Hybrid Haptic Device Modeling and System Identification

Antoine Henri

Department of Mechanical Engineering
McGill University, Montreal

August 2024

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Science

Abstract

The goal of a haptic device is to realize a stable and realistic sense of touch through the harmonious integration of a simulated environment and physical hardware. The ability to realize this objective is hindered by, among many things, the low power density of the actuator, which is most often a brushed DC motor. Coupling this motor with a magnetorheological (MR) brake offers an interesting possibility: an over-actuated hybrid actuator that combines the motor's fast dynamics with the brake's higher torque to supplied current ratio.

To couple a motor with a brake requires careful controller design, canceling out unwanted dynamics without removing too much power, which would reduce the maximum stiffness displayed by the haptic simulation. To design a controller requires a good model of the hybrid actuator. However, the complexity of the motor and the brake, both of which have many unknown parameters, makes model calibration very difficult. As such, in this thesis, a system identification (ID) approach is taken, where data is used to identify model parameters associated with the hybrid actuator. MR brakes display a hysteretic relationship between the produced torque and their angular velocity that is often not modeled. As such, a gray-box model will be used in the system ID process, as opposed to a linear black-box model, so as not to neglect the brake's nonlinear dynamics.

This thesis will first derive a complete dynamic model for the hybrid actuator, interconnecting amplifiers, motor, and brake into one multi-input, multi-output system. The parameters of this model will then be identified with a genetic algorithm, which will minimize the difference between experimental and simulated data. The quality of this system identification algorithm, and of the derived model, will be evaluated by identifying the parameters of the hybrid actuator model and by comparing this nonlinear model's response to the response of a linear black-box model.

Résumé

L'objectif d'un dispositif haptique est d'obtenir une sensation de toucher stable et réaliste grâce à l'intégration harmonieuse d'un environnement virtuel judicieusement simulé et d'une machine soigneusement fabriquée. La réalisation de cet objectif est entravée, entre autres, par la faible densité de puissance de l'actionneur, qui est le plus souvent un moteur à courant continu à balais. Le couplage de ce moteur avec un frein magnétorhéologique (MR) offre une possibilité intéressante : un actionneur hybride sur-actionné qui combine la dynamique rapide du moteur avec la puissance du frein MR.

Le couplage d'un moteur et d'un frein nécessite une conception minutieuse du contrôleur, annulant les dynamiques indésirables sans supprimer trop de puissance, ce qui réduirait la rigidité maximale affichée par la simulation haptique. La conception d'un contrôleur nécessite un bon modèle de l'actionneur hybride. Cependant, la complexité du moteur et du frein, qui ont tous deux de nombreux paramètres inconnus, rend difficile l'étalonnage du modèle. C'est pourquoi est adoptée dans cette thèse une approche d'identification de système où les données sont utilisées pour identifier les paramètres du modèle associés à l'actionneur hybride. Les freins MR présentent une relation hystérétique entre le couple produit et leur vitesse angulaire qui n'est souvent pas modélisée. Ainsi, un modèle « boîte grise » sera utilisé dans le processus d'identification du système, par opposition à un modèle linéaire « boîte noire », afin de ne pas négliger la dynamique non linéaire du frein.

Cette thèse commencera par dériver un modèle dynamique complet de l'actionneur hybride, en interconnectant les amplificateurs, le moteur et le frein en un système à entrées et sorties multiples. Les paramètres de ce modèle seront ensuite identifiés à l'aide d'un algorithme génétique, qui minimisera la différence entre les données expérimentales et les données simulées. La qualité de cet algorithme d'identification du système et du modèle dérivé sera évaluée en identifiant les paramètres du modèle d'actionneur hybride et en comparant la réponse de ce modèle non linéaire à la réponse d'un modèle linéaire « boîte noire ».

For Marie, who was and is always at my side.

Acknowledgements

The financial support provided by Haply Robotics, the Mitacs Accelerate program, and the NSERC Discovery Grant program is graciously acknowledged. Additionally, a big "thank you" to the employees at Haply Robotics for the support, both material and personal, that was provided, as well as the many conversations on haptics that helped shape this thesis.

Thank you to James R. Forbes for supervising this thesis and for supporting my endeavors through tough times.

Thank you to Steven Dahdah, for lending a hand, a lot of his code, and for the many hours of discussion, research focused or not.

Table of Contents

List of Figures

Figure

List of Tables

Table

List of Abbreviations

GA	Genetic Algorithm
MR	Magnetorheological
RMSE	Root Mean Square Error
VE	Virtual Environment
DC	Direct Current
GASSI	Genetic Algorithm for Stable System Identification
ODE	Ordinary Differential Equation
HIL	Hardware-In-the-Loop
SDK	Service Development kit
RL	Resistor-Inductor
CEGA	Computationally Efficient GA
GENOCOP	GEnetic algorithm for Numerical Optimization for COnstrained Problems
SISO	Single-Input, Single-Output
MIMO	Multiple-Input, Multiple-Output
RMSE	Root Mean Square Error
MSD	Mean Signed Deviation
VR	Virtual Reality
ID	Identification

Preface

The contributions of this thesis that are original to the author's knowledge are as follows:

- Section 3.4: Derived a hybrid actuator process model and the model's Jacobian.
- Section 4.1.4: Used specific genetic algorithm implementations and associated constraint handling techniques from the literature to realize system identification of a stable system.

Chapter 1

Introduction

1.1 Motivation

Virtual reality (VR) is a tool for simulation, training and teleoperation and is today more popular than ever. Virtual VR technology displays virtual worlds, or overlays additional information over real images. Innovations in the field of VR are coming out at a rapid pace, but they can't perfectly replicate the real world. Most VR devices can't produce sensations of smell, taste, and, most importantly, touch. Much like VR devices display virtual worlds by interacting with the user's vision, haptic devices simulate the presence of virtual objects and surfaces by stimulating the user's tactile receptors. They are often seen as promising tools to improve surgical training, and teleoperation of dangerous, heavy machinery. It is only natural then to want to improve the realism of such devices. To do so, it is essential to better understand how such devices work, and how they are designed.

Haptic devices and related technologies are designed with many characteristics in mind, but some stand out more than others. In particular, transparency, Z-width and stability, in the context of haptics, are often the primary focus of such endeavors [1]. Many strategies are employed to give a haptic device one, each, or any combination of these characteristics, with the ultimate goal to design a machine that can display to a user, through their sense of touch, the virtual tactile sensations.

Transparent devices often employ inertia [2] or friction [3] cancellation algorithms. This is because a transparent haptic device, when manipulated by the user, should feel as if it's not there. A perfectly transparent haptic device would directly produce the forces corresponding to the impedance of the object in the virtual environment, without delays or undesired force artifacts induced by the mechanical, electronic or software components of the haptic device [4]. Transparency is required to make surgical simulations realistic, for the surgeons to feel their patients, and not the haptic device's interpretation of what their patients should

feel like. It is with this ideal device in mind that haptics device are designed, usually with components that have a low inertial impact, so that force commands signals from the virtual environment aren't attenuated by the device on their way to the user, and that user movements aren't attenuated by the device on their way to the sensors used to communicate the users position and velocity to the virtual environment [1].

A useful characterization of stability in haptic devices is often based on the concept of passivity. Experiments have shown that humans users are considered passive when interacting with haptic devices, and that the usual simulated physical environments represented by the virtual environment are passive [5]. The closed system, where the user interacts with the virtual environment through the haptic device, is thus stable if the haptic device is also passive.

This passivity condition on the haptic display can often be simplified to

$$b > \frac{KT}{2} + B, \tag{1.1}$$

where b is the haptic device's physical damping, T is the rate at which the user's position and velocity is sampled, and K and B are respectively the stiffness and the damping simulated by the virtual environment, as shown in [6]. This passivity approach is conservative, but gives the right intuition on how a haptic device is limited in the types of impedance the virtual environment can hope to display through the device by the device's own physical damping. Designing a device with stability as the main focus often leads to designing a device that is less transparent, and vice versa, since stability inducing strategies often need to introduce some type of virtual impedance to increase the stable region that might interfere with the realism of the simulated environment.

A useful way to describe a haptic device's stability is by defining what impedance the device can display. It is however very difficult to design a device with high Z-width, which is the ratio of the highest impedance the device can display over the lowest impedance the device can display. This is due in part to the nature of the actuators preferred in most haptic devices: electric motors. They are often preferred for their linear impulse response characteristics, and usual fast reactivity to input signal, which leads to increased transparency. However, electric motors have relatively low power densities [1]. This makes it so haptic devices that employ electric motors can't display very hard surfaces, which would be useful for many applications where haptic displays with high impedance are useful: medical simulations, teleoperation of heavy machinery, and others.

Many techniques have been explored to increase the stable impedance range for haptic devices: passivity observers (PO) and passivity controllers (PC), virtual couplings, etc. [1,

6]. These approaches cannot however overcome the physical limitations of electric motors. This is why hybrid haptic devices have often been considered as a solution to achieve both stability and transparency in one haptic device. To attain such a goal, hybrid haptic device combine two or more types of actuators, which are often categorized as either active or passive devices. A classic example of such an active device is the electric motor. Other examples would be linear or pneumatic actuators.

Passive actuators come in many shapes and forms. Some are not controllable, like standard brakes and elastic actuators. Others are, like the magnetorheological brake, or MR brake, the eddy current brake, and the electrorheological brake, or ER brake [1]. These controllable passive actuators affect (1.2), making the haptic's device Z-width wider by augmenting its maximum impedance. This is because you can consider that the passive actuator adds controllable damping to the haptic device:

$$b + b_{passive} > \frac{KT}{2} + B, \tag{1.2}$$

The combination of both the electric motor and the MR brake is often preferred, specifically because of the well-established use of electric motors and the similar voltage demands of the MR brake to the electric motor. The ER brake has, for example, a much higher working voltage of many kilovolts. For the sake of simplicity, it is to be assumed that for the rest of this thesis, when the term hybrid actuator is used, it is used to describe the previously mentioned combination of a direct current brushed electric motor, which will be further called a DC motor, and an MR brake. Control of this type of hybrid actuator is not an easy task. Contrary to the electric motor, the MR brake has nonlinear system dynamics, making it much more difficult to use traditional control strategies that rely on linear system approximations. Linear system approximations of the MR brake cannot however encompass known nonlinear phenomena unique to the MR brake, such as the hysteresis relationship between torque and angular velocity. This hysteretic behavior creates the sensation of stiction present in haptic simulations where MR brakes are used. Some current proposed control strategies for hybrid actuators assume a linear [7] or nonlinear [8] static mapping for the relationship from the current applied to the brake to its torque output. Others use recurrent neural network to predict brake torque [9].

It's been shown in the previously mentioned literature that control strategies that assume these brake models to be valid do produce stable, and sometimes transparent, haptic devices. This thesis proposes that a dynamic model, which has seen promise in other work [10], should instead be considered, to produce a haptic device controller that not only relies on quality metrics from the field of haptics, but also on metrics currently used in control theory, that measure command following and response time.

To produce such controllers, and find such dynamic models, requires a hybrid model to be derived, and for this model's parameters to be deduced.

1.2 Objective

A system, when given an input signal and an initial condition, produces an output signal. A deterministic system will always produce the same output signal for a given input signal and set of initial conditions, while a stochastic system is predictable in the sense that its output respects probabilistic rules.

There are many mathematical representations of systems. Some apply to certain types of systems: for example, transfer functions are used to represent continuous linear time-invariant systems, with the Laplace transform used for continuous systems, and the Z-transform used for discrete systems

Much like transfer functions, state-space representations of systems can be used for both continuous and discrete systems. This thesis will focus on continuous systems, as DC motors and MR brakes exist in a continuous world, and most models, both for the motor and the brake, are defined in continuous time.

State-space representations can also be used for both linear and nonlinear systems, which are ideal for a hybrid actuator model, as the MR brake displays nonlinear behavior. State-space representations are defined by sets of ordinary differential equations, or ODEs, which are usually derived from known natural laws, like Newton's second law, or Kirchhoff's circuit laws.

State-space representations, for the reasons stated above, will be used in this thesis. They will also be used with future work in mind. From a state-space representation can be derived a linearized system model, which in turn can be used for gain-scheduled controllers, which will make use of the latest linear controller design techniques: LQR controllers, $\mathcal{H}_\infty$ controllers, disturbance observers, etc. This representation could also be employed to design nonlinear feedforward controllers, that take advantage of the known dynamics of the actuator to cancel out undesired dynamics.

For a model to be useful, be it in transfer function or state-space form, the model parameters must be known. Finding model parameters can be very difficult. One means to find model parameters is via system identification. In black-box system identification, input-output data is simply fit to an assumed model form. In gray-box system identification, input-output data is used to fit, or calibrate, model parameters.

As such, the focus of this thesis is not to design new system identification methods, but to find the combination of hybrid actuator model and parameter identification technique which

can, when simulating the resulting identified system, reproduce with some degree of fidelity the output response that was displayed by an experimental hybrid actuator. No controller can be designed without confidence in the accuracy of the aforementioned model.

To this end, known models for MR brakes will be used to derive the state-space representation, or process model, of the hybrid actuator as a whole. The same principle will be applied to the chosen parameter identification method, which will be picked from an often used, and well proven method to identify MR brake model parameters, which is the genetic algorithm, or GA.

1.3 Overview

Before any model can de derived, and any parameters deduced, an experimental test bed needs to be built. The hybrid actuator presented in Chapter 2 was built to easily control all inputs and record data from all outputs which might be of interest when building a hybrid actuator model. Chapter 3 takes inspiration from this experimental setup and known dynamic models, for both the DC motor and the MR brake, to derive an interconnected process model. Data collected from the experimental setup and the derived model will then be used by a GA, in Chapter 4, to find the model's parameters and then compare the identified's model response to a linear model's response.

Chapter 2

Experimental Setup

2.1 Motivation

For some brushed DC motors, especially those of sufficient quality, data sheets containing enough information on the motor's characteristics to build an accurate dynamic model can be found. The same cannot be said of MR brakes. As will be seen in Chapter 3, it is possible to model MR brakes in many ways. Modelling an MR brake is not necessarily a problem in itself, as any existing model developed experimentally should be able to be used in simulation to accurately recreate the behavior of an MR brake, and thus be usable to simulate and test potential hybrid actuator controllers. The problem is that there are very few available and useful sets of identified parameters for MR brake dynamic models in the literature that can be used for simulation tasks.

This is not to say that sets of identified parameters for MR brakes do not exist. They do in fact exist, but some of them [9, 11] don't respect conditions on the parameters of MR brake that will be explained in Section 3.4.2, others are missing values for some essential parameters [12], and others apply to a linear MR brake dynamic model [12–14], which is not the type of MR brake used in the fabrication of hybrid actuators used in haptic applications, made from an electric motor and a rotational MR damper.

Some of these previously mentioned experiments take into account an MR brake model that is only valid for a given static value [11, 13, 14]. The brake model needs to be computed at different current set points, and is not adapted for a haptics application where the brake current would always be evolving. As such, even if these models would have a set of usable parameters, the MR brake models they provide are themselves unusable, since a dynamic model is being derived in this thesis. Since the coil found in the brake inherently possesses some type of dynamics electromagnetic behavior, these experiments were deemed unsatisfactory.

Other experiments lead to acceptable sets of parameters and acceptable models, but present some caveats in other ways. For example, the set of parameters found in [15], both for the Dahl and Bouc-Wen models, have values available for all parameters, and these values respect the conditions found in Section 3.4.2, but the value of the parameter n in the Bouc-Wen model, given as $n = 26.2186$, is quite high, and this high exponent significantly increases input to output simulation times. This was tested with the simulation software used for system identification in this thesis and the lengthy simulation times were found too high to produce results in a reasonable span of time.

For the reasons listed above, it was deemed necessary to build an experimental setup, not only to collect input to output data for system identification, but also for future experiments, to design and test hybrid actuator control algorithms.

2.2 Mechanical and Electrical Hardware

The hybrid actuator used in this thesis was built around the Magmotor S23-H-200FX DC brushed motor, with a torque capacity of 0.353 N·m at an excitation current of 2.5 A, and the Placid Industries B6-24-2M Magnetic Particle Brake, with a maximum torque capacity of 0.6779 N·m at excitation current of 0.19 A. No linkage mechanism was placed between the motor and the brake to give mechanical advantage to the motor, as both of their torque capacities were deemed close enough to not warrant the added cost, inertia and nonlinear dynamics that such a mechanism would impart to hybrid actuator response. Indeed, while haptic devices often use capstan drives, instead of gearboxes, to reduce the amount of backlash, which would affect the device's transparency, there is still a small amount of smooth friction caused by the cables in this drive, especially during small movements [2]. Flexible couplings are still used to link the motor to the brake, and the brake to the torque sensor, to account for misalignment errors due to the brake, motor and torque sensors custom designed mounts.

Both the motor and the brake were powered and driven by the Quanser Industries AMPAQ-L2 high-bandwidth, two-channel, linear current amplifier, which possesses a maximum continuous current output of 2.5 A, and a maximum continuous voltage output of ±24 V. The AMPAQ-L2's has an amplifier gain 0f 0.5 A/V, with an output impedance of under 0.2 Ω. This makes it so the amplifier can provide the maximum current needed for the motor and the brake if provided with a command signal of ±5 V. Such command signals are provided by the QUANSER QPID Data Acquisition Device, which is also used to collect all signals provided by the sensors used in the hybrid actuator.

