. Second, a maximum number of iterations is manually imposed to prevent excessively long runs.
- **Computation time:** The average time required for the algorithm to run and converge is measured. For reference, all simulations were performed on a laptop equipped with an Intel Core i7 processor and 16 GB of RAM.
- **Joint-wise relative torque prediction error:** The torque prediction error is evaluated on an unseen test trajectory with r samples, which was not used during training or validation. For each joint j, the error is normalized by the measured torque norm and

computed as

$$e_j = 100 \frac{\sqrt{\sum_{i=1}^{r} \left(\tau_{j,i}^{\text{true}} - \tau_{j,i}^{\text{est}}\right)^2}}{\sqrt{\sum_{i=1}^{r} \left(\tau_{j,i}^{\text{true}}\right)^2}}. \tag{5.1}$$

The resulting value is reported in percent and represents the torque prediction error normalized by the measured torque magnitude of the corresponding joint.

For these experiments, the initial parameter vector for Monte Carlo trial (i) is generated as

$$\boldsymbol{\lambda}_0^{(i)} = \boldsymbol{\lambda}_{true} + p\, \boldsymbol{\lambda}_{true} \boldsymbol{\epsilon}^{(i)}, \qquad \boldsymbol{\epsilon}^{(i)} \sim \mathcal{N}(0, 1). \tag{5.2}$$

This perturbation strategy is suitable for the Monte Carlo analysis because it generates statistically independent initial guesses centered around the true parameter vector. The scalar p defines the relative perturbation level.

5.1. Simulation

In the following section, the proposed inertial parameter identification algorithm is evaluated in a MATLAB simulation environment. In particular, the experiments are designed to evaluate:

1. The behavior of the algorithm under ideal conditions.
2. The influence of the trajectories used during training.
3. The robustness of the algorithm when torque noise is introduced, making the simulation closer to a realistic experimental setting.
4. The comparison with PC-IDIM-OLS as another physically consistent identification method.

Because the simulation environment provides access to the true inertial parameters, the following evaluation criteria are considered for the simulation experiments:

- **AIRM:** The AIRM distance can be used to compare the generalized mass matrix obtained from the identified parameters with the one obtained from the true parameters. This metric is described in Section 2.5.2.

$$d_{\text{AIRM}}(\boldsymbol{M}_{true}, \boldsymbol{M}_{est}) \tag{5.3}$$

- **Frobenius norm:** Similarly, the Frobenius norm is used to quantify the difference between the true and the identified generalized mass matrices. This metric is described in Section 2.5.1.

$$\|\boldsymbol{M}_{true} - \boldsymbol{M}_{est}\|_F \tag{5.4}$$

For reproducibility, the hyperparameters used in this section are summarized in Table 5.1.

Table 5.1.: Optimization hyperparameters used in the Monte Carlo simulation experiments.

Hyperparameter	Value
Optimizer	Adam (Sec. 4.6)
Adam first-moment decay, β_1	0.9
Adam second-moment decay, β_2	0.999
Adam numerical constant, ϵ	1×10^{-8}
Learning rate, η	1×10^{-3}
Mini-batch size	64 samples
Maximum number of iterations	1500
Early-stopping patience	25 iterations
Validation frequency	Every 5 iterations

Unless stated otherwise, the training trajectory used in the simulation experiments corresponds to the best optimized excitation trajectory obtained in Sec. 3.4. Its joint positions, velocities, accelerations, and torques are shown in Fig. A.2, and the corresponding end-effector motion is shown in Fig. 3.3. The validation and test trajectories are selected as the trajectories with the second- and third-lowest optimization losses, respectively.

5.1.1. Ideal Case

The first simulation study evaluates the potential of the proposed method under ideal conditions. In this experiment, the joint positions, velocities, and accelerations are assumed to be noise-free, and the joint torques are computed directly from the true robot model.

The identification procedure was repeated over three Monte Carlo experiments, each consisting of 150 independent trials with different initial parameter guesses. Three perturbation levels were considered, $p \in \{0.05, 0.15, 0.25\}$. Typically, an initial estimate is available from CAD inertial parameters. Therefore, a perturbation level of 5% is considered representative of a realistic starting point. When external hardware, grippers, or other accessories are attached to the robot, a moderate perturbation level of 15% is more appropriate. Finally, a perturbation level of 25% is considered an extreme case, used to evaluate the robustness of the method under poor initialization.

As shown in Table 5.2 , as the perturbation level increases, the final AIRM distance, the Frobenius norm, the number of iterations, and the computation time also increase. Nevertheless, the method consistently converges to physically consistent parameter estimates in 100% of the Monte Carlo trials.

Table 5.2.: Comparison of Monte Carlo simulation experiments under ideal conditions for different initialization perturbation levels.

	$p = 0.05$	$p = 0.15$	$p = 0.25$
Final AIRM	0.0288 ± 0.0166	0.0402 ± 0.0409	0.1076 ± 0.1010
Final Frobenius norm	0.0005 ± 0.0002	0.0015 ± 0.0014	0.0044 ± 0.0034
Avg. iterations	116.53 ± 64.21	690.22 ± 323.22	766.03 ± 446.84
Avg. time	8.8 ± 5.1 s	57.4 ± 27.3 s	62.2 ± 37.95 s
Physical consistency	100%	100%	100%

The joint-wise relative torque error in Table 5.3 remains low for all joints, indicating that the identified models reproduce the inverse dynamics accurately. Since the reported errors are evaluated on a test trajectory that was not used during optimization, they indicate that the identified parameters generalize beyond the training trajectory rather than only fitting it.

Table 5.3.: Joint-wise relative torque prediction error in percent averaged over Monte Carlo trials for different initialization perturbation levels.

Joint	$p = 0.05$ (%)	$p = 0.15$ (%)	$p = 0.25$ (%)
1	0.06 ± 0.02	0.11 ± 0.13	0.31 ± 0.34
2	0.06 ± 0.02	0.15 ± 0.09	0.33 ± 0.23
3	0.05 ± 0.02	0.10 ± 0.14	0.28 ± 0.34
4	0.07 ± 0.02	0.14 ± 0.20	0.42 ± 0.51
5	0.30 ± 0.11	0.58 ± 0.79	1.76 ± 2.32
6	0.10 ± 0.04	0.16 ± 0.17	0.44 ± 0.48
7	0.48 ± 0.23	0.61 ± 0.81	1.50 ± 1.33

Although the identified parameters do not necessarily match the true parameters, they can still describe a dynamically equivalent system. Figures 5.1–5.3 show that the final AIRM distance decreases during optimization for all perturbation levels. This indicates that the proposed algorithm identifies parameters whose generalized mass matrix $\boldsymbol{M}$ is close to the one obtained from the true inertial parameters.

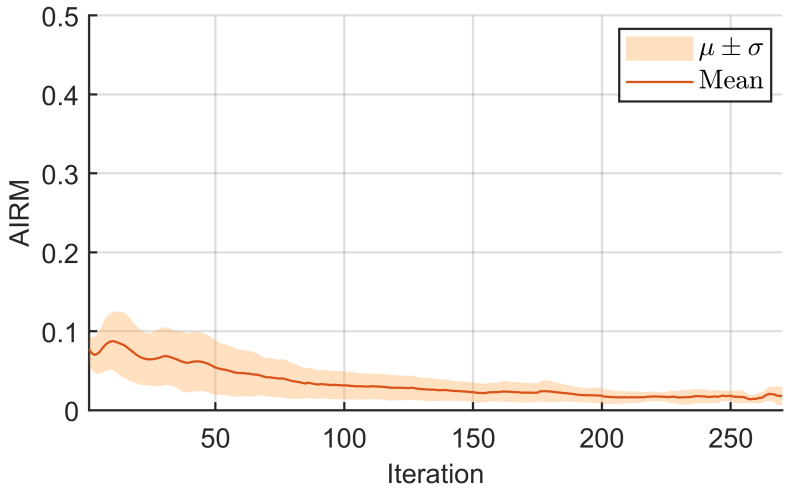

Figure 5.1.: Evolution of the average AIRM distance during training for an initialization perturbation level of $p = 0.05$, averaged over 150 Monte Carlo trials. The shaded band represents one standard deviation around the mean.

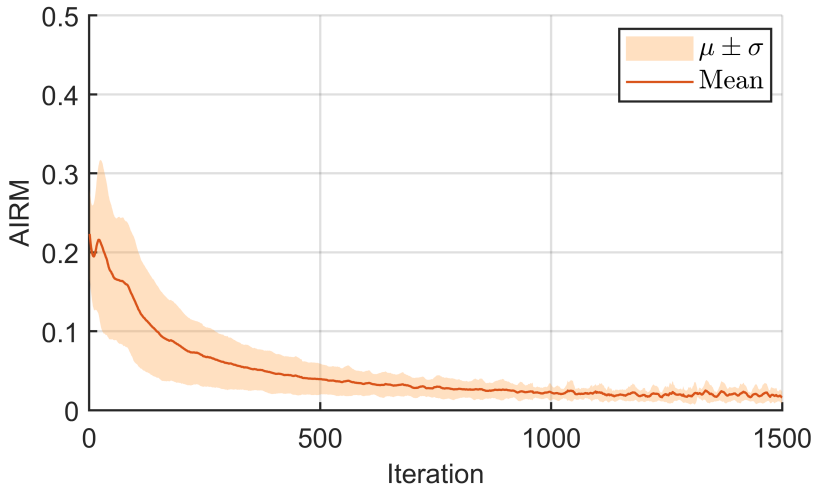

Figure 5.2.: Average AIRM distance during training for $p = 0.15$. Compared with the $p = 0.05$ case, the method requires more iterations but still converges to a low AIRM distance.

