University of
Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study program/specialisation: Robot technology and Signal Processing	Spring semester 2023 Open
Author: Jens Immerstein Trydal	
Program coordinator: Morten Tengesdal Supervisor(s): Morten Mossige	
Title of master's thesis: New control structure for high voltage fields	
Credits(ECTS): 30	
Keywords: control system design, data-driven design reinforcement learning, parameter estimation	Number of pages: 90 + Attachments/other: Stavanger, 15 June 2023

Abstract

In electrostatic painting, a high voltage is applied to the paint, attracting the paint particles toward a grounded work object, drastically increasing the efficiency of the painting process. However, combining high voltage and highly flammable paint is a potential hazard that is reduced by an automatic fire extinguishing system and strict controller limits defined by safety standards. The thesis investigates alternative controller strategies attempting to improve the performance of ABB's high-voltage control system.

The dynamics of electrostatic gas discharge, or corona discharge, is studied to estimate the relation between the applied voltage and the corona current passing through the electrostatic field. However, given the data available for the real-time system, the estimation problem is concluded to be structurally unidentifiable, resulting in the estimators not converging to the actual state of the system. Despite that, a simple estimator is utilized in a current limiting controller. This controller is activated when the system leaves its normal working area. Simulation results indicate that this controller can decrease the amount of unnecessary safety-related stops and reduce the reaction time for actual safety-hazard incidents.

Furthermore, a data-driven approach is selected to model and create a controller for the system generating the high-voltage output. The model of the dynamics of the high-voltage system is created using neural networks and open-loop high-resolution data collected with a self-developed data acquisition program. Then, the estimated model is used in a reinforcement learning environment to create a theoretically optimal controller valid for the entire nonlinear workspace. Due to limited computational resources, and errors in the data, the thesis presents a lower-resolution proof of concept for both the neural network model and controller.

Additionally, the thesis presents a basis of knowledge on ABB's electrostatic painting system, featuring recommendations and suggestions for future work.

Contents

Abbreviations

HVC	High voltage controller
DSP	Digital signal processor
FPGA	Field-programmable gate array
CPU	Central processing unit
AC	Alternating current
DC	Direct current
SoC	System on chip
MPSoC	Multiprocessor system on chip
PI	Proportional integral
PID	Proportional integral derivative
IPS	Integrated processing system
FTP	File transfer protocol
LQR	Linear quadratic regulator
MPC	Model predictive control
RLS	Recursive least squares
ADC	Analogue to digital converter
CSV	Comma separated values
DPM	Dual port memory
LSTM	Long short-term memory
RMSE	Root-mean-squared error
NRMSE	Normalized root-mean-squared error

MPC	Model predictive controller
KCL	Kirchhoff's current law
KVL	Kirchhoff's voltage law
DDPG	Deep deterministic policy gradient
PWM	Pulse-width modulation
AI	Artificial intelligence
LPF	Low pass filter
API	Application programming interface

Chapter 1

Introduction

In automotive painting applications, the paint is typically charged with 50-90 kV relative to the object to be painted. Charging the paint increases the paint's transfer efficiency and reduces the emission from the painting process. An increase in the transfer efficiency, i.e., a reduction in the paint used, impacts both the environment and the economy.

Using a high-voltage application in the potentially explosive atmosphere of a paint booth is a safety hazard. In order to reduce the risk of a spark and corresponding hazard, a controller closely supervises the system, shutting it down if needed.

The performance of ABB's high-voltage controller has not been sufficient for all applications. Thus, this thesis investigates the problem, proposing new solutions and strategies for improving the controller's performance.

The main part of the thesis is split into the following chapters:

- **Chapter 2 - Background:**
  - Analysis of the electrostatic painting process, ABB's hardware and controller, and a description of the new control structure.
- **Chapter 3 - Current controller:**
  - Development and analysis of a current limiting controller.
- **Chapter 4 - Voltage controller:**
  - Development and analysis of a voltage controller using a data-driven approach.
- **Chapter 5 - Reflections and experiences:**
  - Collection of reflections, alternative solutions, and suggestions for further work.
- **Chapter 6 - Summary:**
  - Summary of the central results presented in the thesis.

Chapter 2

Background

This chapter provides some background information and analysis of the following topics fundamental to this thesis:

- Electrostatic painting
- ABB's hardware for electrostatic painting
- ABB's controller solution
- The new control structure

2.1 Electrostatic painting

In traditional spray painting, most paint does not reach its target but is lost to the surrounding environment. Thus electrostatic painting has been developed to increase the amount of paint transferred to the target. The motivation for this is partly economic to save paint, but as the industry is subject to strict environmental regulations, electrostatic paint contributes to satisfying those. Moreover, electrostatic painting is used to improve the paint finish. [3][16]

Figure 2.1 illustrate how negatively charged paint particles are attracted to a workpiece that is grounded.

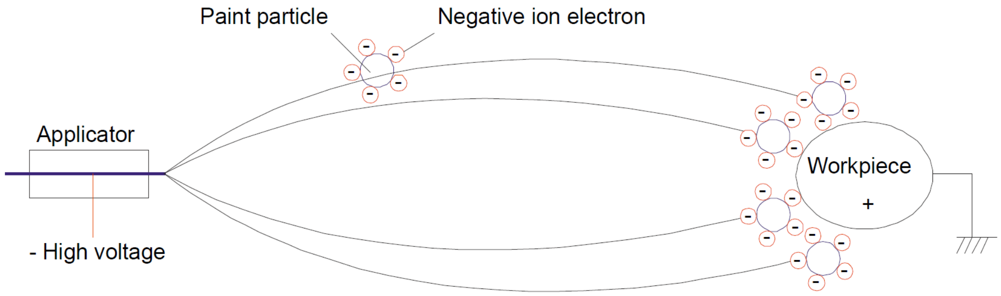


Figure 2.1: This figure displays a sketch demonstrating how paint particles are charged with high voltage so that they are attracted to the target/workpiece. The sketch is sourced from ABB's documentation[3].

High voltage fields, in combination with highly flammable paint, are a potential fire starter. Thus it is essential to reduce the risk of sparks caused by the electrostatic painting equipment. However, due to the characteristics of the high voltage field and painting environment, as of today, it is unfeasible to guarantee that no sparks may appear. The safety norms address this by setting strict requirements for how the electrostatic painting equipment should perform and additional requirements for how a fire extinguishing system handles a potential fire. [6] [7]

2.1.1 Applicators

In ABB's inventory, there are several different paint applicators. They are all based on two fundamental approaches for charging the paint; direct and indirect. For the direct method illustrated in Figure 2.2, the high voltage is applied directly to the bell cup.

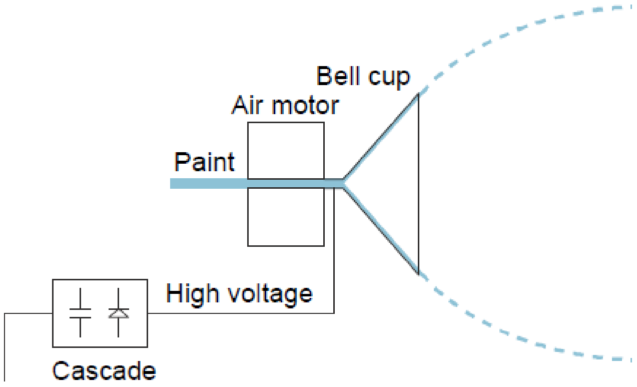


Figure 2.2: This figure displays a sketch of ABB's electrostatic paint system using the direct charge method. The sketch is sourced from ABB's documentation[3].

The bell cup rotates very fast and atomizes the paint into small particles. As the high voltage is applied to the bell cup, the paint is charged before leaving the applicator. This method is suitable when the paint has sufficiently high resistance as a fluid. If the paint is conductive, everything in connection with the paint will be charged to a high voltage. In that case, the paint supply system is isolated from the ground potential. If this is an unpractical solution, the paint is charged using another approach.

For the indirect method, the paint is charged after it is atomized. This is typically done using a ring of electrodes that charge the passing paint particles. This method is generally used when the paint is conductive. Figure 2.3 illustrate an indirect charge setup.

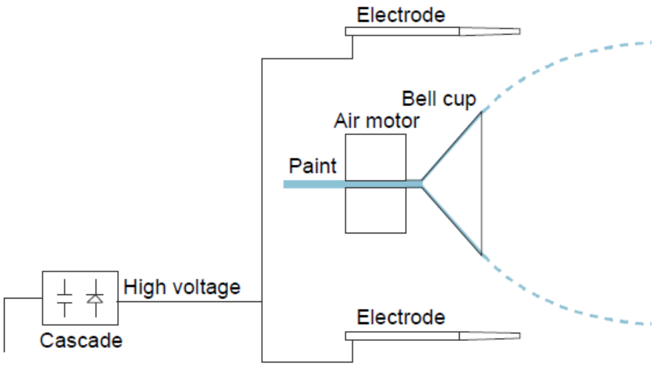


Figure 2.3: This figure displays a sketch of ABB's electrostatic paint system using an indirect charge electrode. The sketch is sourced from ABB's documentation[3].

2.1.2 Paint program

This section introduces some steps and parameters for setting up a paint program. Firstly, plan a robot path. Then define what *”paint brushes”* to use for the robot path by setting parameters defining the configurable forces influencing the paint as it leaves the applicator. Those parameters describe the;

- Amount of fluid/paint.
- Bell atomizer rotating speed.
- Shaping air.
- High voltage setpoint, i.e., attraction force on the paint from the applicator to the work object.

In his master’s thesis[16], Morten Mossige gives a more in-depth introduction to the mentioned parameters for the *”paint brushes”*.

Finally, perform simulation, optimization, testing, and adjustment of safety parameters.

For details on how to set up a painting program, see ABB’s documentation[1][3].

2.1.3 Safety

Safety considerations for electrostatic painting are critical due to the potential hazards. EN 50176 [7] outlines the safety requirements for automatic electrostatic application systems for ignitable liquid coating materials. However, ensuring a guaranteed safe state is accomplished through compliance with EN 16985 [6], which specifies the safety requirements for spray booths used with organic coating materials. In short, the measures for preventing hazardous discharges are insufficient to guarantee no such event. Consequently, a safe state is maintained by an automatic fire extinguishing system.

ABB has integrated several safety functions to prevent hazardous discharges, including *"over current detection"* and *"current slope detection"* [3]. In essence, limits are established for the maximum value and slope of the field current (section 2.1.4). These limits are fitted to each application according to ABB’s instructions. Typical values are permitted, and any deviation triggers a shift to a safe state.

EN 50176 [7] define experiments and requirements for the safety performance of the high voltage control system. The results from these experiments contribute to defining the global maximum limits. Then per [7], local limits are implemented closer to the expected reference values of each application.

2.1.4 Field current

The *"field current"* is defined as the current of electrons that leave the paint applicator, as illustrated in Figure 2.4. This current is indirectly measured with ABB's hardware as described in section 2.2.2. The field current is a product of several factors, most of which are covered in this section.

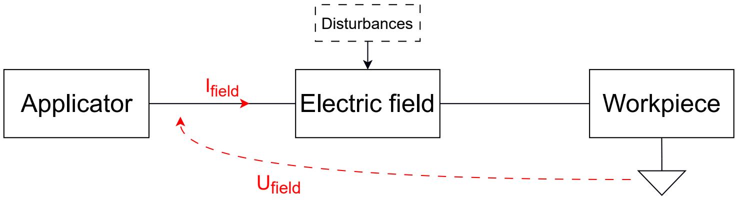


Figure 2.4: Simple illustration of the current that flows through the applicator to the grounded workpiece.

Without digging deeply into the physics of electrostatic gas discharge and corona discharge, a quick overview of the factors influencing the field current is now given.

The field current varies with the following cases and parameters:

- Direct/indirect charging method
  - For the indirect charging method, the paint is charged solely by corona discharge, Whereas for the direct charging method, the paint is directly charged. Thus, the corona discharge current does not encompass the entire measured *"field current"* when using the direct charging method.

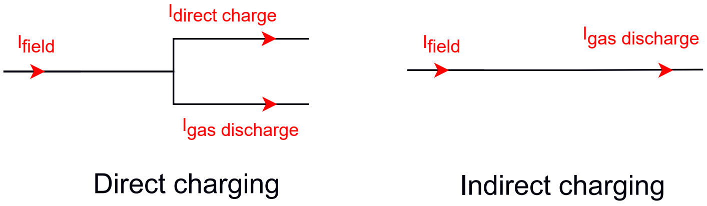


Figure 2.5: Current flow for the charging methods.

- Applicator type
- Contamination on the applicator (This is typically connected to residual paint on the applicator device.)
- Unipolar corona space charge flow[18][16].
  - Geometry of workpiece and applicator influencing the magnetic field lines

- The distance between the workpiece and the applicator
- Applied field voltage, U_{field}
- Air and paint composition influencing the ion mobility of the gas
  - Paint type
  - Paint color
  - Amount of paint
  - Temperature
  - Humidity
  - Air pressure and composition
- Shaping air and draft/airflow in the paint booth

Given real-time knowledge of the variables above, a model could be used to estimate the resulting field current. However, as of today, most of these values are unavailable for ABB's real-time system. Moreover, creating a model based on all these variables is an extensive task. Furthermore, calculating the magnetic field lines, given the varying geometry of the workpiece, is a task that would require a lot of new sensors, computational hardware, and research.

In Morten Mossige's master's thesis[16], some experiments documenting the impact of a few variables influencing the field current were performed. This, however, was done for particular setups not valid for most applications and not replicable due to not documenting the status of all significant variables.

The only related data available in ABB's real-time system are the measured field current and voltage. Deutsch deduced a method for estimating the corona current described by Sigmond [18]. Equation 2.1 shows a simplified version of Deutsch's formula where $U_{(t)}$ is the field voltage, and $U_{0(t)}$ is the voltage at which a current starts to flow, i.e., the gas/paint is charged. And $K_{(t)}$ is a scaling factor. $U_{0(t)}$ and $K_{(t)}$ encompass, i.e., depend on all the previously mentioned variables.

$$I_{Dch(t)} = K_{(t)} \cdot U_{(t)}(U_{(t)} - U_{0(t)}) \tag{2.1}$$

Additionally, Sigmond derived the maximum corona current here simplified into equation 2.2.

$$I_{Sig(t)} = K_{(t)} \cdot U_{(t)}^2 \tag{2.2}$$

Equation 2.3 illustrates how the Simonds maximum corona current equals Deutsch's corona current when $U_{0(t)} = 0$.

$$\lim U_{0(t)} \to 0 \ \Rightarrow I_{Sig(t)} = I_{Dch(t)} \tag{2.3}$$

To summarize, $I_{Sig(t)}$ is only accurate for larger currents where $U_{0(t)} << U_{(t)}$, whereas $I_{Dch(t)}$ is accurate for all cases[18]. Moreover, an increasing current is likely connected to an increasing value for $K_{(t)}$ and decreasing value for $U_{0(t)}$. Furthermore, $K_{(t)}$ and $U_{0(t)}$ are likely connected by being dependent on the same underlying variables.

2.2 ABB's hardware for electrostatic painting

This section is divided into the following subsections:

- System overview
- HVC[1]
- Cascade
- Analytical models

In short, the primary functionality and components of ABB's HVC and Cascade are described. These two devices are the main components of ABB's electrostatic painting equipment. If details are provided, the specific versions described are the HVC-02 rev.5 and the CU930B cascade. When detailed values are provided, they are broad approximations and simplifications, as detailed hardware documentation is confidential ABB property.

2.2.1 System overview

The primary function of the HVC is to deliver a controlled low voltage AC[2] to the Cascade. The cascade then converts the low-voltage AC to a high-voltage DC[3], using a transformer and a Cockcroft-Walton voltage multiplier. A sketch of this relation, showing the most important parts of ABB's electrostatic paint system, is displayed in Figure 2.6.

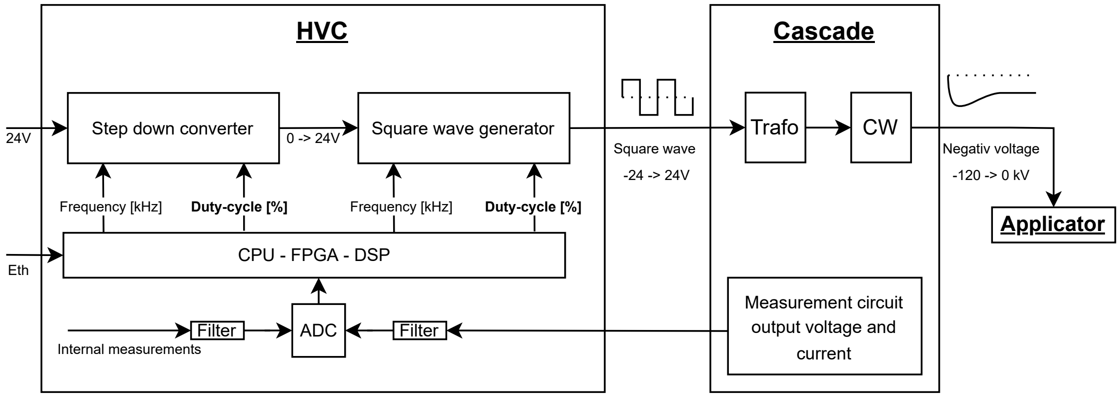


Figure 2.6: The figure display a simplified schematic describing the main parts of ABB's electrostatic painting equipment.

The output voltage and current from the cascade are measured and the analog signal is routed back to the HVC. The controller is typically implemented on the DSP[4]. All signals are routed via the FPGA[5] and are shared with the DSP and CPU[6]. The CPU typically handles communication with the larger system.

[1] High voltage controller
[2] Alternating current
[3] Direct current
[4] Digital signal processor
[5] Field-programmable gate array
[6] Central processing unit

2.2.2 Cascade

ABB's inventory consists of a series of cascades that are based on the same structure. As an input, it receives a square wave AC. The voltage is then increased using a transformer. As a final step, the AC is applied to a Cockcroft-Walton voltage multiplier, drastically increasing the output voltage and making it a DC. Additionally, there is a circuit for measuring the output current and voltage. An illustration of this structure is displayed in Figure 2.7.

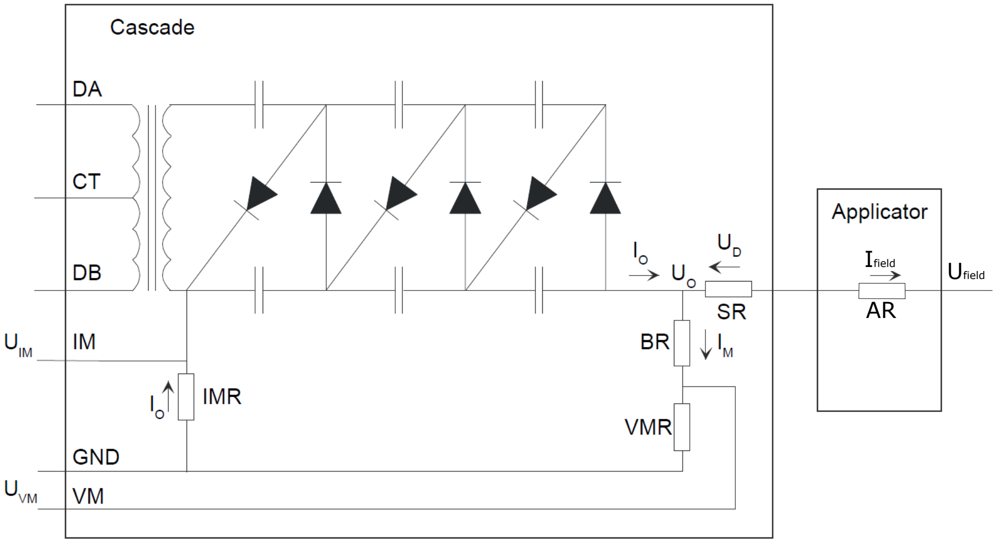


Figure 2.7: The figure display a simplified schematic describing the components of the cascade. The sketch is based on ABB's documentation[3]

$$\begin{array}{rcl@{\qquad}rcl}
BR & - & Bleeder\ resistor & CT & - & Center\ tap\ transformer \\
DA & - & HVC\ output\ 0-24V & DB & - & HVC\ output 0-24V \\
IMR & - & Measurement\ resistor & VMR & - & Measurement\ resistor \\
SR & - & Current\ limiting\ resistor & AR & - & Internal\ resistance\ applicator \\
U_D & - & Voltage\ drop\ SR\ and\ AR & U_0 & - & Cascade\ voltage \\
I_0 & - & Total\ current/Cascade\ current & U_{IM} & - & Voltage\ measured\ over\ IMR \\
I_{field} & - & Current\ through\ applicator & U_{field} & - & Voltage\ at\ applicator\ end \\
I_M & - & Measurement\ circuit\ current & U_{VM} & - & Voltage\ measured\ over\ VMR
\end{array}$$

Measurements

The values of interest, which are I_{field} and U_{field} are not measured directly but can be derived using Equation 2.4a and 2.4b.

$$
\begin{aligned}
&\text{Field current:} && I_{field} = I_0 - I_M \\
& && \Downarrow \\
& && I_{field} = U_{IM}/IMR - U_{VM}/VMR && \text{(2.4a)} \\
&\text{Field voltage:} && U_{field} = U_{VM} + U_{BR} - U_D \\
& && \Downarrow \\
& && U_{field} = U_{VM} + BR \cdot I_M - (SR + AR) \cdot I_{field} && \text{(2.4b)}
\end{aligned}
$$

Filtering

The measured signals, U_{IM} U_{VM}, are routed to the HVC, filtered, and then read by an ADC. The ADC sample frequency is in the range 50-200kHz. The original real-time controller in the DSP (section 2.2.3) samples the signals at 10 kHz before applying a discrete low pass filter with a cut-off frequency of approximately 750Hz. The analog filter consists of higher-order-active filters and simpler 1. order low-pass filters. The resulting cut-off frequency for the analog filters is approximately 1500Hz U_{IM}, and 750Hz for U_{VM}.

Comment on accuracy

The components in the measurement circuit, the analog filters, and the ADC introduce considerable measurement errors. That is both a static and time-varying error. Specifically, the high voltage BR resistor with an ±5 % initial variation introduces a considerable error to the measurements. Calibrating the giga-ohm-sized BR resistor requires specialized instruments that measure giga-ohm values. ABB does currently not employ any online or offline calibration or estimation algorithms. Introducing a calibration routine would increase the similarity and comparability between the different instances.

2.2.3 HVC

This section describes the HVC in more detail. Figure 2.8 points out the covered areas. The DSP, CCPU, and HVC are important components for software implementation, while the AC generation circuit describes some analog hardware in greater detail. Figure 2.9 display an HVC-02 unit.

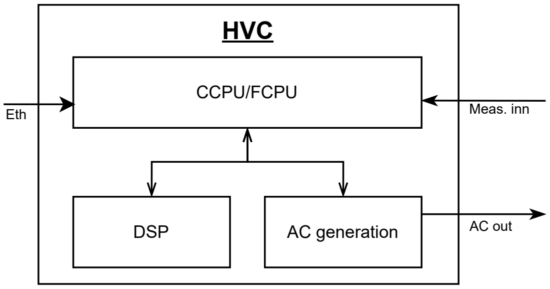


Figure 2.8: Sketch describing parts of HVC-02 that are introduced.