The first two of those signals are analog current sense signals provided by the AMPAQ-

L2, one for each amplifier channel, which measure the amplifier current at a ratio of 2 V/A. The third sensor signal is provided by the encoder integrated in the Magmotor motor, that is the US Digital E3-2048-375-NE-H-D-B Optical Kit Encoder. This encoder has a cycles per revolution ratio of 2048. This angular position signal is recorded by the QPID, which also computes an online angular velocity estimate. The final sensor signal is provided by the Dataforth SCM5B38-05 High-Bandwidth Strain Gauge Input Module, which amplifies, at a ratio of 2 mV/V, the signal coming from the Sensor Development Inc 01324-310 rotary shaft torque sensor, which has a maximum torque rating of 3.53 N·m, which is well above the combined torque capacity of both the motor and the brake. Both the encoder and the strain gauge amplifier are powered by the 5 V power supply integrated in the QPID.

The motor, the brake and the torque sensor are mounted on custom designed, 3D printed parts, which are in turn mounted on 20 mm × 40 mm × 400 mm aluminum extrusion attached to a wooden board. The computer-aided drawings of these mounts can be found in Figure 2.1. The hybrid actuator as a whole is quite heavy, but can be transported easily from one workspace to another in the lab, where it can be clamped onto the side of a table or bench. The final product, built in its entirety except for the load pulley, can be seen in Figure 2.2.

A simple pulley and mass system was attached to the load side of the torque sensor with a clamping shaft adapter, to provide a non-zero load for system identification purposes. Providing randomized electrical input signals to the motor and the brake without such a load would not stimulate all the dynamics found in the hybrid actuator and would render system identification attempts less insightful.

This load system could easily be removed in future experiments to provide an attachment point for a handle or a steering wheel. This would make, with the appropriate software, the hybrid actuator designed for this thesis a completely functional experimental haptic device.

2.3 Software

Three software components were designed for this project. The first one is a data collection application built around Quanser Industries' Hardware-In-the-Loop, or HIL, software development kit, or SDK. This SDK is used to communicate with the QPID Data Acquisition Device, to run programmed sequences of input signals on the motor and brake. These input signals, as well as sensor data, is collected by the QPID and the implemented software in a standard comma-separated values file, or CSV file, to be used by any signal analysis or system identification software subsequently.

The second software package developed for this thesis is a system identification Python library. This package, when provided with a set of input to output signals and a system's

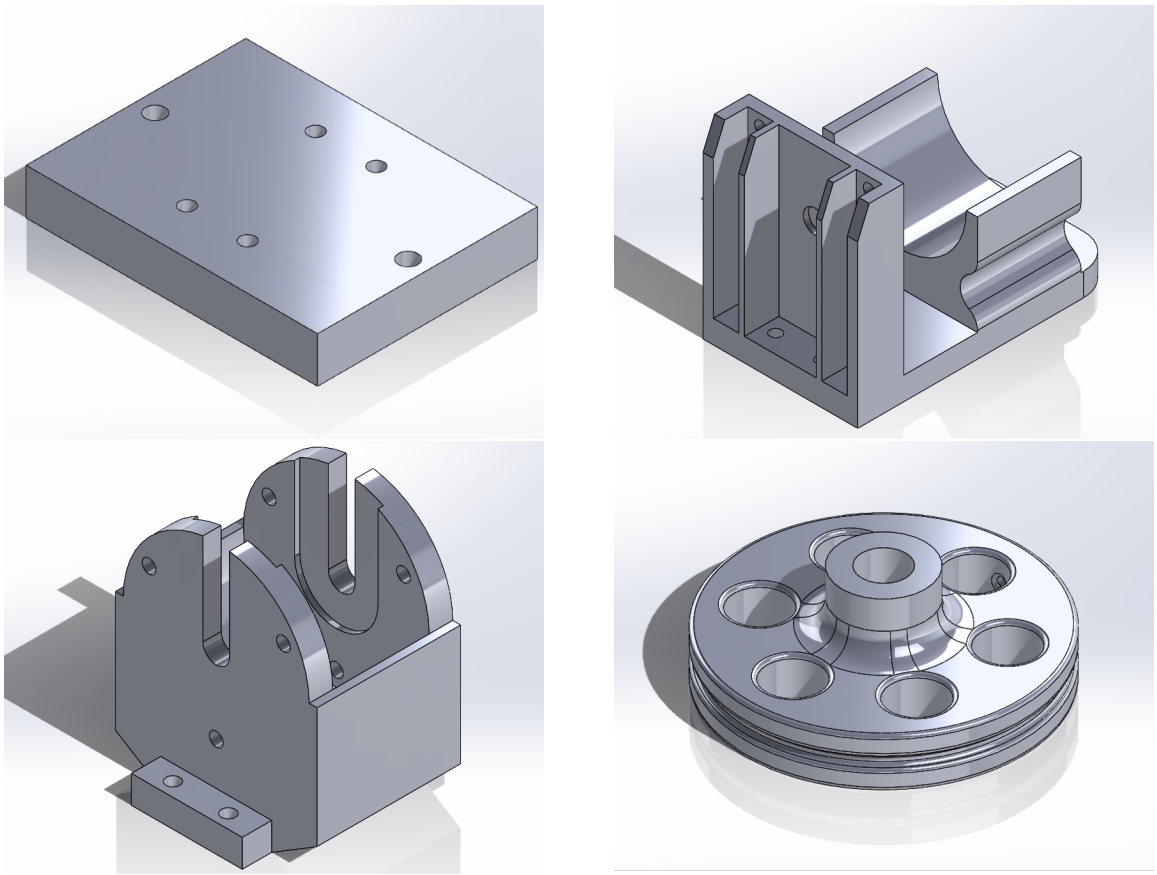

Figure 2.1: Computer-aided drawings of, from top to bottom, and then left to right: the torque sensor mount; the DC motor mount; the MR brake mount; and the load pulley.

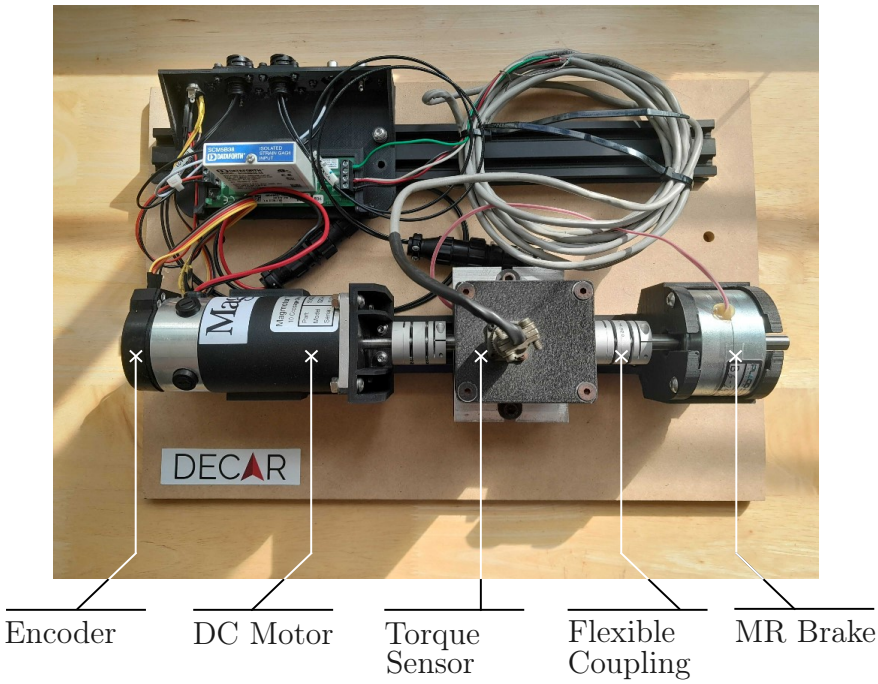

Figure 2.2: An alternative hybrid actuator configuration is shown above with, from left to right: an encoder; a brushed DC motor; a torque sensor; and an MR brake. This configuration is used in Section 4.2.2 for hysteresis and friction measurements. The configuration used in most of this thesis for input to output data collection swaps the position of the torque sensor and the MR brake.

process model, uses an adaptive GA to infer a set of stable system parameters. The Python package is forked from the `pysysid` python library, which implements other system identification algorithms compatible with the popular machine learning package `scikit-learn`.

The third and last application developed for this project is the application used to identify hybrid actuator process model parameters. It combines the GA implemented in the `pysysid` package and the dataset provided by the data collection application mentioned above to do so. This application is also used to compute metrics and comparisons which can help judge both the quality and usefulness of the implemented GA and the derived hybrid actuator process model.

Chapter 3

Hybrid Actuator Model

With the specifications of the experimental setup under study now fully detailed, it is now possible to move on to system identification. There are many ways to do system identification, but one that has not been explored often in the field of haptics is grey-box system identification. Contrary to black-box system identification, grey-box system identification method assumes all systems can be defined by some predetermined set of dynamic equations, for which the parameters, often representing physical quantities like resistance or inertia, can be measured or inferred from experimental data [16].

As mentioned in Section 1.1, the hybrid actuator studied in this thesis is nothing more than a brushed DC motor and an MR brake. Since for a given current, the torque produced by the motor is often much lower than the one produced by the brake, the motor and the brake are often linked together by using a capstan drive, which gives the motor mechanical advantage over the brake. As explained in Section 2.2, the hybrid actuator designed and observed in this thesis does not have such a mechanism; the motor and the brake are linked directly to each other, rotor to rotor. However, such a mechanism is often used in haptic systems, even when only a DC motor is used, and not a hybrid actuator, and in that line of thought the gear ratio n_c which describes the mechanical advantage of the motor over the brake is kept in the following dynamic equations to generalize the possible uses of the following model. Since there is no capstan drive, the motor and the brake always spin in the same direction, at the same rate, and as such, it can be assumed that there is no need to identify this parameter in the next chapters and that its value is $n_c = -1$.

To correctly understand the hybrid actuator dynamics, an explanation of the inner workings of the MR brake is first given in Section 3.1, to better understand what systems found in the brake need to be modelled. An overview of all the electrical systems found in the hybrid actuator will then be given in Section 3.2. Following that, Section 3.3 will go over different possible models for the nonlinear force produced by the brake. Section 3.4 will

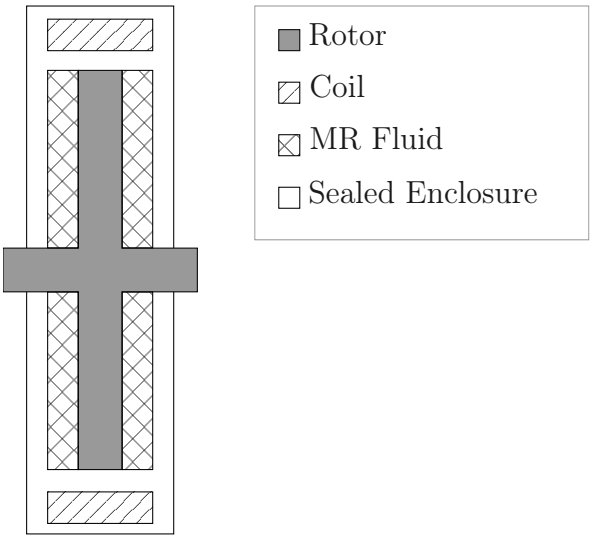

Figure 3.1: Simplified mechanical schematic of the MR brake.

then demonstrate how the motor and the hybrid actuator are interconnected through their mechanical model, and derive the Jacobian and constraints needed by the GA presented in Chapter 4 to compute the value of the hybrid actuator model's parameters.

3.1 Working Principles of an MR Brake

An MR is made from three main components: a sealed enclosure filled with magnetorheological fluid, an electrical coil producing a magnetic field through this fluid, and a rotor that movement is impeded by the changing rheological properties of the fluid, change caused in turn by the fluctuations in the magnetic field [17, 18]. Figure 3.1 demonstrates how these different parts interact with each other:

Many different mechanical, electrical, and magnetic phenomena rule the behavior of the MR brake. Some of them are hard to model. A thorough investigation of the actual physical model of an MR brake would require finite element analysis of both the magnetic field produced by the coil, and some work on the fluid dynamics of the MR fluid. The computational requirements of such a task would produce a model that is not very useful for the eventual purpose of designing a controller for the MR brake and the hybrid actuator, which would require online computing of its response. Finite element analysis of this kind has been done before, but only in magnetostatic conditions [18], which is does not consider the dynamic behavior of the electrical components of the MR brake, which are under consideration in this thesis.

While the dynamics of a DC motor are well known in the literature, there is no similar

consensus when it comes to the dynamics of the MR brake, even for a simplified phenomenological model. As such, the following sections will explore different existing dynamic models used to simulate the behavior of the brake. Section 3.2.2 will go into the electrical dynamics of the brake's coil, while Section 3.3 will explore the force produced by actuating this coil, and Section 3.4 will then explore how this hysteretic force interacts with the rest of the hybrid actuator.

3.2 Electrical Dynamics

Four electrical equations define the electrical behavior of the hybrid actuator. One for the motor, one for the brake, one for the motor's driver, and one for the brake's driver. Throughout this thesis it will be assumed that the dynamics of the systems mentioned above are linear, since experimental results seem to demonstrate that most of the nonlinear dynamics in the hybrid actuator come from the mechanical behavior of the MR brake. As such, the following sections will mostly focus on creating models that are useful when it comes to experimentation, with inputs that can be actuated, and outputs that can be observed through the use of sensors. Different experimental setups might have made some other outputs observable, but the complete electrical model designed in this section will take into account the specific hardware available according to Chapter 2, which provides an angular velocity estimate, and two current signals, one for the motor and one for the brake.

3.2.1 DC Motor

The electrical model for a DC brushed Motor is a well-documented and researched topic. As such, the equations below are taken directly from [19], with slight modifications to consider the relationship between the motor rotor's rotational speed and the MR brake's rotational speed. Thus, the resulting electrical equation for the electrical dynamics of the hybrid actuator is given as

$$\frac{\mathrm{d}i_m(t)}{\mathrm{d}t} = \frac{n_c K_m}{L_m}\omega(t) - \frac{R_m}{L_m} i_m(t) + \frac{1}{L_m} v_m(t), \tag{3.1}$$

where $i_m(t)$ is the current going through the motor's coil, L_m is the coil's inductance, K_m is the back-emf constant, R_m is the coil's inductance, and $v_m(t)$ is the voltage applied on the coil.

The angular velocity in the motor equation is normally $\omega_m(t)$, the rotor's angular velocity, but it is in this case is replaced by $-n_c\omega$, where $\omega(t)$ is the angular velocity on the brake shaft. Haptic applications of hybrid actuators such as the one under study consider a linkage

between the motor and the brake shaft. This linkage, a capstan drive, creates a mechanical advantage n_c for the motor on the brake shaft. Further explanations of the linkage between the DC motor and the MR brake will be detailed in Section 3.4.

Section 3.4 will also consider that the back-emf constant is equal to the motor constant, which is a commonly made assumption. The motor constant will also be indicated with the K_m symbol.

3.2.2 MR Brake

There is no consensus on an electrical model for the dynamics of an MR brake. Sources such as [20] first proposed the model

$$\frac{\mathrm{d}v_a(t)}{\mathrm{d}t} = -\eta(v_a(t) - v_b(t)), \tag{3.2}$$

where η is the cutoff frequency, $v_b(t)$ is the voltage applied to the brake's coil, and $v_a(t)$ is some internal state variable, a filtered voltage. The applied voltage $v_b(t)$ does not itself have an effect directly on the parameters of the hysteresis force produced by the brake, instead, it does so through the first-order filter in (3.2). This what makes the brake torques in Section 3.3 dynamic compared to other MR brake models.

The only issue with (3.2) is that the state variable $v_a(t)$ does not represent a measurable physical quantity. There is no way to measure such a voltage. Indeed, since there are only two electrical terminals on the MR brake, it is impossible to access or manipulate any other voltage than the one found on those two terminals, which is $v_b(t)$.

There is however a physical quantity that will be accessible from those terminals: the current $i_b(t)$ passing through the MR brake. This current is not present in the MR brake electrical model found in [20], which seems out of place since most first-order systems found in electrical engineering relate current and voltage. One of those first-order systems is a resistor-inductor system, or RL system.

The differential equation for such a model, relating an input voltage to an output current, is:

$$\frac{\mathrm{d}i_b(t)}{\mathrm{d}t} = -\frac{1}{L_b}(R_b i_b(t) - v_b(t)), \tag{3.3}$$

where L_b is the coil's inductance and R_b is the coil's internal resistance. Assuming the MR brake's electrical model can be represented by this RL system, and that the potential $v_a(t)$ which did not seem accessible is the voltage across the MR brakes internal resistance

$$v_a(t) = R_b i_b(t).$$

The LR differential equation can be manipulated to take the form

$$\frac{\mathrm{d}v_a(t)}{\mathrm{d}t} = -\frac{R_b}{L_b}(v_a(t) - v_b(t)) \tag{3.4}$$

Comparing (3.2) to (3.4), it follows that

$$\eta = \frac{R_b}{L_b}, \tag{3.5}$$

which is the inverse of the time constant for a resistor-inductor system.

This insight on the possible relationship between $v_a(t)$ and $i_b(t)$ will be useful going forward, as it is possible to access the current $i_b(t)$ with a sensor.