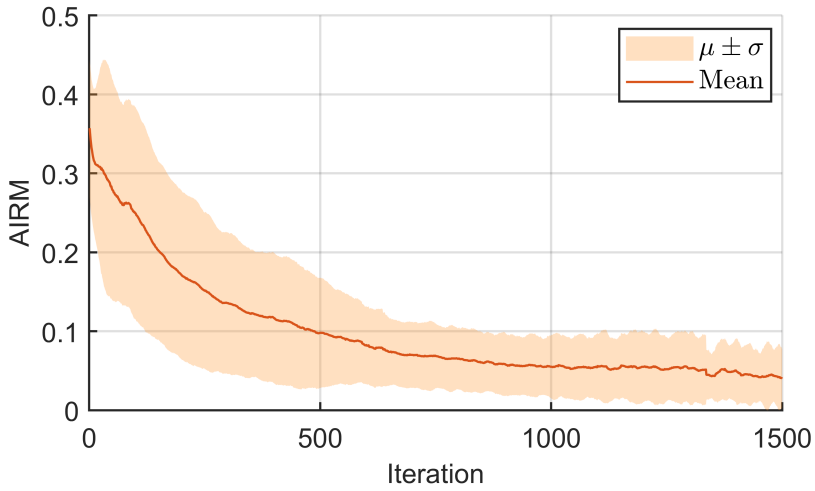

Figure 5.3.: Average AIRM distance during training for $p = 0.25$. The larger initial perturbation leads to a higher initial error and a wider standard-deviation band, but the average AIRM distance decreases consistently over the optimization.

In summary, the ideal-case experiment shows that the proposed method behaves as expected under controlled conditions. The results also show a clear dependence on the initialization. Nevertheless, the AIRM curves indicate that the method consistently moves toward dynamically equivalent inertial parameters. This study should be interpreted as a best-case validation. More realistic conditions are considered in the following sections.

5.1.2. Influence of the Training Trajectory

According to the literature, the choice of training trajectory plays an important role in the parameter identification process, as discussed in Sec. 3.4. In classical regressor-based identification, optimized excitation trajectories are typically used to improve the conditioning of the observation matrix and to make the influence of the inertial parameters more visible in the measured data. Although the proposed approach does not explicitly rely on the classical regressor formulation during the optimization, the training trajectory may still affect the convergence behavior. This simulation experiment investigates whether optimized excitation trajectories provide a measurable advantage over simpler sinusoidal trajectories.

The results obtained in the previous section using the optimal Fourier-based trajectory and a moderate perturbation level of $p = 0.15$ are compared with those obtained using simple sinusoidal trajectories. The validation and test trajectories were kept the same to ensure a fair comparison. The training trajectories include random phase shifts $\boldsymbol{\phi}$ and are generated as

$$\boldsymbol{q}(t) = \boldsymbol{a} + \boldsymbol{b}\sin(2t + \boldsymbol{\phi}), \tag{5.5}$$

$$\dot{\boldsymbol{q}}(t) = 2\boldsymbol{b}\cos(2t + \boldsymbol{\phi}), \tag{5.6}$$

$$\ddot{\boldsymbol{q}}(t) = -4\boldsymbol{b}\sin(2t + \boldsymbol{\phi}). \tag{5.7}$$

The corresponding joint positions, velocities, accelerations, and torques are shown in Fig. A.1.

Table 5.4 summarizes the results obtained with the sinusoidal and optimized trajectories. Both experiments achieve full physical consistency, and the final AIRM distances are very similar. The optimized trajectory leads to a slightly lower Frobenius norm, while the sinusoidal trajectory requires slightly fewer iterations and less computation time. However, the differences between both cases are small.

Table 5.4.: Comparison between sinusoidal and optimized training trajectories under ideal conditions for $p = 0.15$.

	Sinusoidal Trajectory	Optimized Trajectory
Final AIRM	0.0394 ± 0.0301	0.0402 ± 0.0409
Final Frobenius norm	0.0018 ± 0.0017	0.0015 ± 0.0014
Avg. iterations	675.5 ± 294.6	690.22 ± 323.22
Avg. time	55.8 ± 24.9 s	57.4 ± 27.3 s
Physical consistency	100%	100%

The joint-wise relative torque error in Table 5.5 shows that both trajectories lead to comparable prediction accuracy on the test trajectory. The optimized trajectory gives slightly lower errors for some joints, while the sinusoidal trajectory performs similarly for others.

Table 5.5.: Comparison of the joint-wise relative torque prediction error in percent obtained with sinusoidal and optimized training trajectories under ideal conditions for $p = 0.15$. Values are averaged over Monte Carlo trials.

Joint	Sinusoidal trajectory (%)	Optimized trajectory (%)
1	0.15 ± 0.06	0.11 ± 0.13
2	0.14 ± 0.05	0.15 ± 0.09
3	0.12 ± 0.05	0.10 ± 0.14
4	0.20 ± 0.09	0.14 ± 0.20
5	0.59 ± 0.31	0.58 ± 0.79
6	0.10 ± 0.04	0.16 ± 0.17
7	0.63 ± 0.38	0.61 ± 0.81

This experiment indicates that optimized excitation trajectories do not provide a major advantage over simple trajectories in the ideal simulation setting. However, under more realistic conditions, the excitation quality of the training trajectory may become more relevant.

5.1.3. Robustness against Noise

This section evaluates the robustness of the proposed method under different levels of noise. Since the optimization objective is formulated by minimizing the torque prediction error, noise is injected only into the torque measurements, while the joint positions, velocities, and

accelerations are kept unchanged. This allows the effect of corrupted torque to be isolated. The noisy torque vector for Monte Carlo trial (i) is generated as

$$\boldsymbol{\tau}_{\text{noisy}}^{(i)} = \boldsymbol{\tau} + \sigma_\tau \epsilon^{(i)}, \qquad \epsilon^{(i)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{5.8}$$

Three noise levels are considered in this study. The standard case uses a torque noise standard deviation of $\sigma_\tau = 0.05\,\text{Nm}$, which is the same noise level used in BIRDy, an open-source MATLAB toolbox for robot inertial-parameter identification [20]. In addition, a lower noise level of $\sigma_\tau = 0.01\,\text{Nm}$ and a severe noise level of $\sigma_\tau = 0.1\,\text{Nm}$ are considered to analyze robustness under different conditions. The initial parameter vector for all Monte Carlo trials is randomly perturbed using a perturbation level of $p = 0.15$, following Eq. 5.2.

Table 5.6.: Simulation results of the Monte Carlo experiments for different torque-noise levels.

	$\sigma_\tau = 0.01\,\text{Nm}$	$\sigma_\tau = 0.05$ Nm	$\sigma_\tau = 0.1\,\text{Nm}$
Final AIRM	0.2298 ± 0.0728	0.2540 ± 0.1505	0.2936 ± 0.1505
Final Frobenius norm	0.0041 ± 0.0012	0.0037 ± 0.0012	0.0056 ± 0.0015
Avg. iterations	471.06 ± 85.74	425.06 ± 43.77	318.43 ± 136.52
Avg. time	30.41 ± 13.4703 s	23.45 ± 7.38 s	18.63 ± 7.98 s
Physical consistency	100%	100%	100%

The results, summarized in Table 5.6 , show that the proposed method remains physically consistent for all noise levels. As the torque noise increases, both the final AIRM distance and the Frobenius norm increase, especially for the severe noise case with $\sigma_\tau = 0.1\,\text{Nm}$.

The critical factor is not only the absolute noise magnitude, but also its size relative to the torque signal. When the measured torque is small, noise can dominate the signal and lead the optimization to fit noise instead of the true dynamics. This is especially relevant for joint 7, whose measured torque lies only in the range $[-0.1193, 0.1231]$ Nm. Since the gradient contribution of a distal link can also affect the updates of parent-link parameters, as discussed in Sec. 4.4, noise-dominated measurements can propagate through the optimization and degrade convergence.