Figure 2.9: Image of HVC-02 sourced from ABB's documentation[3]

AC generation

Figure 2.10 depicts a simplified circuit version generating the AC output. Further, in Figure 2.11, the different stages of the AC generation are illustrated. The red numbering points out the connection between the different stages in the figures.

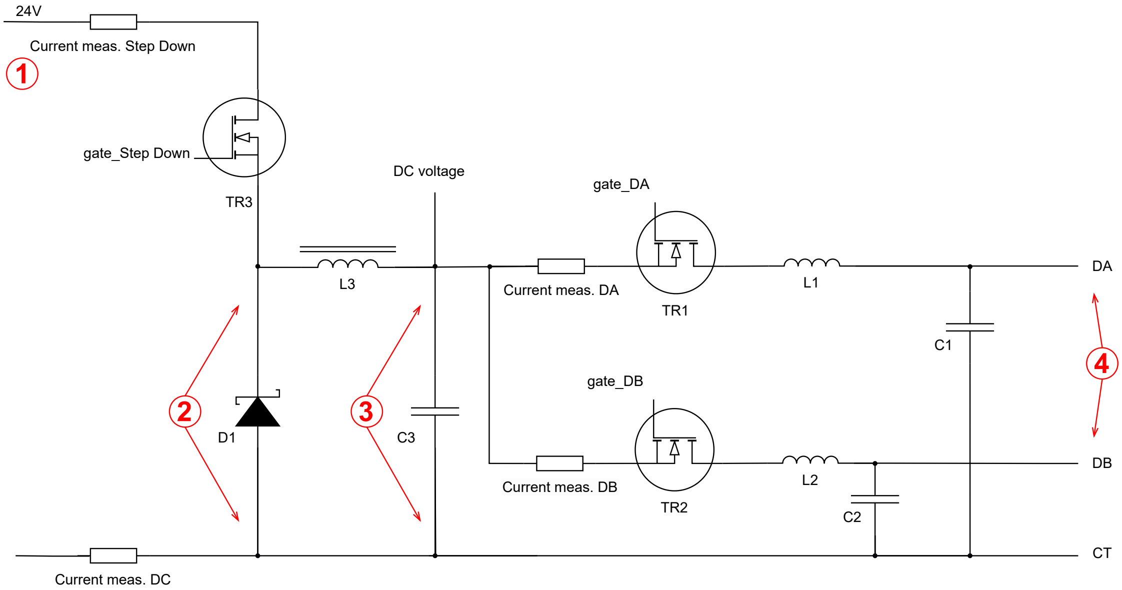


Figure 2.10: Simplified schematic describing some of the internal measured values and the generation of the AC output in the HVC.

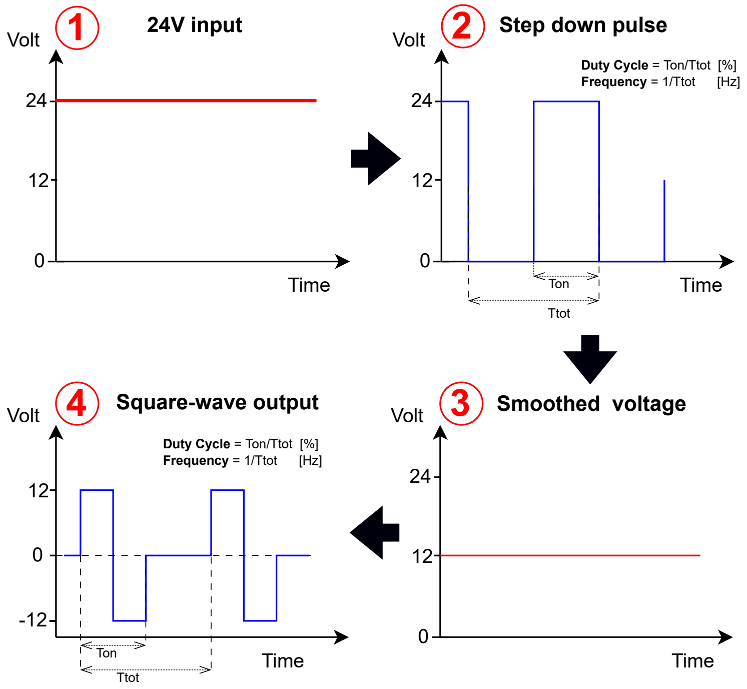


Figure 2.11: Step-by-step sketch of the generation of the HVC's AC output

The three transistors, TR1-3, described in Figure 2.10, generate the HVC's AC output. TR3 is the controlled component in a step-down converter. By adjusting the duty cycle on TR3, the 24V DC input is modified to a DC voltage in the range of 0-24V DC. The frequency of the step-down converter is not adjusted as it matches capacitive and inductive, stationary components. Further on, TR1 and TR2 generate the output AC. The frequency and duty cycle of the output AC is controllable values. However, for the original controller (section 2.3), the duty cycle of TR3 is the only controlled value. For the rest of this thesis, the duty cycles of TR3 and TR1/TR2 will be referred to as duty cycle DC and duty cycle AC.

DSP

The DSP is a simple processor ABB uses on the HVC for the real-time controller. However, it is beginning to age and lacks convenient features like a floating point processor [10], increasing the effort of implementing new features.

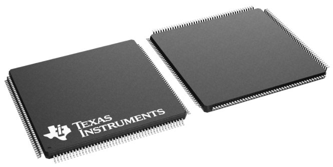


Figure 2.12: Illustrative photo of Texas Instruments TMS32028512-Q1[10]

CCPU

The CCPU is a single-board computer designed explicitly after ABB's requirements. It is a versatile board used on several ABB paint devices.

In addition to memory, a network switch, and several other components, it incorporates an Intel/Altera EP2C5F256 FPGA and a Motorola MPC5200 processor. These components were new about two decades ago. The CCPU operation system is a specialized version of Linux. The HVC board utilizes a CCPU board for higher-level tasks, such as handling interaction with the more extensive ABB system.

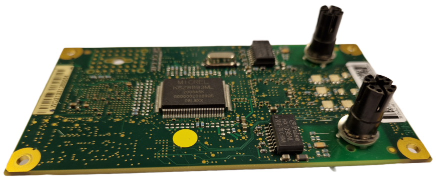


Figure 2.13: Image of a CCPU board.

FCPU

ABB is developing a new board, the FCPU, to replace the older CCPU. This board integrates a Xilinx Zynq™ Ultrascale+™ SoC[7][29]. That is one chip combining CPU and FPGA. The new FCPU board signals a step up in capabilities and possibilities for the programs implemented in ABB's paint application. The FCPU development is in an early prototype stage and ready for experimental testing with the HVC in May 2023.

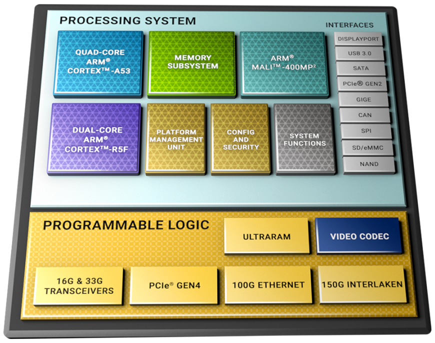


Figure 2.14: Illustrative block diagram describing an Zynq™ UltraScale+™ MPSoC[8][29]

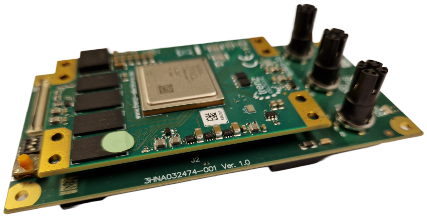


Figure 2.15: Image of the prototype FCPU board.

IPS

IPS[9], or Integrated Processing System, is the program running the paint application part on ABB's paint robots. IPS typically runs on all FCPU and CCPU boards used in the robot controller system. A detailed description is available in ABB's documentation[2].

[7] System on chip

[9] Integrated processing system

2.2.4 Analytical models

This subsection presents:

- Simplified cascade and output model
- Buck converter model
- Higher fidelity models

Simplified cascade and output model

This section presents a simplified model of the system described in section 2.2.1. The model illustrates key dynamics and properties of the system.

In his master's thesis[16], Morten Mossige separately estimated the dynamics of the charge and discharge of the cascade to 1. order models. However, the relationship between charge and discharge was not inspected.

Figure 2.16 present the circuit diagram for a heavily simplified cascade and output model.

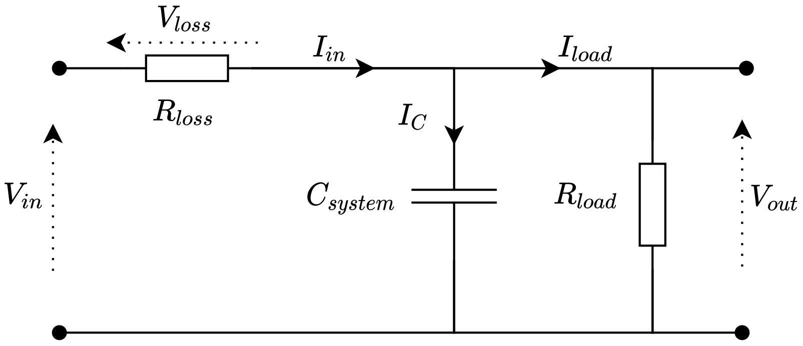


Figure 2.16: Circuit diagram for the simplified cascade and output model.

V_{out} - Cascade output voltage
R_{load} - Measurement circuit, field, applicator, and cable resistance.
C_{system} - Capacitance of cascade, applicator, and cables.
R_{loss} - Resistance representing the internal loss and dynamics of the cascade
V_{in} - Transformed, rectified, and multiplied lossless voltage, i.e., ideal cascade output voltage.

Equation 2.5 presents the relations of the circuit in Figure 2.16 according to Kirchhoff's current and voltage laws.

$$\text{KCL: } I_{in} = I_{load} + I_C, \qquad \text{KVL: } V_{loss} = V_{in} - V_{out} \tag{2.5}$$

Furthermore, Equation 2.6 expands the current relations in the circuit.

$$I_C = C_{system} \cdot \frac{dV_{out}}{dt}, \quad I_{load} = \frac{V_{out}}{R_{load}}, \quad I_{in} = \frac{V_{loss}}{R_{loss}} \tag{2.6}$$

Equation 2.7 inserts the relations from Equation 2.6 into the KCL equation.

$$\frac{V_{loss}}{R_{loss}} = \frac{V_{out}}{R_{load}} + C_{system} \cdot \frac{dV_{out}}{dt} \tag{2.7}$$

Equation 2.8 rearranges Equation 2.7 with respect to $\frac{dV_{out}}{dt}$.

$$\frac{dV_{out}}{dt} = \frac{V_{loss}}{R_{loss}C_{system}} - \frac{V_{out}}{R_{load}C_{system}} \tag{2.8}$$

Further, in Equation 2.9 the KVL relation from Equation 2.5 is inserted for V_{loss}.

$$\frac{dV_{out}}{dt} = \frac{V_{in} - V_{out}}{R_{loss}C_{system}} - \frac{V_{out}}{R_{load}C_{system}} \tag{2.9}$$

Equation 2.10 takes the Laplace transform of Equation 2.9, assuming all R and C values are constants.

$$s \cdot V_{out}(s) = \frac{V_{in}(s)}{R_{loss}C_{system}} - \frac{V_{out}(s)}{R_{loss}C_{system}} - \frac{V_{out}(s)}{R_{load}C_{system}} \tag{2.10}$$

Equation 2.11 rearrange Equation 2.10 with respect to $\frac{V_{out}(s)}{V_{in}(s)}$.

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{\frac{R_{load}}{R_{loss}+R_{load}}}{\frac{R_{loss}R_{load}C_{system}}{R_{loss}+R_{load}} \cdot s + 1} = \frac{K}{\tau s + 1} \tag{2.11}$$

Where the system's time-constant τ and gain k are given in Equation 2.12.

$$\tau = \frac{R_{loss}R_{load}C_{system}}{R_{loss} + R_{load}}, \qquad K = \frac{R_{load}}{R_{loss} + R_{load}} \tag{2.12}$$

The relations in Equation 2.12 indicate what impact changing the C_{system} and R_{load} values have for the dynamics of the system.

Higher fidelity models

[15] [14] describes a higher fidelity model of the Cocroft-Walton voltage multiplier. Furthermore, an extensive tutorial and review of modeling and controlling buck converters are given by [12].

2.3 ABB's controller solution

The ABB controller setup has been thoroughly described in ABB's official documentation [3]. Thus, a more high-level overview is presented here.

As shown in Figure 2.17, the controller comprises two PI[10]-controllers that manipulate a shared input signal. These controllers regulate the duty cycle DC of the step-down converter in the HVC, as described in section 2.2.3.

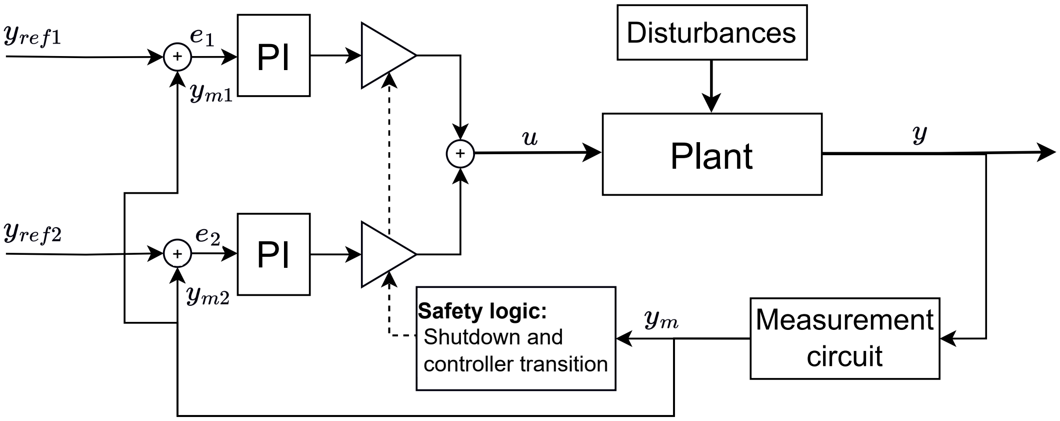


Figure 2.17: Sketch illustrating ABB's present controller. Signals marked with 1 are connected to the voltage. Signals marked with 2 are connected to the current. m indicates a measured value, u the input, e the reference deviation, and y the plant output.

Furthermore, a customized logic block is implemented to weigh and combine the controller outputs into a unified signal. The voltage controller typically dominates, but the current controller gradually takes over when the measured current surpasses a predetermined limit. Additionally, this logic block manages an enable/disable signal to the HVC, which immediately shuts down the system if the current or its derivative surpasses the established safety threshold.

2.3.1 Improvement potential

- The controller parameters are static, optimizing the controller's performance for certain workspaces only. Thus the stability and performance of the controller are reduced when the controlled system change from the optimized area.
- The framework for switching between the controllers is not optimized for all systems, resulting in undesirable behavior like oscillation between the controllers.
- The current controller has a slow/weak activation phase reducing its ability to compensate for fast disturbances in the field current. Which might lead to unnecessary

[10] Proportional integral

safety shutdowns by triggering global maximum values for the current.

- The measurements used by the controller are uncalibrated values with static error in the size of a minimum of ±5%.
- The manipulated variable, the duty cycle DC allegedly has a low resolution. The controller has been observed oscillating between neighboring values of the discrete output, indicating a low resolution but also a poorly optimized controller.

These statements are based on a qualitative controller analysis and input from ABB employees familiar with the controller application. These assumptions are the basic motivation for the work presented in this thesis.

2.4 The new control structure

The controller challenge is split into two main tasks based on the observations from section 2.3.

- A new voltage controller.
- A new current controller strategy focused on reducing any unnecessary triggering of the safety limits.

Figure 2.18 present the new control structure where the current controller is in series with the voltage controller. The main goal is to increase the high-voltage controller's robustness and performance.

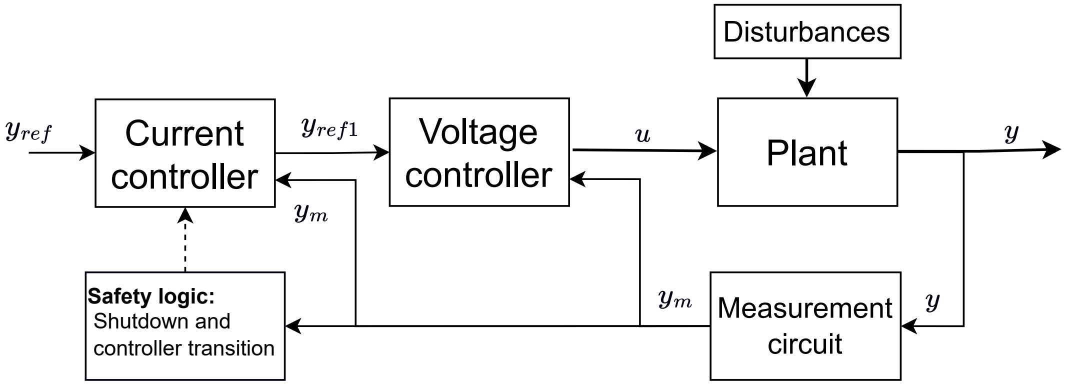


Figure 2.18: The controller challenge with the current and voltage controller in series. m indicates a measured value, 1 a value connected to the voltage, u the input, ref the reference, and y the plant output values.

Chapter 3

Current control

This chapter explores the relationship between the field voltage and field current (section 2.1.4), trying to utilize information about the system to create a controller for limiting the disturbance on the field current.

The chapter is split into two main sections:

- Parameter estimation
- Current limiting controller

3.1 Parameter estimation

This section analyzes the relationship between the field current and voltage. Furthermore, some attempts at estimating the parameters describing this relationship are made.

3.1.1 Overview and analysis

As described in section 2.1.4, the field current is defined by equations 3.1 and 3.2, but best by Equation 3.2.

$$I_{Sig(t)} = K_{(t)} \cdot U_{(t)}^2 \tag{3.1}$$

$$I_{Dch(t)} = K_{(t)} \cdot U_{(t)} \cdot [\, U_{(t)} - U_{0(t)} \,] \tag{3.2}$$

$I_{(t)}$ − Field current

$K_{(t)}$ − Current scaling variable

$U_{(t)}$ − Applied field voltage

$U_{0(t)}$ − Current onset voltage threshold

Valid/typical range:

$I_{(t)} \; \exists \, [0, 1000] \, \mu A$

$U_{(t)} \; \exists \, [0, 100] \, kV$

$U_{0(t)} \, \exists \, [0, 100] \, kV$

$K_{(t)} \; \exists \, [0, 1] \, -$

Limiting relations

$U_{0(t)} =< U_{(t)}$

The variables $I_{(t)}$ and $U_{(t)}$, are measured while $K_{(t)}$ and $U_{0(t)}$ are unknown variables. Additionally, $U_{(t)}$ is a manipulatable variable, making $I_{(t)}$ manipulatable as well.

In general, Equation 3.2 is a non-linear equation with two unknown variables, which analytically does not resolve to a unique solution. This is clarified by doing some substitutions:

Inserting: $I/U = x_1, \; K = x_2, \; (U - U_0) = x_3$

Thus: $x_1 = x_2 \cdot x_3$ Where x_1 is known

Trying to solve for x_2 and x_3 has no unique answer but infinite solutions. An assumption must be made about one of them before calculating the other. On the other hand, Equation 3.1 gives a unique answer but is a rough approximation when $U_{0(t)} > 0$.

A system's structural identifiability is defined in [13]. In short, the parameters of a system are structurally unidentifiable if the set of parameters that solve the system are not unique, countable, or finite. Thus, given the conclusion that Deutsch's formula has infinite solutions for the parameters. The conclusion must be that the system is structurally unidentifiable.

3.1.2 Estimation of Sigmond's Formula

Estimating $K_{(t)}$ in from equation 3.1 is a trivial matter with a unique solution for all values of $I_{(t)}$ and $U_{(t)}$. Thus the only consideration is to reduce the impact of any measurement/process noise.

Low pass filter

A basic low-pass filter, as described in Equation 3.3, is an appropriate and computationally inexpensive solution to the problem.

$$K_{est(n)} = A \cdot k_{est(n1)} + (1-a) \cdot \frac{I_{(n)}}{U_{(n)}^2} \tag{3.3}$$

3.1.3 Estimation of Deutsch's Formula

Estimating the unknown variables in Equation 3.2 is challenging and might not have any satisfactory solutions, as indicated in section 3.1.1. Despite that, this subsection does a study into alternative strategies attempting this.

Dual low-pass filter

A simple approach is the dual low-pass filter displayed in Equation 3.4. The relationship between the convergence speed of the filters is of utmost importance. In addition to the filter, a framework of limitations keeps the estimates in between the valid relations described in Equation 3.5. The pseudocode for this filter is given in Algorithms 1.

$$\begin{aligned} U_{0est(n)} &= a \cdot U_{0est(n-1)} + (1-a) \cdot (U_{(n)} - \frac{I_{(n)}}{K_{est(n-1)}}) \\ K_{est(n)} &= b \cdot k_{est(n-1)} + (1-b) \cdot \frac{I_{(n)}}{U_{(n)} \cdot (U_{(n)} - U_{0est(n-1)})} \end{aligned} \tag{3.4}$$

$$\begin{aligned} U_{0est(t)} \to U_{(t)} \quad &\Rightarrow \quad I_{(t)} \cong K_{est(t)} \ \cdot \ U_{(t)} \\ U_{0est(t)} \to 0 \quad &\Rightarrow \quad I_{(t)} \cong K_{est(t)} \ \cdot \ U_{(t)}^2 \end{aligned} \tag{3.5}$$

The relation between the parameters a and b is essential to the filter's performance. Increasing the convergence speed of one filter over the other gives better performance only if that reflects the actual state of the system. I.e., the variable with the most variance should have the fastest converging filter.

Algorithm 1 Dual low pass filter with limitation framework

```
Initialize filter and define parameters:
$a = -$
$b = -$
$exp_{max} = -$ % Abs. max = 2, This value indirectly sets $min(U_{0est})$.
while Estimating do
    Read measurements $I_{(n)}$ and $U_{(n)}$
    if $I_{(n)} \approx 0$ then
        $U_{0\,update} = U_{(n)}$
        $K_{update} = I_{(n)}/U_{(n)}$
    else
        $U_{0\,update} = U_{(n)} - \frac{I_{(n)}}{K_{est(n-1)}}$
        $K_{update} = \frac{I_{(n)}}{U_{(n)} \cdot (U_{(n)} - U_{0est(n-1)})}$
    end if
    if $U_{0\,update} > U_{(n)}$ then
        $U_{0\,update} = U_{(n)}$
    else if $U_{0\,update} < 0$ then
        $U_{0\,update} = 0$
    end if
    if $K_{update} > I_{(n)}/U_{(n)}$ then
        $K_{update} = I_{(n)}/U_{(n)}$
    else if $K_{update} < I_{(n)}/U_{(n)}^{exp_{max}}$ then
        $K_{update} = I_{(n)}/U_{(n)}^{exp_{max}}$
    end if
    $U_{0\,est.(n)} = a \cdot U_{0\,est.(n-1)} + (1-a) \cdot U_{0\,update}$
    $K_{est.(n)} = b \cdot U_{0\,est.(n-1)} + (1-b) \cdot K_{update}$
end while
```

Figure 3.1 and 3.2 depict the estimation results from two sets of filter parameters compared to the true values and the K values given Sigmond's formula. The figures highlight the performance when the parameters are well-adjusted and what might happen when the actual state does not match the filters.