3.2.3 Drivers

The goal of this analysis of the hybrid actuator is to better understand its dynamics. Better understanding of a system's dynamics, whether it's for the sake of controller design or state estimation, is always improved when there is concern for systems exogenous to the system under analysis. These other systems might themselves affect the dynamics of the system under analysis, since no real system has an infinitely small output impedance or infinitely big input impedance. This is why it is important to model the dynamics of the drivers which transform the low power command signals from a computer or a microcontroller to the high power voltages at the input of both the DC motor and the MR brake in the hybrid actuator.

For this study of drivers, two categories will be studied: open-loop and closed-loop amplifiers. Some driver specifications might be very clear and explicit with models, parameters or their input responses, which would indicate in which of the previously mentioned categories they fall into. Others might not, which is the case in this study, since the drivers used in the experimental setup shown in Chapter 2 have no clear indications in their specifications on their input to output responses. As such, both driver models will be detailed in this section, as first order approximations. This approximation is chosen for the sake of reducing the complexity of parameter inference in Chapter 4, but higher order approximations could be made.

The process model for a first order, linear, open-loop amplifier driving the DC motor has the transfer function

$$\frac{V_m(s)}{V_{cmd_m}(s)} = \frac{K_{cmd_m}\omega_{cmd_m}}{s + \omega_{cmd_m}} \tag{3.6}$$

with $V_m(s)$ the Laplace transform of the driver's output voltage, $V_{cmd_m}(s)$ the Laplace trans-

form of the driver's the command signal, ω_{cmd_m} the driver's cutoff frequency, and K_{cmd_m} the driver's DC gain.

The ODE governing its dynamic behavior would thus be

$$\begin{aligned} \dot{v}_{m_{open}}(t) &= -\omega_{cmd_m} v_m(t) + K_{cmd_m}\omega_{cmd_m} v_{cmd_m}(t) \\ &= f_{md_{open}}(v_m(t)) + K_{md_{open}} v_{cmd_m}(t), \end{aligned} \tag{3.7}$$

with $v_m(t)$ the driver's output voltage, and $v_{cmd_m}(t)$ the driver's the command signal. A first order, linear, closed-loop amplifier has the transfer function

$$\frac{V_m(s)}{E(s)} = \frac{K_{cmd_m}\omega_{cmd_m}}{s + \omega_{cmd_m}}, \tag{3.8}$$

with

$$E(s) = K_{v2i} V_{cmd_m}(s) - I(S), \tag{3.9}$$

the Laplace transform of the error between the current command, which is the voltage command multiplied by the driver's input gain K_{v2i}, and the motor's current.

Combining (3.8) and (3.9) results in the state-space realization

$$\begin{aligned} \dot{v}_{m_{closed}}(t) &= -\omega_{cmd_m} v_m(t) - K_{cmd_m}\omega_{cmd_m} i_m(t) + K_{v2i} K_{cmd_m}\omega_{cmd_m} v_{cmd_m}(t) \\ &= f_{md_{closed}}(v_m(t), i_m(t)) + K_{md_{closed}} v_{cmd_m}(t). \end{aligned} \tag{3.10}$$

Both (3.7) and (3.10) can apply directly to the driver used to power the DC and the MR brake, with the appropriate change in parameters if a different driver is used for this later machine. Since both of these models are however linear and can thus only be an approximation of the real dynamic behavior of the drivers, it might be the case that while both the DC motor and the MR brake use the same driver, the identified parameters for each individual driver might be different. This is because the MR brake and the motor possess different input impedance, which will affect the internal dynamics of the drivers, since their output impedance can never be fully close to zero.

3.3 Brake Hysteresis Models

It is clear that the MR brake has an inertial component to the force it produces, since, like the DC motor, the brake possesses a rotor with some undetermined inertia J_b. For any

given hysteretic force model, this inertial model will stay the same, and equal to

$$\tau_{b_{inertial}} = J_b \frac{\mathrm{d}\omega(t)}{\mathrm{d}t}. \tag{3.11}$$

There is however no consensus in the literature on which model that should be used to model the damping force τ_b produced the by the MR brake, or in simpler term, the force produced by the effect of both the MR brake's magnetorheological fluid on the brake's rotor and the effect of its coil on this fluid. It has been observed experimentally that this damping force displays a hysteretic behavior.

This section will explore different known dynamic models for such a hysteretic force. It should be known that some static models for the brake force do exist, as shown in [21]. However, the approach of this thesis is directed towards dynamic models, first because, as shown in Section 3.2.2, the brake displays dynamic behavior when it comes to the interaction of the voltage applied at its input and the current passing through its coil, but also because dynamic models could help in the design of an eventual hybrid actuator controller.

3.3.1 Voltage-Dependent Modified Bouc-Wen Model

The voltage-dependent Bouc-Wen model first found in [20] assumes that torque produced by a rotational MR brake is of the form

$$\tau_b(v_a(t), \omega(t), z(t)) = \alpha(v_a(t))z(t) + c(v_a(t))\omega(t), \tag{3.12}$$

with the values of $\alpha(v_a)$ and $c(v_a)$ given by

$$\begin{aligned} \alpha(v_a) &= \alpha_0 + \alpha_1 v_a(t), \\ c(v_a) &= c_0 + c_1 v_a(t), \end{aligned} \tag{3.13}$$

and the hysteretic differential equation being

$$\begin{aligned} \frac{\mathrm{d}z(t)}{\mathrm{d}t} &= f_{z_{bouc-wen}}(\omega(t), z(t)) \\ &= -\gamma|\omega(t)|z(t)|z(t)|^{(n-1)} - \beta\omega(t)|z(t)|^n + A\omega(t), \end{aligned} \tag{3.14}$$

with γ, β and A determining the hysteretic loop shape.

The state variable $z(t)$ found above is an evolutionary variable that does not represent a physical quantity, but is only there to reproduce the hysteretic behavior of the brake.

Using the assumption made in Section 3.2.2, τ_b can be reformulated to

$$\tau_{b_{bouc-wen}}(i_b(t), \omega(t), z(t)) = \alpha(i_b(t))z(t) + c(i_b(t))\omega(t), \tag{3.15}$$

with no effect on its computed value since $v_a(t) = R_b i_b(t)$ and R_b can be included in the value of scalars c_1 and α_1.

It is to be noted that there exists other versions of this voltage-dependent Bouc-Wen model, where the $\alpha(i_b(t))$ and $c(i_b(t))$ are not first order polynomials of i_b, but higher order polynomials or combinations of exponential functions. There are however many sources that do use the first order polynomials, and thus to reduce the number of model combinations and time spent computing parameters in Chapter 4, we will assume that both of the previously mentioned functions are always first order polynomials.

3.3.2 Modified Dahl Model

The Dahl model does not differ much from the Bouc-Wen model. In essence, it follows the same structure. The hysteretic behavior is determined by the ODE [22]:

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f_{z_{dahl}}(\omega(t), z(t)) = \sigma\omega(t)(1 - z(t)sgn(\omega(t))), \tag{3.16}$$

with σ used to change the hysteretic loop shape. The hysteresis torque applied by the Dahl model is similar to the one produced by the Bouc-Wen model,

$$\tau_{b_{dahl}}(i_b(t), \omega(t), z(t)) = \alpha(i_b(t))z(t) + c(i_b(t))\omega(t), \tag{3.17}$$

with $\alpha(i_b(t))$ and $c(i_b(t))$ also first polynomials of $v_a(t)$ in the original source, and thus will be first order polynomials of $i_b(t)$ in this thesis as per the discussion in Section 3.2.2.

3.3.3 Modified LuGre Model

The hysteresis equation for the LuGre model is given by [23]

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f_{z_{lugre}}(\omega(t), z(t)) = \omega(t) - \zeta|\omega(t)|z(t), \tag{3.18}$$

with the hysteresis torque being

$$\tau_{b_{lugre}}(i_b(t), \omega(t), z(t)) = \alpha(i_b(t))z(t) + c(i_b(t))\omega(t) + \epsilon(i_b(t))\frac{\mathrm{d}z(t)}{\mathrm{d}t}. \tag{3.19}$$

Statements made in [21] indicate that $\alpha(i_b(t))$, $c(i_b(t))$, and $\epsilon(i_b(t))$ share the same form as

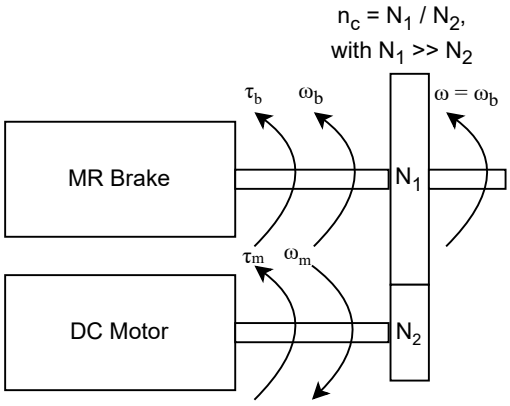

Figure 3.2: Mechanical schematic of the hybrid actuator.

the first order polynomials in the definition of τ_b for the Bouc-Wen and Dahl models.

3.4 Interconnected Hybrid Actuator Model

3.4.1 Process Model

The most important equation to correctly define the behavior of the hybrid actuator as a whole is defined by Newton's second law, according to which the mechanical behaviors of both the motor and the brake can't be separated, since their respective rotating parts are both attached to the same shaft through a capstan drive of gear ratio n_c. This mechanical equation is

$$-n_c\tau_{m_{inertial}} + -n_c\tau_{m_{damping}} + \tau_{b_{inertial}} + \tau_b = -n_c\tau_{m_{electro-mechanical}} + \tau_u, \tag{3.20}$$

with $\tau_{m_{inertial}}$ the inertial torque applied by the motor, $\tau_{m_{damping}}$ the viscous damping torque applied by the motor, $\tau_{b_{inertial}}$ the inertial torque applied by the brake, $\tau_{m_{electro-mechanical}}$ the electromechanical torque applied by the motor, and τ_u the load torque applied by the haptic device's user. Equation (3.20) is equivalent to

$$n_c^2 J_m \frac{\mathrm{d}\omega(t)}{\mathrm{d}t} + n_c^2 B_m \omega(t) + J_b \frac{\mathrm{d}\omega(t)}{\mathrm{d}t} + \tau_b(i_b(t), \omega(t), z(t)) + n_c K i(t) = \tau_u(t). \tag{3.21}$$

The physical configuration of the motor, brake, and capstan drive are displayed in Figure 3.2. Defining $J = n_c^2 J_m + J_b$, (3.21) becomes

$$\frac{\mathrm{d}\omega(t)}{\mathrm{d}t} = -\frac{n_c K}{J} i(t) - \frac{n_c^2 B_m}{J}\omega(t) - \frac{1}{J}\tau_b(i_b(t), \omega(t), z(t)) + \frac{1}{J}\tau_u(t). \tag{3.22}$$

Combining the state space equations from the previous sections, we can derive the complete hybrid actuator process model, with

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= \mathbf{f}\left(\mathbf{x}(t),\, \mathbf{u}(t)\right) \\
&= \mathbf{f}_{\mathbf{x}}(\mathbf{x}(t)) + \mathbf{B}\mathbf{u}(t) \\
\begin{bmatrix} \frac{\mathrm{d}v_m(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}v_b(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}i_m(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}i_b(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}\omega(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}z(t)}{\mathrm{d}t} \end{bmatrix} &= \begin{bmatrix} f_{md}(v_m(t), i_m(t)) \\ f_{bd}(v_b(t), i_b(t)) \\ -\frac{R_m}{L_m} i_m(t) \;+\; \frac{n_c K_m}{L_m}\omega(t) + \frac{1}{L_m} v_m(t) \\ -\frac{R_b}{L_b} i_b(t) + \frac{1}{L_b} v_b(t) \\ -\frac{n_c K}{J} i_m(t) \;-\; \frac{n_c^2 B_m}{J}\omega(t) - \frac{1}{J}\tau_b\left(i_b(t), \omega(t), z(t)\right) \\ f_z(\omega(t), z(t)) \end{bmatrix} \\
&\quad + \begin{bmatrix} K_{md} & 0 & 0 \\ 0 & K_{bd} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{J} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_{cmd_m}(t) \\ v_{cmd_b}(t) \\ \tau_u(t) \end{bmatrix}
\end{aligned} \tag{3.23}$$

and

$$\mathbf{y}(t) = \begin{bmatrix} i_m(t) & i_b(t) & \omega(t) \end{bmatrix}^T, \tag{3.24}$$

with $\mathbf{y}(t)$ the measured output vector, $\mathbf{u}(t)$ the controllable input vector, $\mathbf{x}(t)$ the state vector, and $\mathbf{f}\left(\mathbf{x}(t),\, \mathbf{u}(t)\right)$ the process model. Explanations on the nature of the load torque input $\tau_u(t)$ are given in Appendix A.

The selection of motor and brake electrical models $f_{md}(v_m(t), i_m(t))$ and $f_{bd}(v_b(t), i_b(t))$, as well as the brake's output torque 54 $\tau_b(i_b(t), \omega(t), z(t))$ and hysteresis equation $f_z(\omega(t), z(t))$, will be made in Chapter 4, where initial experimental results will help to choose from the different choices expressed here in Chapter 3.

3.4.2 Constraints

Realistic models not only define the dynamic behavior of a system, but also constraints on its parameters. In most cases, these constraints can be determined from physical intuition. These constraints are there most of the time to assure that both the motor and brake models are passive and thus have stable input to output responses.

It will be assumed, that except for explicitly mentioned exceptions, that all parameters in the process model found in equation (3.23) should be positive.

For example, physical intuition shows that positive parameters make sense when it comes to the parameters in the motor's and the brake's electrical model. A negative resistance would produce energy, and thus be an active component in the circuit. Going forward, it will thus be assumed that all resistance and inductance values are positive, as well as the motor constant K_m. The same assumption can be made when it comes to the mechanical parameters, the inertia and the viscous damping, found in (3.22).

Two exceptions can however be considered. The first concerns the motor and brake drivers. Considering the (3.7) and (3.10), there is a possibility that the drivers induce instability in the system as a whole, because of the amplification done by both drivers, which introduces more energy in the hybrid actuator than the energy provided by the driver's input signals. A detailed stability analysis of what values the parameters of both driver models need to possess could be made by deriving the poles of the $\frac{I_m(s)}{V_{cmd_m}}(s)$ and $\frac{I_b(s)}{V_{cmd_b}}(s)$ transfer functions. However, since the hybrid actuator possesses nonlinear dynamics in (3.22). For example, the feedback effect of τ_b could not be considered in the analysis, since its Laplace transform would involve convolutions. Those would impede the use of traditional stability analysis techniques in the frequency domain which rely on the transfer functions under analysis to be rational functions. For the reasons stated above, the parameters in the driver models will be considered positive.

The second exception appears when considering the passivity of the hysteretic force τ_b. Experimental results from [24] demonstrate that the modified Bouc-Wen model for MR brakes belongs to the Class 1 of Bouc-Wen models as defined in [25]. The following derivations will assume the conditions for the MR brake model to be part of this Class 1 are respected. This implies that

$$\begin{aligned} \gamma + \beta &\geq 0 \\ \gamma - \beta &\geq 0 \\ |z| &\leq z_{max} \\ n_b &\geq 1 \end{aligned}, \tag{3.25}$$

with $z_{max} = max(|z(0)|, z_0)$ and $z_0 = \sqrt[n]{\frac{A}{\beta+\gamma}}$. One of the conditions for this Bouc-Wen model to belong in Class 1 also requires that $\alpha(i_b(t)) > 0$. This statement must be interpreted with cushion, however, because the original Bouc-Wen model takes into account $\alpha(i_b(t))$ and $c(i_b(t))$ are constant coefficients, and not functions of some exogenous signal.

With this in mind, the same assumption will be made for $c(i_b(t))$, meaning $c(i_b(t)) > 0$, since physical intuition shows us that a negative damping coefficient would necessarily make the system non-dissipative, and thus necessarily non-passive. Experimental results from [24]

show it is possible for both $\alpha(i_b(t))$ and $c(i_b(t))$ to be positive if the input voltage is kept positive. To that effect, it will be assumed that $c_0 > 0$, $c_1 > 0$, $\alpha_0 > 0$ and $\alpha_1 > 0$.

When it comes to the Dahl model, the same assumption will be made for the coefficients of $c(i_b(t))$, as well as $\alpha(i_b(t))$, for the same reasons as stated above for the Bouc-Wen model: negative damping could lead to a non-dissipative system, and because the constant coefficient equivalent to $\alpha(i_b(t))$ is per the definition of the model, constrained to be positive [26]. The original Dahl model also states that $\sigma > 0$, which will be assumed to be true for the modified Dahl model as well.

All the parameters in the LuGre model will be assumed positive as well, for the reasons stated above, but also based on the original LuGre definitions from [26].

3.4.3 Jacobian

Because of the absolute value functions found in (3.14), process model linearization is impossible. Indeed, the absolute function is not differentiable, in this case specifically when $\omega(t) = 0$ and $z(t) = 0$. As such, we can approximate (3.14), the hysteresis differential equation, as was done in [27]. In particular,

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f_{z_{bouc-wen}}(\omega(t), z(t)) = \omega(t)(A - (\tanh(\rho z(t))z(t))^n(\beta + \gamma \tanh(\rho\omega z(t)))). \quad (3.26)$$

The higher the value of ρ, the better the approximation, as $\tanh(\rho t) \to \mathrm{sgn}(t)$ as $\rho \to \infty$.