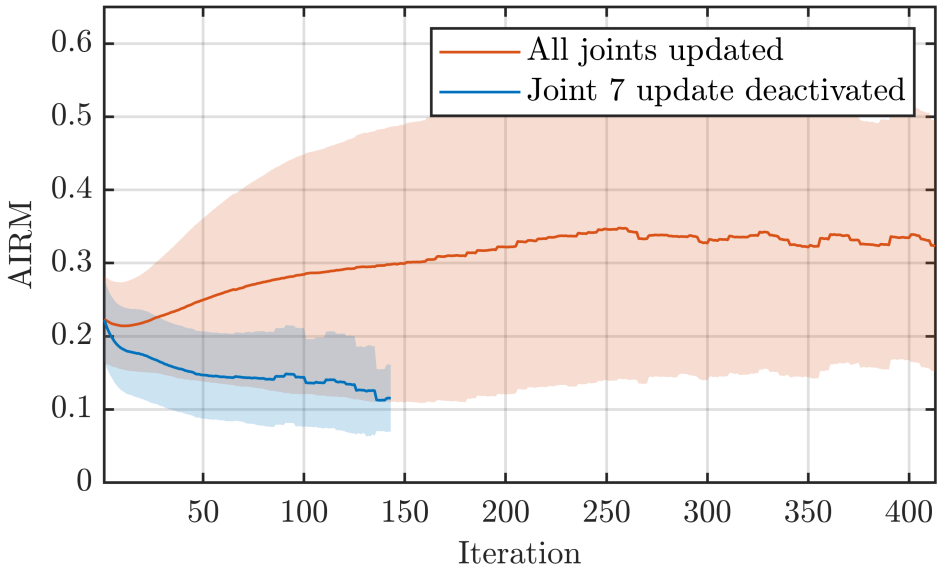

Figure 5.4.: Average AIRM distance during training for the third Monte Carlo trial with $\sigma_\tau = 0.1\,\mathrm{Nm}$. The full parameter update is compared with a modified update in which the joint 7 update is deactivated. This leads to improved convergence behavior and final accuracy.

Figure 5.4 illustrates this effect by comparing two minimization cases: a full parameter update, in which all 70 parameters are optimized, and a modified update, in which the gradient-based update associated with link 7 is deactivated and the corresponding diagonal element of the weighting matrix $\mathbf{W}$ is set to zero. The comparison shows that excluding unreliable gradient information can improve convergence and reduce the final AIRM distance.

Table 5.7.: Effect of torque-noise level on the joint-wise relative torque prediction error in percent in simulation, averaged over Monte Carlo trials.

Joint	$\sigma_\tau = 0.01$ Nm (%)	$\sigma_\tau = 0.05\,\mathrm{Nm}$ (%)	$\sigma_\tau = 0.1\,\mathrm{Nm}$ (%)
1	0.41 ± 0.13	0.52 ± 0.18	0.60 ± 0.17
2	0.33 ± 0.08	0.38 ± 0.11	0.45 ± 0.12
3	0.39 ± 0.12	0.50 ± 0.19	0.59 ± 0.20
4	0.57 ± 0.19	0.76 ± 0.240	0.90 ± 0.28
5	3.38 ± 0.81	3.02 ± 1.50	4.80 ± 1.79
6	0.69 ± 0.22	0.81 ± 0.29	1.07 ± 0.240
7	2.85 ± 0.23	3.58 ± 1.19	4.30 ± 1.68

Table 5.7 shows that the proposed method remains robust when torque noise is introduced during identification. For most joints, the relative torque prediction error stays below 1% across the tested noise levels, indicating good generalization to the unseen test trajectory.

Even in the severe-noise case with $\sigma_\tau = 0.1\,\mathrm{Nm}$, the errors remain bounded. These results also highlight the importance of signal preprocessing, since filtering or window averaging can improve gradient reliability and make the optimization more stable.

5.2. Real-Robot Results

After simulation, the proposed method is tested using data collected from the real FR3 robot. The evaluation focuses primarily on torque prediction performance over unseen trajectories.

The real-robot dataset consists of measured joint positions, joint velocities, and joint torques recorded from the FR3 manipulator. Friction is not modeled explicitly in this work. Joint accelerations are obtained by numerically differentiating the measured joint velocities and are preprocessed before being used. Because both numerical differentiation and torque measurements are sensitive to noise, the measured signals are filtered using a window-averaging method before parameter identification.

The proposed method is initialized from the CAD inertial parameters and its performance is evaluated by comparing the predicted torques against the measured torques on both trajectories. The hyperparameters used in this section are summarized in Table 5.8.

Table 5.8.: Optimization hyperparameters used for the real-robot data experiments.

Hyperparameter	Value
Optimizer	Adam (Sec. 4.6)
Adam first-moment decay, β_1	0.9
Adam second-moment decay, β_2	0.999
Adam numerical constant, ϵ	1×10^{-8}
Learning rate, η	1×10^{-4}
Mini-batch size	64 samples
Maximum number of iterations	2500
Early-stopping patience	25 iterations
Validation frequency	Every 5 iterations

5.2.1. Inertial Parameter Fine-Tuning

In the following section, the proposed algorithm is evaluated as a fine-tuning procedure for the CAD inertial parameters. The optimization is initialized from the CAD inertial parameters, which are slightly perturbed with $p = 0.05$ according to Eq. 5.2. The objective is to tune the inertial parameters such that the predicted joint torques better match the measured torques. This represents the typical practical use case, where an existing CAD-based model is available but requires adjustment. The training trajectory, shown in Fig. A.3, is used for parameter identification.

Across the Monte Carlo trials, the method preserves physical consistency in 100% of the cases. On average, convergence is reached after 192.3 iterations, corresponding to 32.74 seconds.

Fig. 5.5 compares the joint-wise relative torque prediction error before and after optimization. For all optimized joints, the average error either decreases or remains approximately unchanged. The larger relative errors for joints 5 and 7 should be interpreted carefully, since these joints have small measured torque magnitudes. For example, joint 5 varies only between approximately -0.9973 Nm and 0.5671 Nm Therefore, even small absolute prediction errors can produce large relative errors after normalization.

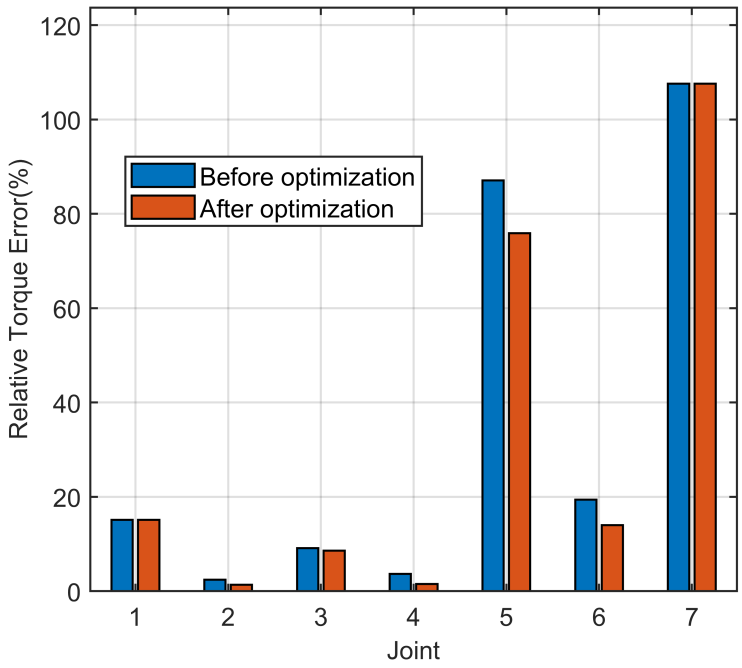

Figure 5.5.: Joint-wise relative torque prediction error before and after optimization in the fine-tuning experiment. Bars show the mean over 150 Monte Carlo trials.

Table 5.9.: Joint-wise relative torque prediction error in percent before and after optimization in the parameter-tuning experiment.

Joint	Before optimization (%)	After optimization (%)
1	15.13 ± 0.07	15.12 ± 0.02
2	2.30 ± 0.69	1.32 ± 0.03
3	9.05 ± 0.27	8.59 ± 0.04
4	3.56 ± 0.97	1.50 ± 0.32
5	87.03 ± 0.90	75.94 ± 0.57
6	19.41 ± 0.84	14.00 ± 0.29
7	107.58 ± 0.00	107.58 ± 0.00

Table 5.9 shows that the optimization reduces the joint-wise relative torque prediction error

for most joints. The largest improvements are observed for joints 2, 4, 5, and 6. These results indicate that the proposed method can refine the CAD inertial parameters and improve torque prediction on unseen real-robot data without compromising physical consistency.

5.2.2. Payload Adaptation

An important application of inertial-parameter identification is payload adaptation. This is relevant when a new device is attached to the robot arm or when the robot manipulates objects with unknown inertial properties. In such cases, adapting the inertial parameters can improve the dynamic model of the combined robot-payload system. To evaluate this use case, an experiment was performed with an unknown payload attached to the end effector. The experimental setup is shown in Fig. 5.6 and the training trajectory is shown in Fig. A.4.

Figure 5.6.: Experimental setup for payload adaptation on the FR3 robot, with an unknown payload attached to the end effector.