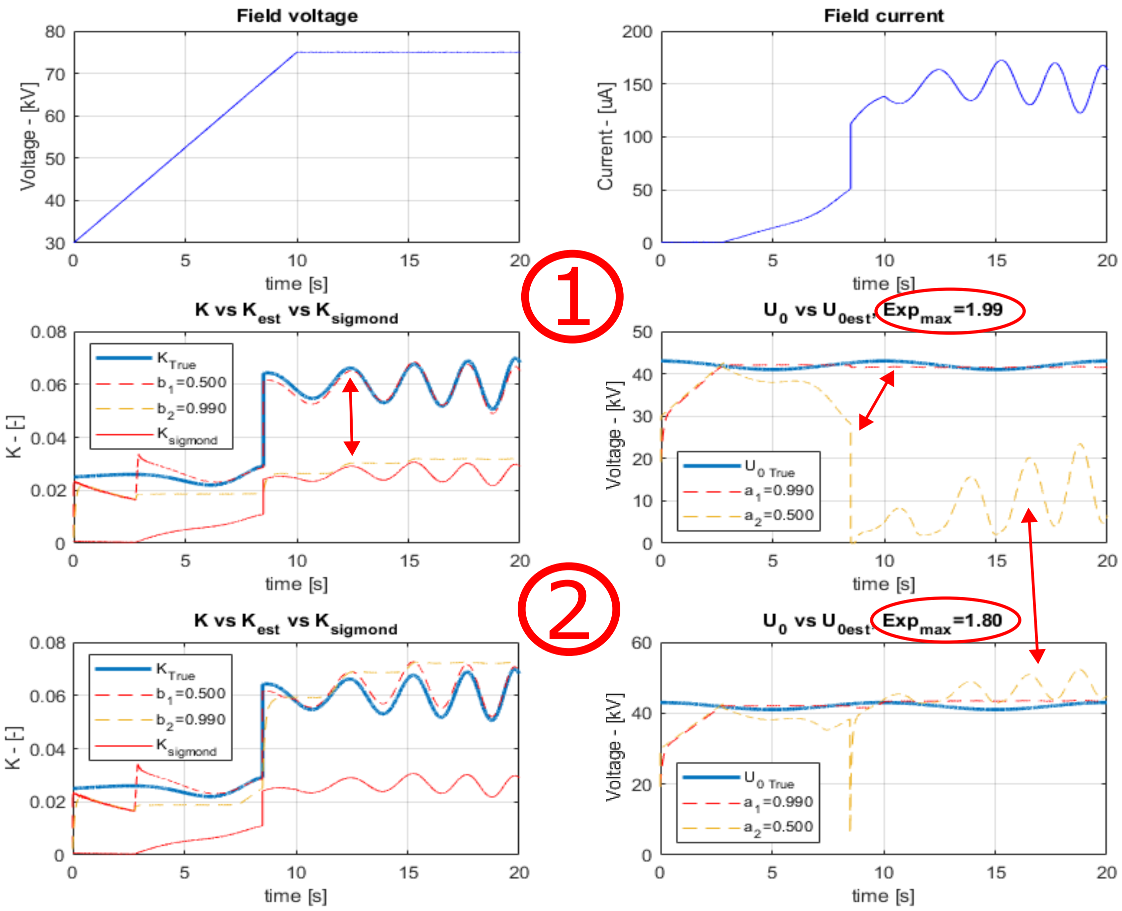


Figure 3.1: Dual low pass filter used for estimation of the unknown variables $U_{0(t)}$ and $K_{(t)}$ with two sets of filter parameters. The filter coefficients marked with number one for the filter converging fast for $K_{(t)}$, whereas those marked with two represent the filter converging faster for $U_{0(t)}$. The plots below the 1. marking illustrate how the filter converging faster for $K_{(t)}$ performs best when $K_{(t)}$ is the variable that varies the most. For the plots below the 2. marking, the Exp_{max} parameter is reduced, giving the filter converging fast for $U_{0(t)}$, i.e., is poorly adjusted to the system, better performance.

These results clarify that the filter will not automatically converge toward the true values. However, given an assumption that U_0 is varying less, the appropriate parameters might give an estimate closer to the true state than simply using Sigmond's formula. Making the correct assumptions are crucial for the performance of the filter.

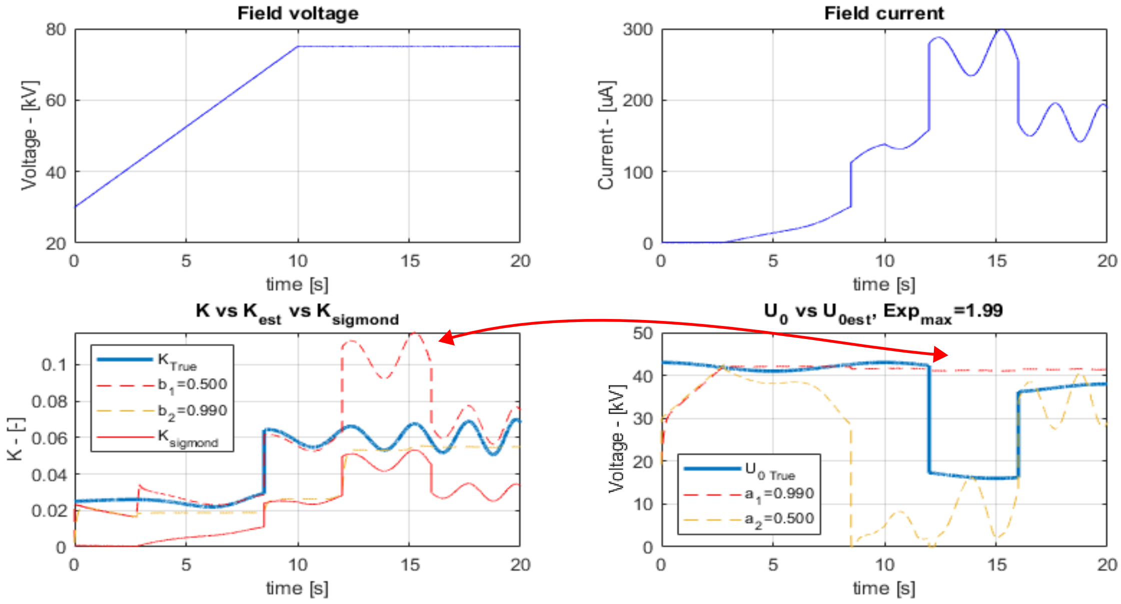


Figure 3.2: Dual low pass filter used for estimation of the unknown variables $U_{0(t)}$ and $K_{(t)}$. The red arrow highlights how the filter converges fast for $K_{(t)}$ believing that the change is happening to $K_{(t)}$ and not $U_{0(t)}$ as is the actual case. The same can be seen the other way around for the other filter.

Recursive least squares filter

The RLS[1] filter[11] solves a least squares problem estimating the parameters α_n that minimizes the error $e_{(n)}$. Figure 3.3 illustrates such a scheme where $u_{(n)}$ is the input, $\hat{y}_{(n)}$ the estimated output, and $y_{(n)}$ the actual output of the system.

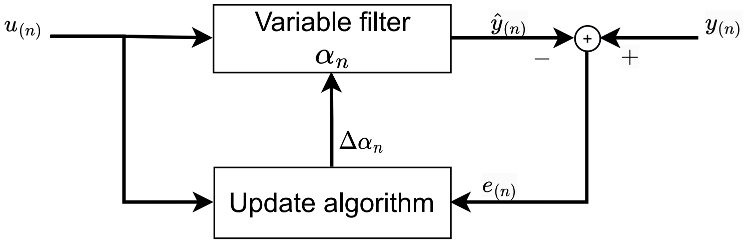


Figure 3.3: Illustration of the fundamental concept behind the RLS filter.

Equation 3.6 derives an expression for the output as a linear combination of an input signal vector and a parameter vector, suitable for use with the RLS algorithm. Equation 3.7 presents a version of the RLS without any matrix inversions in the recursive algorithm.

[1] Recursive least squares

Given:
$$I_{(n)} = K_{(n)} \cdot U_{(n)} \cdot [\, U_{(n)} - U_{0(n)} \,]$$
$$\frac{I_{(n)}}{U_{(n)}} = K_{(n)} \cdot U_{(n)} - K_{(n)} \cdot U_{0(n)}$$

Redefine:
$$y_{(n)} = I_{(n)}/U_{(n)}, \quad u_{(n)} = \begin{bmatrix} U_{(n)} \\ \text{-1} \end{bmatrix}, \quad \alpha_{(n)} = \begin{bmatrix} K_{(n)} \\ K_{(n)} \cdot U_{0(n)} \end{bmatrix} \tag{3.6}$$

Thus:
$$y_{(n)} = \alpha^T_{(n)} u_{(n)}$$

Recursive algorithm:

$$L_{(n)} = \frac{P_{(n-1)} u_{(n)}}{\lambda + u^T_{(n)} P_{(n-1)} u_{(n)}}$$

$$P_{(n)} = \frac{1}{\lambda} \left(P_{(n)-1} - L_{(n)} u^T_{(n)} P_{(n-1)} \right)$$

$$\hat{\alpha}_{(n)} = \hat{\alpha}_{(n-1)} + L_{(n)} \left(y_{(n)} - u^T_{(n)} \hat{\alpha}_{(n-1)} \right)$$

Initialization:

$$P_{(n_0)} = \left(\textstyle\sum_{i=1}^{n_0} u_{(i)} u^T_{(i)} \right)^{-1} \tag{3.7}$$

$$\alpha_{(n_0)} = P_{(n_0)} \sum_{i=1}^{n_0} u_{(i)} y_{(i)}$$

λ – Forgetting factor

Figure 3.4 presents the results from an experiment using the RLS filter on simulated data.

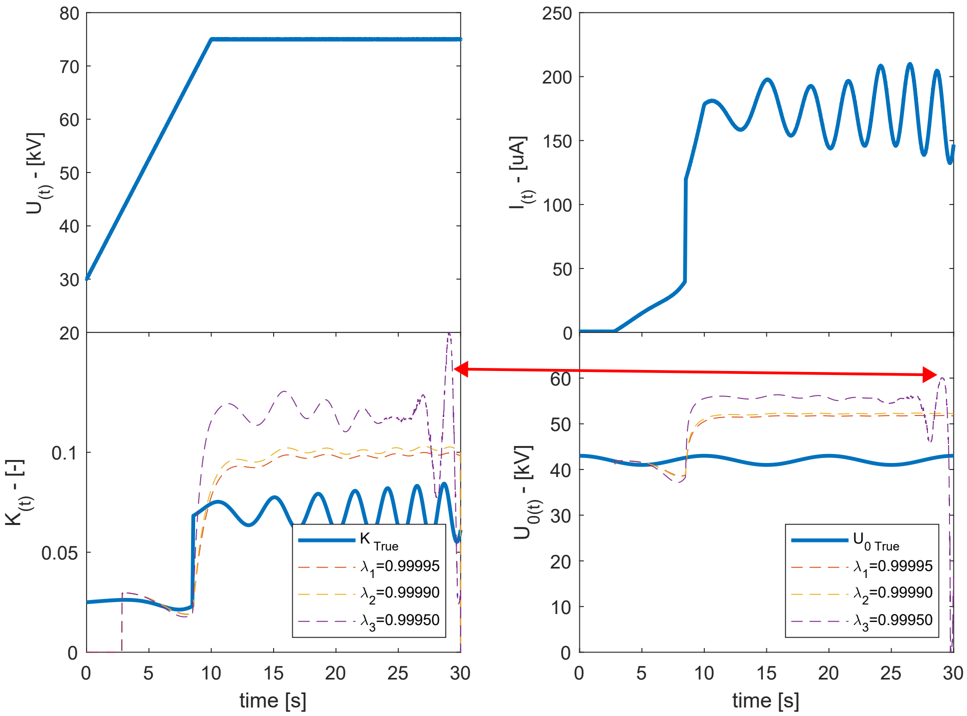


Figure 3.4: RLS filter used for estimation of the unknown variables $U_{0(t)}$ and $K_{(t)}$. The plots contain the results from filters using three different forgetting factor values. The red arrow highlights where one filter becomes numerically unstable due to a high forgetting factor. The plots illustrate how the RLS filters do not converge to the true solution.

The experiment confirms the expected results that the RLS filter does not converge toward the true parameter values. Additionally, an numerical instability of the algorithm is revealed. RLS filters based on adaptive Lattice-Ladder algorithms[11] have better properties with respect to numerical stability and would thus be a better algorithm to use. However, as the results confirm that the system is unidentifiable, improving the filter is deemed unnecessary.

3.1.4 Conclusion

The experiments with different filters confirm what was implied in section 3.1.1. That the system, given the current knowledge, is structurally unidentifiable. Thus the most predictable estimation strategy is Sigmond's formula, which gives a unique answer. However, an estimation based on Deutsch's formula, with manually fitted parameters, might provide an estimate closer to the actual state of the system, as Sigmond's formula is more of a worst-case estimate.

Thus the two most viable options are:

1. Estimate using Sigmond's formula and a single low pass filter.
2. Estimate using Deutsch's formula and a dual low pass filter with constraints.

The performance of these two options can best be decided by measuring the performance of a controller implementing them.

3.2 Current limiting controller

This section describes a current-limiting controller based on the field voltage and field current relationship. Furthermore, the controller is tested in simulations using different estimates.

3.2.1 Motivation

The current limiting controller is an attempt at satisfying the following ABB requirements:

- Reduction unnecessary shutdowns due to the field current triggering global maximum limits.
- Faster reaction time to safety incidents.
- Simple implementation in ABB's system.
- Minimized adjustment of the field voltage, as this might impact the overall paint result.

The controller aims to improve the double PI-controller described in section 2.3. The PI controllers' main weakness, with respect to the above-stated requirements, is a weak entry on the current controller part, making it less efficient at reducing fast current spikes.

An improvement is to be achieved by addressing any significant current deviation more aggressively.

3.2.2 Controller description

The controller's core is a voltage setpoint calculation based on the reference current and the estimated disturbance parameters. This relation is illustrated in Figure 3.5.

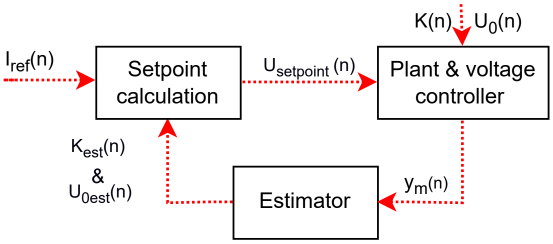


Figure 3.5: The current limiting controller setup illustration. The variables in the figure have been introduced in section 3.1.

Another essential part of the controller is the framework describing when and how the controller should be activated and deactivated. This framework is outlined in Algorithm 2, describing the controller algorithm, and Figure 3.6, illustrating how the controller is supposed to work.

Algorithm 2 Current limiting controller

```
Initialization of parameters, estimators, etc.

while controller active do

    # Various variable, measurement and estimation updates
    $I_{upper\,lim} = I_{ref(n)} * factor$
    $I_{lower\,lim} = I_{ref(n)} * factor$
    $U_{upper\,lim} = U_{ref(n)} * factor$
    ............
    ............

    # Standard safety shutdown
    if $I_{(n)} > I_{max}$ OR $I'_{(n)} > I'_{max}$ then
        Disconnect power
    end if

    # Activation/deactivation framework
    if $I_{(n)} < I_{lower\,lim}$ OR $U_{(n)} > U_{upper\,lim}$ then
        Controller_on = False
        $U_{setpoint(n-1)} = U_{(n)}$
    end if
    if $I_{(n)} > I_{upper\,lim}$ then
        Controller_on = True
    end if

    # Setpoint calculation
    if Controller_on then
        $U_{setpoint(n)} = \frac{U_{0est(n)}}{2} + \frac{\sqrt{K_{est(n)} \cdot (K_{est(n)} \cdot U^2_{0est(n)} + 4 \cdot I_{ref(n)})}}{2 \cdot K_{est(n)}}$
        if $U_{setpoint(n)} > U_{ref(n)}$ then
            $U_{setpoint(n)} = U_{ref(n)}$ # Controller stability measure
        end if
    else
        $U_{setpoint(n)} = a \cdot U_{setpoint(n-1)} + (1-a) * U_{ref(n)}$
    end if

end while
```

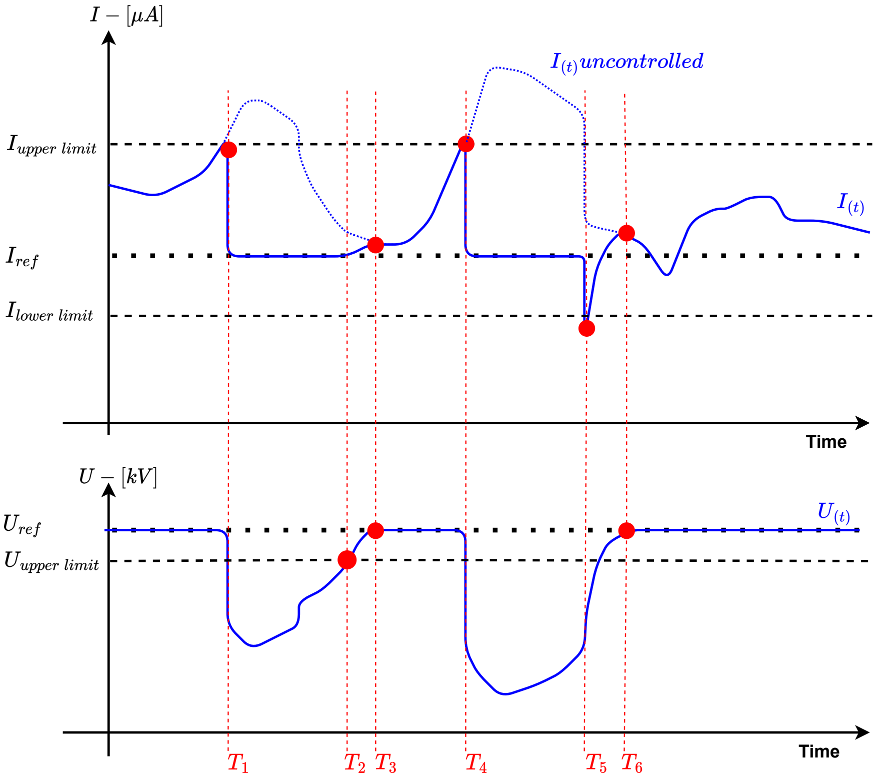


Figure 3.6: Sketch illustrating the main stages of the current limiting controller. At the time T_1, $I_{(t)} > I_{upper\,limit}$, and the controller is activated, lowering the voltage forcing $I_{(t)} = I_{ref}$. At the time T_2, $U_{(t)} > U_{upperlimit}$, activating a smooth transition decoupling the current limiting controller, thus returning $U_{(t)} \to U_{ref}$. At the time T_3, this transition is completed. Furthermore, at time T_4, the controller is activated once again. Eventually, at time T_5, a sudden drop in $I_{(t)}$, where $I_{(t)} < I_{lower\,limit}$, activate a smooth transition away from the current limiting controller. Finally, at the time T_6, the current limiting controller is decoupled.

3.2.3 Testing and evaluation

The tests document how important variables affect the controller's performance. A standard test is used to evaluate the controllers' performance across the range of possible plants by doing a series of simulations iterating the plant's U_0 value (x-axis in the standard test result plots). Thus presenting the controllers' performance across the entire workspace.

The testing and test description is split into eight parts:

1. Test setup.
2. Impact of the model assumptions.
3. Impact of the estimator convergence speed.
4. Impact of the system's capacitance.
5. Impact of the disturbance step size.
6. Impact of the reference voltage.
7. Functionality illustration.
8. Comparison to I-controller.

Test setup

To evaluate the controller's performance, the focus is on how the controller output, i.e., the estimated $U_{setpoint}$, converges towards the optimal $U_{setpoint}$. The accuracy of the estimates is essential for the performance of the controller. A step in the field current is the worst-case incident that maximizes the estimator's error. Thus a step in the current is used to test the controller's performance. The performance indicators are:

- $T_{rise\,63.2\%}$ = Time until $\Delta U_{est.set} = 0.632 \cdot \Delta U_{optimal.set}$
- $T_{settle\,\pm 5\%}$ = Time until $\Delta U_{est.set}$ is within ±5% of $\Delta U_{optimal.set}$
- $Overshoot_{\%} = \dfrac{\Delta U_{max\,est.set} - \Delta U_{optimal.set}}{\Delta U_{optimal.set}} \cdot 100\%$

The controller is tested with the following estimation methods:

- Deutsch's formula using a dual low pass filter.
- Sigmond's formula using a single low pass filter.
- The true values of Deutsch's formula are assumed to be known, resulting in the optimal version of the controller.

For simplicity, all the dynamics of the HVC and Cascade are assumed to be simplified by another controller into a maximum change rate for the field voltage $U_{(n)}$. The system's capacitance and the current $I_{(n)}$ gives that maximum change rate.

The following setup and assumptions are used in the subsequent test if not stated otherwise.

- Controller and estimator sample rate: $Ts = 1ms$.
- $U_{ref} = 80kV$, $I_{ref} = 200\mu A$, $I_{upper\,threshold} = 1.1 * I_{ref}$
- Disturbance: A step in $I(n)$ of $+20\%$.
- Maximum discharge rate: $\dfrac{dU_{(n)}}{dt} = \dfrac{I_{(n)}}{C}$, where C is the capacitance of the system (section 2.2.4). C is assumed to be a constant typically in the range of 100-2000pF. C is assumed to be 200 pF for this testing. For simplicity, the maximum charge rate is set to a conservative three times the discharge rate. The charge rate is of less importance for the current limiting controller.
- The controller deactivation thresholds are disconnected during the tests to uncover the full extent of any undesirable properties.

Impact of the model assumptions

Figure 3.7 presents the performance data for the controller using four different estimators. The estimators make varying assumptions about the systems' U_0 value, influencing their performance.