A similar approximation can be made for the Dahl model, that being

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f_{z_{dahl}}(\omega(t), z(t)) = \sigma\omega(t)(1 - z(t)\tanh(\rho\omega(t))). \quad (3.27)$$

Similarly, for the LuGre model,

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f_{z_{lugre}}(\omega(t), z(t)) = \omega(t) - \zeta\omega \tanh(\rho\omega(t))z(t). \quad (3.28)$$

The resulting linearized process model, or Jacobian, can be described as

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_{md}}{\partial v_m} & 0 & \frac{\partial f_{md}}{\partial i_m} & 0 & 0 & 0 \\ 0 & \frac{\partial f_{bd}}{\partial v_b} & 0 & \frac{\partial f_{bd}}{\partial i_b} & 0 & 0 \\ \frac{1}{L_m} & 0 & -\frac{R_m}{L_m} & 0 & \frac{n_c K}{L} & 0 \\ 0 & \frac{1}{L_b} & 0 & -\frac{R_b}{L_b} & 0 & 0 \\ 60 & 0 & -\frac{n_c K}{J} & -\frac{\partial \tau_b}{\partial i_b} & -\frac{n_c^2 B_m}{J} - \frac{\partial \tau_{b_{bouc-wen}}}{\partial \omega} & -\frac{\partial \tau_{b_{bouc-wen}}}{\partial z} \\ 0 & 0 & 0 & 0 & \frac{\partial f_z}{\partial \omega} & \frac{\partial f_z}{\partial z} \end{bmatrix}. \quad (3.29)$$

The undefined partial derivatives for the hysteresis dynamic equation and hysteretic force in (3.29) are as follows, when considering the Bouc-Wen model is used:

$$\frac{\partial f_{z_{bouc-wen}}}{\partial \omega} = A - (\tanh(\rho z(t))z(t))^n[\beta + \gamma \tanh(\rho\omega(t)z(t)) - \gamma\rho\omega(t)z(t)(1 - \tanh^2(\rho\omega(t)z(t)))], \tag{3.30}$$

$$\begin{aligned}\frac{\partial f_{z_{bouc-wen}}}{\partial z} = &- n(\tanh(\rho z(t))z(t))^{n-1}(\rho z(t)(1 - \tanh^2(\rho z(t))) + \tanh(\rho z(t)))(\beta + \tanh(\rho\omega(t)z(t))) \\ &- \gamma\rho\omega(t)(\tanh(\rho z(t))z(t))^n(1 - \tanh^2(\rho\omega(t)z(t))),\end{aligned} \tag{3.31}$$

$$\frac{\partial \tau_{b_{bouc-wen}}}{\partial i_b} = \frac{\alpha_1}{J} z(t) - \frac{c_1}{J}\omega(t), \tag{3.32}$$

$$\frac{\partial \tau_{b_{bouc-wen}}}{\partial \omega} = \frac{1}{J} c\left(i_b(t)\right), \tag{3.33}$$

$$\frac{\partial \tau_{b_{bouc-wen}}}{\partial z} = \frac{1}{J} \alpha\left(i_b(t)\right). \tag{3.34}$$

The same partial derivatives can de derived from (3.17) and (3.27) for the Dahl model, and from (3.19) and (3.28) for the LuGre model.

The undefined partial derivatives for the motor driver model are as follows:

$$\frac{\partial f_{md_{closed}}}{\partial v_m} = -\omega_{cmd_m}, \tag{3.35}$$

$$\frac{\partial f_{md_{closed}}}{\partial i_m} = -K_{cmd_m}\omega_{cmd_m}. \tag{3.36}$$

The same partial derivatives can be derived for the open-loop driver configuration with (3.7). It is trivial to derive the partial derivatives for both the closed- and open-loop configurations for the brake, as they are the same as the ones used for the motor.

The Jacobian found in (3.29) will be used for computational purposes in Chapter 4, as some integration algorithms used to simulate the input to output response of a system require not only the process model of that system, but also its Jacobian.

Chapter 4

Genetic Algorithm Based Nonlinear System Identification

Genetic algorithms at their core are direct search algorithms which solve optimization problems. This implies that, given an objective function to minimize, they will evaluate the latter using multiple randomly generated set of parameters, combining and modifying these sets of parameters over multiple generations to go over a given search space until a satisfactorily low value is attained for one of these sets of parameters. Such an algorithm can be used for system identification, which is a specific type of optimization problem with specific objective functions to ensure that the identified system can reproduce the input to output response of the real world system.

Now that a model for the hybrid actuator as been established, it is necessary to try to identify a set of parameters for this system. This will help in assessing if this model can indeed reproduce a hybrid actuator's experimentally recorded output response. There are many methods to identify the parameters of a system when given such a model. It has been shown that the parameters of an MR brake can be successfully identified with a GA [28]. Since the GA can be applied to any process model for which a set of parameters needs to be identified, the following section will proceed to, instead of exploring other system identification algorithms, improve on this existing algorithm, for ease of use, but also to accommodate for the hybrid actuator model as a whole, which possesses more parameters, more inputs and more outputs than the MR brake by itself.

An appropriate GA needs to be designed to identify such a set of parameters. The next few sections, specifically Section 4.1.1, 4.1.2, and 4.1.3, will go over the different algorithms and techniques that inspired the genetic algorithm for stable system identification, or GASSI, that was designed to identify the parameters of the hybrid actuator built in Chapter 2. This latter algorithm is to be found in Section 4.1.4 and will be used in Section 4.3 to identify the

parameters of the hybrid actuator under study. It is however necessary to fully understand which models, from the many options presented in Chapter 3, are to be identified, and from these models, how each parameter should be estimated before they can be identified, because, as with many iterative optimization techniques, an initial guess needs to be given to the GA before it can infer the desired set of parameters. These choices, both of models and initial parameter guesses, will be made in Section 4.2.

4.1 Working Principles of Genetic Algorithms

Before we explore different GAs and related techniques, it is of the utmost importance to go over a brief explanation of some concepts and vocabulary related to the use of GAs. The first and most important keyword when it comes to the iterative process of the GA, which mimics the process of evolution as based on Darwin's theory, is the chromosome. The chromosome is the set of parameters given to the objective function to evaluate its value. The value of the objective function for a given chromosome is often called that chromosome's fitness, or fitness score. Using the chromosomes' fitness scores, the GA oversees the evolution of a group of chromosomes, which is called a population. Each chromosome evolves individually and through interaction with others. These interactions, which happen during the crossover and mutations steps of the algorithm, are named genetic operators.

When it comes to system identification, a chromosome is a set of system parameters. Each chromosome thus represent a different system, with dynamics that differ greatly or not so much. The chromosomes will change, or evolve, through the iterations of the algorithm, each iteration also being called a generation.

The notation which will be used from now on to represent a chromosome will be as follows:

$$\mathbf{l}_{ij} = \{l_{ij1}, l_{ij2}, \ldots, l_{ijp}\},$$

with $l_{ijk} \in \mathbb{R}$ the value of parameter k in chromosome j during generation i, $i \in [1, 2, \ldots, m]$, $j \in [1, 2, \ldots, n]$, $m \in \mathbb{N}$ is the number of generations the GA will go through, $n \in \mathbb{N}$ is the number of chromosomes in the population, and $p \in \mathbb{N}$ is the number of parameters to be identified. For the sake of brevity, a population will be represented as

$$\mathbf{P}_i = \{\mathbf{l}_{i1}, \mathbf{l}_{i2}, \ldots, \mathbf{l}_{in}\}, \tag{4.1}$$

with P_i being the population of chromosomes at generation i.

Now that this basic notation is set, the inner workings of the GA will be easier to explain.

4.1.1 Classic Genetic Algorithm Optimization

GA algorithms have been around since the 1960s [28], but the specific version of the GA which is of interest in this thesis, the real-coded GA, uses floating point numbers instead of a binary representation for each of the parameters found in a chromosome.

To better understand what makes some GA used in the literature and in this thesis adaptive, it is necessary to first understand what iterative steps the classic GA goes through to navigate to a solution.

The standard algorithm for a real-coded GA, while it may undergo some slight variations, is given in Algorithm 1 [29, 30].

Algorithm 1 Classic Genetic Algorithm

$i \leftarrow 0$
Initialize $\mathbf{P}_i$
Evaluate $\mathbf{P}_i$
while termination condition not reached **do**
    $i \leftarrow i + 1$
    Select $\mathbf{P}_i$ from $\mathbf{P}_{i-1}$
    Cross over the chromosomes in $\mathbf{P}_i$
    Mutate the chromosomes in $\mathbf{P}_i$
    Evaluate $\mathbf{P}_i$
end while

There are many ways to implement the steps in Algorithm 1: initialization, evaluation, selection, crossover, mutation and the termination condition, as is shown in reviews in the literature [30].

The purpose of this thesis is not to go explain every one of these steps, but instead to understand how the algorithm shown in Section 4.1.4 was designed. For that purpose, we will instead try to understand what role each step serves in reaching the optimization goal of the algorithm.

4.1.1.1 Initialization

Initialization is the step that sets the initial parameter values for all chromosomes in the population, at generation 0. The number of chromosomes in the population which will evolve through the iterations of the algorithm is one of the GA's inevitable hyperparameters. More chromosomes in the population will ensure a better coverage of the search space, but will lead to longer convergence times, as the evaluation, crossover and mutation steps iterate over more and more chromosomes. As with any optimization algorithm that needs to be

initialized, a good first estimate of the parameters will lead to the algorithm converging at a faster rate.

4.1.1.2 Evaluation

Evaluation is computing the value of the objective function, or fitness, for every chromosome.

4.1.1.3 Selection

This fitness value is used in the selection step, where chromosomes from population $\mathbf{P}_{i-1}$ are copied into population $\mathbf{P}_i$. Chromosomes with higher fitness values will have a higher probability to be placed into the next generation of chromosomes. This probability is first calculated, and then a sampling algorithm picks chromosomes based on each chromosome's previously computed probability. Both the probability calculation mechanism and the sampling algorithm can be chosen from many options [30].

An example of a probability calculation mechanism is proportional selection. This method calculates the probability by using the formula

$$p_{ij} = \frac{e_{ij}}{\sum_{k=1}^{n} e_{ik}}, \tag{4.2}$$

where e_{ij} is the value of the objective function evaluated at generation i for the j^{th} chromosome, and P_{ij} is this chromosome's probability to get picked for the next generation by the sampling algorithm.

4.1.1.4 Crossover

The crossover step then combines the parameters from randomly selected pairs of chromosomes to form offspring, with the hope that the chromosomes that have been favored during selection pass down their favorable parameters to the next generation.

A possible crossover operator, the arithmetical crossover [31], combines chromosomes α and β of generation i, replacing them in generating $i+1$ with their offspring. The parameters of their offspring are generated via:

$$l_{i+1,\alpha,k} = \lambda l_{i\alpha k} + (1-\lambda) l_{i\beta k} \qquad \forall k \in [1, 2, \cdots, p] \tag{4.3}$$

$$l_{i+1,\beta,k} = \lambda l_{i\beta k} + (1-\lambda) l_{i\alpha k} \qquad \forall k \in [1, 2, \cdots, p] \tag{4.4}$$

The coefficient λ is chosen as a hyperparameter of the genetic algorithm, by giving it a

constant value or an initial value, which then diminishes with every generation.

4.1.1.5 Mutation

The mutation step is there to explore or recuperate parameter values. It changes the parameter values of randomly picked chromosomes. This step ensures that the algorithm does not converge earlier on a suboptimal solution.

4.1.1.6 Termination condition

The termination condition determines when the algorithm stops. This is often done by detecting a certain number of generations, an objective function evaluated under a certain predetermined value or small changes in the objective function value between two consecutive generations. All of these methods require some hyperparameter, a threshold, to indicate when the previously mentioned conditions are reached.

4.1.2 Adaptive Genetic Algorithm for System Identification

Now that the general structure of the GA is known, a choice needs to be made on how the GA will be implemented, specifically for system identification. To narrow the choices of implementation, articles in the literature where GAs are used for identification of the MR brakes Bouc-Wen model parameters were first selected. Many of them exist, but some are not very specific on how exactly the GA is implemented [9, 12]. Others are more thorough when it comes explaining how the GA was implemented, but used an implementation of the GA which requires to choose many hyperparameters, that apply to different steps of the GA [13, 14, 32]. Preliminary simulation results of the hybrid actuator, based on the MR brake model and parameters shown in [9], seemed to indicate that the model's numerical integration time is quite high, possibly due to model nonlinearities and complexity. As such, the GA implementation used for this thesis needed to be succinct in its use of hyperparameters, to reduce time spent running the GA multiples times to narrow down the set of hyperparameters that would lead to convergence.

The algorithm presented in this section is taken directly from [11], where it was presented as the computationally efficient GA, or CEGA. It was the main inspiration for the algorithm presented in Section 4.1.4, which only adds a few elements to the algorithm presented in this section, elements which will be presented in the next few sections.

The main difference between the CEGA with the GA presented in Section 4.1.1 is that the algorithm presented in this section is adaptive, meaning it doesn't use hyperparameters to tune how the crossover, mutation and terminations conditions behave. It also skips the

selection step, and instead goes through an elitism step. The role of the usual selection step is separated into the crossover and elitism step, which only happens after the crossover and mutation steps. These changes in the overall structure of the algorithm can be seen in Algorithm 2.

Algorithm 2 Computationally Efficient Genetic Algorithm

```
i ← 0
Initialize P_i
while termination condition not reached do
    Evaluate P_i
    Save elite chromosome l_ij
    i ← i + 1
    Cross over the chromosomes in P_i
    Mutate the chromosomes in P_i
    Replace r_elite% chromosomes in P_i with l_ij
end while
```

A set of experimental data needs to be collected for the CEGA to identify a set of parameters for the Bouc-Wen model of an MR brake. [11] considers that the dynamic model for the MR brake has for an input the rotor's angular velocity ω, and, as an output, the brake torque τ_b. This dataset will be used to: first, compute the output of the system corresponding to a chromosome's parameters by feeding the numerical simulation of that system the experimental input; and second, compare the simulated output to the experimental output during the objective function evaluation, to reach the ultimate goal of the CEGA, which is to converge towards a set of parameters that minimizes this difference.

4.1.2.1 Initialization

To do so, each chromosome's is first initialized, choosing the value for each parameter from an initial range, range determined by analysis of experimental responses of the system under study. For example, the minimum possible value for the viscous damping of an MR brake would be zero, and its maximum could be estimated as the following slope,

$$c_0 = \frac{\tau_b^{max} - \tau_b^{min}}{\omega^{max} - \omega^{min}}, \tag{4.5}$$

where the maximum values and minimum values for brake torque and angular velocity are recorded experimentally, with a motor applying a sinusoidal displacement on the brake set a fixed input current.

4.1.2.2 Evaluation

The evaluation step here stays the same in essence: each chromosome's fitness score it evaluated based on the algorithm's objective function. However, this fitness score is not simply the objective function evaluated for chromosome $\mathbf{l}_{ij}$, but instead a normalized version of this value. This normalization is usually hidden by the selection probability calculation step of the selection process, by computing the probability in (4.2). This computation does not however happen in the CEGA since it has no selection step.

The fitness score computation first starts by simulating the output response of the system corresponding to each chromosome, given the MR brake's process model, the chromosome's parameters and the experimental input data set.

The objective function value, which in this case is the averaged sum of normalized square error between the simulated and experimental data, is then calculated for each chromosome in generation i:

$$e_{ij} = \frac{1}{n_d} \sum_{k=1}^{n_d} |\bar{\omega}_k| \left(\frac{\tau^k_{b_{sim}} - \tau^k_{b_{exp}}}{\Delta\tau_{b_{exp}}} \right)^2 \forall j \in [1, 2, \cdots, n], \tag{4.6}$$

with

$$|\bar{\omega}_k| = \frac{2(\omega^k_{exp} - \min_i\{\omega^i_{exp}\})}{\max_i\{\omega^i_{exp}\}} - 1, \tag{4.7}$$

and

$$\Delta\tau_{b_{exp}} = \max_i\{\tau^i_{b_{exp}}\} - \min_i\{\tau^i_{b_{exp}}\}, \tag{4.8}$$

where n_d is the number of data points in the input to output dataset, the i^{th} experimental angular velocity is ω^i_{exp}, the i^{th} experimental brake torque is $\tau^i_{b_{exp}}$, and the i^{th} simulated brake torque is $\tau^i_{b_{sim}}$. e_{ij} is also informally named as the error of chromosome j in generation i, for the sake of simplicity, but also to distinguish it from the fitness fit_{ij}.

Each chromosome's fitness score is computed and normalized using Algorithm 3:

Algorithm 3 CEGA Fitness Normalization Algorithm

$fit_{ij} \leftarrow 1 - \frac{e_{ij}}{\max_k\{e_{ik}\}} \forall j \in [1, 2, \cdots, n]$
$fit_{ij} \leftarrow fit_{ij} - \min_k\{fit_{ik}\} \forall j \in [1, 2, \cdots, n]$
$fit_{ij} \leftarrow \frac{fit_{ij}}{\max_k\{fit_{ik}\}} \forall j \in [1, 2, \cdots, n]$

The resulting $fit_{ij} \in [0, 1] \forall j \in [1, 2, \cdots, n]$.