The joint-wise relative torque prediction error in percent before and after optimization is shown in Fig. 5.7. The corresponding numerical values are reported in Table 5.10.

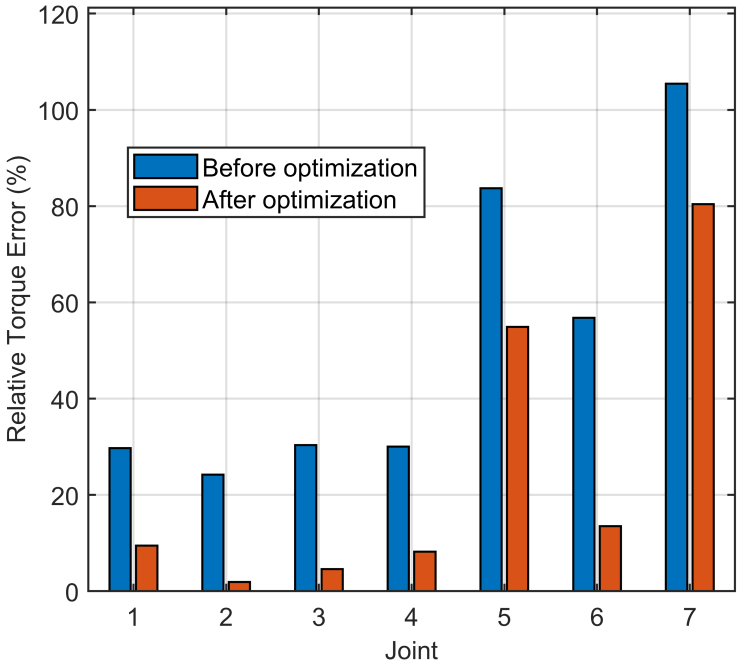

Figure 5.7.: Joint-wise relative torque prediction error before and after optimization in the unknown-payload adaptation experiment. Bars show the mean over 150 Monte Carlo trials.

Table 5.10.: Joint-wise relative torque prediction error in percent before and after optimization in the payload-adaptation experiment.

Joint	Before optimization (%)	After optimization (%)
1	29.72 ± 0.72	9.47 ± 0.17
2	24.22 ± 0.80	1.90 ± 0.37
3	30.26 ± 0.75	4.58 ± 0.09
4	30.02 ± 0.81	8.18 ± 1.43
5	83.69 ± 0.55	54.92 ± 0.92
6	56.81 ± 0.82	13.50 ± 1.99
7	105.44 ± 0.07	80.43 ± 0.10

The results show a clear reduction in the relative torque prediction error for all joints after optimization. This indicates that the proposed method can adapt the inertial parameters to the modified robot dynamics introduced by the unknown payload. The largest improvements are observed for joints 1–4 and 6, while joints 5 and 7 remain with larger errors. A similar behavior was also observed in the simulation section.

5.2.3. Comparison against PC-IDIM-OLS

In this section, the proposed algorithm is compared against PC-IDIM-OLS on the real-robot dataset. The comparison is performed for the two practical use cases considered in the previous sections.

Tables 5.11 and 5.12 compare the proposed method with PC-IDIM-OLS on the real-robot dataset. Both methods preserve physical consistency in all experiments. In the parameter-tuning case, both methods achieve comparable joint-wise relative torque prediction errors, while the proposed method requires considerably less computation time.

Table 5.11.: Comparison between the proposed method and PC-IDIM-OLS on the real-robot dataset for parameter tuning and payload adaptation.

		Our Approach	PC-IDIM-OLS
Parameter tuning	Avg. time	32.74 ± 5.37 s	78.28 s
	Avg. iterations	192.3 ± 27.26	–
	Physical consistency	100%	100%
Payload adaptation	Avg. time	194.00 ± 45.22 s	84.14 s
	Avg. iterations	1499.60 ± 318.33	–
	Physical consistency	100%	100%

The higher computation time of the proposed method in the payload-adaptation case can be explained by its initialization. The optimization is initialized from the CAD inertial parameters and therefore starts farther from a good parameter estimate, requiring more iterations to converge. In contrast, PC-IDIM-OLS solves a constrained least-squares problem directly and is not affected by the initialization.

Table 5.12.: Joint-wise relative torque prediction error in percent for the proposed method and PC-IDIM-OLS on the real-robot dataset. Results are shown for parameter tuning and payload adaptation.

Joint	Parameter tuning		Payload adaptation	
	Our approach (%)	PC-IDIM-OLS (%)	Our approach (%)	PC-IDIM-OLS (%)
1	15.12 ± 0.02	16.52	9.47 ± 0.17	9.23
2	1.32 ± 0.03	0.95	1.90 ± 0.37	1.35
3	4.48 ± 0.04	8.08	4.58 ± 0.09	4.49
4	1.50 ± 0.32	1.25	8.18 ± 1.43	1.37
5	75.94 ± 0.57	74.29	54.92 ± 0.92	50.30
6	14.00 ± 0.29	13.90	13.50 ± 1.99	8.54
7	107.58 ± 0.00	99.82	80.43 ± 0.10	95.04

Figure 5.8 provides a visual comparison of the joint-wise relative torque prediction error for the proposed method and PC-IDIM-OLS on the real-robot dataset. In both experiments, the two methods achieve comparable torque-prediction performance. For some joints, the

proposed method gives lower relative errors, while for others PC-IDIM-OLS performs slightly better.

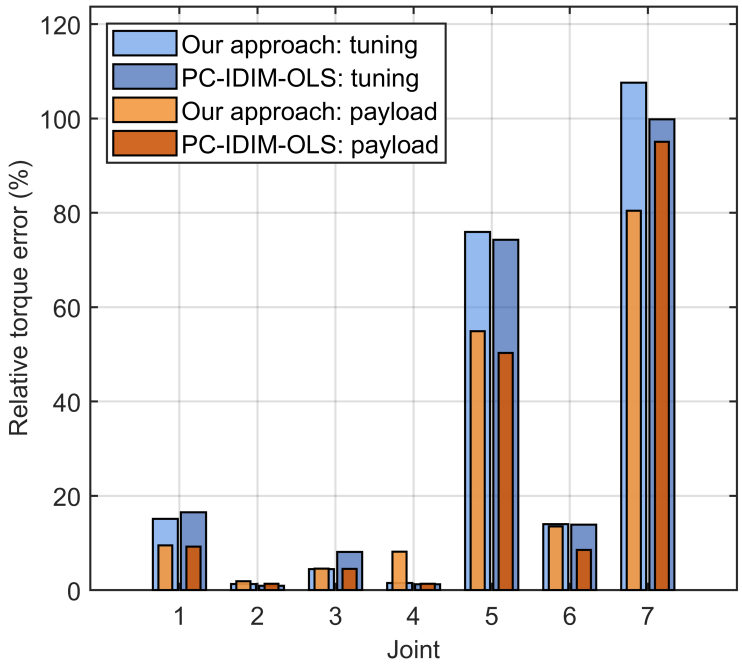

Figure 5.8.: Comparison of joint-wise relative torque prediction error in percent for the proposed method and PC-IDIM-OLS on the real-robot dataset.

The relative torque prediction error should be interpreted carefully. Joints 5 and 7 have small measured torque magnitudes, approximately in the range of -1 to $1\,\mathrm{Nm}$. Therefore, even small absolute torque errors can lead to large relative errors after normalization. In addition, when the torque signal is small, measurement noise can dominate the signal.

A similar consideration applies to joint 1. As discussed in Sec. 3.2, most of the inertial parameters of the first link do not affect the inverse dynamics model of the FR3. As a result, the optimizer has limited freedom to reduce the prediction error associated with joint 1, making large improvements more difficult to achieve.

6. Conclusion and Future Work

6.1. Conclusion

This thesis investigated whether physically consistent inertial parameters can be identified using an unconstrained gradient-based optimization method. The proposed parameterization guarantees positive masses and density-realizable inertia matrices by construction. Across all simulation and real-robot experiments, the optimization produced physically valid parameter sets in 100% of the experiments.

In the ideal noise-free simulation, the method preserved physical consistency in all Monte Carlo trials and achieved low joint-wise relative torque prediction errors on unseen test trajectories. The decreasing AIRM distance shows that the identified parameters produce a generalized mass matrix close to the one obtained from the true inertial parameters. This indicates that the algorithm not only fit the training torques, but also finds a dynamically equivalent model. These results highlight the potential of the proposed algorithm.

The noise experiments showed that the method is sensitive to the signal-to-noise ratio of the measured torques. Although physical consistency was preserved for all tested noise levels, low torque magnitudes combined with measurement noise produced unreliable gradients and degraded convergence. This was especially visible for joint 7, where disabling the corresponding parameter update improved the optimization behavior.

The real-robot experiments represent the main practical result of this thesis. Two practical use cases of inertial-parameter identification were considered: fine-tuning an existing CAD-based model and adapting the model to an unknown payload. In both cases, the proposed method improves the torque prediction performance, indicating that it can refine the robot model and adapt to changes in the robot dynamics.