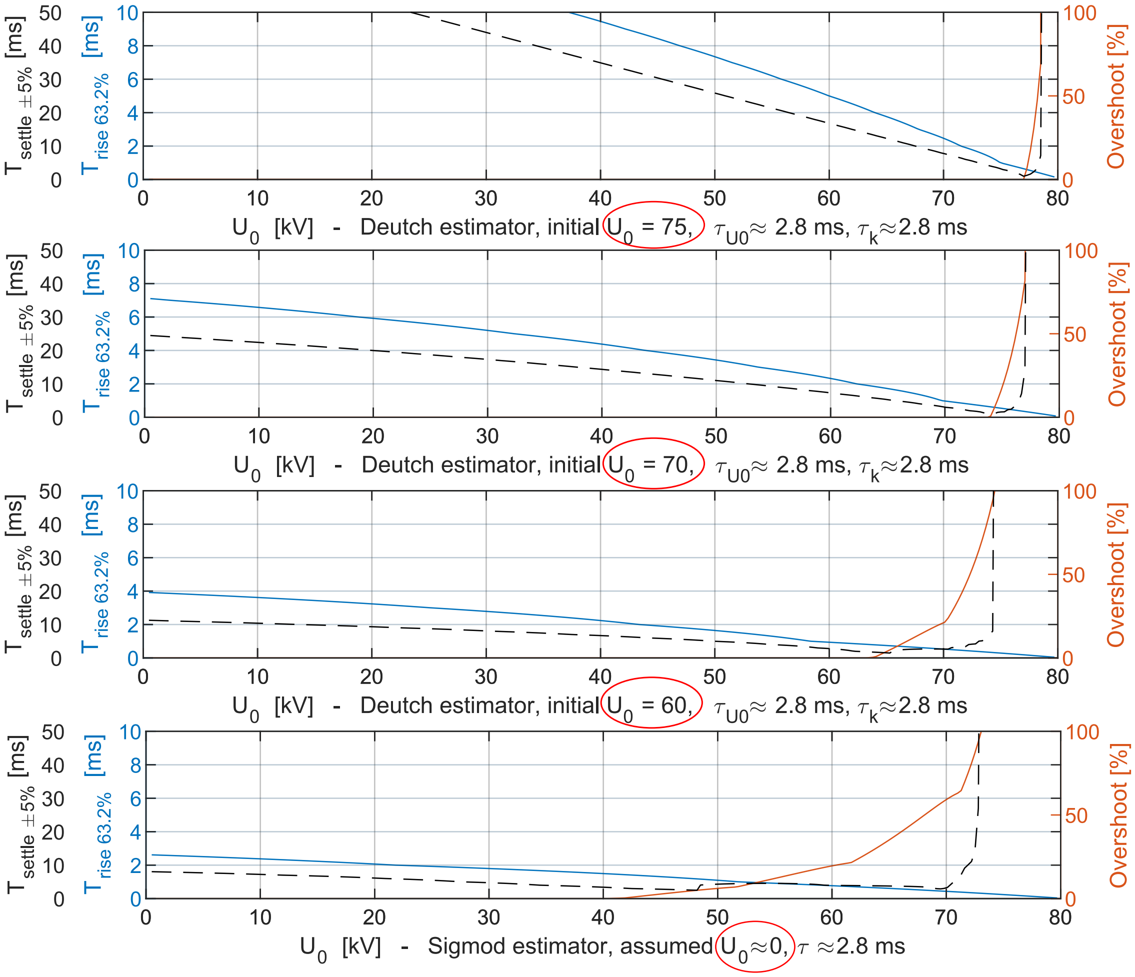


Figure 3.7: This figure highlights how changing the initial conditions of the estimators, i.e., changing the assumptions about the underlying model, affects the controller's performance. Note that the last instance uses Sigmond's formula, and its estimators time constant is not directly comparable to the other estimators.

The results in Figure 3.7 present how the choice of U_0 affects the performance of the controllers. Note that with an estimator assuming $U_0 \approx U = 80kV$, the range of U_0 values where the performance is good shrinks considerably. On the other hand, using an estimator assuming $U_0 \approx 0$ performs better for a wide range of U_0 values.

Impact of the estimator convergence speed

Figure 3.8 presents the performance data for the controller using estimators with increasing convergence speeds.

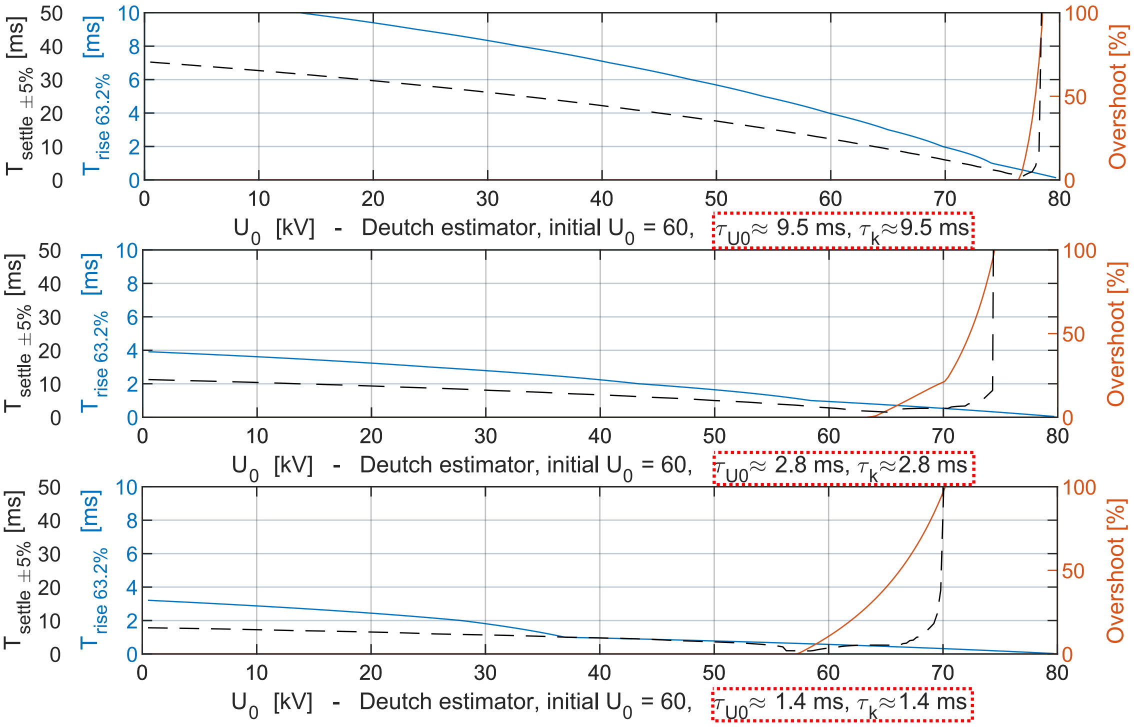


Figure 3.8: This figure highlights how the filter time constant/convergence speed affects the controller's performance.

By comparing figures 3.7 and 3.8, it is evident that the estimators' convergence speed affects the controllers' performance similarly to how the model assumptions do. A slower estimator reflects $U_0 \to U$, while the fastest estimator gives the best performance where the assumed U_0 equals the actual U_0.

Impact of the system's capacitance

The discharge rate for U, is given by $\frac{dU_{(n)}}{dt} = \frac{I_{(n)}}{C}$, where C is the system's capasitance. Thus a system's capacitance influence how fast it is possible to lower the voltage. Figure 3.9 illustrates how a system's capacitance influences the controller's performance.

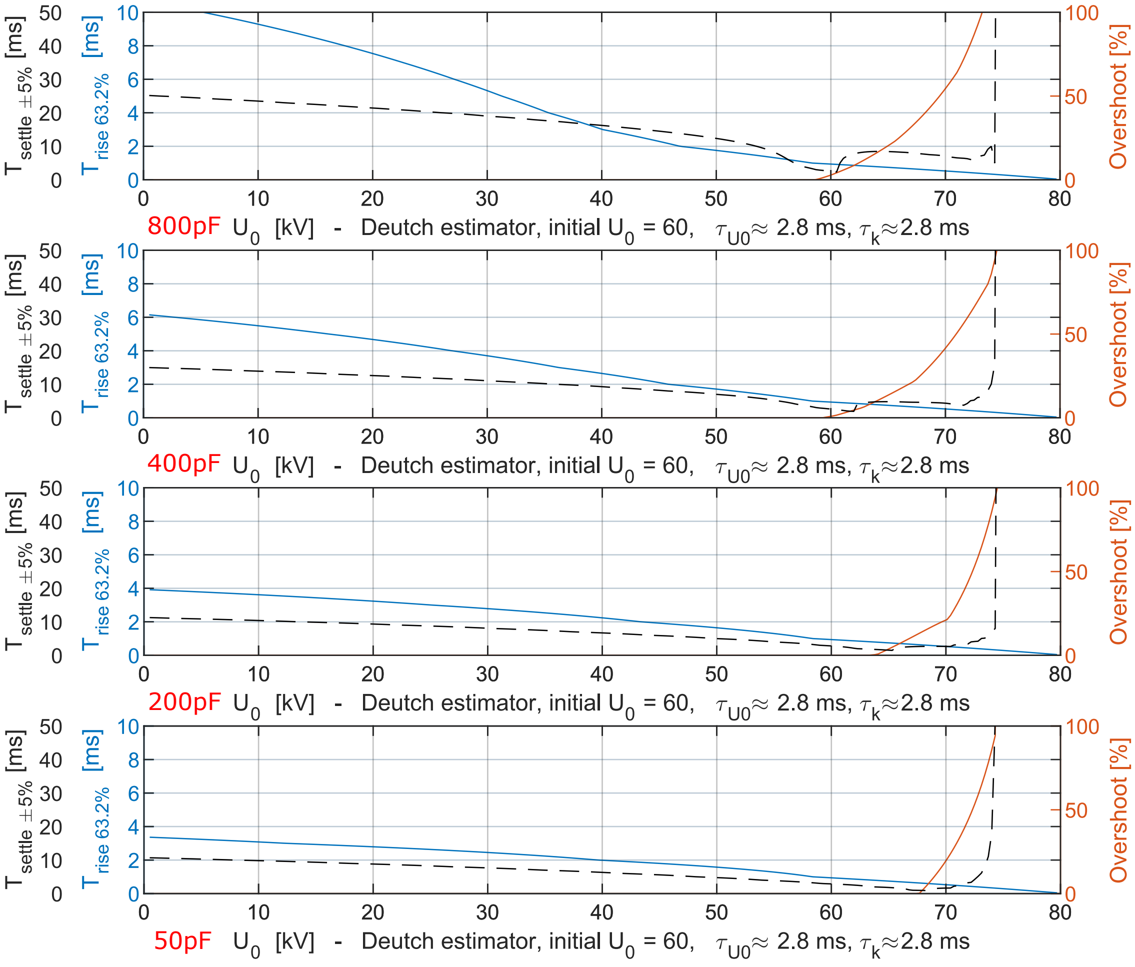


Figure 3.9: This figure highlights how the controller performs for systems with various capacitances.

The results from Figure 3.9 indicate that the faster the voltage change rate, the better the controller's performance.

Impact of the disturbance step size

Figure 3.10 presents the results from where larger steps in the current are applied to the system.

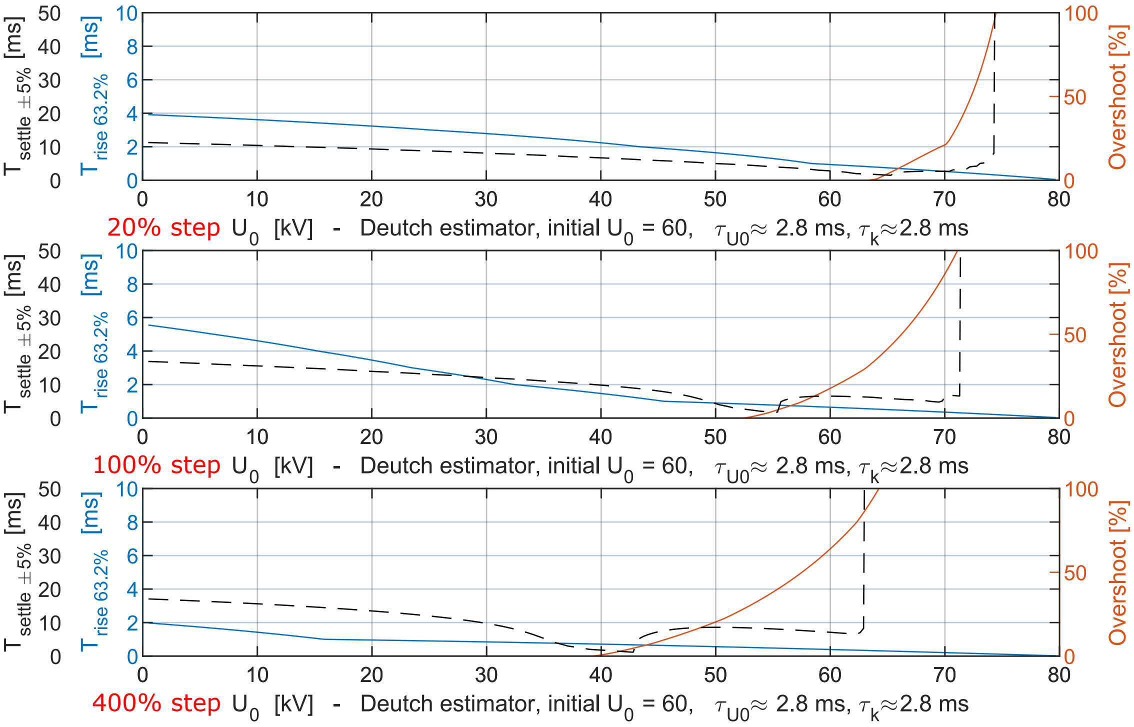


Figure 3.10: This figure highlights how the controllers' performance is affected by the size of the disturbance, i.e., the current step.

A larger step in the current generally requires a large voltage reduction. The large current also contributes to a faster discharge of the system. Thus when, as depicted in Figure 3.10, the step size is increased, the range of U_0 with good performance is reduced, while the performance in the remaining workspace is increased. An effect similar to what is described in Figure 3.8, where the impact of the convergence speed of the estimators is illustrated.

Impact of the reference voltage

Figure 3.11 presents the controller's performance for a series of U_{ref} values and a corresponding set of U_0 assumptions for the estimator.

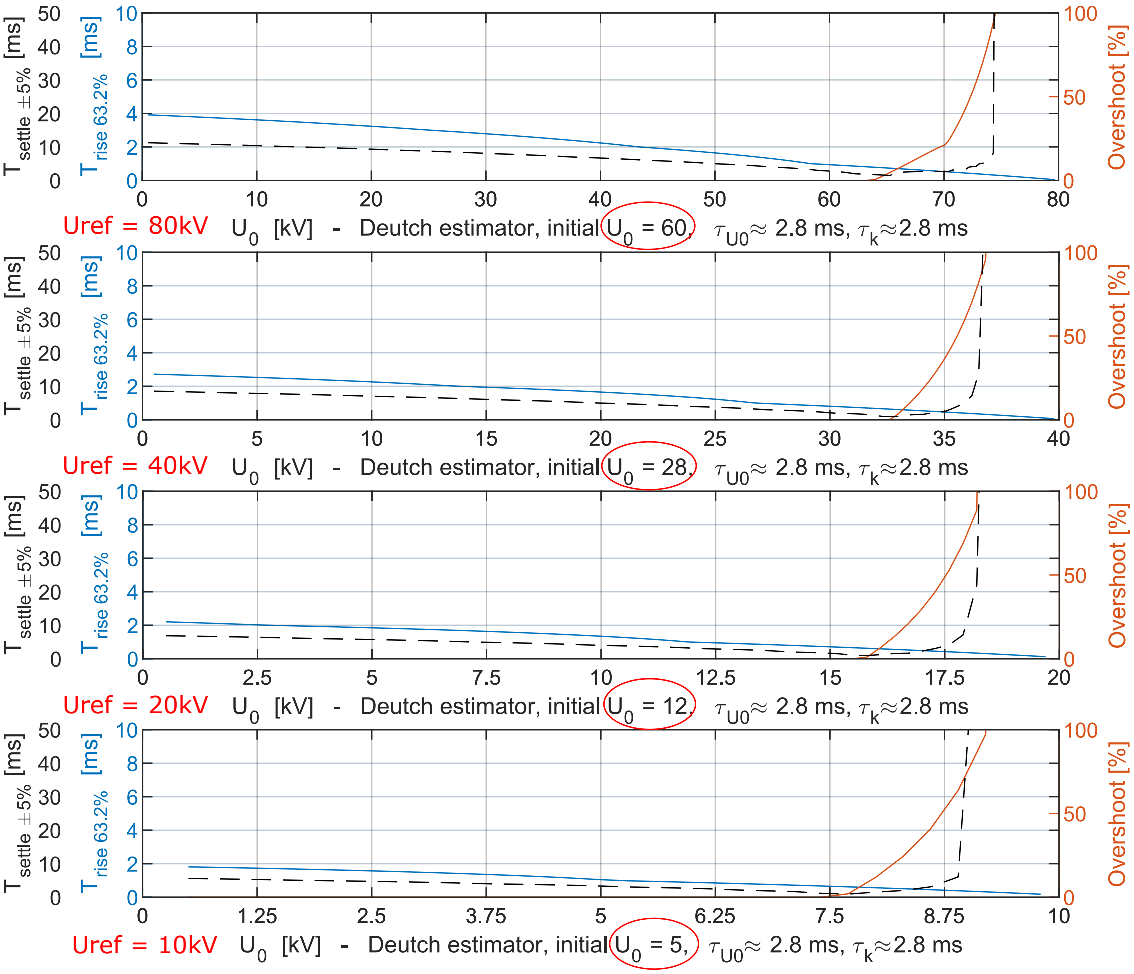


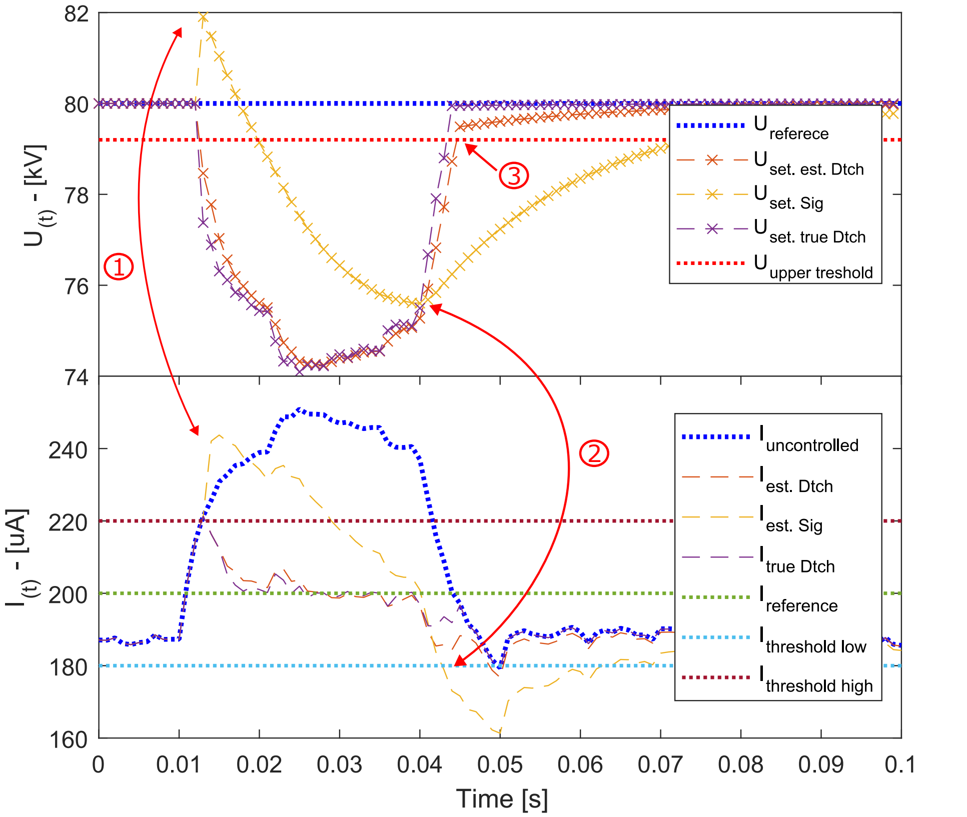
Figure 3.11: This figure display the controllers' performance for a series of U_{ref} values and corresponding assumptions for the estimators U_0 value.

Figure 3.11 illustrates that a lower reference voltage generally increases the controller's performance when the current is kept at a constant level, i.e., a lower voltage gives a proportionally higher maximum rate of change for the voltage when the current remains unchanged. Additionally, the figure illustrates how the performance characteristics of the controller remain approximately constant when the U_0 assumptions for the estimator are set by the constant relation to U_{ref}:

$$U_0 = (U_{ref})^a, \text{ where } a \in (0-1)$$

Functionality illustration

Figure 3.12 illustrates how the controller reacts to current disturbance. The figure highlights both the activation and deactivation of the controller and what a large estimation error might do to the controller. This test differs from the other tests by having active deactivation limits and not a clean current step.



Ts = 1.0 ms, Deutch filter: $\tau_{U0} \approx$ 2.0 ms, $\tau_k \approx$2.0 ms, Sigmond filter: $\tau \approx$19.5 ms

Figure 3.12: Illustration comparing a controller with a slow estimator based on Sigmond's formula to a controller with a faster estimator based on Deutsch's formula to a controller using the true parameters of the system. Point 1 highlights how the calculated $U_{setpoint}$ might go wrong when the estimator's error is significant. In this case, the error is attributed to the slowness of the estimator. Points 2 and 3 highlight the decoupling of the current limiting controller when the I and U deactivation thresholds are breached.

Under standard conditions, the reverse voltage highlighted by point 1 in Figure 3.12 will not appear. It is stopped by the conditional statement in the controller algorithm, lines 29-31 in Algorithm 2. However, it is included to illustrate the possible effects of estimation errors caused by the combination of a slow estimator and wrong assumptions. Other artifacts connected to the estimation error are overshoot and oscillations around the optimal $U_{setpoint}$.

Comparison to I-controller

The results of the current limiting controller algorithm closely resemble those of a simple integral (I)-controller. Figure 3.13 illustrates how an I-controller performs similarly to the current limiting controller. However, the current limiting controller excels in that its performance remains similar across a larger workspace. Thus the controller would compare more to an I-controller with gain scheduling. This is illustrated in Figure 3.14, where the current step size is changed from the standard experiment. The figure emphasizes that the I-controller's performance changes while the controller's performance remains. This comparison also reflects how the current limiting controller compares to the existing PI-controller from section 2.3.

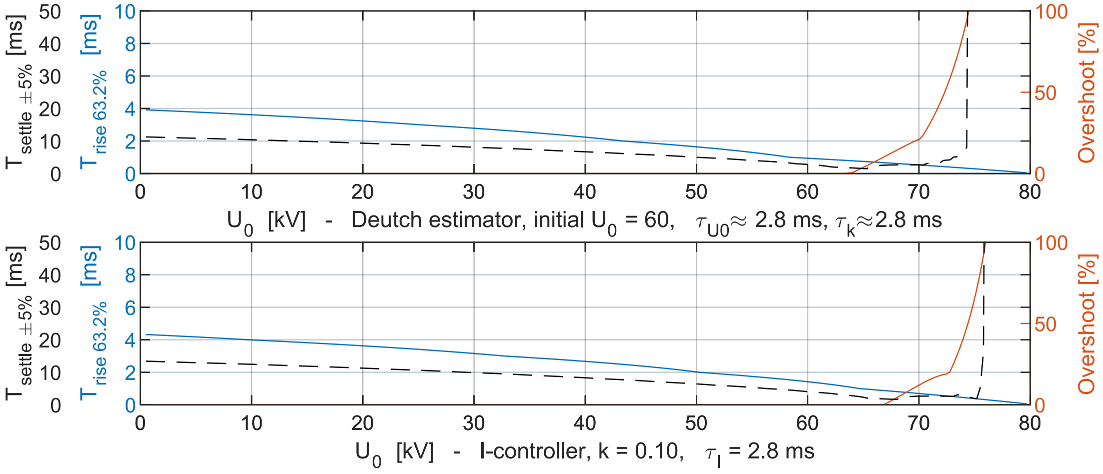


Figure 3.13: Comparison to the I-controller under the standard testing conditions.