4.1.2.3 Elitism

This step replaces the selection step in Algorithm 1. Its purpose is to counteract the destructive effects of the crossover, but mostly the mutation step, which in some situations could bring all the chromosomes further away from the ideal solution. The only hyperparameter of the CEGA, except for the number of chromosomes, is found in this step. To be precise, the elitism is separated into two parts, and thus could be considered to be 2 steps. The first part is the selection of the i^{th} generation's best chromosome $\check{\mathbf{l}}_i$. To find such a chromosome, the smallest error value for that generation must be determined. This smallest error value

$$\check{e}_i = \min_j fit_{ij} \tag{4.9}$$

is then compared to the smallest error value from the previous generation, $\check{e}_{i-1}$. If $\check{e}_i < \check{e}_{i-1}$, then $\check{\mathbf{l}}_i = \mathbf{l}_{ik}$ such that $k = \arg\min_j e_{ij}$. If not, then $\check{\mathbf{l}}_i = \check{\mathbf{l}}_{i-1}$ and the value of $\check{e}_i$ is now changed to the value of $\check{e}_{i-1}$.

The error is used here since it is the only metric that can be used to compare objective function values between generations. In the CEGA, the fitness value is normalized over the other fitness values in the generation, rendering comparison between generations with the fitness score inaccurate.

The second part of the elitism step is the replacement step. This step comes after the mutation step, which might erase the progress of the algorithm by modifying the better fitting chromosomes. Some randomly selected chromosomes, an r_{elite} percentage of the population, is replaced by $\hat{\mathbf{l}}_i$, to keep the good genetics of this elite chromosome in the population. This percentage should be chosen such that $n\frac{r_{elite}}{100}$ is a whole number, with a low enough value such that the exploitation potential of the elitism step doesn't overshadow the exploitation potential of the mutation step.

4.1.2.4 Crossover

Similar to the arithmetical crossover show in (4.3) and (4.4), the proposed crossover operator serves also as a selection operator. Instead of just combining the genetics of the parent chromosomes and giving these genes to the equivalent offspring, only one offspring is produced, replacing the parent with the least fitness with a combination of both parents, favoring the parameters of the fittest parent when computing the genes of the offspring. This technique is demonstrated in Algorithm 4.

Algorithm 4 CEGA Crossover Algorithm

for n times **do**
  Pick chromosomes $\mathbf{l}_{i\alpha}$ and $\mathbf{l}_{i\beta}$, with α,β chosen randomly from $[1, 2, \cdots, n]$
  if $fit_{i\alpha} > fit_{i\beta}$ **then**
    $l_{i+1,\beta,k} = \lambda l_{i\alpha k} + (1-\lambda) l_{i\beta k} \forall k \in [1, 2, \cdots, p]$
  else
    $l_{i+1,\alpha,k} = \lambda l_{i\beta k} + (1-\lambda) l_{i\alpha k} \forall k \in [1, 2, \cdots, p]$
  end if
end for

Here λ is not a hyperparameter but is instead $\in \mathcal{U}(0,1)$ and is randomly generated at each generation of the CEGA. This is in part what makes the CEGA an adaptive algorithm. There is no need to tune the ratio λ.

4.1.2.5 Mutation

The mutation algorithm is given in Algorithm 5.

Algorithm 5 CEGA Mutation Algorithm

for n times **do**
  Pick chromosomes $\mathbf{l}_{i\alpha}$ with α chosen randomly from $[1, 2, \cdots, n]$
  if $fit_{i\alpha} < r$ **then**
    $l_{i+1,\alpha,k} = l_{i\alpha k} + r g_k \forall k \in [1, 2, \cdots, p]$
  end if
end for

Here $r \in \mathcal{U}(0,1)$ and is generated at each generation of the CEGA, while $g_n \in \mathcal{N}(0, \sigma_k^2)$ and is generated for each parameter of $l_{i\alpha k}$ when it is picked by the mutation algorithm. σ_k is the standard deviation of the value of parameter k in the initial population of chromosomes.

4.1.2.6 Termination condition

The CEGA's termination condition is based on the running mean of the error,

$$\bar{e}_i = \frac{1}{i-g} \sum_{k=g}^{i} \check{e}_k, \tag{4.10}$$

where g in this case is the first generation in the CEGA where $\check{e}_k$ changes from one value to the other. As such, $g > 1$.

The CEGA uses this running mean to stop the iterative process when the expression

$$((i > i_{\min}) \wedge (\bar{e}_i \leq r_{\text{termination}} \check{e}_0)) \vee (i > i_{\max}) \tag{4.11}$$

is true, with $i_{\min}$ the number of generations before the CEGA can stop, with$i_{\max}$ the maximum number of generations the algorithm can pass through, and $r_{\text{termination}}$ the ratio between the initial value of $\check{e}_k$ and the running mean used to determine when the error has settled.

4.1.3 Constraint Handling in Genetic Algorithms

An adaptive GA has been presented in Section 4.1.2. This method has shown real promise when it comes to system identification. Early tests of such a technique revealed real promise when it comes to recreating the parameters of a simulated system. This means that the CEGA in itself, while it may have a hard time identifying the parameters of an experimental system, since the system's model might be inaccurate, and the experimental data may be noisy, is able to do its job in an ideal environment, where the user knows that the data fed to the algorithm is exactly the result of the model for which it is trying to find the parameters.

This was the case most of the time: the data from a simulated mass-damping-spring system was fed to the CEGA, which had access to that's systems process model, and a set of parameters similar to the ones first given to the simulated system would be produced. However, the algorithm would sometimes never converge when a change was made to the CEGA's few hyperparameters, for example when a change was made to the number of chromosomes or the seed provided for the random number generators. The cause of this problem was determined to be unbounded output responses. For some chromosomes, the value of the simulated force output would get so big that it would be greater than the maximum floating point number value, and the numerical integration algorithm would then stall and keep trying to simulate the system's output indefinitely. It was then determined that this was caused by a negative parameter value, for example a negative viscous damping value, which would, instead of dissipating energy from the velocity input, introduce more energy into the simulated system.

The ideal GA would not let the parameters being identified converge towards such values, which causes instability. The authors of [11] insured this undesired behavior would not happen by initializing each parameter's value above zero. However, the random mutation operator, as presented in Algorithm 5, can bring the parameter's value out of their initial estimated range. This does not always happen, since the change made on each parameter is bounded at each generation by the standard deviation of the original population, but it can happen after a few generations.

This introduces the need for constraining the possible parameter values that the GA should allow. Constraints are not a new concept in the GA literature, but a specific strategy needs to be applied, one that is appropriate for the kind of problem which the GA is trying to solve in this thesis, which is system identification.

There are many ways to constrain the evolution and mutation of the sets of parameters, or chromosomes, in genetic algorithms. A few of these constraint handling methods specifically designed for GAs will be shown in this section. This will not however be a comprehensive review, which already exists in the GA literature [33–35]. This section will instead strive to demonstrate algorithms from some of the most widely recognized categories of constraint handling methods and explain why the one chosen for the algorithm designed specifically for this thesis was picked over the others.

4.1.3.1 Feasibility preserving representation and operators

One way to constrain the value of each parameter in the GA's chromosomes is to pick specific genetic operators, and the spaces where the parameter values can be found. Some of these techniques explore a combination of convex spaces, which are set by linear constraints, and closed operators, to keep the resulting sets of GA solutions within the constraints. One such strategy is the GEnetic algorithm for Numerical Optimization for COnstrained Problems, or GENOCOP [30, 36]. The issue with such this strategy is that it requires, to define some of its crossover and mutation operators, domain constraints of the following shape:

$$l_k^{\max} \geq l_{ijk} \geq l_k^{\min} \qquad \forall i \in [1, 2, \ldots, m], j \in [1, 2, \ldots, n], k \in [1, 2, \ldots, p] \tag{4.12}$$

Most parameters, when it comes to system identification, require one-sided constraints, which are imposed by some impossibility when to the laws of physics. For example, a negative resistance value in a circuit with no active components. It is however possible to determine a theoretical upper bound for some parameters, based experimentally measured quantities. As mentioned in Section 4.1.2, this is done [11] with to determine a theoretical maximum damping value for an MR brake Bouc-Wen model. However, there should be a distinction between bounds which define a region outside which the identified parameter values would be infeasible, due to physical laws, and bounds which define outside which the identified parameter values would be improbable, due to measured experimental results. The initial estimated parameter range in the CEGA defines the second type of region, and this should stay as it is, since this incites the algorithm not to wander too far out of this region, for the sake of exploitation, but to still wander out in some occasions, for the sake of exploration. The GENOCOP method does not make this distinction and would treat improbable solutions as infeasible, and this is why it was not used in this thesis.

4.1.3.2 Repair algorithms

Other constraint handling techniques, instead of totally impeding the movement of chromosome towards infeasibility, instead repair the chromosomes, meaning they push them back towards the feasible space when they reach the infeasible space. The approach in [37] is based on the gradient of inequality and equality constraints. This method, instead of changing the GAs operator to constrain the identified parameters, instead computes a repair vector which is added to the corresponding chromosome to repair chromosomes which are found added to The inequality constraints are a bit more permissive when it comes to this algorithm, as there is no need to provide two-sided inequalities for every parameter being identified. However, this method is not adaptive: it requires tuning some hyperparameters. The hyperparameters in question are a repair rate, which determines the probability that an infeasible chromosome is repaired when detected as such, and a minimum adjustment scale, which is used to determine when the iterative process to repair chromosomes has failed.

4.1.3.3 Penalty functions

Both of the constraint handling methods can be quite intrusive when it comes to the overall structure of a GA. They often disrupt the genetic operators, or make additional changes to the chromosomes before or after the genetic operators, which require that the algorithm as a whole be designed around these constraint handling methods. They are thus as a design tool less practical, giving less leeway to the designer to pick and choose each of the GA' steps. Penalty functions don't suffer from this same problem. They instead only modify the evaluation step, by penalizing the fitness score of chromosomes that possess infeasible parameters.

In keeping with the spirit of the previous section, Section 4.1.2, it would be preferable to find a constraint handling technique which is also adaptive, to reduce the number of hyperparameters that need to be tuned in order for the GA to converge to a set of parameters that respects the inequality and equality constraints set by the user of the GA. It should not enforce the use of two-sided inequalities for every parameter. Such a technique is demonstrated in [38].

This constraint handling methods works in three steps. The first step is to give each chromosome, at generation i, an infeasibility score. Then, chromosomes that define the boundaries of the search space are selected, which helps in computing how penalties should be applied to infeasible chromosomes in the last step.

The notation used for constraints in [38] is as follows. Inequality constraints are defined

as

$$g_k(\mathbf{l}_{ij}) \leq 0 \qquad \forall k = [1, 2, \ldots, m_c], \tag{4.13}$$

and equality constraints as

$$h_k(\mathbf{l}_{ij}) = 0 \qquad \forall k = [m_c + 1, m_c + 2, \ldots, m_c + m_h]. \tag{4.14}$$

With these constraints the infeasibility score can be calculated. Constraint violation values are first calculated

$$c_k(\mathbf{l}_{ij}) = \begin{cases} \max(0, g_k(\mathbf{l}_{ij})), & \text{if } 1 \leq k \leq m_c \\ \max(0, |h_k(\mathbf{l}_{ij}) - \delta|), & \text{if } m_c + 1 \leq k \leq m_c + m_h \end{cases}, \tag{4.15}$$

with δ a small tolerance value, since no chromosome will probably possess the exact value of the equality constraint.

From (4.15), the infeasibility score for a chromosome $\mathbf{l}_{ij}$ will be given by

$$i(\mathbf{l}_{ij}) = \frac{\sum_{k=1}^{m_c+m_h} \frac{c_k(\mathbf{l}_{ij})}{\max_q\{c_q(\mathbf{l}_{ij})\}}}{m_c + m_h}. \tag{4.16}$$

It is to be noted that a population with all feasible chromosomes, meaning $i(\mathbf{l}_{ij}) \forall j \in [1, 2, \ldots, n]$, doesn't need to go through the rest of the constraint handling algorithm, as none of the chromosomes need to be penalized.

Now that an infeasibility score can be computed, indicators for the scale of the penalty to apply to the infeasible chromosomes need to be selected. The indicators are the most extreme points of the current search space at generation i. The first indicator is the best individual $\check{\mathbf{l}}_i$, which is the feasible chromosome with the lowest objective value $\check{e}_i$, or, if there are no feasible chromosomes, the chromosome with the lowest infeasibility score.

The second indicator is the worst infeasible solution $\hat{\mathbf{l}}_i$. When all infeasible chromosomes have higher objective function values e_{ij} higher than $\check{\mathbf{l}}_i$, this chromosome is either the chromosome with the highest infeasibility score, or the chromosome with the highest infeasibility score and the highest e_{ij} value, when two or more chromosomes share the highest infeasibility score. When some infeasible chromosomes have lower objective function values e_{ij} higher $\check{\mathbf{l}}_i$, the worst infeasible solution is the one with the highest infeasibility and a value of e_{ij} under $\check{e}_i$. If there exist two such chromosomes with equal highest infeasibility values, the worst infeasible solution is the one with the lowest objective function value e_{ij}. This second indicator objective function value is $\hat{e}_i$.

The third indicator of a population's overall infeasibility is the chromosome $\breve{\mathbf{l}}_i$, that

possesses the highest objective function value $\check{e}_i$.

With the indicator chromosomes now established, the penalty can now be applied to the objective function value of each infeasible chromosome.

The first penalty only applies if there exists an infeasible chromosome with an objective function value that is lower than $\check{e}_i$. If so, all infeasible chromosomes are penalized, their objective function value shifted from $e_i j$ to $\prime e_{ij}$, according to the following formula:

$$\tilde{i}(\mathbf{l}_{ij}) = \frac{i(\mathbf{l}_{ij}) - i(\check{\mathbf{l}}_i)}{i(\hat{\mathbf{l}}_i) - i(\check{\mathbf{l}}_i)} \tag{4.17}$$

$$e'_{ij} = e_{ij} + \tilde{i}(\mathbf{l}_{ij})(\check{e}_i - \hat{e}_i) \tag{4.18}$$

When the first penalty is not applied, $e'_{ij} = e_{ij}$. The second penalty is applied as follows:

$$e''_{ij} = e'_{ij} + \gamma_{\text{penal}}|e'_{ij}| \left(\frac{\exp(2\tilde{i}(\mathbf{l}_{ij})) - 1}{\exp(2) - 1} \right) \tag{4.19}$$

$$\gamma_{\text{penal}} = \begin{cases} \frac{\check{e}_i - \check{e}_i}{\check{e}_i}, & \text{if } \hat{e}_i \leq \check{e}_i \\ 0, & \text{if } \hat{e}_i = \check{e}_i \\ \frac{\check{e}_i - \hat{e}_i}{\hat{e}_i}, & \text{if } \hat{e}_i > \check{e}_i \end{cases} \tag{4.20}$$

The fitness score for each chromosome is then computed with the following equation, whether they were penalized or not:

$$fit_{ij} = \max_j e''_{ij} - \begin{cases} e''_{ij}, & \text{if } i(\mathbf{l}_{ij}) \neq 0 \\ e_{ij}, & \text{if } i(\mathbf{l}_{ij}) = 0 \end{cases} \tag{4.21}$$

4.1.4 Genetic Algorithm for Stable System Identification

The problem that is being solved in this thesis, which is the successful identification of a hybrid actuator's model parameters, can only be solved through a system identification algorithm that can respect a specific set of specifications. The algorithm designed in this paper, through combination of previously used techniques found in the literature, will be designated as a genetic algorithm for stable system identification, or GASSI.

The first important specification for this algorithm is that it needs to be adaptive. This is not a requirement per se for a GA, as many GA used for system identification are not adaptive, and they can without a shadow of a doubt produce a set of parameters that can be used to represent the system being identified with a satisfactory degree of utility [13, 14,

32]. However, a system identification algorithm should be useful in the sense that it should be versatile and easy to implement in an engineering pipeline, where system identification is only but a step in the overall engineering process. Since the motivation for this thesis originally came from collaboration with an industrial partner with the intent purpose of eventually designing a product, the system identification tool developed throughout the project needed to be intuitive, without the need for unnecessary guess work when it comes to hyperparameter tuning. The system identification algorithm designed here is meant to be used not just as a research tool, but also as an engineering tool. This is where the algorithm described in Section 4.1.2 comes into play as the principal inspiration of the GASSI.

However, changes needed to be made to this original algorithm to account for the differences in the exact problem being solved. First, the CEGA, as a system identification algorithm, needed to be generalized to work with not only single-input, single-output systems, or SISO system, but also multiple-input, multiple-output systems, or MIMO systems. This can be explained by the fact that the hybrid actuator is designed as a combination of different systems, each with their own collection of inputs and outputs that feedback and interact with each other. It is to be expected that such a system will be MIMO. As with most of the changes made to the algorithm found in [11], the CEGA, the necessary modification to account for MIMO systems was applied to the evaluation step of this specific GA. To that effect, a technique inspired by [39] was applied to modify the objective function, in such a way that it is the mean value of the normalized least-square error of each output

$$\begin{aligned} e_{ij} = & \frac{1}{3}\sqrt{\sum_{k=1}^{n_d}\left(\frac{i^k_{m_{sim}} - i^k_{m_{exp}}}{\max_t |i^t_{m_{exp}}|}\right)^2} \\ & + \frac{1}{3}\sqrt{\sum_{k=1}^{n_d}\left(\frac{i^k_{b_{sim}} - i^k_{b_{exp}}}{\max_t |i^t_{b_{exp}}|}\right)^2} \\ & + \frac{1}{3}\sqrt{\sum_{k=1}^{n_d}\left(\frac{\omega^k_{sim} - \omega^k_{exp}}{\max_t |\omega^t_{exp}|}\right)^2} \\ & \forall j \in [1, 2, \cdots, n], \end{aligned} \tag{4.22}$$

with equal weight given to each of the normalized least-square error of each of the hybrid actuator model output. Other weights could be given if model accuracy was to be prioritized for one output over the other.