Although both real-robot experiments show improved torque prediction, the remaining relative errors are still relatively high for high-precision model-based control. Based on the simulation results, this can be partly explained by the quality of the measured signals: in the ideal noise-free case, the method achieved much lower errors, while noisy torques, velocities, and accelerations degraded the optimization performance. This suggests that improved signal preprocessing could reduce the remaining errors and further improve generalization.

Overall, the proposed method represents a promising alternative to existing physically consistent inertial-parameter identification approaches, especially when physical validity should be guaranteed without relying on constrained optimization. The main findings can be summarized as follows:

- The proposed parameterization guarantees physically valid inertial parameters at every optimization step.

- In the absence of noise, the method identifies parameters whose generalized mass matrix is close to the one obtained from the true inertial parameters.
- The method showed limited dependence on optimized excitation trajectories, unlike classical identification methods.
- The method can refine CAD inertial parameters and improve torque prediction on unseen real-robot trajectories.
- The method can also adapt the inertial parameters to changes in the robot dynamics caused by a payload with unknown inertial properties.
- In the parameter-tuning experiment, the proposed method required less computation time than PC-IDIM-OLS. In the payload-adaptation experiment, however, it required more time due to the larger mismatch between the CAD initialization and the true robot-payload dynamics.
- The main limitations are the sensitivity to the quality of the measured signals, including joint torques, velocities, and accelerations, and the absence of explicit modeling of friction.

6.2. Future Work

Future work should address friction modeling, since friction is not captured by the inertial model and can explain part of the remaining torque-prediction error on real-robot data. A possible extension is to combine the proposed inertial parameter identification method with neural-network-based friction identification, for example by using one neural network per joint to learn the corresponding friction torque. This is motivated by the ability of neural networks to capture nonlinear friction behavior and learn relationships that may be difficult to describe with classical friction models.

A second direction is to improve the preprocessing strategy used before identification. The experiments showed that noisy measurements can produce unreliable gradients and degrade convergence behavior. Better filtering and differentiation methods are therefore expected to improve the identification performance.

Finally, the computational efficiency of the method could be further improved. Although the computational cost was reduced by precomputing the kinematic quantities before optimization, the evaluation of the gradients with respect to the inertial parameters remains one of the main bottlenecks. A C++ implementation and parallelized gradient computation could most likely make full-batch optimization feasible.

A. Appendix

This appendix provides additional plots of the trajectories and measured signals used in the simulation and real-robot experiments. It also includes selected analytical derivatives of the reparameterized inertia matrix. The derivatives with respect to w_1, w_2, w_3, a, b, and c are not listed explicitly, since their expanded symbolic expressions are very large. These derivatives are computed symbolically in MATLAB during the implementation.

Derivatives with Respect to μ and CoM Parameters

$$\frac{\partial^k \mathbf{M}_k}{\partial \mu_k} = e^{\mu_k} \begin{bmatrix} 1 & 0 & 0 & 0 & z_k & -y_k \\ 0 & 1 & 0 & -z_k & 0 & x_k \\ 0 & 0 & 1 & y_k & -x_k & 0 \\ 0 & -z_k & y_k & {y_k}^2 + {z_k}^2 & -x_k y_k & -x_k z_k \\ z_k & 0 & -x_k & -x_k y_k & {x_k}^2 + {z_k}^2 & -y_k z_k \\ -y_k & x_k & 0 & -x_k z_k & -y_k z_k & {x_k}^2 + {y_k}^2 \end{bmatrix}. \tag{A.1}$$

$$\frac{\partial^k \mathbf{M}_k}{\partial x_k} = e^{\mu_k} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -y_k & -z_k \\ 0 & 0 & -1 & -y_k & 2x_k & 0 \\ 0 & 1 & 0 & -z_k & 0 & 2x_k \end{bmatrix}. \tag{A.2}$$

$$\frac{\partial^k \mathbf{M}_k}{\partial y_k} = e^{\mu_k} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2y_k & -x_k & 0 \\ 0 & 0 & 0 & -x_k & 0 & -z_k \\ -1 & 0 & 0 & 0 & -z_k & 2y_k \end{bmatrix}. \tag{A.3}$$

$$\frac{\partial^k \mathbf{M}_k}{\partial z_k} = e^{\mu_k} \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 2z_k & 0 & -x_k \\ 1 & 0 & 0 & 0 & 2z_k & -y_k \\ 0 & 0 & 0 & -x_k & -y_k & 0 \end{bmatrix}. \tag{A.4}$$

Trajectories

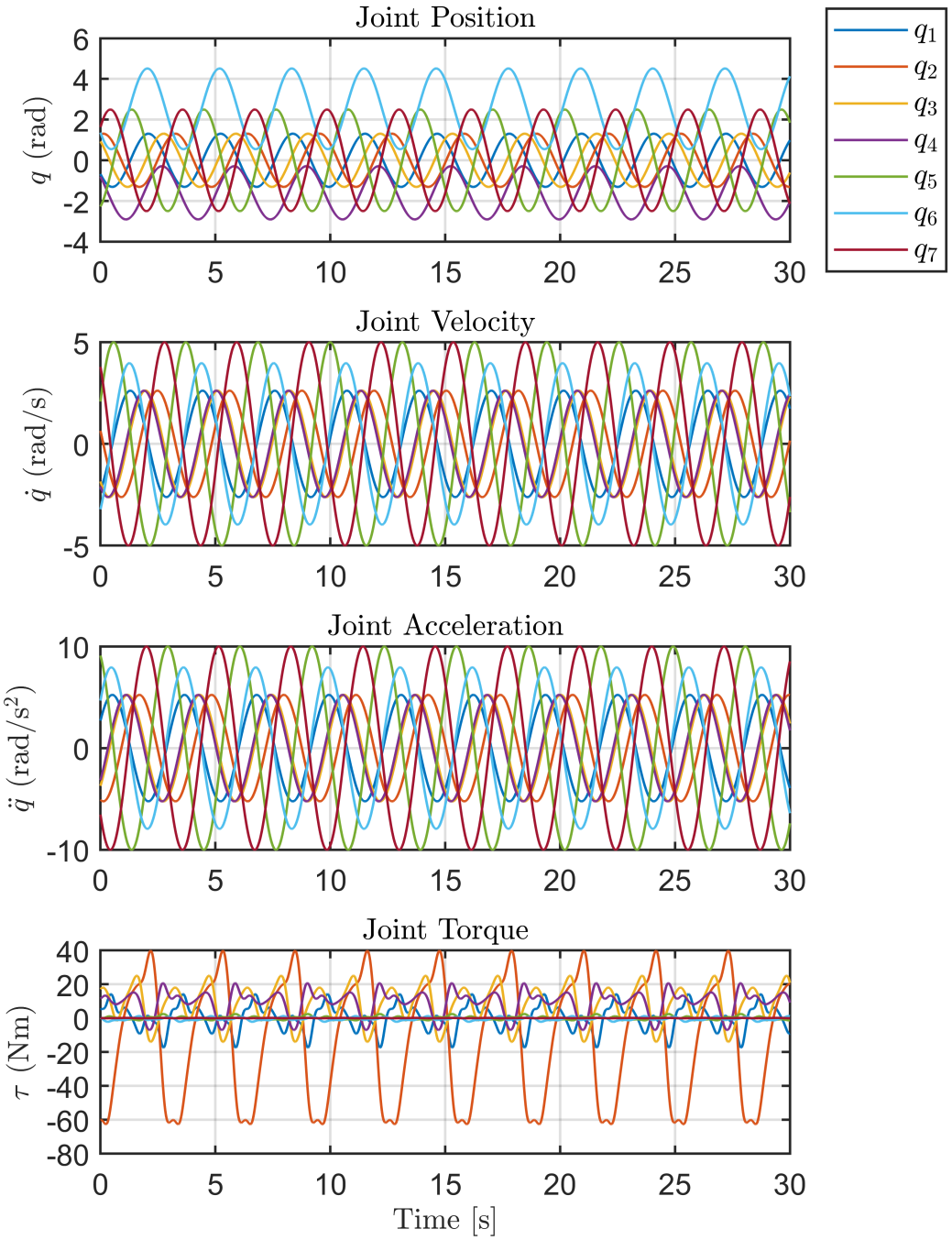

Figure A.1.: Sinusoidal joint positions, velocities, accelerations, and corresponding torques for all seven joints during the Monte Carlo simulations.