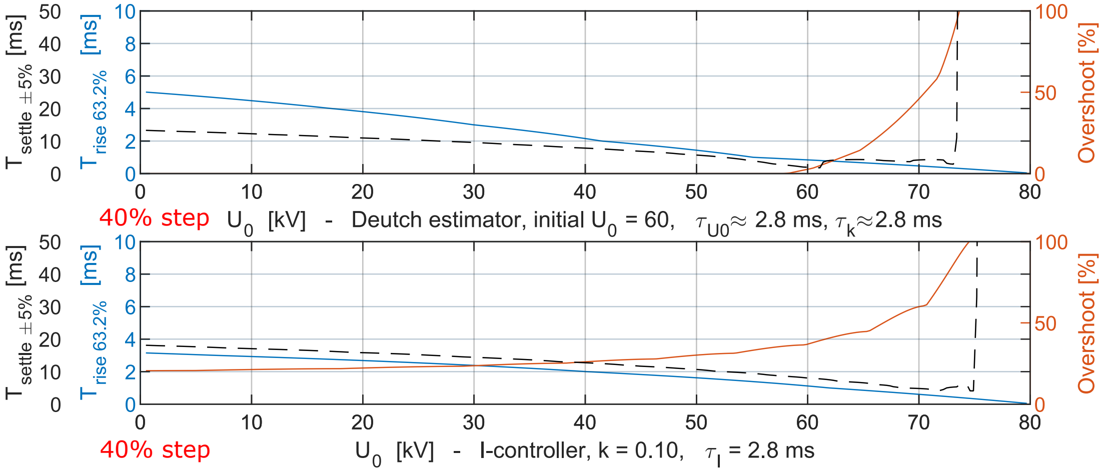


Figure 3.14: Comparison to the I-controller with the reference- current at 200 μA, and a current step of 50% up to 300 μA. The performance indicators suggest that the I-controller adapts poorly to a larger disturbance. Note how the overshoot and settle time has drastically increased for the entire range of U_0.

3.2.4 Configuration and implementation considerations

Conceptual configuration protocol:

1. Plan and configure the painting program.
2. Do tests throughout the programs' path to determine the $U_0(t)$ values.
   - Use these results to configure the controllers', i.e., the estimators' parameters.
3. Log the field current in a series of test runs.
   - These test runs should include the most likely variations in the process. Different paint types, extrema positions of the workpiece, temperature, humidity, etc.
4. Based on the resulting field current curves, a rough reference current and limit factors are added to the paint program.
   - If the paint program is uniform, the current reference is typically a constant value for the entire program. Otherwise, the current reference would have different constant values for the corresponding regions of the program.

Implementation considerations:

The current limiting controller has a computationally simple algorithm suitable for implementation on the fixed point processor, the DSP described in section 2.2.3.

Alternatively, the algorithm might be implemented directly on the FPGA on the CCPU board (section 2.2.3).

As mentioned in section 2.2.2, the measurements from the HVC are filtered with a low pass filter, limiting their usable resolution to a maximum of 0.75-1.5 kHz. Making this a limiting factor for the current limiting controllers' update frequency.

The paint jobs are sometimes stopped and restarted mid-program. This might be an issue if the stop/restart leads to field currents deviating from the reference, activating the current limiting controller. It could be solved by delaying the activation of the current limiting controller after restarts. However, this is most likely not an issue but rather an area where the current limiting controller could contribute to keeping the system running.

3.2.5 Conclusion

The simulation test indicates that the current limiting controller is a robust solution superior to the existing controller. The controller effectively reduces current disturbances, limits the corresponding voltage disturbance with an activation/deactivation framework, and remains stable for a significant part of the workspace. The choice of model assumptions, estimator convergence speed, and the system's maximum voltage change rate significantly influence the controller's performance. A considerable improvement over the existing controller design is likely accomplished by configuring the controller as instructed in section 3.2.4. However, this is unverified as implementation and testing on an actual system still need to be performed.

Overall, the proposed controller demonstrates a potential for enhancing the electrostatic painting system's performance by decreasing unnecessary safety-related stops and reducing the reaction time for actual safety-hazard incidents.

Chapter 4

Voltage control

This chapter addresses the controller problem for the generation of a high-voltage output using the HVC and Cascade. The chapter consists of the following sections:

1. Data acquisition
2. Model estimation
3. Reinforcement learning controller

If details are provided or data collected, the specific versions described/used are the HVC-02 rev.5 and the CU930B cascade.

4.1 Data acquisition

This section describes the data acquisition and processing performed to capture the dynamics of the HVC and Cascade. Furthermore, the resulting datasets' coverage, validity, and error are addressed. The section is split into the following parts:

- Test setup
- Data acquisition program overview
- IPS logging and HVC signals
- Python test executor
- Matlab data processing
- Results and conclusion

4.1.1 Test setup

The test setup consists of a test computer communicating with two CCPU boards mounted on an HVC and another I/O board. The HVC is used for logging measurement data and adjusting the input signals, the duty cycle values described in section 2.2.3. The I/O board controls a series of pneumatic actuators that connect/disconnect resistors to the cascade output, simulating a change in the field current.

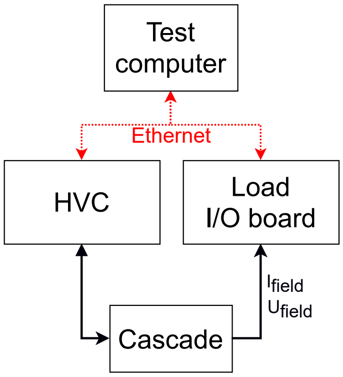


Figure 4.1: The test setup, where the cascade output is connected to a variable load controlled by an I/O board.

The test computer controls the setup by sending commands to the HVC and I/O board while the measurement data is logged on the HVC CCPU board. After a test sequence, the HVC saves the data locally in a CSV file, which the test computer retrieves using FTP[1]. The data is then processed and stored for further exploitation. Figure 4.2 illustrates the relations between the test computer and the internal HVC components.

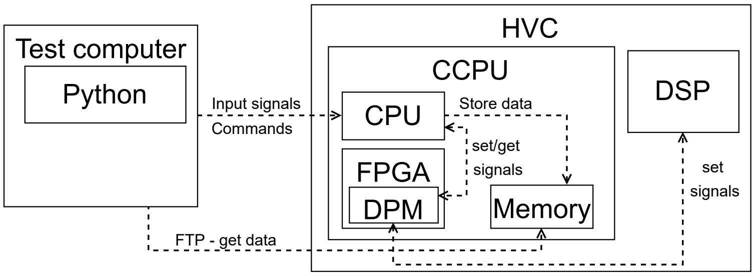


Figure 4.2: Illustration of the test signals, data transfer, and data storage for the test computer and HVC. The DPM[2] in the HVC is a resource shared by the CPU, FPGA, and DSP. The FPGA logs the measurements to the DPM, from where the CPU collects the data storing it in another location.

[1] File transfer protocol

4.1.2 Data acquisition program overview

The data acquisition program is split into two main levels. That is the main program that runs on the test computer and the software running on the CCPU boards. Figure 4.3 presents a detailed sequence diagram describing these relations. The main program and select features on the CCPU boards have been specifically developed for this data acquisition task and are described in more detail.

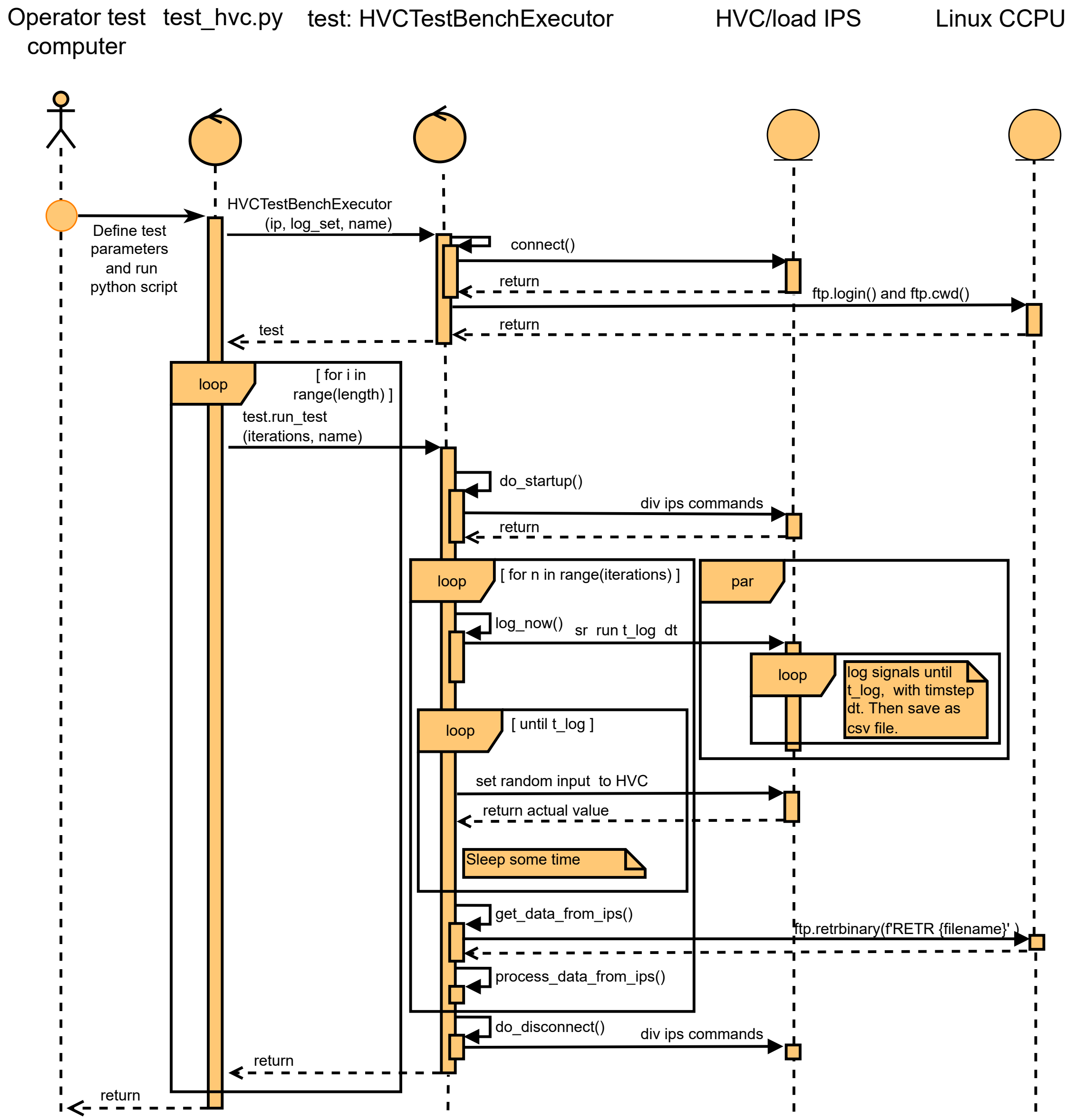

Operator test computer
test_hvc.py
test: HVCTestBenchExecutor
HVC/load IPS
Linux CCPU
Define test parameters and run python script
HVCTestBenchExecutor (ip, log_set, name)
connect()
return
ftp.login() and ftp.cwd()
test
return
loop
[for i in range(length)]
test.run_test (iterations, name)
do_startup()
div ips commands
return
loop
[for n in range(iterations)]
par
log_now()
sr run t_log dt
loop
log signals until t_log, with timstep dt. Then save as csv file.
loop
[until t_log]
set random input to HVC
return actual value
Sleep some time
get_data_from_ips()
ftp.retrbinary(f'RETR {filename}')
process_data_from_ips()
do_disconnect()
div ips commands
return
return

Figure 4.3: A simplified sequence diagram illustrating how the data acquisition program works. The format of the sequence diagram is based on articles from IBM[4], and diagrams.net [8]

4.1.3 IPS logging and HVC signals

Within IPS's (section 2.2.3) existing framework for data logging, signals are only logged at a minimum timestep of 16 ms. To capture the dynamics of the system, faster logging is required. However, an unofficial IPS feature, SR log, is used for logging faster than the 16 ms intervals. This function is limited to a minimum log interval of 1 ms and a maximum of 10 seconds duration. However, a 1 ms log interval is unfeasible as the logging thread is put at the lowest priority in an environment with several other high-priority tasks. The original log-related functions are used in the IPS command line interface using the following input:

1. sr toggle |signal name| - Select signals to be logged.
2. sr run |log duration [s]| |log interval [ms]| - Start logging and save to CSV[3] file.

To enhance logging performance, the SR log features are upgraded in the following ways:

- Increasing possible sample rate by switching to timers counting nanoseconds.
- Increased priority for the thread running sr log.
- Add two timestamps to each sample: the actual and desired times.
- Increase the maximum log duration to 100 seconds.

The modified sr run function:

- sr run |log duration [ms]| |log interval [μs]| - Start logging and save to CSV file.

Additionally, to log the desired data, new IPS signals/devices are defined for logging unfiltered raw data from the HVC. Furthermore, new functions and signals for directly setting the HVC's duty cycle inputs are also implemented. Furthermore, a manual testing mode is created, decoupling the standard regulator and allowing manual control.

All these edits are compiled into an experimental IPS version running on the HVC's CCPU board for testing.

[3] Comma separated values

4.1.4 Python test executor

The ”HVCTestBenchExecutor” class introduced in Figure 4.3 is at the core of the data acquisition program. This section gives a closer description of some essential features/-functions of this class.

The ”HVCTestBenchExecutor” has access to a framework of dictionaries describing:

- All available signals and their scaling and bias for transforming the raw data to correct physical units.
- All possible input test signals, their allowed range, step size, unit, etc.
- Calibrated values for the measurement circuit of the test bench cascade.
- Calibrated values for the load resistors.

Input signal limitation framework

The *”Input signal framework”* algorithm sends the duty cycle input commands to the HVC and load selection command to the I/O board. In short, the algorithm selects a random input signal to excite with a piecewise random value before reiterating after a random normal-distributed pause.

The input is limited by the CU930B cascade, whose maximum output values are 90 kV and 1 mA. However, these limits are somewhat exceeded to improve the valid range for the resulting model. The output is limited by a:

- Predefined range of allowed input values.
- Set of predefined maximum step for the input values when the output voltage is $<$ 70 kV, 70 $<$ 90 kV, and $>$ 90 kV.
- No input increase when the output voltage is more than 110kV. Then the input is cut by a large step.
- The variable load is not allowed to go below $100M\Omega$

Thus, the extrema values of the output voltage are limited to staying below 120 kV, which should be within the range of what the cascade is designed to handle. No documentation confirms that exceeding 90kV is safe, but the components are supposed to be robustly designed. No deterioration or malfunctions of the equipment has been observed during testing.

Process data from IPS

The class method $process\_data\_from\_ips()$ does some processing of the raw data loaded from the HVC. Algorithm 4 presents the method's pseudo-code. In short, the method generates data for the variable load connected to the cascade output, transforms the raw data into physical units, and calculates values of interest whose relations are described in section 2.2.2.

Algorithm 3 process_ data_ from_ ips(self, dataframe, file_name):

```
 1: # Create resistive load data
 2: dataframe["load"] = initial load
 3: for all  load steps do
 4:       Define start and stop values encompassing the step.
 5:       step_index = find_step( dataframe["curent_measurment"][start:stop])
 6:       dataframe["load"][step_indx:-1] = new load value
 7: end for
 8:
 9: # Do scaling and add bias values to the raw data
10: for all  raw signals do
11:       dataframe["signal"] = dataframe["signal"] * signal["scale"] + signal["bias]
12: end for
13:
14: # Calculate values of interest
15: dataframe["total_current"] = dataframe["$U_{IM}$"]/self.CASCADE["IMR"]
16: dataframe["field_current"] = ...
17: dataframe["cascade_output_voltage"] = ...
18: dataframe["total_cascade_load"] = ...
19: etc...
20:
21: # Save dataframe as csv file
22: dataframe.to_csv(directory+file_name+time)
```

The find_step() function from line 5 in Algorithm 4 utilizes the Ruptures[28] Python library. The function finds the step in the measured current and assumes this corresponds to the load step. Unfortunately, the accuracy of this algorithm is questionable, as the dynamics of the load steps impact on current vary somewhat. Thus, using data generated by this algorithm for model estimation risks introducing considerable error.

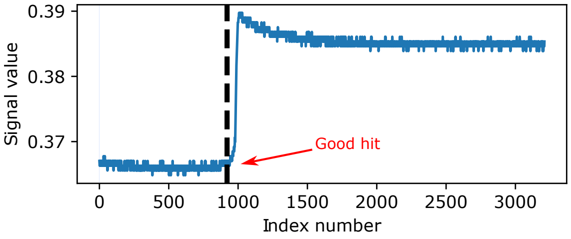


Figure 4.4: Accurate identification of a decrease in the load.

The troubles connected to the step detection algorithm are illustrated by figures 4.4, 4.5 and 4.6. Figure 4.4 and 4.5 illustrate reasonably good identification of the connection of the load. However, as the load connection per se happens before any change in the current, finding precisely when the load step happens for a corresponding current step is tricky. Missing even by a little might introduce a considerable error to the data.

Figure 4.6 present a case where the load step's impact on the current is not noticeable for the step detection algorithm. Then a standard value is used, adding even more error to the data.

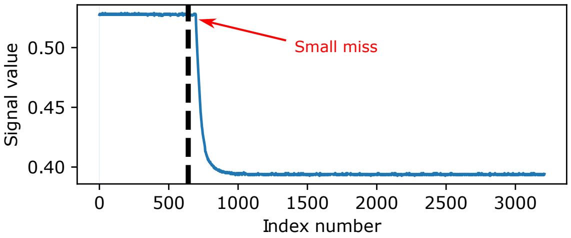


Figure 4.5: Inaccurate identification of an increase in load.

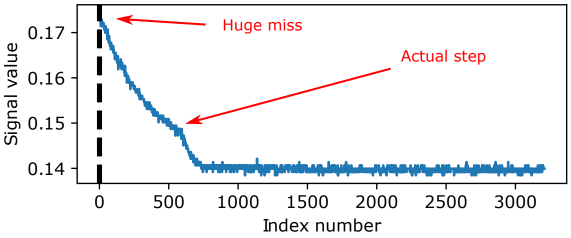


Figure 4.6: The step detection algorithm misses the current step, reverting to a standard fallback value.

Other features worth mentioning

- Automatic calibration algorithm finding the optimal sample time value for the ”sr run” function to achieve the desired mean sampling time.
- Optional plotting for debugging purposes.

4.1.5 Matlab data processing

A Matlab script is used to process the data, preparing it for further use. The data is imported from CSV files created by the Python test executor program. Then, the data is resampled, normalized, split into smaller pieces, and combined into data structures suitable for model estimation. The pseudo-code in Algorithm 4 gives a more detailed description of the data preprocessing.

Algorithm 4 Matlab -> Import and process data

```
Define the name and unit for the signals to import.
Define root filename, number of files, and the desired timestep.
Define the length of each file in seconds.
Define how many times to split each file.
Define normalization scaling factors.

% Import and resample data:
while Files to import do
    Import data from the file into a table.
    Convert to timetable format using the original unsynchronized timestamps.
    Resample and synchronize the data to the desired timestep.
    Append the data into alternative data structures.
end while

% Normalize and format data:
for all imported files do
    Normalize the data of the current file.
    for number of splits do
        Select the appropriate data and format it into alternative data structures.
    end for
end for
Save the data structures and normalization factors into a Matlab data file.
```

4.1.6 Results and conclusion

Results

This section presents the parameters used in the data gathering and processing resulting in two final datasets. Further, the resulting workspace coverage and actual sample time of the datasets are graphically presented.

- **Key parameters:**
  - Timestep:
    - Desired logging timestep: 250 [μs]
    - Desired synchronized data timestep: Typically 1-10[ms] (selectable in Matlab post-processing).
  - Input limitation parameters
    - Duty cycle AC:
      - Allow values: 0-100 %
      - Max stepsize when voltage: <70 kV=30%, 70<90 kV=3%, >90 kV=1.2%.
      - Normalization divisor: 100, original unit [%]
    - Duty cycle DC:
      - Allow values: 0-50 %
      - Max stepsize when voltage: <70 kV=10%, 70<90 kV=2%, >90 kV=0.8%.
      - Normalization divisor: 100, original unit [%]
  - Normalization
    - Field voltage:
      - Normalization divisor: 120 000, original unit [V].
    - Cascade/Total current:
      - Normalization divisor: 0.002, original unit [A]
    - Total resistive load:
      - Normalization divisor: 1000, original unit [MΩ]
    - Optional, more signals are available but are not used in this thesis. Further scaling parameters are available in the attached code.
- **Resulting datasets:**
  - *”No load stepping dataset”*

  In this dataset, the load variable is not excited while logging. Thus the error from the step detection algorithm mentioned in section 4.1.4 is omitted. Figures 4.7, 4.8, 4.9 and 4.10 present the datasets coverage of the workspace. Each point in the figures represents a unique set of input variables. Figure 4.11 presents the distribution time between each sample for the dataset.

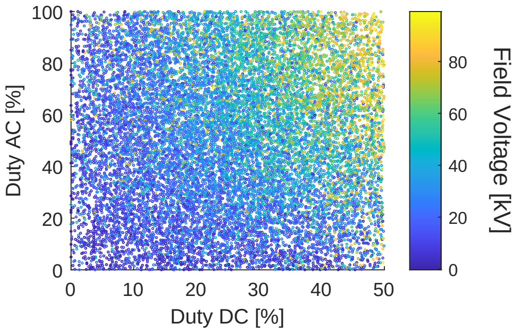


Figure 4.7: Distrubuiton of data for the ”no load stepping” dataset.

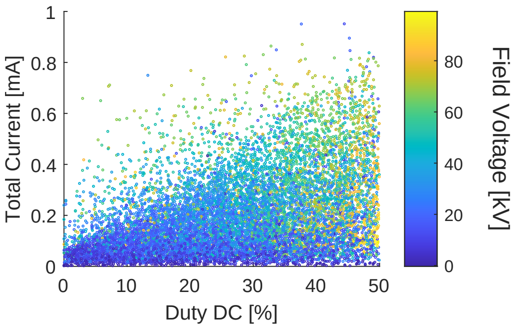


Figure 4.9: Distrubuiton of data for the ”no load stepping” dataset.

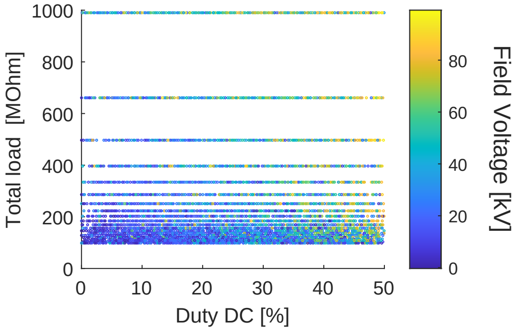


Figure 4.8: Distrubuiton of data for the ”no load stepping” dataset.