The second modification to the evaluation step was the addition of the penalty function constraint handling algorithm from [38], as presented in Section 4.1.3. The penalty function is applied directly after (4.22) to all infeasible chromosomes, but the fitness is not computed

with (4.21), but instead with Algorithm 3. The reason for this choice is two-fold. First, (4.21) was not used because the equation in [38] does not seem to fit with graphical results showing the effects of the fitness calculation on an example population of chromosomes, which might be due to a badly worded explanation of how the fitness should be calculated. Second, the fitness normalization algorithm from [11] was chosen instead, as the mutation and operators of the CEGA rely on fitness values between 0 and 1, which is not guaranteed by the fitness computation as found in [38].

Other minor modifications were made to the evaluation step. These are made with the intent of dealing with numerical integration results of unstable systems. The constraint handling algorithm only reacts to unstable systems, and is not proactive, compared to the GENOCOP for example, which means that some chromosomes will sometimes fall outside the pre-established constraints and can thus correspond to unstable systems.

Two events can happen during the evaluation step, where the parameters for each chromosome is are used to build the corresponding system, which is then numerically integrated. These errors in numerical integration can impede the convergence of the GA to a suitable solution, if not dealt with as is done in the GASSI. First, the algorithm sometimes stalls during numerical integration. Whether this is due to bad conditioning from specific combinations of systems parameters or due to those parameters corresponding to unstable systems, matters not. What matters is that these chromosomes should be flagged, and the corresponding parameters discarded, without needing to restart the GA completely. To do so, a timeout mechanism was put in place, to stop the numerical integration after a set amount of time. A good measure of how long the algorithm should wait before it times out is at least 3 times over the time it takes to numerically integrate a system that falls within the initial parameter range. Any chromosome that requires numerical integration over that amount of time is flagged. The second event happens when numerical integration produces an output response that is unbounded. This type of event can be detected when calculating (4.22), as the resulting value should go over the maximum floating point value, and thus can be checked to have a value of either positive or negative infinity. If so, the corresponding chromosome should be flagged. When any of those two events is detected and flagged, the GA should give the corresponding faulty chromosomes a fitness score of zero. This will ensure that those chromosomes, if they are randomly picked during the mutation and crossover steps, will always be modified and hopefully brought within the feasible zone by the respective operators, which is not guaranteed when only using the constraint handling methods.

This error detection sequence that follows numerical integration is the last change that makes the GASSI deviate from its original source, the CEGA. Nonetheless, before the GASSI can be used on experimental data, its functionality as a system identification method needs

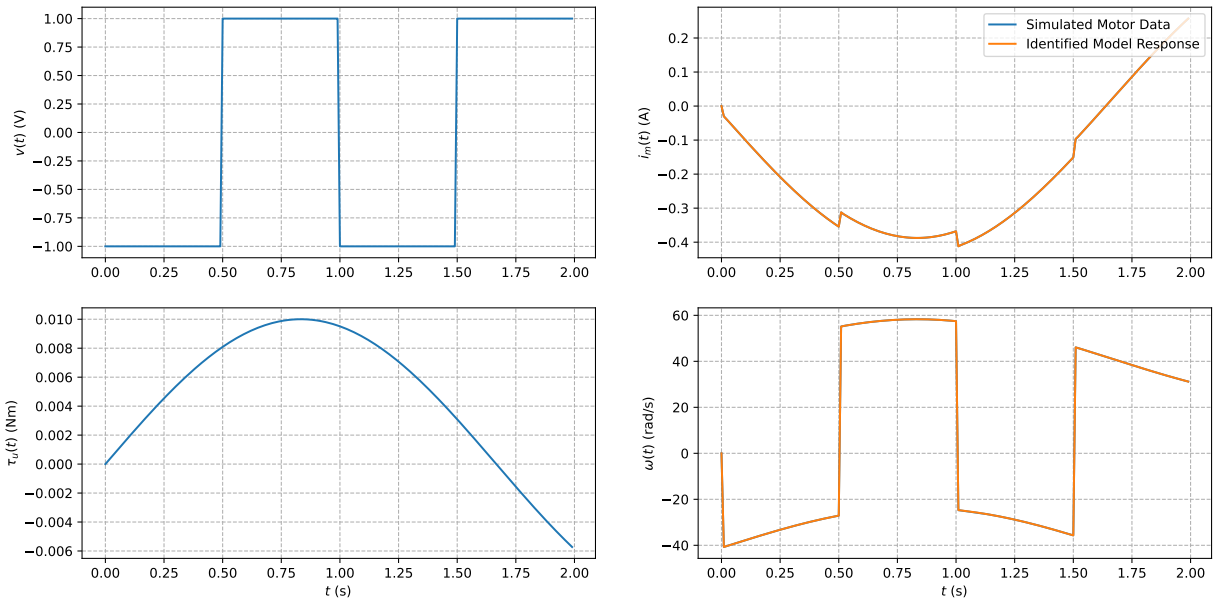

Figure 4.1: Comparison between a training dataset from a simulated DC motor and the response of the identified DC motor model to that training dataset. The identified response completely covers the experimental training data.

to be validated. To do so, a simple DC motor was modeled and simulated. The motor's process model, and the aforementioned simulated data were fed to the GASSI, with initial parameter estimates ranging from one order of magnitude below the real parameters of the simulated motor, to one order of magnitude above. The GASSI ran for $m = 50$ generations, with a population of $n = 400$ chromosomes, and an elitism replacement rate of $r_{\text{elite}} = 1\%$. Fig. 4.1 and Fig. 4.2 show the results of training and validating the motor model identified by the GASSI. As can be seen, the identified response is able to follow the simulated response quite well, with a training error given by (4.22) of 0.0000267, which confirms that the GA implementation designed in this thesis can be used for stable system identification, when provided with an ideal model and input to output data.

4.2 Choosing a Model and Finding Initial Estimates of Physical Parameters

As will be shown in Section 4.3, the implementation of GA used for system identification in this thesis, the GASSI, has its limitations. The one limitation that hinders the collection of experimentally identified sets of model parameters the most is computation time. While it would be ideal to collect parameters for different driver and brake hysteresis models, time limitations only allowed for the choice of one such model where multiple choices were

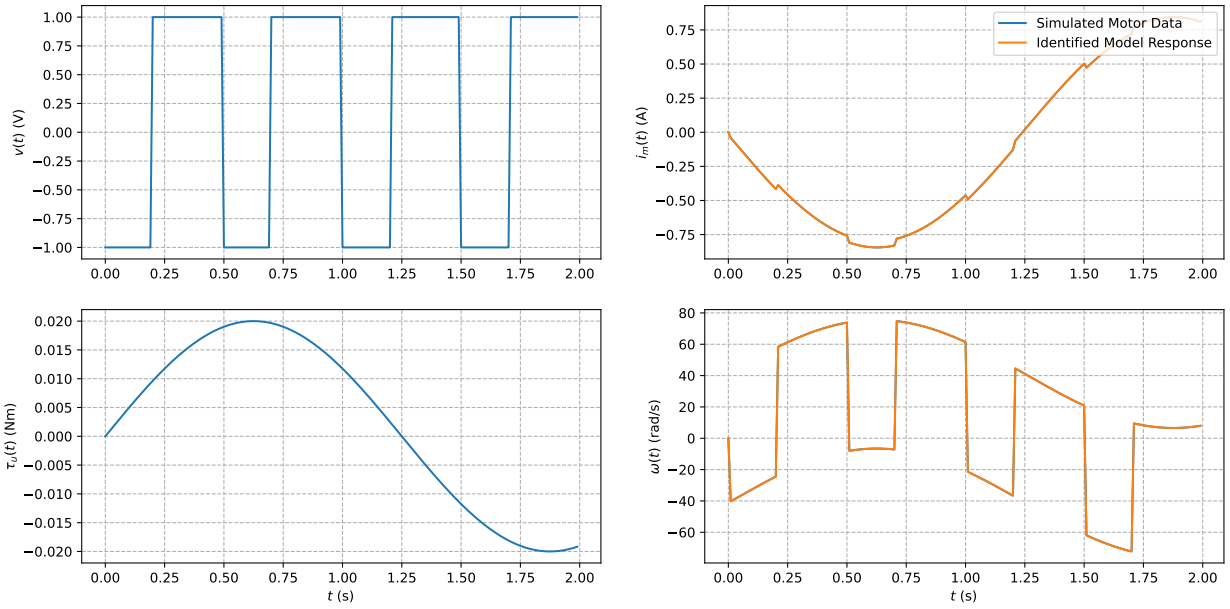

Figure 4.2: Comparison between a validation dataset from a simulated DC motor and the response of the identified DC motor model to that validation dataset. The identified response completely covers the experimental validation data.

available. The most obvious choices based on previously used models in the literature are further detailed in Chapter 3. This chapter will explain why one model was chosen over another, and how initial values for such parameters were chosen.

4.2.1 Electrical Systems

The selection of initial parameters ranges for the electrical systems was fairly easy. Inductance and resistance for the DC motor were available in its specification sheets. Resistance was also available for the MR brake.

With such parameters in hand, it was easy to simulate the electrical response of the motor and the brake, ignoring the effects of back-emf. To do so, a driver model was first chosen. The closed-loop model seemed to be the appropriate choice, as experimental data demonstrated error integration in the electrical response from command voltage v_{cmd_m} to current i_m.

Given the data sheet parameters, and a conservative guess for the MR brake inductance, with the same order of magnitude as the DC motor inductance, simulation results for the brake and motor were compared to experimental data. The initial value for the driver cutoff was given as 400 rad/s, and the driver gain at 10.

As can be seen in Fig. 4.3, a sufficiently accurate estimate of the motor's model parameters can be achieved by inspection. Slowly changing the driver parameters, and adjusting the

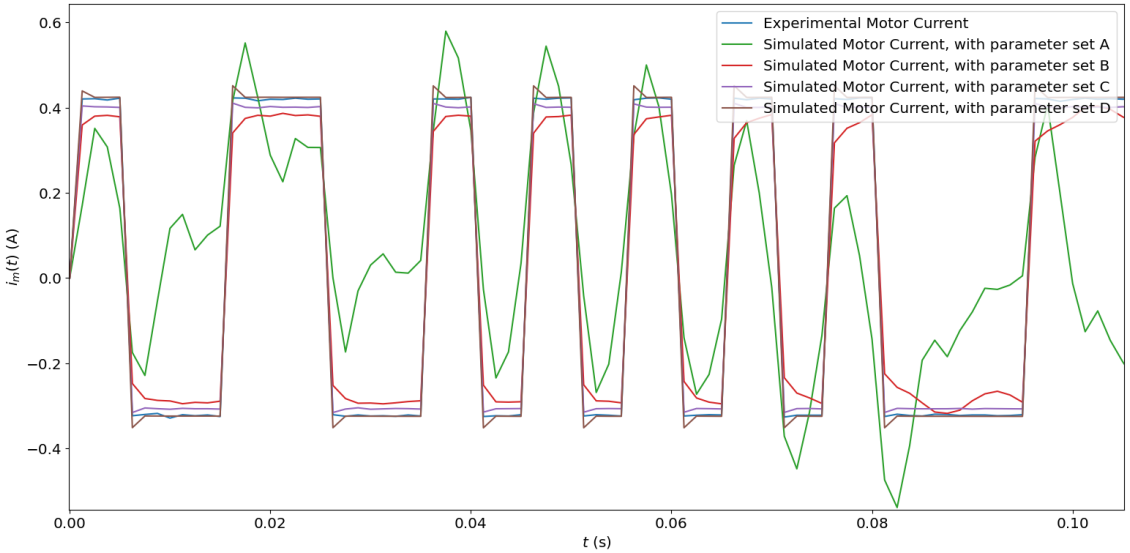

Figure 4.3: The experimental electrical response was iteratively compared to the output response of different simulated systems. The change in parameter values between each iteration is recorded in Table 4.1.

Parameter Set	K_{cmd_m} (unitless)	$\omega_{cmd_m}(rad/s)$	R_m (Ω)	L_m (mH)	K_m (N·m/A)
A	10.0	400	1.18	4.80	0.141
B	29.6	1046	3.44	0.912	0.373
C	112	1828	6.61	1.18	0.0597
D	379	4517	0.255	0.768	0.0985

Table 4.1: The electrical parameter estimates were adjusted by trial and error, taking note of the changes in overshoot and response time of the electrical output response between each iteration to adjust the corresponding parameters.

resistances and inductances of the motor and brake, led to a satisfying approximation of the electrical parameters of the hybrid actuator.

A variation of 10% over and under the guessed parameter values approximated using the method explained above was then used as the initial parameter range given to the GASSI.

4.2.2 Hysteresis Model

The choice of dynamic hysteresis model was based on results found in the literature. As shown in [15] and [21], the Bouc-Wen model seems to be a clear winner. [15] shows that identification of parameters by comparison of experimental and simulated, in respect to both the frequency and amplitude of both signals, yields smaller accumulated differences for the Bouc-Wen model than the Dahl model. [21] shows that the voltage-dependent Bouc-Wen model demonstrated lower normalized error norms to both the modified Dahl and LuGre models, and comparable normalized errors to the modified algebraic, and non-dynamic model, presented in that paper. The voltage-dependent Bouc-Wen model also outperformed the Dahl and LuGre models in terms of mean and average deviation from experimental data at all tested velocities, notably with the most consistent average deviation out of all the models presented in [21], dynamic or not.

With the Bouc-Wen model selected, good guesses for the initial value of its parameters is now crucial to ensure fast convergence of the GASSI. However, it is hard, by observing experimental data, to estimate the values of the hysteresis parameters in the Bouc-Wen model by observing experimental data, as was stated in [11]. Most of the parameters in this model do not represent physical quantities, but instead exist as a way to adjust the phenomenological behavior of the model.

An estimate for values of c_0 and c_1 might be guessed from an angular velocity ω to brake torque τ_b graph, but that graph can only be produced by placing the torque sensor between the motor and the brake, and recording τ_b at constant velocities, with a constant brake current i_b.

The experimental setup in Chapter 2 was modified to fulfill the requirements stated above, and the data collection software as well. The angular velocity of the brake was controlled by a PID controller driving the motor command voltage. However, friction from the brushes in the torque sensor introduced nonlinear behavior in the recorded angular velocity to torque relationship, which overrode the hysteresis curve. An attempt at producing a model for the observed friction was made, comparing steady-state angular velocity to torque, with the intent of exploring friction models as presented in [40]. This did not bear fruit, as the even in steady-state, both the angular velocity and torque signals were very noisy. This behavior can be seen in Fig. 4.4. The cause of this noise might have been electrical interference, or

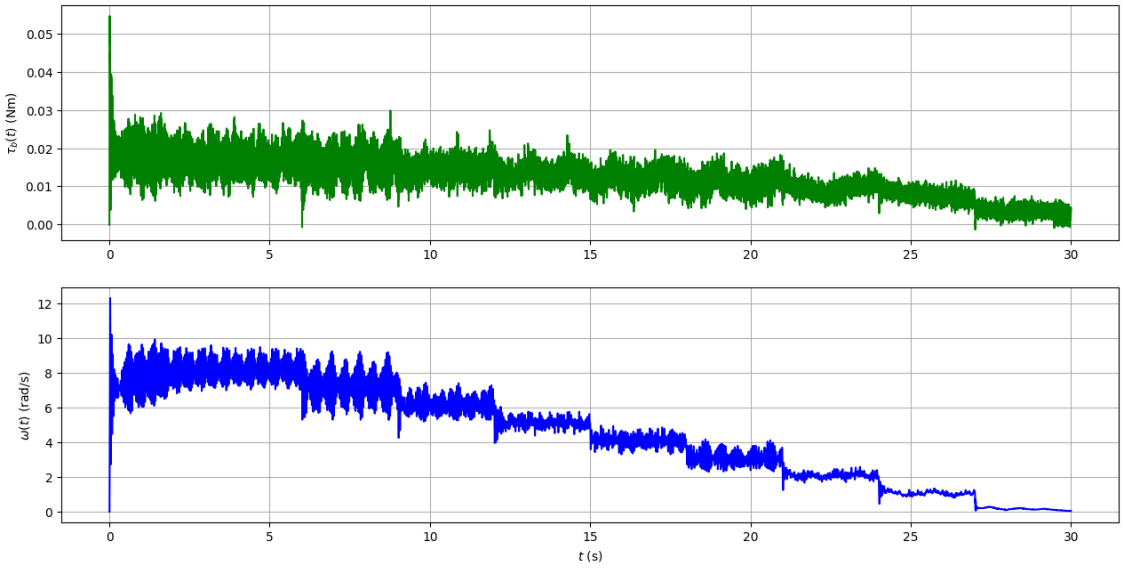

Figure 4.4: Both the torque and angular velocity responses where very noisy, much more than in previous work where torque sensor captured responses of rotational MR brakes were shown [15]. Some of this noise might actually be cyclical friction patterns caused by torque sensor brush geometry defects.

cyclical friction behavior, caused by the buckling and unbuckling of the brushes in the torque sensor.

As such, attempts at modeling this friction were left for future work. The choice of initial parameter ranges for the Bouc-Wen model were thus simply left to trial and error.