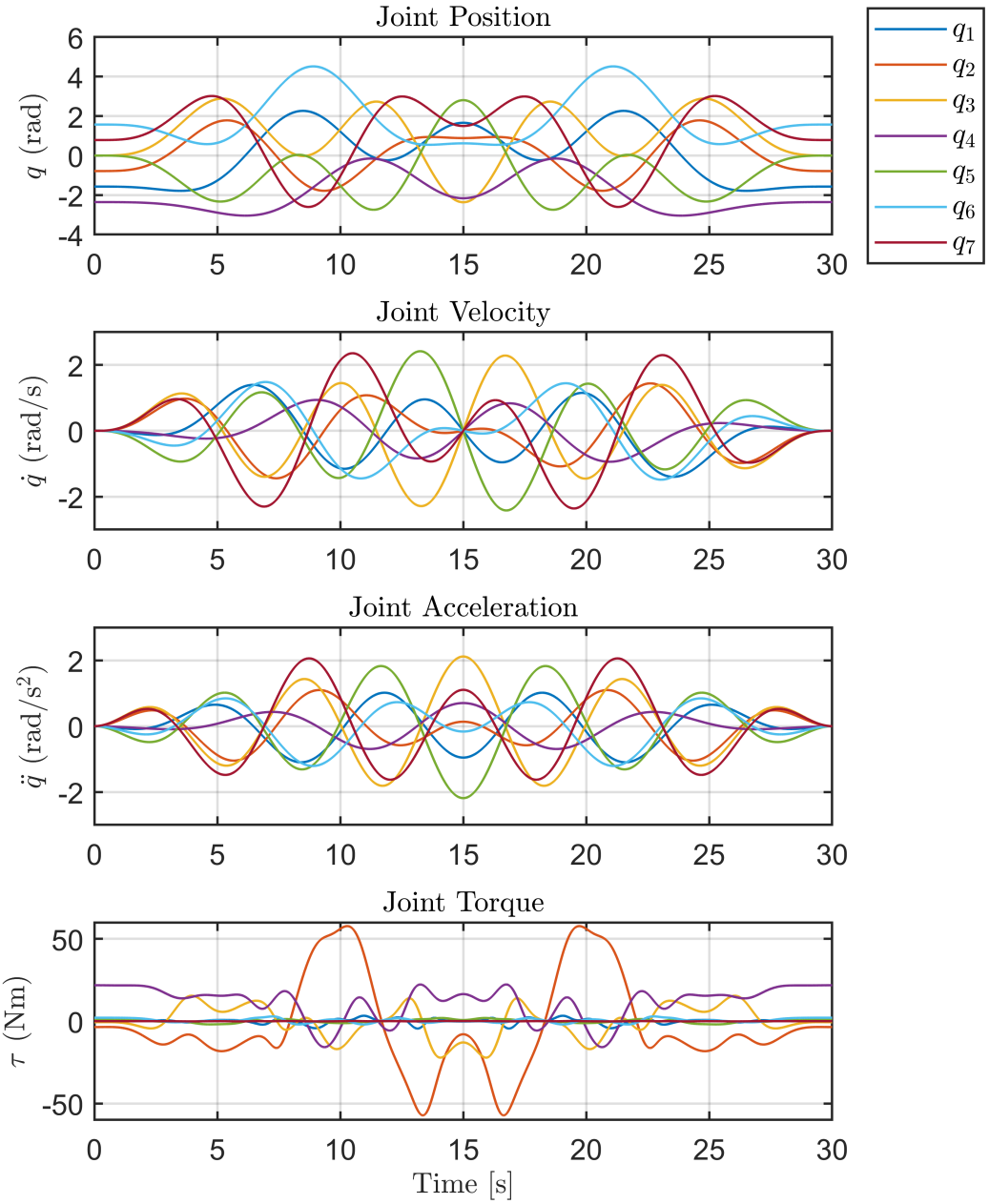

Figure A.2.: Joint positions, velocities, accelerations, and corresponding torques generated by the Fourier-based optimal trajectory for all seven joints.

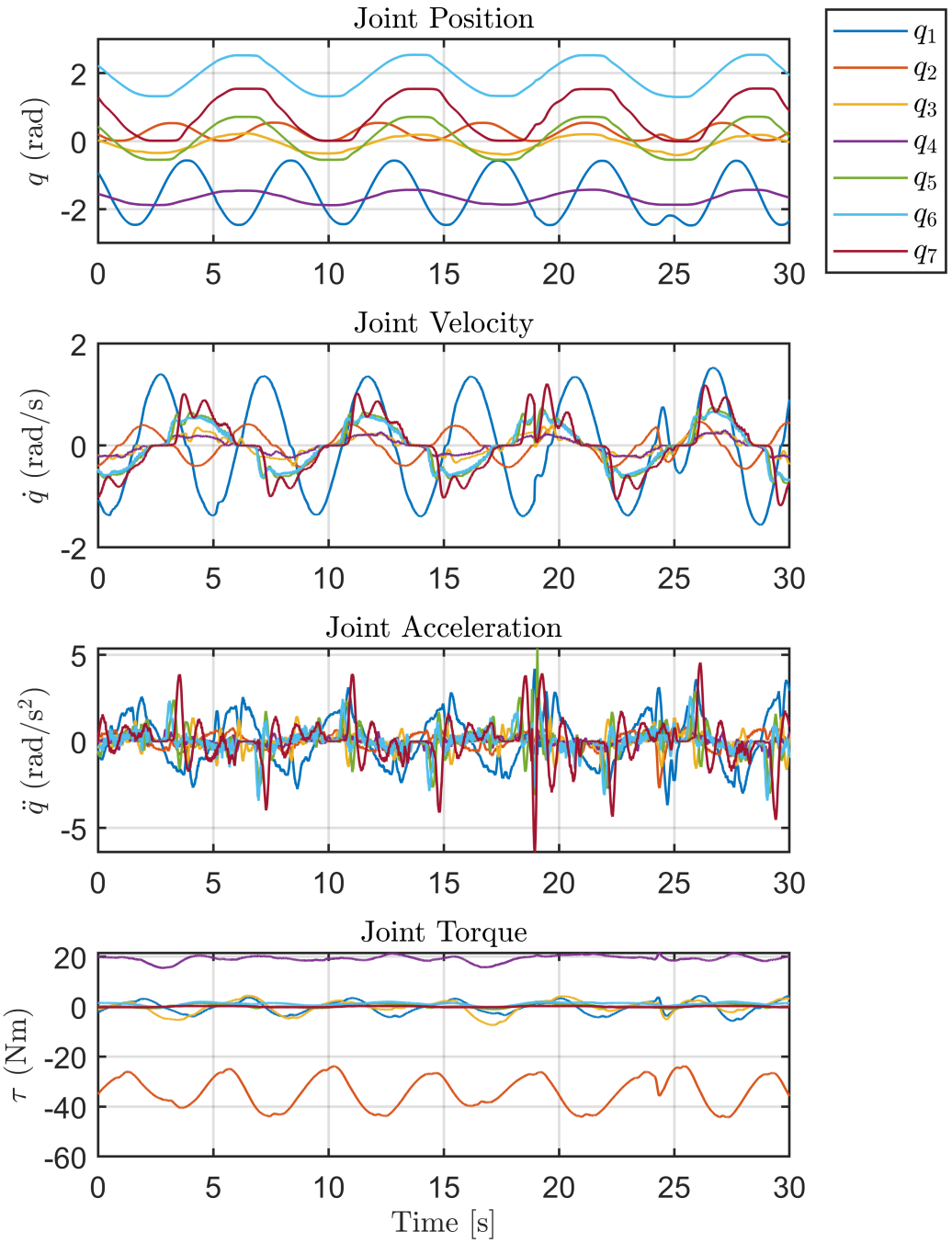

Figure A.3.: Measured joint positions, velocities, numerically differentiated accelerations, and measured joint torques for all seven joints of the real-robot training trajectory used in the parameter-tuning experiment.