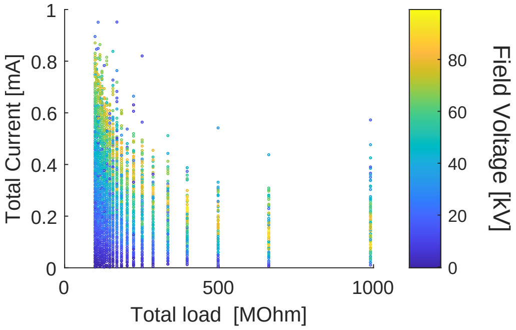


Figure 4.10: Distrubuiton of data for the ”no load stepping” dataset.

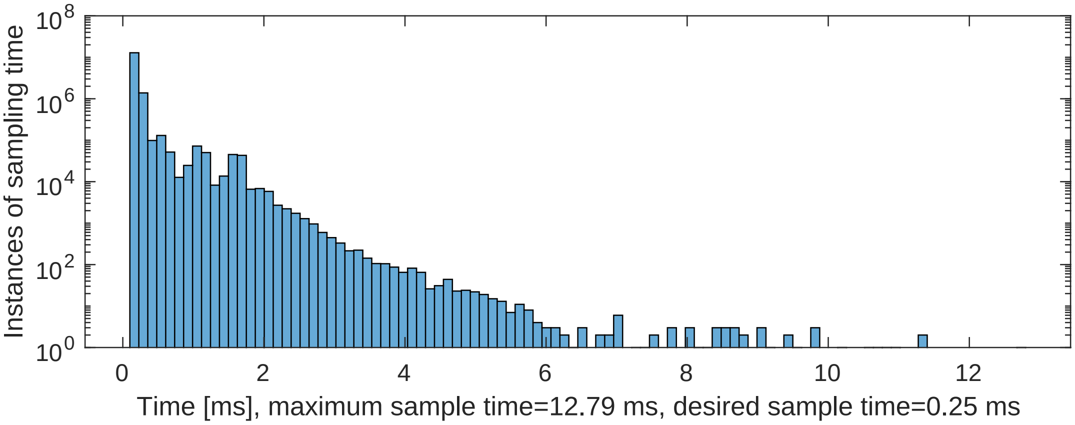


Figure 4.11: Histogram presenting the distribution of time between each sampling instance. Note that the Y-axis is logarithmic to emphasize the extreme values.

– *"Load stepping dataset"*

In this data set, the load variable is extensively excited while logging. The step detection algorithm from section 4.1 creates the load variable, adding error to the dataset. Figures 4.12, 4.13, 4.14 and 4.15 present the datasets coverage of the workspace. Each point in the figures represents a unique set of input variables. In figures 4.14 and 4.15, points appear in between the static load levels of the variable load; this is connected to the interpolation from downsampling the data (described in section 4.1.5). Figure 4.16 presents the distribution time between each sample for the dataset.

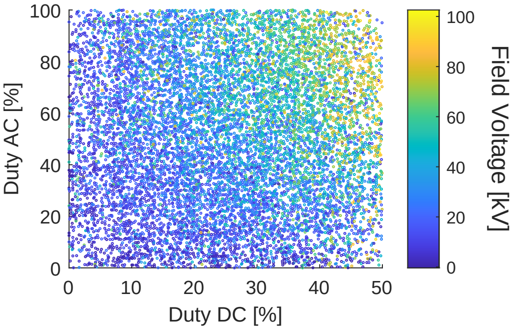


Figure 4.12: Distrubuiton of data for the load stepping dataset.

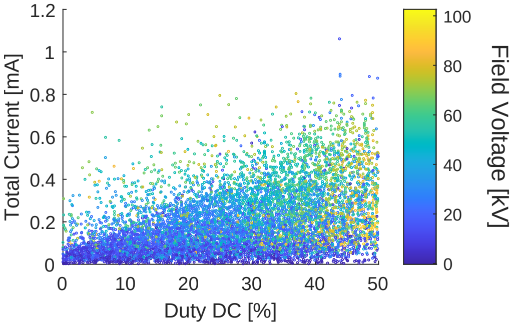


Figure 4.14: Distrubuiton of data for the load stepping dataset.

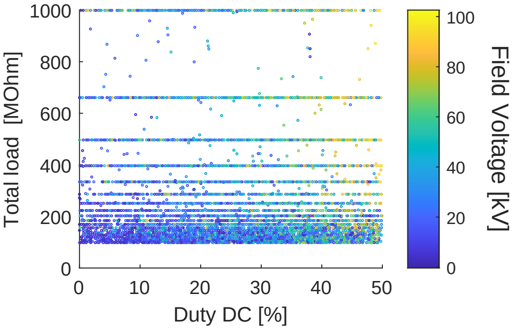


Figure 4.13: Distrubuiton of data for the load stepping dataset.

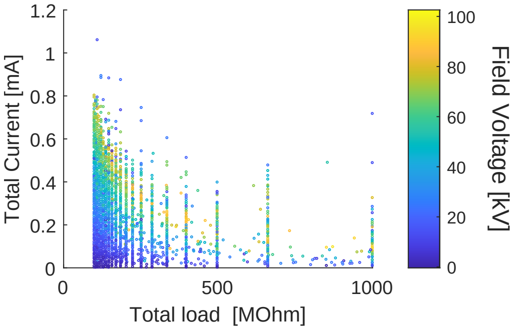


Figure 4.15: Distrubuiton of data for the load stepping dataset.

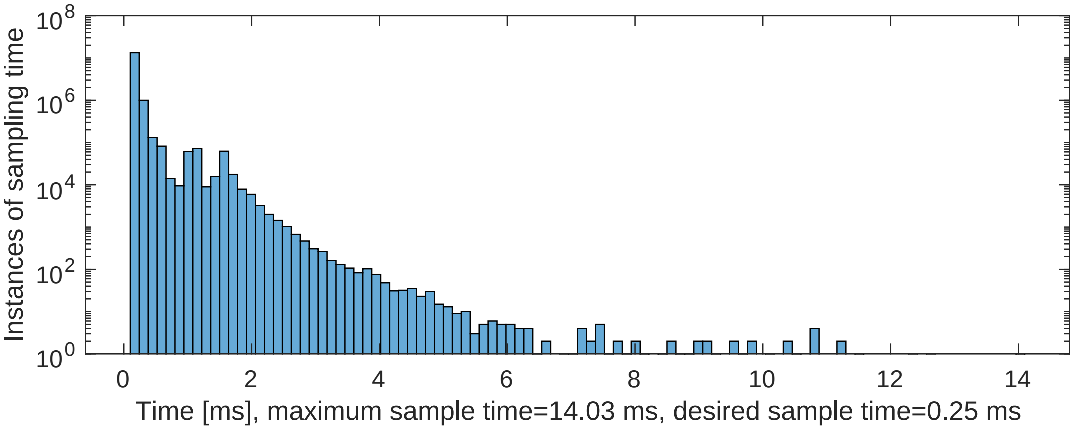


Figure 4.16: Histogram presenting the distribution of time between each sampling instance. Note that the Y-axis is logarithmic to emphasize the extreme values.

Conclusion

The results from Figure 4.11 and 4.16 illustrate the shortcomings of the SR logging function (section 4.1.3) and the CCPU platforms' real-time capabilities. These significant offsets from the desired sampling time introduce a substantial amount of error to the data. This error only is reduced by downsampling the data to exclude the information gaps.

Furthermore, the datasets without any load stepping are likely the best due to the limitations of the step detection algorithm.

Considering the data's validity, it should be noted that the data is collected from a test bench where a variable resistive load simulates the field current as a disturbance. The data does not encompass the dynamics for systems with another total capacitance value. The simplified cascade model in section 2.2.4 illustrates how the capacitance influences the system's dynamics. Thus, depending on how the data is utilized, it might only be valid for the test-bench conditions.

To summarize the conclusion, the data acquisition program generally works but has two critical shortcomings: the step detection algorithm and SR-logging sample time jitter. Furthermore, it should be noted that, in general, the data only represents the testbench conditions.

For comments on the sampling time issue, go to section 5.2.9.

4.2 Model estimation

This section describes the estimation of a model for simulating the HVC and Cascade using two different neural networks. As described in Figure 4.17 the model shall utilize the input vector $u(n)$ and the process disturbance vector $w(n)$ to estimate the output vector $y_{est}(n)$. The variables are defined by Equation 4.1.

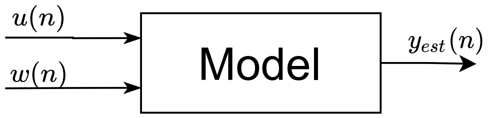


Figure 4.17: Illustration of a simulation model.

$$\begin{array}{lll} u(n) & - & \text{The two duty cycle inputs to the HVC} \\ y_{est}(n) & - & \text{The estimated output values: field voltage, output current, plus optional.} \\ w(n) & - & \text{The total resistive load connected to the cascade output.} \\ & & \text{This includes the internal load contributed by the measurement circuit.} \end{array} \tag{4.1}$$

4.2.1 LSTM-based model

The dynamics of the HVC and cascade can be expressed as a general nonlinear state space model, as illustrated in Figure 4.18. The essence is that the output directly depends on the current input and earlier output value. The LSTM[4] network structure is suitable because of its recurrent nature, internally using the last output, combined with the new input data, for calculating the next output. The LSTM excels at capturing long-term dependencies, which is suitable considering that some dynamics might be slow relative to the simulation timestep. [9]

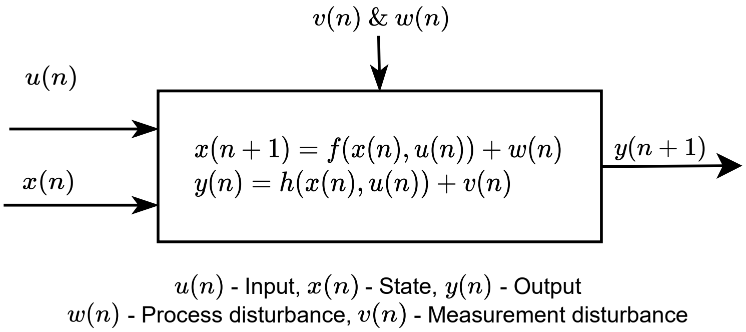

$v(n)$ & $w(n)$
$u(n)$
$x(n)$
$x(n+1) = f(x(n), u(n)) + w(n)$
$y(n) = h(x(n), u(n)) + v(n)$
$y(n+1)$
$u(n)$ - Input, $x(n)$ - State, $y(n)$ - Output
$w(n)$ - Process disturbance, $v(n)$ - Measurement disturbance

Figure 4.18: Illustration of a non-linear state space model.

[4] Long short-term memory

For this project, the neural network model is trained using Matlab's deep learning framework and proposed architecture for LSTM networks.[24]. Figure 4.19 presents the deep neural network structure used for the LTSM model.

4.2.2 LSTM results

The datasets from section 4.1.6 are utilized as the data for testing and training. As a standard, the test data is a random selection of 15% separated from the original datasets. The output model from each training sequence is the model with the lowest RMSE[5] on the test data. The training sequence typically overtrains, but the final output model will be the model with the best results on the testing data. The datasets are down-sampled to 1000 samples per second, i.e., a timestep of 1 ms.

The optimal network size

The optimal size of the neural network is the one with the lowest number of neurons still able to represent the data adequately.

Figure 4.20 presents the results of an experiment where the same dataset is used to train neural networks of increasing size. The networks are trained until they are overfitted on the training data, and then the output network with the best performance on the validation data is selected. The validation data is a 15% partition of the original dataset, which for this experiment is the "no load stepping dataset".

The results in Figure 4.20 indicate that six neurons for each internal layer of the deep neural network (Figure 4.19) is sufficient for representing the data.

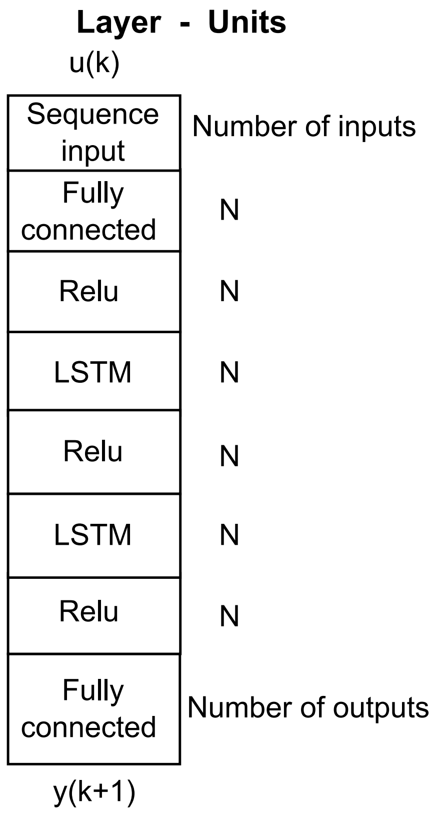


Figure 4.19: Deep neural network structure for the LSTM-based model consisting of fully connected perceptron layers, LSTM layers, and Relu activation function layers. N is the number of units/perceptron in each layer of the deep neural network.

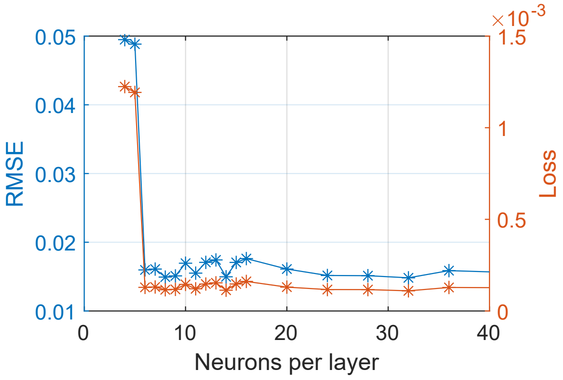


Figure 4.20: Loss and RMSE versus the number of neurons/perceptrons in each layer.

[5] Root-mean-squared error

Dataset comparison

The performance of models resulting from the different datasets is compared. Each dataset file is split into 24 smaller sequences of 2.5 seconds. The models are trained using the same training configuration and network size. A total of five training sequences are completed producing the following results:

- Mean results for the *"no load stepping dataset"*:
  - RMSE: 0.016
- Mean result for the *"load stepping dataset"*:
  - RMSE: 0.020

These results illustrate how the "no load stepping" dataset best fits the data. The extra error for the "load stepping" dataset is likely connected to inaccurate step detection of the load variable.

No initial condition

The LSTM layers of the model are internally dependent on their last output; when there have not been any outputs, it thus poorly reflects the real systems behavior. This could be a problem if the model is used for reinforcement learning. Figure 4.23 highlights how this makes the LSTM model perform worse in the initial phase of a simulation.

4.2.3 Nonlinear state-space neural network model

In late 2022 Matlab introduced a framework for estimating nonlinear state-space models using measured time-domain data[23]. The underlying model is a deep neural network of fully connected neuron layers as illustrated in Figure 4.21. The model is trained to predict the output of the actual system, given the input values and the current state. The subsequent output is estimated using the estimated output of the last iteration and the current input. This nonlinear state-space model is suitable for use both for prediction and simulation. The model is initialized simply by selecting an initial state.

Algorithm 5 describes a procedure created to train a model using Matlab's nlssest()[23] function. This algorithm was created to improve the training speed by training on a randomly selected small proportion of the data.

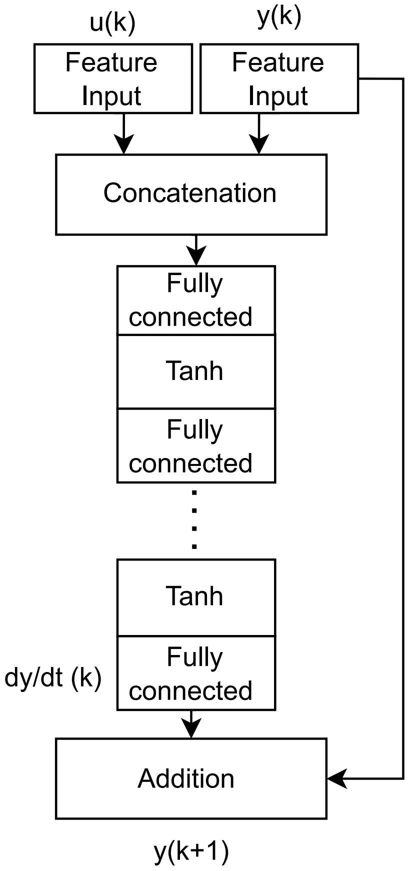


Figure 4.21: Structure of the neural network used in Matlabs "Non-linear state-space models"

Algorithm 5 Matlab -> Train neural network for non-linear state-space model

```
Import normalized data formatted as a cell array of experiments.
Define names and units for the input and outputs.
Define the nonlinear state space neural network model with input, output, timestep,
and network size.
Define the training options for the neural network.
while Training do
    Train/retrain the model with a randomly selected subset of the data using a
    pre-defined experiment for validation.
    if Validation error < goal OR training manually stopped OR timeout then
        stop training
    else
        Decrease the initial learning rate and increase the number of epochs for the
        next training sequence.
    end if
end while
Save the best-trained model.
```

Training issues

Matlab's nlssest()[23] training function is generally slow and does not utilize GPU resources. The slow training is connected to manually inserting the last estimated output back into the network input. For the LSTM case, this is done more efficiently internally. This has been especially evident when attempting to train a model with a resolution down toward 1 ms. Then for this case, with the available hardware resources, and datasets, the training algorithm struggles to converge to an acceptable accuracy.

4.2.4 Validation

This section presents the validation and consequent analysis of the two proposed model structures. The validation data is a separate dataset composed of data not used in training. The validation data cover the entire workspace similarly to the training datasets presented in section 4.1.6. The models are trained using the *"no load stepping"* dataset with a 10 ms timestep.

The performance of the models is measured using the NRMSE[6] value transformed into a percentage where 100% is a perfect fit, 0 % is a model estimating the output to be a straight line equal to the mean of the reference output, and a negative value worse than the mean of the reference output. The calculation of the NRMSE value is presented in Equation 4.2.

$$NRMSE = 100 \cdot \left(1 - \frac{||y - \hat{y}||}{||y - mean(y)||}\right) \tag{4.2}$$

Where y is the true/reference output and $\hat{y}$ is the estimated output.

Overall results

The overall results are presented in the table below. Additionally, the results from a 20-second sequence are presented in Figure 4.22.

Overall validation results		
Model	Field voltage	Total current
lstm	92.91 %	83.94 %
nlss	93.11 %	65.14 %

These results indicate that both models are reasonably accurate for modeling the field voltage output of the system. However, the models both struggle to model the total current output accurately.

[6] Normalized root-mean-squared error

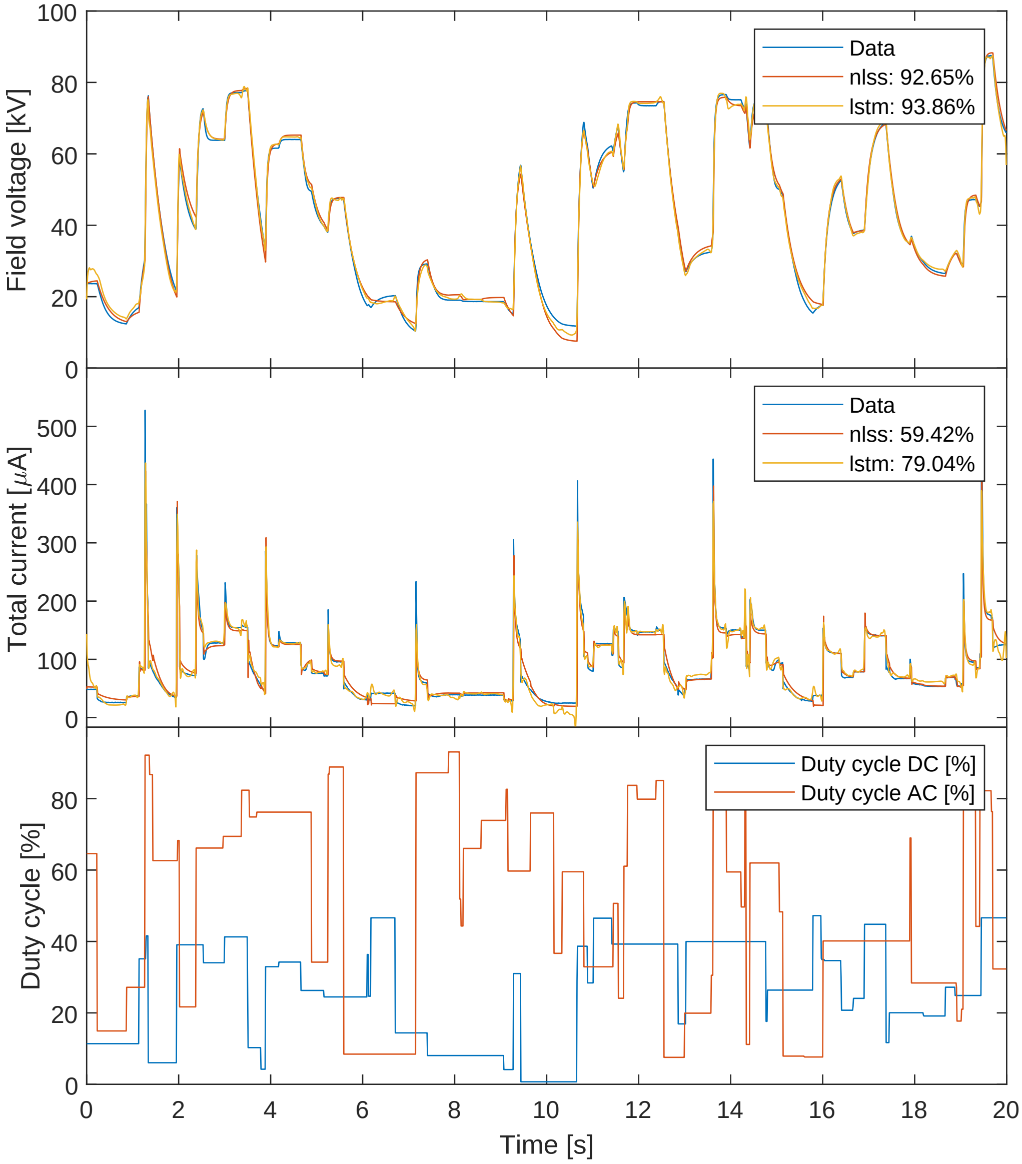


Figure 4.22: Comparison of the LSTM- and nonlinear state-space neural network models to data collected from the real world system.

Analysis of models

Figure 4.23 present the results from a 1-second sequence where behavior contributing to the error of the models is highlighted.