4.3 Experimental Results

With a hybrid actuator process model derived, the initial parameter ranges now set, and the GASSI implemented, it is now time to identify the hybrid actuator parameters. Before doing so, input and output was gathered on the experimental setup shown in Chapter 2, at a sampling rate of 800 Hz, over 1.25 seconds. The voltage command signals applied on the motor and brake drivers were pseudo-random binary sequences, also known as PRBS, with an amplitude of 0.75 V and an offset of 0.1 V for the motor command signal, and an amplitude of 0.07 V and an offset of 0.07 V for the brake command signal. The user torque was applied by a mass of approximately 250 grams, attached to a cable, which was in turn attached to the load pulley. The user torque was not actuated by the hybrid actuator control system, but instead by the reaction of the load pulley system to the angular velocity output of the hybrid actuator. More information on this feedback behavior can be learned in Appendix A.

PRBS signals are usually recommended for system identification [16], as they cover wide frequency spectrums and thus stimulate a wide range of dynamic resonances in the system being identified. The PRBS signal applied on the motor is slightly offset. This is not ideal, since the average angular velocity for a hybrid actuator in most haptic applications is zero, but this offset was necessary to keep the load pulley, which was wound for five turns before the start the data collection process, from completely unspooling. The transition from a wound load pulley to an unwound load pulley would have changed the displayed user torque set point during the data collection process. Having a constant set point on all inputs and outputs helps in the identification of a linear model, which is done in Section 4.3.2. The offset on the brake voltage can be explained by the fact that the brake voltage needs to stay positive, as shown by the constraints on its hysteresis model set in Section 3.4.2, that state that $\alpha(i_b(t))$ and $c(i_b(t))$ need to stay positive too.

4.3.1 GA Identified Model and its Response

With the experimental data in hand, the GASSI implementation, as presented in Section 4.1.4 and Section 2.3, was set compute a set of parameters over $m = 40$ generations, with $n = 300$ chromosomes and a replacement rate of $r_{\text{elite}} = 0.5\%$. It was found that it could compute such a set of parameters once every ten hours when set with the previously mentioned hyperparameters and initial parameter ranges.

The GA seems to be able to identify a model for the linear parts of the hybrid actuator model, which are the electrical current outputs, that follows experimental data quite closely. The same cannot be said for the nonlinear parts of the hybrid actuator model, which produce the angular velocity response. This can be seen in Fig. 4.5, where experimental data is compared to the output of the identified model.

Output Signal	RMSE (unitless)	MSD (unitless)
$i_m(t)$	0.0149	0.00159
$i_b(t)$	0.0709	0.00155
$\omega(t)$	0.179	0.0157

Table 4.2: Error metrics between a training dataset from the experimental hybrid actuator and the response of the identified hybrid actuator model to that training dataset show that the GASSI identified the linear outputs of the hybrid actuator system quite well, but performed less ideally when it comes to the nonlinear output.

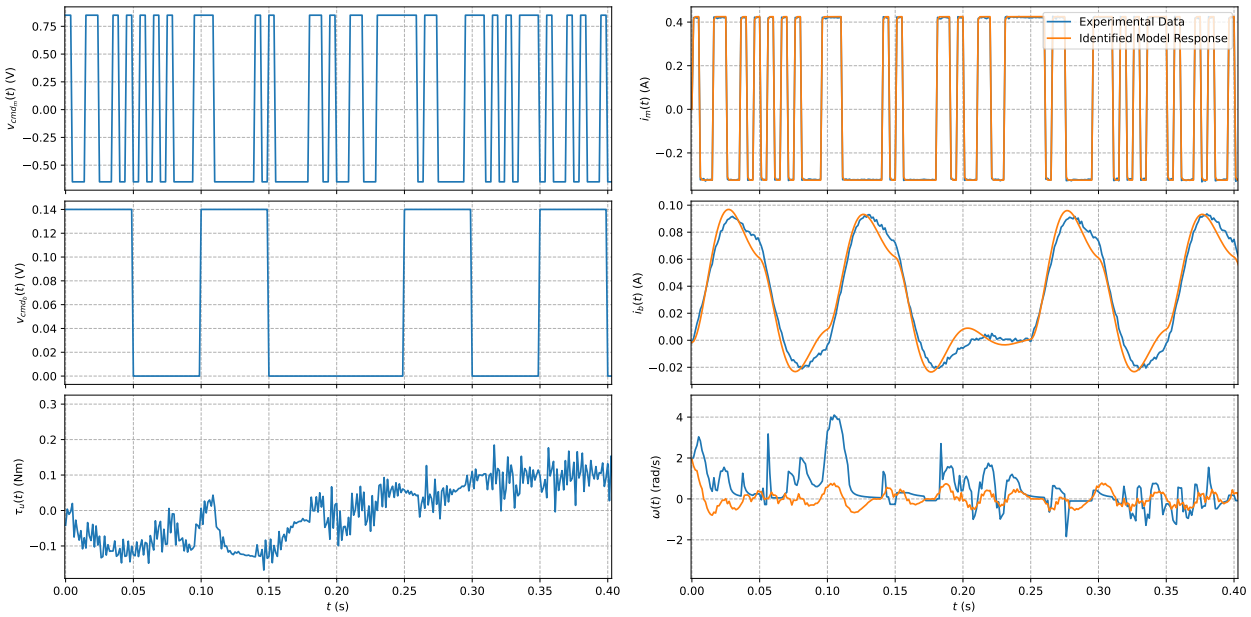

Figure 4.5: Comparison between a training dataset from the experimental hybrid actuator and the response of the identified hybrid actuator model to that training dataset.

Table 4.2 was computed using normalized output data. It shows that the linear outputs from the identified model are satisfyingly unbiased when compared to the training experimental data, as shown by the mean signed deviation, or MSD. Variance seems to be quite low too, as shown by the RMSE, or root mean squared error. Both error metrics are however one order of magnitude higher for the nonlinear output. The differences found in the experimental and identified velocity output suggest either

- a bad guess or approximation for the initial values of the parameters for both the hysteresis and mechanical model;
- or a bad choice of hysteresis model.

To better understand the mechanical properties of the system, an identification approach where the torque sensor is attached to the motor directly could be used, to identify the motor model first and then identify the hybrid actuator system as a whole.

Finding a solution for the hysteresis modeling problems would be somewhat more complicated, since it will always be hard to say if the initial parameter range or the choice of model is wrong.

One way to see if the choice of parameters is correct would be to

- put the torque sensor between the motor and the brake, as shown in Fig. 2.2;
- use the motor to excite a constant velocity as an input to the brake model;
- fix the brake voltage v_b, and at steady state, the brake current i_b, at a constant value;
- and observe and inspect the relationship between the velocity and the brake torque $\tau_b(t) = \alpha(i_b(t))z(t) + c(i_b(t))\omega(t)$ which will be measured by the torque sensor.

The relationship between $\omega(t)$ and $\tau_b(t)$ should be a hysteresis curve. To observe this curve, the torque should only be measured when $\omega(t)$ reaches a constant value, so that the inertial effects disappear. The brake model could then be simulated, and the simulated hysteresis curve could then be matched to the experimental curve by trial and error, similarly to what was done with the output currents in Section 4.2.1. The initial guess for the hysteresis parameters γ, β and A would then be tuned so that both the experimental curves and simulated curves match somewhat closely for a few fixed values of the brake current i_b.

This experimental approach, the initial results of which can be seen in Section 4.2.2, was however shown not to produce the desired hysteresis curve, but instead the clear results of unforeseen friction effects in the hybrid actuator, the source of which was narrowed down to the torque sensor. Past results found in the literature suggested that while friction is a consideration in such haptic devices, their effect should have been negligible in a 1-DOF device such as the one used in this thesis.

4.3.2 Comparison with a Linear, ARMAX Identified Model

Even though the results of the system identification process are corrupted by the unforeseen friction, there is still a possibility that they could be useful for controller design, if they outclass the accuracy of another baseline identification method. To do so, the quality of the model derived in this thesis and the associated GASSI method was compared to a linear model and linear model identification method. The ARMAX methods implemented in the `pysysid` Python package were tested on some experimental data, to see how well a linear system representation could reproduce the output response of a hybrid actuator. Fig. 4.6 and Fig. 4.7 show comparisons of the identification results of both the GASSI and ARMAX method, and demonstrate their respective error to the experimental datasets.

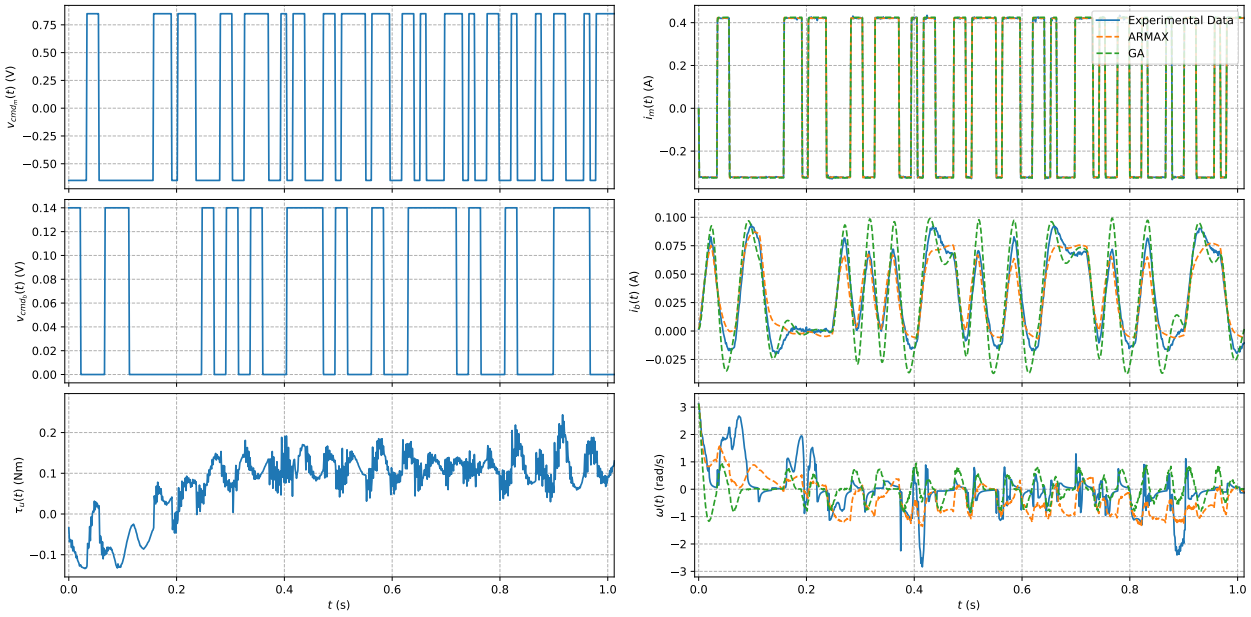

Figure 4.6: Comparison between a validation dataset from the experimental hybrid actuator and the response of both the GASSI and ARMAX identified hybrid actuator models to that training dataset.

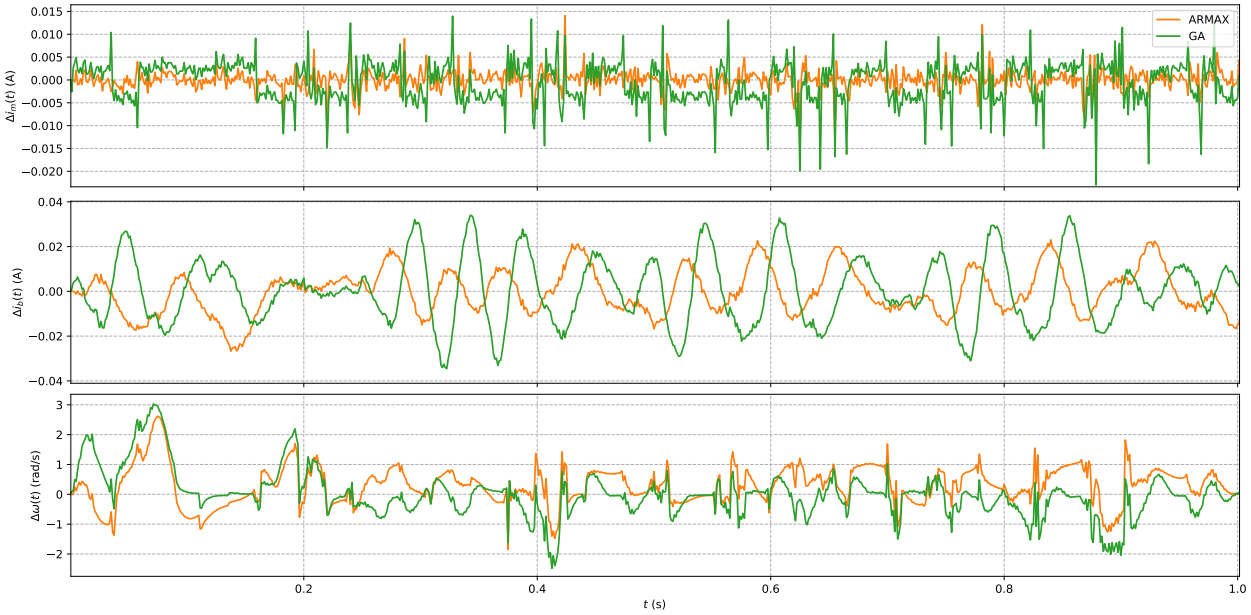

Figure 4.7: Comparison of the error between a validation dataset from the experimental hybrid actuator and the response of both the GASSI and ARMAX identified hybrid actuator models to that training dataset.

The results in Fig. 4.6, Fig. 4.7 and Table 4.3, which was computed with normalized output data, show that both the GASSI and the ARMAX identification methods could

Output Signal	RMSE, GASSI (unitless)	MSD, GASSI (unitless)	RMSE, ARMAX (unitless)	MSD, ARMAX (unitless)
$i_m(t)$	0.0112	-0.00123	0.00528	0.000247
$i_b(t)$	0.136	0.000805	0.1024	0.0114
$\omega(t)$	0.151	-0.0652	0.119	0.0147

Table 4.3: Error metrics between a validation dataset from the experimental hybrid actuator and the response of both the GASSI and ARMAX identified hybrid actuator models to that training dataset. The ARMAX model outperforms the GASSI identified nonlinear model in all metrics, with comparable metrics however when it comes to the electric current responses.

use some tuning, with both of them able to reproduce the linear outputs, like the motor and brake current, but not showing as much accuracy when it comes to angular velocity modeling. While understandable for the linear model, which doesn't take into account the nonlinear hysteresis torque from the brake, this result is not as encouraging when it comes the GA modeled hybrid actuator. This unexpected lesser performance of the GASSI identified, nonlinear model can be explained by the undesired friction in the experimental setup, which was not modeled and hindered the estimation of hysteresis parameters, which were to be given to the GASSI for its required range of initial parameters.

As it stands, the ARMAX identified model outperforms the GASSI identified model, and that, according to all metrics, as shown in Table 4.3. Still, when it comes to angular velocity, the RMSE and MSD of both models is too low for any of them to be used for controller design.

Chapter 5

Conclusion

5.1 Closing Remarks

The goal of this thesis was to produce a better haptic device. An up and coming avenue in the world of haptics, when it comes to stable haptic devices, are hybrid actuators. As such, this thesis tried to define a dynamic model of such an actuator, for the purpose of better controlling haptic devices and make them not only more stable, but also more transparent.

Building a model requires experimental data, and Chapter 2 presented how a hybrid actuator was built for the specific purpose of recording input and output data. Chapter 3 focused on deriving the hybrid actuator's model, from known mathematical representations of the motor and the brake.

As was seen in Chapter 4, parameter identification provided some promising results, especially when it came to the linear parts of the hybrid actuator process model. The response of both modelled electrical currents seemed to replicate with fidelity the experimental data, and did so with a similar degree of accuracy when compared to electrical currents modelled by ARMAX identified transfer functions. The nonlinear parts of the hybrid actuator model did not display the ability to replicate the experimental angular velocity response accurately. The cause of this discrepancy, and lack of performance from the nonlinear model is two-fold. First, undesired friction has introduced unmodeled torque in the nonlinear part of the process model. Second, this undesired friction has rendered accurate estimations of the hysteresis parameters infeasible, which meant that the GASSI could not compute the hybrid actuator model parameters as well as it could have.

5.2 Future Work

Such results seem to indicate that dynamic modelling of hybrid actuators doesn't have potential as a tool for controller design. Still, by going back to the roots of the problems

that were encountered in the making of this thesis, it would be possible to deliver on the derived dynamic model's promise of improving haptic device controller design.

A more precisely crafted experimental setup is at the heart of such an inquiry. Without reliable, noise free data, the quality of the identified model parameters can only go so far. Machining the mounts for the brake, the motor, and the torque sensor, instead of 3D printing them, would help in the alignment of all rotating parts in the actuator, which would reduce the undesired frictions which were discussed in Chapter 4. Changing the Sensor Development Inc. torque sensor for a more modern torque sensor could also reduce friction, as there is a possibility that the contact brushes of the torque sensor might have improved geometries, which reduce friction.

While reducing friction by adjusting the experimental setup is within the realm of possibilities, it is not possible however to entirely remove it. As such, it would be useful, to understand the dynamic behavior of the hybrid actuator completely, to model this friction. Models such as the ones found in [40], some of which are continuous and differentiable, could be investigated.