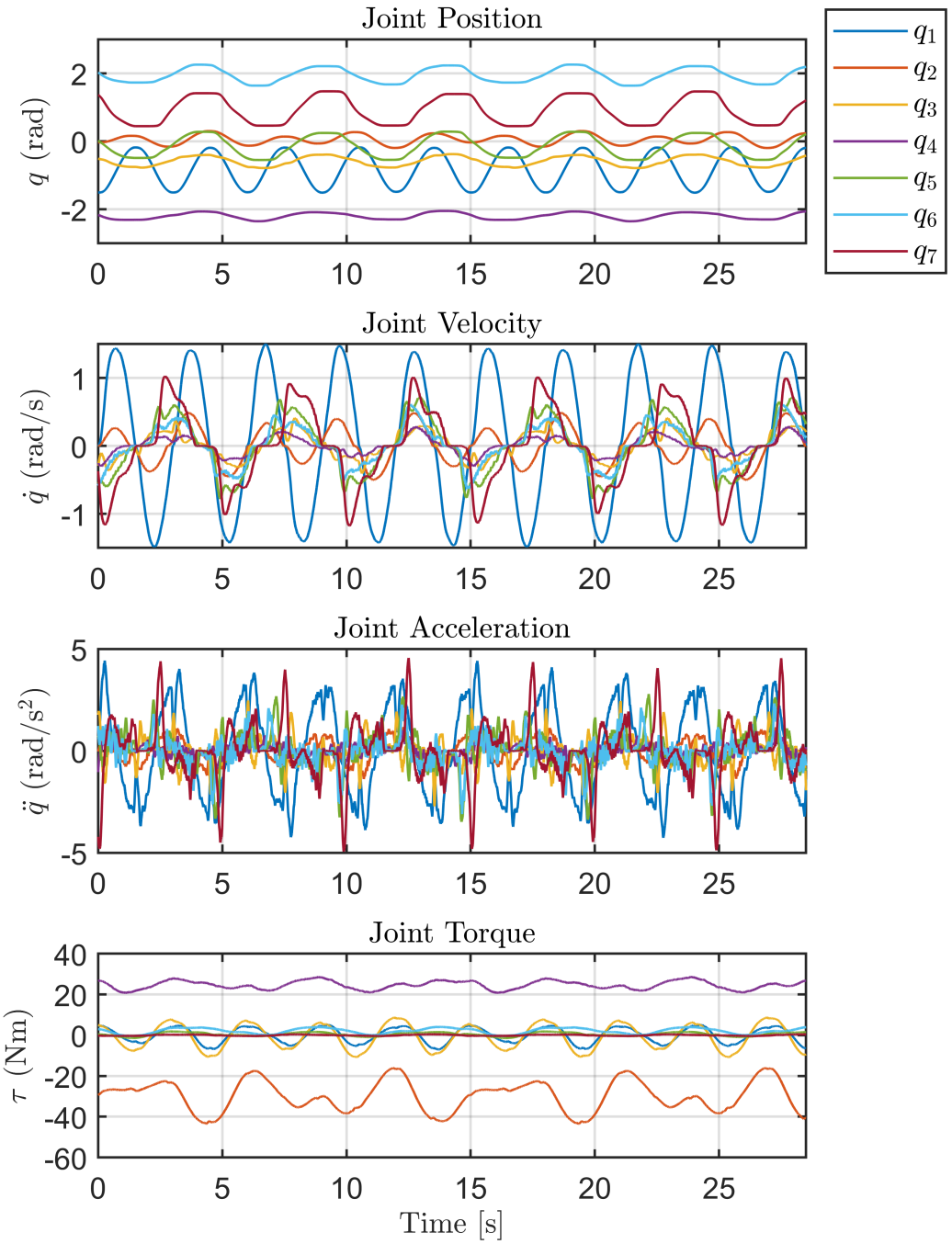

Figure A.4.: Measured joint positions, velocities, numerically differentiated accelerations, and measured joint torques for all seven joints of the real-robot training trajectory used in the payload-adaptation experiment.

B. Franka Research 3

The Franka Research 3 is a seven-degree-of-freedom collaborative robotic manipulator designed for robotics research. Its integrated joint torque sensors, together with joint position and velocity measurements from the control interface, provide the data required for the inertial-parameter identification experiments in this thesis.

This appendix collects the main parameters used to model the Franka Research 3. These include the modified Denavit–Hartenberg parameters, the standard inertial parameters extracted from the URDF model, joint limits and one base inertial parameter set obtained using the reduction procedure described in Section 3.2.

Figure B.1.: Franka Research 3 from Franka Robotics. Image source: Husarion [38].

Joint Limits and Default Configuration

	Def. Pos. (rad)	$q_{\min}$ (rad)	$q_{\max}$ (rad)
q_1	$-\pi/2$	- 2.744	2.744
q_2	$-\pi/4$	-1.784	1.784
q_3	0	-2.900	2.900
q_4	$-3\pi/4$	-3.042	-0.152
q_5	0	-2.806	2.806
q_6	$\pi/2$	0.5445	4.5169
q_7	$\pi/4$	-3.0159	3.016

Table B.1.: Default joint configuration and position limits of the Franka R3.

Joint	$\dot{q}_{\max}$ (rad/s)	$\ddot{q}_{\max}$ (rad/s^2)	$\tau_{\max}$ (Nm)
1	2.620	10	87
2	2.620	10	87
3	2.620	10	87
4	2.620	10	87
5	5.260	10	12
6	4.180	10	12
7	5.260	10	12

Table B.2.: Velocity, acceleration, and torque limits of the Franka R3 .

Modified Denavit–Hartenberg Parameters

Link	$\boldsymbol{\alpha_i}$	$\boldsymbol{a_i}$	$\boldsymbol{d_i}$
1	0	0	0.333
2	$-\pi/2$	0	0
3	$\pi/2$	0	0.316
4	$\pi/2$	0.0825	0
5	$-\pi/2$	-0.0825	0.384
6	$\pi/2$	0	0
7	$\pi/2$	0.088	0

Table B.3.: Modified Denavit–Hartenberg parameters of the Franka R3.

Standard Inertial Parameters

Parameter	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6	Link 7
m	2.9275	2.9355	2.2449	2.6156	2.3271	1.8170	1.2543
mX	0.0000	0.0093	0.0914	−0.1201	−0.0037	0.1085	0.0057
mY	−0.0531	−0.2182	−0.0108	0.1649	0.0681	−0.0746	0.0108
mZ	−0.1130	0.0259	−0.0650	−0.0223	−0.2264	−0.0185	−0.0203
XX	0.0293	0.0584	0.0260	0.0451	0.0756	0.0087	0.0009
YY	0.0268	0.0254	0.0253	0.0346	0.0699	0.0207	0.0008
ZZ	0.0073	0.0780	0.0228	0.0572	0.0184	0.0256	0.0003
XY	0.0000	0.0009	0.0029	0.0208	−0.0056	0.0107	−0.0001
XZ	−0.0001	0.0040	−0.0002	0.0091	−0.0039	0.0025	0.0002
YZ	−0.0040	−0.0023	−0.0024	0.0004	0.0173	−0.0021	0.0004

Table B.4.: Standard inertial parameters of the Franka R3.

Lambda Parameters

Parameter	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6	Link 7
μ	1.0741	1.0769	0.8087	0.9615	0.8446	0.5972	0.2266
x	0.0000	0.0032	0.0407	−0.0459	−0.0016	0.0597	0.0045
y	−0.0181	−0.0743	−0.0048	0.0630	0.0293	−0.0410	0.0086
z	−0.0386	0.0088	−0.0290	−0.0085	−0.0973	−0.0102	−0.0162
ω_1	−0.0778	−2.0174	0.0255	0.7494	−0.3782	1.1721	−0.8127
ω_2	−1.5610	−2.1220	−1.1784	0.8385	−1.3277	−0.1688	−0.9066
ω_3	0.1123	0.0762	0.7770	2.2615	−0.7734	−0.5045	−1.6587
a	0.1424	0.1994	0.1195	0.1906	0.2119	0.1259	0.0223
b	0.0605	0.1523	0.1116	0.1227	0.1142	0.0408	0.0071
c	0.0495	0.0379	0.0686	0.0307	0.0047	0.0165	0.0071

Table B.5.: Lambda parameters obtained by mapping the CAD inertial parameters of the Franka R3 from the standard parameter vector $\boldsymbol{\pi}$ to the physically consistent parameter vector $\boldsymbol{\lambda}$.

Base Inertial Parameters

It is worth noting that the base parameter set is not unique, since different regrouping strategies can lead to dynamically equivalent base parameter sets.

Parameter	Value	Parameter	Value	Parameter	Value
b_1	0.0327	b_{16}	−0.5655	b_{31}	−0.0543
b_2	0.0093	b_{17}	2.0115	b_{32}	−0.0210
b_3	−3.3950	b_{18}	0.6658	b_{33}	0.0361
b_4	0.0009	b_{19}	0.7860	b_{34}	0.0089
b_5	1.1412	b_{20}	0.1731	b_{35}	0.0025
b_6	0.0009	b_{21}	0.0091	b_{36}	−0.0021
b_7	0.0040	b_{22}	0.0004	b_{37}	0.0057
b_8	−0.0023	b_{23}	−0.0037	b_{38}	0.0108
b_9	0.7525	b_{24}	0.0866	b_{39}	0.0001
b_{10}	0.0115	b_{25}	0.0362	b_{40}	0.0003
b_{11}	0.0175	b_{26}	0.0489	b_{41}	−0.0001
b_{12}	0.1487	b_{27}	−0.0056	b_{42}	0.0002
b_{13}	0.0010	b_{28}	−0.0039	b_{43}	0.0004
b_{14}	−0.0002	b_{29}	0.0173		
b_{15}	−0.0024	b_{30}	0.2189		

Table B.6.: Base inertial parameters of the Franka Research 3.

Bibliography

[1] Y. Han, J. Wu, C. Liu, and Z. Xiong. "An Iterative Approach for Accurate Dynamic Model Identification of Industrial Robots". In: *IEEE Transactions on Robotics* 36.5 (2020), pp. 1577–1594. DOI: 10.1109/TRO.2020.2990368.

[2] S. M. Mamedov and S. M. Mikhel. "Practical Aspects of Model-Based Collision Detection". In: *Frontiers in Robotics and AI* 7 (2020). DOI: 10.3389/frobt.2020.571574.