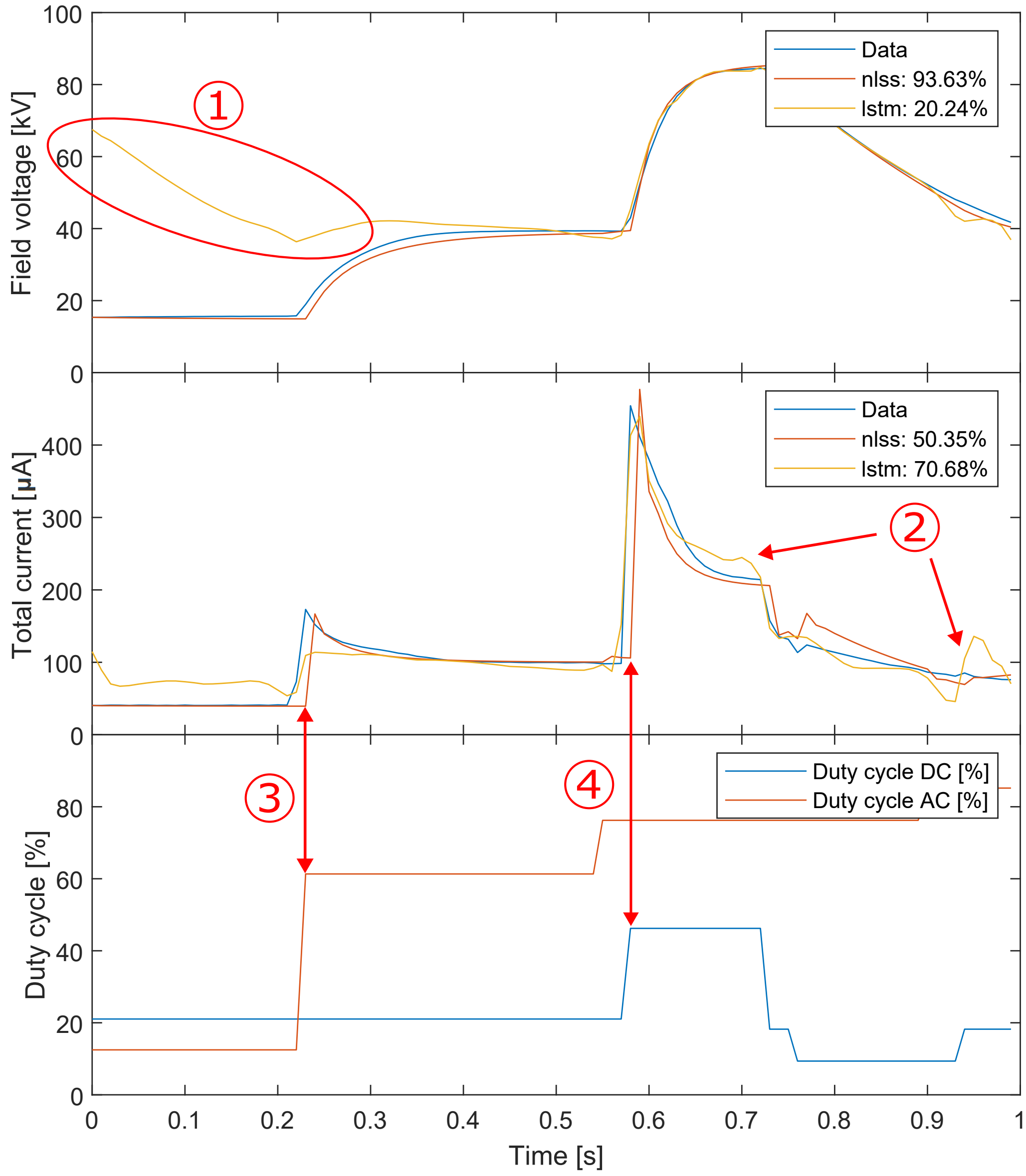


Figure 4.23: Comparison of the LSTM- and nonlinear state-space neural network models to data collected from the real world system.

The behavior highlighted in Figure 4.23:

1. Error caused by the lack of an initial condition for the LSTM model.
2. Examples of uncharacteristic behavior likely to be connected to a long-term memory effect in the LSTM model. The models seem to anticipate change before the input is excited, suggesting that the timing of input excitation performed by the data acquisition program is not as random as it should be. Even if the behavior is not linked to any learned time dependency, the behavior signal is uncharacteristic of the actual system and, thus, exceptionally bad.

3. & 4. Point out how downsampling/interpolation and/or logging errors cause the response of the input excitation to appear before the actual input excitation. The fact that the LTSM model, unlike the nonlinear state-space model, is able to predict this reiterates what is highlighted by point 2.

Another observation that can be made from Figure 4.23 is that the dynamics of the total current variable are much faster than the relatively slow change of the field voltage variable. Thus a 10 ms timestep is insufficient for representing a fast change in the total current not directly connected to the slower field voltage. This is likely the main reason behind the worse simulation results for the total current variable.

4.2.5 Conclusion

The trained LSTM model appears overtrained, utilizing long-term dependencies only found in data collected from the test bench. This, in combination with the lack of a standard initialization option, make the LSTM model unsuitable for use as a simulator.

On the other hand, the neural network model on a strictly nonlinear state-space model format gives better results. The error observed for this model does not appear uncharacteristic and is likely connected to the data. Thus this model is suitable for further use.

Due to the low model resolution, i.e., the 10 ms timestep, these results are only valid as proof of concept. A timestep of 1 ms or less, as is the timestep of the existing controller, is a more appropriate value for future implementations. The timestep of 10ms was selected to speed up the training process.

4.3 Reinforcment learning controller

This section describes the training and performance of a neural network controller trained on a model using reinforcement learning. The controller setup is illustrated by Figure 4.24.

Matlab's Reinforcement Learning Toolbox[26], with its documentation and references to scientific literature, is the basis for the reinforcement learning experiment performed in this section. A fundamental introduction to reinforcement learning is given by [19].

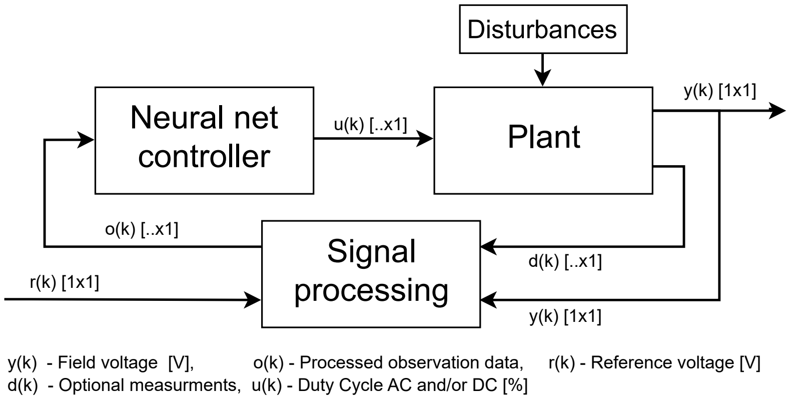

Disturbances
Neural net controller
u(k) [..x1]
Plant
y(k) [1x1]
o(k) [..x1]
Signal processing
d(k) [..x1]
r(k) [1x1]
y(k) [1x1]
y(k) - Field voltage [V],
d(k) - Optional measurments,
o(k) - Processed observation data,
u(k) - Duty Cycle AC and/or DC [%]
r(k) - Reference voltage [V]

Figure 4.24: Illustration of the neural network controller utilizing observation data to generate input values for the plant.

4.3.1 Setup

Training environment

The controller training/test environment consists of a few basic components:

- Plant model
- Reset function:
  - For each training run, the initial setpoint, initial load/disturbance, and initial state are updated with random values within the valid range of the model.
- Training challenge:
  - Addition of random steps and sinusoid signals in the reference and disturbance for the controller to learn to follow the reference and reject the disturbance. The setup of the reference and disturbance in the training environment will affect the trained controllers' disturbance rejection and reference following performance.
- Stop simulation signal:
  - Stop the simulation if the model's valid range is broken, the accumulated negative reward is too large, or the simulation time has been reached.

Reward function

The reward function is split into a cost- and a penalty part. The cost part is similar to that of LQR[7]/MPC[8] controllers, fine-tuning the neural network to reflect such controllers' performance. Additionally, a cost dependent on the discrete integration of the error with saturation limits is added to reduce static error. The cost of error change The penalty function is more of a rough framework indicating exceptionally bad behavior, speeding up convergence in training. Algorithm 6 present the general version of the reward function.

The recommendations given by Matlab[22] inspired the development of the reward function.

Algorithm 6 Reward function:

```
reward_function($y_k, y_{k-1}, y_{ref\_k}, u_k, u_{k-1}$,$int_e$)
    % The $Q$, $P$, $e_{lim1/2}$ and $y_{min/max}$ cost/penalty constants/matrices are predefined

    $e_k = y_{ref\ k} - y_k$
    $dy_k = y_{k-1} - y_k$
    $du_k = u_{k-1} - u_k$
    $Cost = e_k \cdot Q_e \cdot e_k + dy_k \cdot Q_{dy} \cdot dy_k + u_k^T \cdot Q_{du} \cdot du_k + int_e \cdot Q_{int} \cdot int_e$
    $Penalty\_e = (e_k > e_{lim1}) \cdot P_{e1} \cdot (e_k > e_{lim1})^2 + (e_k > e_{lim2}) \cdot P_{e2} \cdot (e_k - e_{lim2})^2$
    $Penalty\_y = (y_{min} > y_k) \cdot (y_{min} - y_k)^2 \cdot P_y + (y_k > y_{max}) \cdot (y_k - y_{max})^2 \cdot P_y$
    $Reward = -(Cost + Penalty\_e + Penalty\_y)$
Return $Reward$
```

Observations for the controller

The controller should be valid and robust for different setups and hardware variations. Thus, the reinforcement learning controller should not utilize signals inherently changing between system instances.

For instance, the current measurements are connected to the system capacitance. However, it is also connected to internal loss, and the system capacitance might probably be estimated using the derivative of the output field voltage and the current, arguing that the current measurement should be included.

However, as the accuracy of the current estimates of the plant model was low (section 4.2.4), the standard observations sent to the controller are as follows:

1. The field voltage

[7] Linear quadratic regulator

[8] Model predictive controller

2. The derivative of the field voltage (optional LPF[9])
3. The reference voltage
4. The derivative of the reference voltage (optional LPF)
5. The error between reference and field voltage
6. The integral of the error with saturation limits
7. Optional (not included) - The total current

Reinforcement learning algorithms and training configuration

The DDPG[10][21] and twin-delayed DDPG (TD3)[27] algorithm is selected per Matlab's guidance document for the selection of reinforcement learning algorithms [25]. Specific details on the training configurations used are available in the Matlab code in the appendix. The adjustment and configuration of the parameters are based upon Matlab's documentation, courses, and examples[26] and an informed trial and error process.

4.3.2 Results

Due to limited computational resources, and errors in the data, the reinforcement learning environment utilizes a low-resolution (10 ms timestep) model of the painting system. The specific training parameters and reward weights can be found in the appended Matlab reinforcement learning files and trained controllers files.

Figures 4.25 and 4.26 present a performance illustration and the training progress using the TD3 algorithm for a controller with two actuator variables. Similarly, Figure 4.27 presents the performance illustration of a controller with one actuator variable trained using the DDPG algorithm.

The results from these figures are promising; however, there is room for improvement in the training configuration and the reward function weights. From Figure 4.26, it is visible that the training algorithm was converging very slowly and still improving when stopped. The training lasted approximately 15 hours, making it a slow process. Figures 4.25 and 4.27 prove that the controller works to a degree, but exactly how well is unspecified. To accurately determine the controllers' performance, quantitative indicators are needed. Thus the results only represent a proof of concept for the reinforcement learning solution.

[9] Low pass filter

[10] Deep deterministic policy gradient

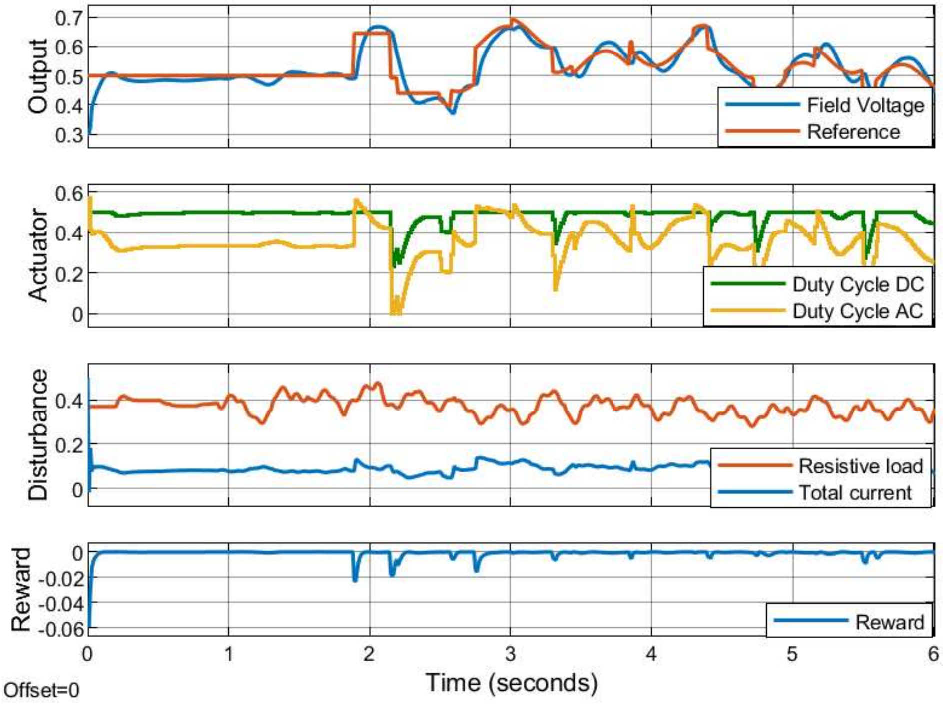


Figure 4.25: Simple performance illustration of a two-input controller trained using the TD3 algorithm. The training progress of this controller is presented in Figure 4.26.

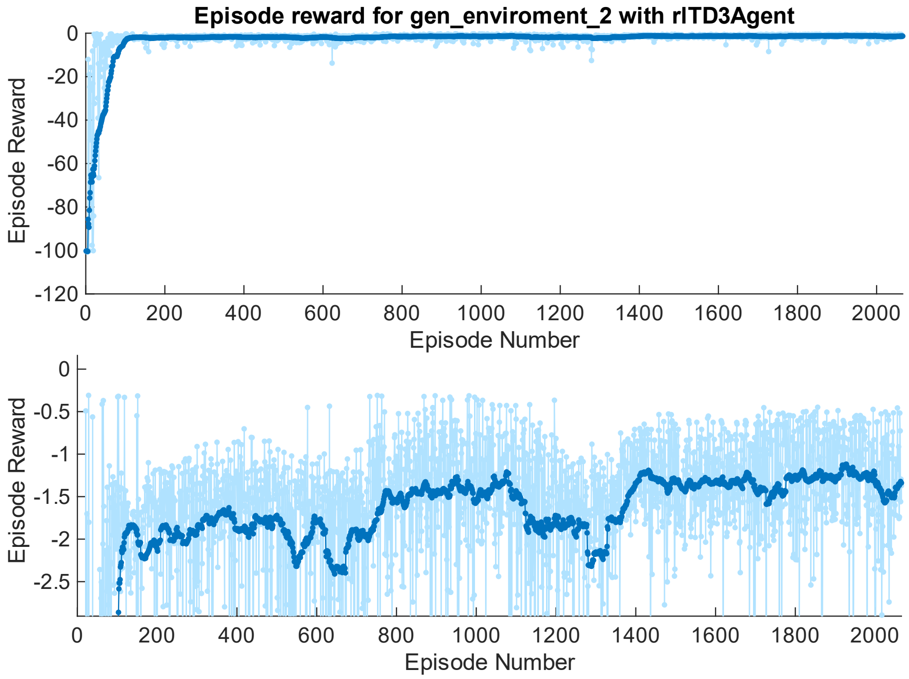


Figure 4.26: Illustration of the training progress for the controller using a TD3 algorithm. The performance of the controller when the training was stopped is displayed in Figure 4.25

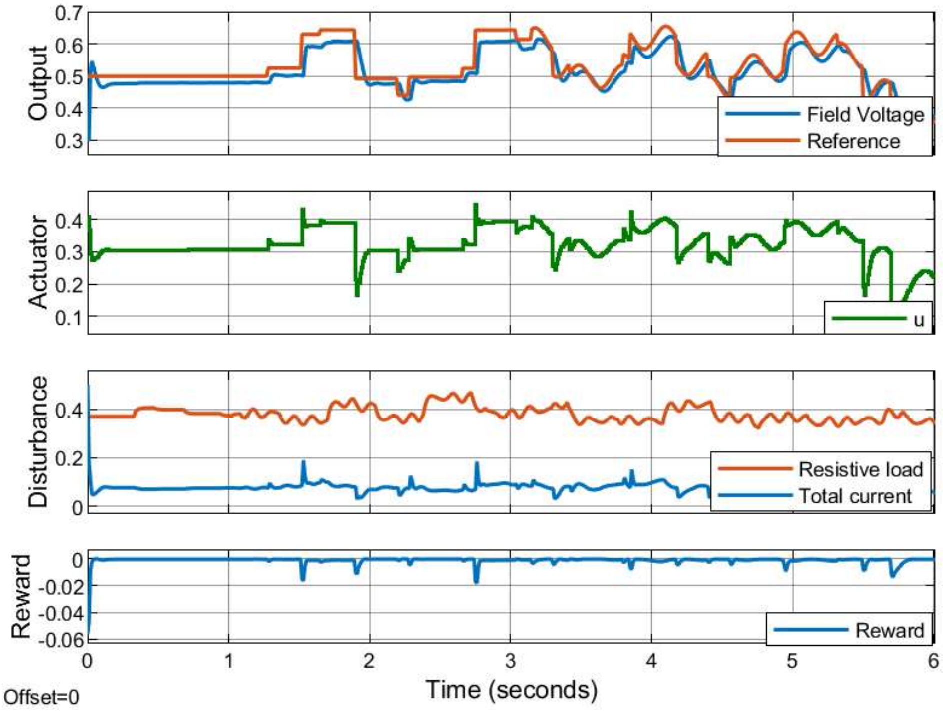


Figure 4.27: Simple performance illustration of a one-input controller trained using the DDPG algorithm.

4.3.3 Conclusion

The reinforced learning approach gives promising results; however, further optimization of the reward function weights and the training parameters must be performed before the controller is ready for deployment. Additionally, a controller trained on a higher-resolution model as well as quantitative performance indicators is necessary for further development.

The result presents what is likely a non-optimal controller and highlights a primary challenge of the reinforcement learning approach, the time-consuming optimization and training process.

Implementation considerations

The trained reinforcement learning controller consists of a relatively small neural network and signal processing part feeding the neural network with observation data. For this observation data to be uniform across different instances of the electrostatic painting system, the signal processing part should accept configuration options like applicator type and measurement circuit resistor and calibration values.

The most suitable hardware for deploying the controller is the FPGA on the FCPU platform. However, depending on the final complexity of the controller, it should also be possible to implement the neural network on the older Intel FPGA on the CCPU.

Chapter 5

Reflections and experiences

This chapter contains reflection and lessons learned through the work with the master thesis. Furthermore, suggestions for further work and alternative solutions not included in the main chapters are presented. The chapter is split into the following sections:

- The workflow and direction choices
- Alternative solutions, further work, and reflections
- Summary of further work

5.1 The workflow and direction choices

In the project's initial phase, the focus was on the following three elements.

- Talk with ABB employees and gather info on challenges related to the high voltage system.
- Study electrostatic gas discharge and corona currents and the painting process.
  - It was concluded that detection and inhibition of sparks were impossible, given the available hardware. The limiting factors are the filtered measurements with a cut-off frequency of about 1 kHz, a system discharge time of hundreds of milliseconds, and spark transition time in the range of microseconds[20]. It became clear that the most feasible strategy for safety was to reduce the risk by following the safety standards, i.e., depending on an automatic fire extinguishing system. It also became clear that the system could be improved by not only using global absolute maximum safety limits. In addition, to the

global maximum limits, a current reference could be introduced, enabling earlier action and reduced reaction times. This suggestion has been implemented in section 3.2.4.

  - Estimation of the field currents to the field voltage was raised as a key point that could solve some of the difficulties ABB experienced on the existing controller (section 2.3).

- Study the existing ABB controller and hardware to map out the restrictions and opportunities given by the hardware.
  - To study and understand ABB's hardware and software, the work resulting in the data acquisition program was started (section 4.1). It was deemed an effort worth the investment as a high-resolution data-gathering system did not exist. Additionally, the work would give me experience working with ABB's software easing the potential implementation of a new controller algorithm at a later stage.
  - A conclusion was that the DSP would not be sufficient for more advanced controllers but that the new FCPU platform offered better performance.
  - The existence of FPGAs gave rise to a focus on developing a neural network controller to be implemented directly in the FPGA.

The project's second phase focused on developing the data acquisition program and training a system model. The progression on this front slowed considerably when experimentation with model training and optimization began, initiating the third phase.

In the third phase, the focus was to estimate and utilize the relation between the corona current and the applied field voltage. Even though the initial study indicated that a converging estimation was impossible, the search for solutions continued. A series of experiments trying out different algorithms and approaches were made, some of which are presented in chapter 3. The final product of this work was the current limiting controller.

In the fourth phase, the focus return to the model estimation and a neural network controller. Experimentation and research on training parameters were time-consuming and did not provide results ready for direct implementation.

The project has been influenced by not having a clearly defined solution to implement and test. Instead, the direction of the thesis has appeared through a research process. Especially reinforcement learning and model estimation has been time-consuming endeavors. By having a clearly defined path from the start and focusing on a narrower field of study, more applicable results and conclusions might have been achieved.

5.2 Alternative solutions, further work, and reflections

This section contains a series of reflections, alternative solutions, and suggestions for further work on existing solutions.

5.2.1 Measurements and calibration

As mentioned in section 2.2.2, a considerable error is introduced to the measurements. This section elaborates on some possible approaches to improve this.

Initial calibration

Component variations introduce measurement uncertainty that might be reduced by an initial calibration, removing the offset error. Initial calibration is crucial for high-voltage components, like the BR resistor with ±5% initial variation. Calibrating the giga-ohm-sized BR resistor requires specialized instruments that measure giga-ohm values. Nevertheless, noise and drift still leave measurement uncertainty after calibration.

Internal calibration

In addition to the initial calibration, online calibration/estimation can further improve the measurement accuracy. For instance, the system's actual output can be predicted by utilizing a cascade model. The final estimate can be updated depending on the ratio of the uncertainty associated with the model and the uncertainty inherent in the measurements.

An alternative, more straightforward approach would involve defining a test setup with a predetermined input to the cascade, resulting in a known static output. Then, it is possible to recalibrate by comparing the measured output value with the expected output value. This simpler scheme may be suitable for specific applications where an extensive model-based approach may not be feasible or appropriate.

5.2.2 Improved field current parameter estimation

As mentioned in section 2.1.4, there is most likely some connection between U_0 and K. Collecting large amounts of data while changing the underlying variables and then analyzing that data for a connection between U_0 and K, could be a future project. If a relationship between these two parameters were to be determined, it might be possible to estimate better the relationship between the field voltage and field current. However, considering all the variables influencing the system, this is a very large task.