This is not meant to imply that the process model used in this thesis is perfect. Adjustments can be made, and in many parts of the model. The driver models, could be further enhanced by using higher-order linear models, or introducing known nonlinearities. Parameters for other hysteresis models could be identified, to see which model works best in the overall process model.

There are improvements to be made on the software used for system identification. While the GA presents some advantages over other system identification methods when it comes to controller design, the current implementation presented in this thesis is painfully slow. Some changes to the numerical integration implementation need to be made for this method to be usable in an industrial setting, and to compete with other system identification algorithms.

With the changes mentioned above put in place, it is entirely possible that the model derived in this thesis, whose parameters have been identified with the GASSI, could be used to derive a hybrid actuator controller. Such a nonlinear model shows the most promise for gain-scheduled controller design. A passivity analysis of the process model derived in Chapter 3 and the correctly designed gain-scheduled controller could assure that the closed-loop system is stable, as was shown in [41] and [42], for example. This kind of rigorous approach to controller design could lead to improvements not only in the stability of haptic devices, due to the inherent damping from the MR brake, but also to an improved impedance range and transparency, due to the fast dynamics an optimally designed controller.

Appendices

Appendix A

User Hand Dynamics

The sole purpose of modeling a hybrid actuator is to better understand how to control the torque it produces. Since the objective of this thesis is to design and identify a model of the hybrid actuator to be eventually used for control of this same actuator as a haptic force-feedback device, it is necessary to understand what torque output we want to control, or more precisely, what torque the device's user will feel. Newton's third law tells us that the torque the hybrid actuator will enact on the user's hand is exactly equal and opposite to the torque being produced by the user's hand. Thus, the torque that the user feels, and the torque we want to control, is really τ_u, as derived in (3.22). Now a problem arises: in the process model defined by (3.23), we consider τ_u to be the input to our system. If we also consider it to be an output of the system, controller design either becomes trivial, since the input is fed directly to the output, or it becomes impossible, as there is no way to directly actuate the torque produced by the user's hand. Thus, we need to reconsider how the user's hand affects our system.

Haptic feedback literature presents an interesting modelling convention from which we can get inspiration: the impedance approach to force-feedback [2]. In this approach, the user's hand position and velocity are collected by sensors. This data is then used to compute which force the haptic device should produce, by considering the dynamics of the virtual environment it is trying to reproduce, and the dynamics of the haptic device itself. Thus, a virtual causality is created, whereas a velocity passed through an impedance generates a force.

We will consider a similar pattern for the hybrid actuator which we are trying to control. Our theory here is that the user does not impart a torque directly upon the hybrid actuator, but instead does so from the wrist to the fingertips through the hand impedance. An angular velocity and an angular position, which we will call the user angular velocity ω_u and user angular position θ_u, are considered for visualization purposes to be the wrist angular velocity

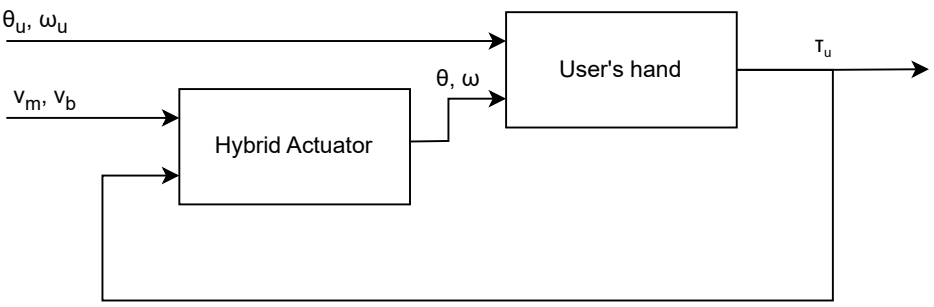

Figure A.1: Relationship between hybrid actuator and user hand dynamics

and wrist angular position. The difference between these physical quantities and the velocity and angular position of the fingertips, which are equal to ω and θ, the hybrid actuator shaft's angular velocity and position, should cause a torque produced by the hand's stiffness and damping, since the user's fingertips are on the output shaft of the hybrid actuator. Let us thus consider the hand to be a simple mass-damper-spring system of 2nd order. Further, elbow and shoulder dynamics could be represented similarly with a 4th or 6th order system. The dynamics of such a system can be represented using the form

$$\tau_u = B_u(\omega_u - \omega) + K_u(\theta_u - \theta), \tag{A.1}$$

as was shown in [43], for example. The diagram in Fig. A.1 helps visualize the new-found complexity of designing a controller for the hybrid actuator.

While it is possible to model the hybrid actuator position and velocity output using the known voltage inputs, as well as the user torque, which we can measure when using a torque sensor on the actuator shaft, it is much more difficult however to model the hand dynamics, as they depend on a lot of unknown physical quantities. The coefficients J_u, B_u and K_u of the user's hand vary with how the hand is gripping the actuator shaft, but for the sake of simulating the hybrid actuator's behavior these properties will be set to constant values.

Choosing a non-resonant damping coefficient of $\xi = 0.707$, an inertia J_u greater by an order of magnitude or two than J_m and J_b, and a response time of $T_r = 0.237$, which is determined from hand response time values found in [44], the two other coefficients could be computed using

$$\begin{aligned} \omega_n &= \frac{3}{T_r}, \\ B_u &= 2\xi\omega_n J_u, \\ K_u &= \omega_n^2 J_u. \end{aligned} \tag{A.2}$$

Bibliography

[1] M. Lacki and C. Rossa, "Towards the Ideal Haptic Device: Review of Actuation Techniques for Human-Machine Interfaces", in *Human-robot interaction: control, analysis, and design*, Cambridge Scholars, 2020, ch. 3, pp. 45–75.

[2] V. Hayward and K. E. Maclean, "Do it yourself haptics: Part I", *IEEE Robotics & Automation Magazine*, vol. 14, no. 4, pp. 88–104, Dec. 2007. [Online]. Available: `https://ieeexplore.ieee.org/document/4437756/` (visited on 04/29/2024).

[3] Dong-Soo Kwon and Ki Young Woo, "Control of the haptic interface with friction compensation and its performance evaluation", in *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000) (Cat. No.00CH37113)*, vol. 2, Takamatsu, Japan: IEEE, 2000, pp. 955–960. [Online]. Available: `http://ieeexplore.ieee.org/document/893142/` (visited on 07/02/2024).

[4] O. Baser and E. I. Konukseven, "Utilization of motor current based torque feedback to improve the transparency of haptic interfaces", en, *Mechanism and Machine Theory*, vol. 52, pp. 78–93, Jun. 2012. [Online]. Available: `https://linkinghub.elsevier.com/retrieve/pii/S0094114X12000225` (visited on 07/02/2024).

[5] N. Hogan, "Controlling impedance at the man/machine interface", in *Proceedings, 1989 International Conference on Robotics and Automation*, 1989, 1626–1631 vol.3.

[6] D. Weir and J. Colgate, "Stability of haptic displays", in *Haptic Rendering: Foundations, Algorithms, and Applications*, 2008, pp. 123–156. [Online]. Available: `https://www.scopus.com/inward/record.uri?eid=2-s2.0-84881427247&partnerID=40&md5=a7bbc53674a6092e7627322119fec831`.

[7] C. Rossa, J. Lozada, and A. Micaelli, "Design and control of a dual unidirectional brake hybrid actuation system for haptic devices", *IEEE Transactions on Haptics*, vol. 7, no. 4, pp. 442–453, 2014.

[8] O. Baser, H. Gurocak, and E. I. Konukseven, "Hybrid control algorithm to improve both stable impedance range and transparency in haptic devices", en, *Mechatronics*, vol. 23, no. 1, pp. 121–134, Feb. 2013. [Online]. Available: `https://linkinghub.elsevier.com/retrieve/pii/S0957415812001766` (visited on 07/04/2024).

[9] M. G. Karabulut, S. F. Kuecuekoglu, and M. I. C. Dede, "Training and modelling the non-linear behavior of an mr brake by using rnn and lstm", in *ACTUATOR; International Conference and Exhibition on New Actuator Systems and Applications 2021*, 2021, pp. 1–4.

[10] P. Dills, N. Colonnese, P. Agarwal, and M. Zinn, "A Hybrid Active-Passive Actuation and Control Approach for Kinesthetic Handheld Haptics", in *2020 IEEE Haptics Symposium (HAPTICS)*, Crystal City, VA, USA: IEEE, Mar. 2020, pp. 690–697. [Online]. Available: https://ieeexplore.ieee.org/document/9086310/ (visited on 07/24/2024).

[11] N. Kwok, Q. Ha, M. Nguyen, J. Li, and B. Samali, "Bouc–Wen model parameter identification for a MR fluid damper using computationally efficient GA", en, *ISA Transactions*, vol. 46, no. 2, pp. 167–179, Apr. 2007. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0019057807000237 (visited on 07/05/2024).

[12] C.-J. Lin, H.-T. Yau, C.-Y. Lee, and K.-H. Tung, "System Identification and Semiactive Control of a Squeeze-Mode Magnetorheological Damper", *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 6, pp. 1691–1701, Dec. 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6595023/ (visited on 07/08/2024).

[13] Y. Liu, S. Yang, and Y. Liao, "A Quantizing Method for Determination of Controlled Damping Parameters of Magnetorheological Damper Models", en, *Journal of Intelligent Material Systems and Structures*, vol. 22, no. 18, pp. 2127–2136, Dec. 2011. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1045389X11425278 (visited on 07/05/2024).

[14] X. Xiaomin, S. Qing, Z. Ling, and Z. Bin, "Parameter Estimation and its Sensitivity Analysis of the MR Damper Hysteresis Model Using a Modified Genetic Algorithm", en, *Journal of Intelligent Material Systems and Structures*, vol. 20, no. 17, pp. 2089–2100, Nov. 2009. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1045389X09343789 (visited on 07/05/2024).

[15] K. A. Alenborn, A. Klausen, S. S. Tordal, and H. R. Karimi, "Firefly optimization used to identify hysteresis parameter on rotational MR-damper", in *2014 International Conference on Mechatronics and Control (ICMC)*, Jinzhou: IEEE, Jul. 2014, pp. 2302–2307. [Online]. Available: https://ieeexplore.ieee.org/document/7231979/ (visited on 07/08/2024).

[16] L. Ljung, *System Identification: Theory for the User* (Prentice Hall information and system sciences series). Prentice Hall PTR, 1999. [Online]. Available: https://books.google.ca/books?id=nHFoQgAACAAJ.

[17] W. Li and H. Du, "Design and Experimental Evaluation of a Magnetorheological Brake", *The International Journal of Advanced Manufacturing Technology*, vol. 21,

no. 7, pp. 508–515, May 2003. [Online]. Available: http://link.springer.com/10.1007/s001700300060 (visited on 07/27/2024).

[18] C. Rossa, A. Jaegy, J. Lozada, and A. Micaelli, "Design Considerations for Magnetorheological Brakes", *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 5, pp. 1669–1680, Oct. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6680747/ (visited on 07/27/2024).

[19] S. Stuart, *DC Motors, Speed Controls, Servo Systems: An Engineering Handbook*. Elsevier Science, 2013.

[20] B. F. Spencer, S. J. Dyke, M. K. Sain, and J. D. Carlson, "Phenomenological Model for Magnetorheological Dampers", en, *Journal of Engineering Mechanics*, vol. 123, no. 3, pp. 230–238, Mar. 1997. [Online]. Available: https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9399%281997%29123%3A3%28230%29 (visited on 07/18/2024).

[21] İ. Şahin, T. Engin, and Ş. Çeşmeci, "Comparison of some existing parametric models for magnetorheological fluid dampers", *Smart Materials and Structures*, vol. 19, no. 3, p. 035 012, Mar. 2010. [Online]. Available: https://iopscience.iop.org/article/10.1088/0964-1726/19/3/035012 (visited on 07/09/2024).

[22] Q. Zhou, S. Nielsen, and W. Qu, "Semi-active control of three-dimensional vibrations of an inclined sag cable with magnetorheological dampers", *Journal of Sound and Vibration*, vol. 296, no. 1, pp. 1–22, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022460X0600201X.

[23] R. Jiménez and L. Álvarez-Icaza, "LuGre friction model for a magnetorheological damper", en, *Structural Control and Health Monitoring*, vol. 12, no. 1, pp. 91–116, Jan. 2005. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/stc.58 (visited on 07/28/2024).

[24] B. F. Spencer, S. J. Dyke, M. K. Sain, and J. D. Carlson, "Phenomenological model for magnetorheological dampers", *Journal of Engineering Mechanics*, vol. 123, no. 3, pp. 230–238, 1997.

[25] F. Ikhouane and J. Rodellar, "Systems with hysteresis: Analysis, identification and control using the bouc-wen model", *Systems with Hysteresis: Analysis, Identification and Control using the Bouc-Wen Model*, Sep. 2007.

[26] F. Ikhouane, V. Mañosa, and G. Pujol, "Minor loops of the dahl and lugre models", *Applied Mathematical Modelling*, vol. 77, pp. 1679–1690, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0307904X19305190.

[27] D. Li and Y. Wang, "Parameter identification of a differentiable bouc-wen model using constrained extended kalman filter", *Structural Health Monitoring*, vol. 20, no. 1,

pp. 360–378, 2021. eprint: https://doi.org/10.1177/1475921720929434. [Online]. Available: https://doi.org/10.1177/1475921720929434.

[28] G. Quaranta, W. Lacarbonara, and S. F. Masri, "A review on computational intelligence for identification of nonlinear dynamical systems", en, *Nonlinear Dynamics*, vol. 99, no. 2, pp. 1709–1761, Jan. 2020. [Online]. Available: http://link.springer.com/10.1007/s11071-019-05430-7 (visited on 07/23/2024).

[29] J. Bosworth, N. Y. Foo, and B. P. Zeigler, "Comparison of genetic algorithms with conjugate gradient methods", NASA, Tech. Rep., 1972.

[30] F. Herrera, M. Lozano, and J. Verdegay, "Tackling Real-Coded Genetic Algorithms Operators and Tools for Behavioural Analysis", *Artificial Intelligence Review*, vol. 12, no. 4, pp. 265–319, 1998. [Online]. Available: http://link.springer.com/10.1023/A:1006504901164 (visited on 07/24/2024).

[31] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, en. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996. [Online]. Available: http://link.springer.com/10.1007/978-3-662-03315-9 (visited on 07/30/2024).

[32] B. A. Negash, W. You, J. Lee, and K. Lee, "Parameter identification of Bouc-Wen model for Magnetorheological (MR) fluid Damper by a Novel Genetic Algorithm", en, *Advances in Mechanical Engineering*, vol. 12, no. 8, p. 168 781 402 095 054, Aug. 2020. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1687814020950546 (visited on 08/03/2024).

[33] S. Salcedo-Sanz, "A survey of repair methods used as constraint handling techniques in evolutionary algorithms", *Computer Science Review*, vol. 3, no. 3, pp. 175–192, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574013709000379.

[34] I. Rahimi, A. H. Gandomi, F. Chen, and E. Mezura-Montes, "A Review on Constraint Handling Techniques for Population-based Algorithms: From single-objective to multi-objective optimization", en, *Archives of Computational Methods in Engineering*, vol. 30, no. 3, pp. 2181–2209, Apr. 2023. [Online]. Available: https://link.springer.com/10.1007/s11831-022-09859-9 (visited on 08/02/2024).

[35] O. Kramer, "A Review of Constraint-Handling Techniques for Evolution Strategies", en, *Applied Computational Intelligence and Soft Computing*, vol. 2010, pp. 1–11, 2010. [Online]. Available: http://www.hindawi.com/journals/acisc/2010/185063/ (visited on 08/02/2024).

[36] Z. Michalewicz and C. Z. Janikow, "Handling constraints in genetic algorithms.", in *Icga*, vol. 398, 1991, pp. 151–157.

[37] P. Chootinan and A. Chen, "Constraint handling in genetic algorithms using a gradient-based repair method", *Computers & Operations Research*, vol. 33, no. 8, pp. 2263–2281, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030505480500050X.

[38] R. Farmani and J. Wright, "Self-adaptive fitness formulation for constrained optimization", en, *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 5, pp. 445–455, Oct. 2003. [Online]. Available: http://ieeexplore.ieee.org/document/1237163/ (visited on 07/30/2024).

[39] A. Dupuis, M. Ghribi, and A. Kaddouri, "Multiobjective genetic estimation of dc motor parameters and load torque", in *2004 IEEE International Conference on Industrial Technology, 2004. IEEE ICIT '04.*, vol. 3, 2004, 1511–1514 Vol. 3.

[40] J. Na, Q. Chen, and X. Ren, "Friction Dynamics and Modeling", en, in *Adaptive Identification and Control of Uncertain Systems with Non-smooth Dynamics*, Elsevier, 2018, pp. 11–18. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780128136836000039 (visited on 07/05/2024).

[41] J. Forbes and C. Damaren, "Design of gain-scheduled spr controllers using numerical optimization", Aug. 2008.

[42] A. Walsh and J. R. Forbes, "A very strictly passive gain-scheduled controller: Theory and experiments", *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 6, pp. 2817–2826, 2016.

[43] L. L. Kovács and J. Kövecses, "Dynamics of coupled haptic systems", in *2015 IEEE World Haptics Conference (WHC)*, 2015, pp. 286–292.

[44] A. K. Chouamo, S. Griego, and F. S. M. Lopez, "Reaction time and hand dominance", *The Journal of Science and Medicine*, 2021.