[3] A. Hereid and A. D. Ames. "FROST: Fast Robot Optimization and Simulation Toolkit". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2017, pp. 4552–4559. DOI: 10.1109/IROS.2017.8202230.

[4] S. Khorshidi, M. Dawood, B. Nederkorn, M. Bennewitz, and M. Khadiv. "Physically-Consistent Parameter Identification of Robots in Contact". In: *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 677–683. DOI: 10.1109/ICRA55743.2025.11128710.

[5] C. G. Atkeson, C. H. An, and J. M. Hollerbach. "Estimation of Inertial Parameters of Manipulator Loads and Links". In: *The International Journal of Robotics Research* 5 (3 Sept. 1986), pp. 101–119. DOI: 10.1177/027836498600500306.

[6] S. Traversaro, S. Brossette, A. Escande, and F. Nori. "Identification of fully physical consistent inertial parameters using optimization on manifolds". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 5446–5451. DOI: 10.1109/IROS.2016.7759801.

[7] K. Yoshida, K. Osuka, H. Mayeda, and T. Ono. "When is the set of base parameter values physically impossible?" In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*. Vol. 1. 1994, 335–342 vol.1. DOI: 10.1109/IROS.1994.407372.

[8] O. Rodrigues. "Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire". In: *Journal de Mathématiques Pures et Appliquées* 5 (1840), pp. 380–440.

[9] Z. Nurlanov. *Exploring SO(3) Logarithmic Map: Degeneracies and Derivatives*. Technical Report. Technical University of Munich, 2021. URL: https://cvg.cit.tum.de/_media/members/demmeln/nurlanov2021so3log.pdf (visited on 06/17/2026).

[10] G. Garofalo, C. Ott, and A. Albu-Schäffer. "On the closed form computation of the dynamic matrices and their differentiations". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 2364–2359. DOI: 10.1109/IROS.2013.6696688.

[11] J. Englsberger. "Combining reduced dynamics models and whole-body control for agile humanoid locomotion". PhD thesis. Technische Universität München, 2016.

[12] M. Gautier. "Dynamic Identification of Robots with Power Model". In: *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*. Vol. 3. Albuquerque, NM, USA: IEEE, 1997, pp. 1922–1927. DOI: 10.1109/ROBOT.1997.619069.

[13] M. Gautier, A. Janot, and P. Vandanjon. "DIDIM: A new method for the dynamic identification of robots from only torque data". In: *2008 IEEE International Conference on Robotics and Automation*. 2008, pp. 2122–2127. DOI: 10.1109/ROBOT.2008.4543520.

[14] M. Gautier and P. Poignet. "Extended Kalman filtering and weighted least squares dynamic identification of robot". In: *Control Engineering Practice* 9.12 (2001), pp. 1361–1372. ISSN: 0967-0661. DOI: https://doi.org/10.1016/S0967-0661(01)00105-8. URL: https://www.sciencedirect.com/science/article/pii/S0967066101001058.

[15] A. Janot, P.-O. Vandanjon, and M. Gautier. "A Generic Instrumental Variable Approach for Industrial Robot Identification". In: *IEEE Transactions on Control Systems Technology* 22.1 (2014), pp. 132–145. DOI: 10.1109/TCST.2013.2246163.

[16] Z.-H. Jiang, T. Ishida, and M. Sunawada. "Neural Network Aided Dynamic Parameter Identification of Robot Manipulators". In: *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*. 2006, pp. 3298–3303. DOI: 10.1109/ICSMC.2006.384627.

[17] P. M. Wensing, S. Kim, and J.-J. E. Slotine. "Linear Matrix Inequalities for Physically-Consistent Inertial Parameter Identification: A Statistical Perspective on the Mass Distribution". In: *IEEE Robotics and Automation Letters* 3.1 (2018), pp. 60–67. DOI: 10.1109/LRA.2017.2729659.

[18] M. Gautier and W. Khalil. "A direct determination of minimum inertial parameters of robots". In: *Proceedings. 1988 IEEE International Conference on Robotics and Automation*. 1988, 1682–1687 vol.3. DOI: 10.1109/ROBOT.1988.12308.

[19] H. Mayeda, K. Yoshida, and K. Osuka. "Base parameters of manipulator dynamic models". In: *IEEE Transactions on Robotics and Automation* 6.3 (1990), pp. 312–321. DOI: 10.1109/70.56663.

[20] Q. Leboutet, J. Roux, A. Janot, J. R. Guadarrama-Olvera, and G. Cheng. "Inertial Parameter Identification in Robotics: A Survey". In: *Applied Sciences* 11.9 (May 2021). DOI: 10.3390/app11094303.

[21] C. Pham and M. Gautier. "Essential parameters of robots". In: *[1991] Proceedings of the 30th IEEE Conference on Decision and Control*. 1991, 2769–2774 vol.3. DOI: 10.1109/CDC.1991.261862.

[22] M. Gautier. "Numerical calculation of the base inertial parameters of robots". In: *Proceedings., IEEE International Conference on Robotics and Automation*. 1990, 1020–1025 vol.2. DOI: 10.1109/ROBOT.1990.126126.

[23] T. Lee, J. Kwon, P. M. Wensing, and F. C. Park. "Robot Model Identification and Learning: A Modern Perspective". In: *Annual Review of Control, Robotics, and Autonomous Systems* 7 (1 July 2024), pp. 311–334. DOI: 10.1146/annurev-control-061523-102310.

[24] M. Gautier and W. Khalil. "Direct Calculation of Minimum Set of Inertial Parameters of Serial Robots". In: *IEEE transactions on robotics and Automation* 6 (3 June 1990), pp. 368–373. DOI: 10.1109/70.56655.

[25] M. Gautier, S. Briot, and G. Venture. "Identification of consistent standard dynamic parameters of industrial robots". In: *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. 2013, pp. 1429–1435. DOI: 10.1109/AIM.2013.6584295.

[26] M. Gautier and G. Venture. "Identification of standard dynamic parameters of robots with positive definite inertia matrix". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 5815–5820. DOI: 10.1109/IROS.2013.6697198.

[27] C. D. Sousa and R. Cortesão. "Physical feasibility of robot base inertial parameter identification: A linear matrix inequality approach". In: *The International Journal of Robotics Research* 33.6 (2014), pp. 931–944.

[28] K. Ayusawa, G. Venture, and Y. Nakamura. "Real-time implementation of physically consistent identification of human body segments". In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 6282–6287. DOI: 10.1109/ICRA.2011.5979903.

[29] J. Jin and N. Gans. "Parameter identification for industrial robots with a fast and robust trajectory design approach". In: *Robotics and Computer-Integrated Manufacturing* 31 (2015), pp. 21–29. ISSN: 0736-5845. DOI: 10.1016/j.rcim.2014.06.004.

[30] J. Swevers, C. Ganseman, D. B. Tukel, J. de Schutter, and H. V. Brussel. "Optimal robot excitation and identification". In: *IEEE Transactions on Robotics and Automation* 13.5 (1997), pp. 730–740. DOI: 10.1109/70.631234.

[31] K. J. Park. "Fourier-based optimal excitation trajectories for the dynamic identification of robots". In: *Robotica* 24.5 (2006), pp. 625–633. DOI: 10.1017/S0263574706002712.

[32] J. Swevers, C. Ganseman, J. De Schutter, and H. Van Brussel. "EXPERIMENTAL ROBOT IDENTIFICATION USING OPTIMISED PERIODIC TRAJECTORIES". In: *Mechanical Systems and Signal Processing* 10.5 (1996), pp. 561–577. ISSN: 0888-3270. DOI: https://doi.org/10.1006/mssp.1996.0039.

[33] W. Wu, S. Zhu, X. Wang, and H. Liu. "Closed-Loop Dynamic Parameter Identification of Robot Manipulators Using Modified Fourier Series". In: *International Journal of Advanced Robotic Systems* 9 (2012), p. 29. DOI: 10.5772/45818.

[34] W. Rackl, R. Lampariello, and G. Hirzinger. "Robot excitation trajectories for dynamic parameter estimation using optimized B-splines". In: *2012 IEEE International Conference on Robotics and Automation*. 2012, pp. 2042–2047. DOI: 10.1109/ICRA.2012.6225279.

[35] D. Kostic, B. de Jager, M. Steinbuch, and R. Hensen. "Modeling and identification for high-performance robot control: an RRR-robotic arm case study". In: *IEEE Transactions on Control Systems Technology* 12.6 (2004), pp. 904–919. DOI: 10.1109/TCST.2004.833641.

[36] G. Antonelli, F. Caccavale, and P. Chiacchio. "A systematic procedure for the identification of dynamic parameters of robot manipulators". In: *Robotica* 17.4 (1999), pp. 427–435. DOI: 10.1017/S026357479900140X.

[37] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations*. 2015. URL: https://arxiv.org/abs/1412.6980.

[38] Husarion. *Franka Research 3*. URL: https://husarion.com/tutorials/ros-equipment/franka-research-3/ (visited on 06/22/2026).