5.2.3 Reportedly a too low resolution on the duty cycle DC

If this is the case, it could be improved by using a higher resolution timer for the PWM[1] signal. The DSP creates the current PWM. An option is to implement a timer on the FPGA, replacing the one on the DSP. This is probably the simplest approach if inspecting the problem as an isolated task.

An alternative approach is to make the control system a two-input-one-output system, which has been attempted using the reinforcement learning approach.

5.2.4 Relflections on the data-driven approach

The selection of the purely data-driven approach was primarily motivated by the approach's potential to solve any controller problem, creating the optimal controller valid for the entire workspace.

Another motivation for this approach was that it allowed for the easy development of a multiple-input controller, utilizing both the AC and DC duty cycle inputs from section 2.2.3. Which again had the potential for solving the reportedly low resolution on the input signals. Another contributing factor was the fact that the available hardware, the DSP on the CCPU(section 2.2.3) would likely be insufficient for more advanced controller concepts, whereas the FPGA on the CCPU could probably be used for implementing a compact neural network controller.

The data-driven approach introduced new requirements and challenges, all extensive tasks.

- Collecting high-resolution valid data.
- Finding and evaluating suitable approaches for estimating a model.
- Finding a suitable approach for training a controller.

The main experiences drawn with respect to the data-driven approach are:

- The importance of reducing error in the data as any error is passed on.
- The need for lots of computational power when handling large datasets, performing estimation, and training neural networks.
- The configuration and selection of the reinforcement learning setup and neural network models tend to be a time-consuming pseudo-scientific trial-and-error process.

[1] Pulse-width modulation

- The controller must be created robust to handle errors in the training environment model and, when deployed, variations in the controlled systems.

Testing this approach has made it evident that the data-driven approach is not an easy solution for all problems but a possible approach with its fair share of challenges. Given this perspective, the more traditional analytical model-based approaches might appear as the preferred way to go.

5.2.5 Note on AI in safety-related systems

[17] discuss using AI[2] for safety-related closed-loop systems. They explore a variety of possible uses for AI in control systems and what implications it may have for safety-critical systems. For the AI solution presented for the voltage controller, it is suggested to use a supervision framework to ensure the safety-related states. In our case, this is reflected in the framework described for the current limiting controller. In other words, the AI part would not be directly involved in ensuring a safe state. But the safety system would manually discharge the high voltage.

5.2.6 Better data-driven modeling and control approach

The Sparse Identification of Nonlinear Dynamics algorithm, or SINDy[5], has advantages over the neural network training algorithms used in this thesis. This method gives more interpretable results by finding the governing terms and relations in the system, producing equations that can be compared to an analytical model of the system. Additionally, it has been shown that SINDy typically requires less data and is more computationally effective. The data-driven approach utilized in this thesis appears inferior to the more state of art SINDy approach. Thus for future work, it is recommended to check out this line of research.

[2] Artificial intelligence

5.2.7 The traditional model-based voltage controller

In this thesis, the voltage controller work focuses on the data-driven approach. However, the more traditional approach would be analytical modeling and controller design. This section suggests an alternative approach to the one chosen in this thesis. The process could look like this:

1. Set up theoretical models for the buck converter, ac-generator with transformer, and Cocrfot-Walton voltage multiplier (section 2.2.4).
    - Find the expected numerical values of the components. (The exact data for the cascade has been unavailable as it is not ABB property.)
2. Select controller structure. Here is a proposal:
    - An internal PID[3] controller with feedforward from reference and disturbance for the buck converter. Then an outer PI/PID controller for the output field voltage.
3. Analyse stability, robustness, and performance using traditional control theory methods.
    - Evaluate the improvements if modifying ABB's hardware is allowed. The main limitations are heavy low-pass analog filters on the measurements and possibly low resolution on the duty cycle output.
    - Evaluate computational cost and impact of the controller sampling speed. Compare with the available computational power.

Other options for a controller are:

- LQR/MPC with integral action.
- Indirect or direct model reference adaptive control.

However, due to a high system capacitance and a corresponding ”slow” response on the field voltage, more advanced control strategies that a basic PID controller might not increase performance. The traditional PI or PID controllers are a good choice, primarily because they are well-known, have simple algorithms, and are relatively straightforward to configure and reconfigure. Even though other controllers might perform better, these arguments raise the threshold for selecting a more untraditional approach.

[3] Proportional integral derivative

5.2.8 The cascade discharge problem

Motivation

The rate at which it is possible to discharge the cascade is essential for the painting systems' performance. For instance, when cleaning the paint applicator in series production, the production line halts for a few seconds waiting for the paint applicator to discharge. Furthermore, being able to discharge the painting system rapidly would increase the performance results of the controller concerning safety norms. Which again would allow for a higher applicator voltage, i.e., increasing the systems transfer efficiency.

Thus with this motivation in mind, a simple proposal for a cascade discharge circuit is proposed. This proposal should be the subject of further study to determine if the solution is feasible.

Another aspect is that the components realizing the proposed circuit and its development might be relatively expensive. Thus a closer analysis of the potential savings weighted against the cost should be performed.

Cascade discharge circuit

Figure 5.1 describe a proposal for a circuit discharging the cascade. Note that the type of transistors should be the depletion mode type, "normally on" transistors.

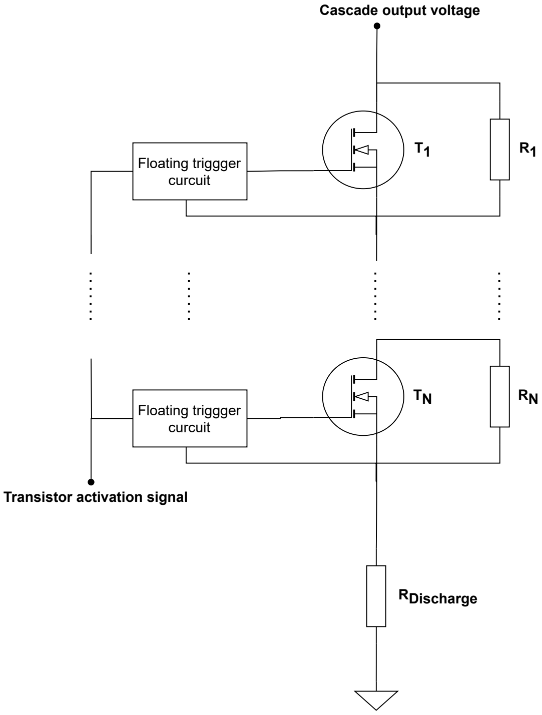

Cascade output voltage
Floating triggger curcuit
T1
R1
Floating triggger curcuit
TN
RN
Transistor activation signal
RDischarge

Figure 5.1: Simplified schematic of the proposed cascade discharge circuit. Note that depletion mode transistors are likely the best solution, even though the figure depicts the enhancement type.

The number of blocks, N, divides the maximum applied voltage from the cascade and sets the circuit's components requirements. The relations described in Equation 5.1 describe the requirements and performance of the proposed circuit.

N - Number of blocks in the circuit, C - Cascade and system capasitace
$\tau_{Discharge}$ - Discharge time constant, $V_{O\,max}$ - Cascade output voltage
$V_{block\,max}$ - Maximum voltage expected to be applied to each block.
$I_{off\,max}$ - The maximum current flowing through the circuit in the off state.
$I_{on\,max}$ - The maximum current flowing through the circuit in the on state

$$\begin{aligned}
I_{off\,max} &= \frac{V_{O\,max}}{R_{off}} = \frac{V_{O\,max}}{\Sigma_{i=1}^{N} R_i + R_{Discharge}},\ where\ R_{Discharge} << \Sigma_{i=1}^{N} R_i.\\
I_{on\,max} &= \frac{V_{O\,max}}{R_{on}} = \frac{V_{O\,max}}{\Sigma_{i=1}^{N} R_{DS\,on\,i} + R_{Discharge}},\ where\ \Sigma_{i=1}^{N} R_{DS\,on\,i} << R_{Discharge}.\\
\tau_{Discharge} &= \frac{1}{R_{off}\cdot C}, \qquad V_{block\,max} \approx \frac{V_{O\,max}}{N}
\end{aligned} \tag{5.1}$$

5.2.9 The sampling time jitter problem

The sampling time jitter problem (section 4.1.6) of the SR logging function was only discovered in the last days before the thesis deadline. Thus, no fix was applied.

However, early on, sampling time jitter was confirmed not to be a problem. An experiment was performed by sending log commands through the IPS command console. In contrast, the data acquisition program sends the commands through the ICI-server interface (API[4] for IPS). The ICI server does not run on the same thread as the command console. Thus the commands sent through the ICI server are not prioritized by the operating system, resulting in the observed jitter on the sampling time. The fix for this is simply adjusting the thread priority of the ICI server.

This was discovered and considered early on, but somehow due to human error, the problem was wrongly assumed fixed and not sufficiently confirmed to be working. This oversight was only discovered when placeholder plots were updated in the last leg of the thesis.

A real effort has been put into training a model with 1 ms a timestep using the nonlinear state space neural network model. However, it might be that the error introduced by the sampling time jitter (Figure 4.11) was a primary factor reducing the convergence rate of the training.

5.2.10 Alternative IPS datalogger

As mentioned in section 4.1.3, the standard IPS is only able to log signals with a timestep of 16ms. To circumvent this, a few options were studied and attempted. In addition to the option used in the solution, another alternative solution was developed. Figure 5.2 illustrate how this solution utilizes a dual port memory on the FPGA for storing the data.

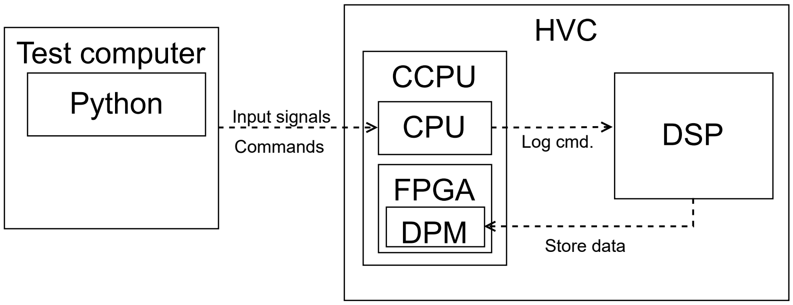


Figure 5.2: Illustration of logging program setup when data is stored directly in some dual port memory on the FPGA.

[4] Application programming interface

The limiting factor for this solution is the memory size of the DPM, which is approximately 10kB or 5k data samples for the CCPU. This solution allowed for both synchronized and fast sampling. When logging is initiated, the results are read via customized IPS commands, slicing the entire data pack into smaller pieces, as IPS cannot transmit more than 1460 bytes per transmission. However, when logging all the relevant signals, the maximum duration of the logged sequence becomes so short that each test is limited to a single step response.

As these limitations became apparent, the other option described in section 4.1.3 was selected as the most appropriate solution as this allows for logging longer data sequences.

The two solutions have a partly supplementary set of pros and cons. The pros for this alternative solution are a relatively small jitter and synchronized sampling intervals, and the cons are the available memory size. Vise versa for the main solution, the pros are the available memory, and its cons are the relatively large jitter and variations in the logging interval.

5.2.11 Better actuator utilization

As mentioned in section 4.1.4, the range of input values it is safe to use is limited by the maximum output voltage of the cascade. The solution presented utilized for data collection is by no means optimal. Thus, a strategy to get more data coverage should be implemented to improve the workspace of the model and the resulting reinforcement learning controller.

5.2.12 Nonlinear controller performance test

In order to gain interpretable performance data of a reinforcement learning controller, a validation framework using intuitive indicators should be created. This framework should calculate performance parameters covering the entire workspace, producing a clear image of the controllers' capabilities. Suitable performance indicators for reference following and disturbance rejection might be the resulting cut-off frequencies and corresponding phases when exiting the controller with an increasing frequency reference and disturbance signal. Additionally, a test measuring the robustness of the controller to a plant deviating from the training environment should be implemented.

5.2.13 Note for the direct charge paint applicator

In section 2.1.1 it is stated that the high voltage is applied to the atomizer bell cup. This is according to ABB's documentation [3], but this is not the case. The high voltage is applied to the paint prior to the atomizer.

5.3 Summary of further work

This section contains a summary of suggestions for further work related to the solutions presented in this thesis and other suggestions related to ABB's electrostatic painting system.

- Train and deploy the neural network controller:
  - Fix the sampling time jitter issue of the SR logging function.
  - Train a higher resolution model, i.e., smaller timesteps.
  - Configure the reward function with appropriate weights.
  - Train the controller.
  - Create a standard test framework for evaluating/validating the controller's performance, robustness, and stability.
  - Deploy the controller on an FPGA using Matlab/Xilinx/Intel tools for neural network deployment.
- Real training environment:
  - Connect the training environment directly to the SoC on the new FCPU board for real-time control. This eliminates the in-between model error and drastically enhances the quality of the training/testing environment.
- Deploy and test the current limiting controller:
  - Integrate the current limiting controller in series with the existing PI voltage controller using the structure presented in Figure 2.18.
  - Do a series of tests comparing the performance to the old controller structure. Performance criteria should include stability and robustness indicators like avoiding unnecessary shutdowns, disturbance rejection properties, and overcurrent shutdown response time.
  - Based on these findings, consider further use of the algorithm.
- Remove the hardware limitations:
  - Increase resolution of the duty cycle controlling the DC-DC buck converter.
  - Increase the cutoff frequency of the analog filter for the measurements.
  - Implement a calibration routine for the cascade measurement circuit.
- Develop and deploy a traditional model-based controller for the field voltage:
  - Follow recommendations in section 5.2.7.
- Evaluate and develop a solution for rapidly discharging the high-voltage system. A possible solution is described in section 5.2.8.
- Create an external monitoring program that classifies wear and tear, indicates the need for cleansing the paint applicator, and raises alarms on unexpected behavior.
- Investigate the characteristics of the electrostatic gas discharge for relations improving the parameter estimation of the current limiting controller (section 5.2.2).
- Use the SINDy[5] approach for the data-driven control and modeling (section 5.2.6).

5.3.1 Recomendation for ABB

This is a recommendation for ABB's direction regarding the electrostatic painting system. Emphasis has been placed on low-risk solutions.

1. Deploying and testing the current limiting controller is a relatively small task that might introduce rather significant improvements to the system with regard to adherence to safety standards, the controllers' robustness and stability, disturbance rejection, and a more intuitive configuration procedure.
2. Consider the need for better performance for the voltage controller. Implement the proposed hardware changes and deploy a simple model-based PID controller if necessary.
3. If a priority for the company is to develop a painting application with very high transfer efficiency, i.e., using a higher voltage and moving the paint applicator closer to the work object, the cascade discharge unit solves this problem. By drastically decreasing the system's discharge time, the safety standards allow more liberal safety and control parameters.
4. Apply ABB's data logging program Parati as a standard solution for monitoring and reporting on long-term behavior for the electrostatic paint application.

Chapter 6

Summary

This chapter sums up the central results presented in the thesis.

The thesis presents a detailed overview of the theory behind electrostatic painting and ABB's painting system. ABB's high voltage controller is analyzed, and a new control structure is proposed. The new structure includes a current limiting controller related to safety, in series, with a field voltage controller for generating the high voltage output.

For the development of the current limiting controller, an attempt is made to estimate the relation between the electrostatic field voltage and the resulting field current (corona current). However, the estimation problem is concluded to be structurally unidentifiable, i.e., the estimate does not converge to the actual state. Despite this, the current limiting controller utilizes a simple estimator for estimating the field voltage setpoint that reduces the field current, forcing it to a reference value. The characteristics of the current limiting controller are thoroughly reviewed, and a comparison to an I-controller highlights its potential advantage over the existing solution. That is a more robust and aggressive controller for limiting spikes and disturbance in the field current. However, these are only indications, and deployment and testing remain before a final conclusion can be made.

For the voltage controller, a data-driven approach is selected. A data acquisition program is specifically developed to gather high-resolution data from ABB's system. This program has some challenges introducing additional errors into the data. The data is used to create a neural network-based nonlinear state space model of the controlled system. This model is utilized for reinforcement learning to create an optimal controller with a performance similar to MPC. Due to the error in the data, computationally heavy algorithms, and a slow iterative parameter optimization process, optimal high-resolution models and controllers are not presented. Instead, low-resolution (10 ms timestep) models and non-optimal reinforcement learning controllers are presented as proof of concept.

Lastly, the steps necessary for finalizing the controllers, recommendations for ABB, alternative solutions, and other directions of inquiry are presented.

Bibliography

[1] ABB Robotics AS. Operating manual painting powerpac, robotstudio 2020.2, 3hac054618-001 rev.j. Available on request.

[2] ABB Robotics AS. Reference manual ips structure, v4.60.-, 3hna025397-001 rev.17. Available on request.

[3] ABB Robotics AS. Unit description paint, irb paint robots, 3hna012856-001 rev.38, chapter 10 electrostatic painting. Available on request.

[4] Donald Bell. Explore the uml sequence diagram. `https://developer.ibm.com/articles/the-sequence-diagram/`, Last accessed 2. March 2023.

[5] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences of the United States of America*, 113(15):3932–3937, 2016. https://www.pnas.org/content/pnas/113/15/3932.full.pdf.

[6] Spray booths for organic coating material – Safety requirements. Standard, European Committee for Standardization, Rue de la Science 23, B-1040 Brussels, 2018.

[7] Stationary electrostatic application equipment for ignitable liquid coating material - Safety requirements. Standard, European Committee for Electrotechnical Standardization, Avenue Marnix 17, B - 1000 Brussels, 2009.

[8] Kymberly Fergusson. Create uml sequence diagrams in draw.io. `https://drawio-app.com/create-uml-sequence-diagrams-in-draw-io/`, Last accessed 2. March 2023.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[10] Texas Instruments. Automotive c2000™ 32-bit mcu. `https://www.ti.com/product/TMS320R2812-Q1/part-details/TMS320R2812PGFQ`, Last accessed 15. March 2023.

[11] Dimitris K. Manolakis John G. Proakis. *Digital Signal Processing*. Pearson, 2014.

[12] Santanu Kapat and Philip T. Krein. A tutorial and review discussion of modulation, control and tuning of high-performance dc-dc converters based on small-signal and large-signal approaches. *IEEE Open Journal of Power Electronics*, 1:339–371, 2020.

[13] Sébastien Lagrange, Nicolas Delanoue, and Luc Jaulin. Injectivity analysis using interval analysis: Application to structural identifiability. *Automatica*, 44(11):2959–2962, 2008.

[14] A. Lamantia, P. Maranesi, and L. Radrizzani. The dynamics of the cockcroft-walton voltage multiplier. In *21st Annual IEEE Conference on Power Electronics Specialists*, pages 485–490, 1990.

[15] A. Lamantia, P.G. Maranesi, and L. Radrizzani. Small-signal model of the cockcroft-walton voltage multiplier. *IEEE transactions on power electronics*, 9(1):18–25, 1994.

[16] M. Mossige. Automated electrostatic painting; principles and models, 2005. Available on request.

[17] Julius Schöning, Adrian Riechmann, and Hans-Jürgen Pfisterer. Ai for closed-loop control systems: New opportunities for modeling, designing, and tuning control systems. In *2022 14th International Conference on Machine Learning and Computing (ICMLC)*, ICMLC 2022, page 318–323, New York, NY, USA, 2022. Association for Computing Machinery.

[18] R.S. Sigmond. The unipolar corona space charge flow problem. *Journal of Electrostatics*, 18(3):249–272, 1986.

[19] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction, second edition.* The MIT Press, 2018.

[20] N. R. Svendsen. Automated electrostatic painting; safety and control, 2005. Available on request.

[21] Inc The MathWorks. Deep deterministic policy gradient agents. `https://uk.mathworks.com/help/reinforcement-learning/ug/ddpg-agents.html?searchHighlight=DDPG&s_tid=srchtitle_DDPG_1`, Last accessed 13. May 2023.

[22] Inc The MathWorks. Define reward signals. `https://uk.mathworks.com/help/reinforcement-learning/ug/define-reward-signals.html`, Last accessed 13. May 2023.

[23] Inc The MathWorks. Estimate nonlinear state-space model using measured time-domain system data. `https://se.mathworks.com/help/ident/ref/nlssest.html`, Last accessed 13. May 2023.

[24] Inc The MathWorks. Long short-term memory neural networks. `https://se.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html`, Last accessed 13. May 2023.

[25] Inc The MathWorks. Reinforcement learning agents. `https://uk.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html`, Last accessed 13. May 2023.

[26] Inc The MathWorks. Reinforcement learning toolbox. `https://uk.mathworks.com/products/reinforcement-learning.html`, Last accessed 13. May 2023.

[27] Inc The MathWorks. Twin-delayed deep deterministic policy gradient agents. `hhttps://uk.mathworks.com/help/reinforcement-learning/ug/td3-agents.html`, Last accessed 13. May 2023.

[28] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

[29] Xilinx. Zynq ultrascale mpsoc. `https://www.xilinx.com/products/silicon-devices/soc/zynq-ultrascale-mpsoc.html`, Last accessed 15. March 2023.

Attachments

OneDrive folder

The attached files are located in a OneDrive folder available until September 2023 with the following link:

- Click on this line to access the OneDrive folder.

After September 2023, it will only be available on request.

An outline of the file structure of the attached OneDriev folder is illustrated by Figure 1.

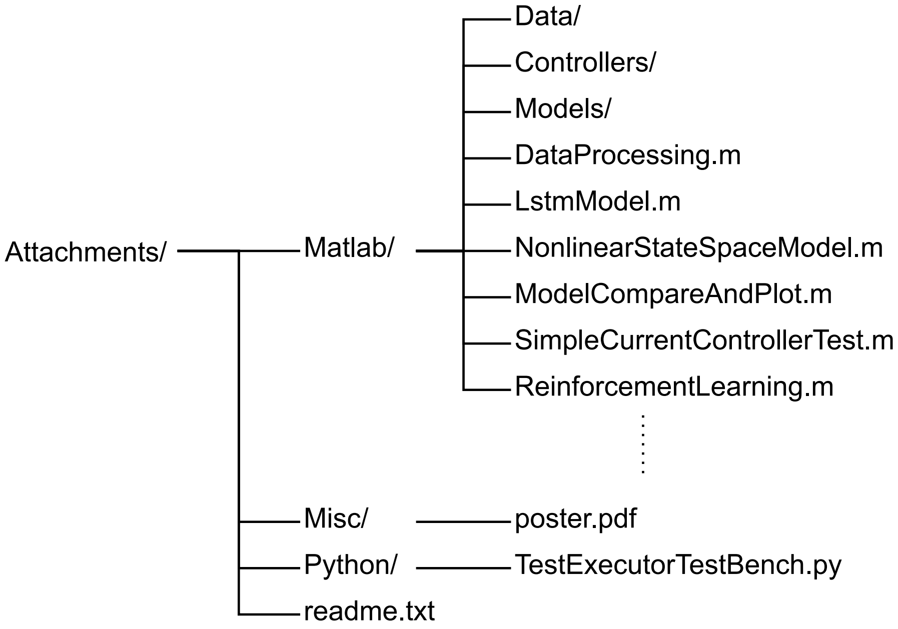


Figure 1: Structure of the attached OneDrive folder.