

FH Aachen

Faculty Electrical Engineering and Information Technology

Information Systems Engineering

Field of specialisation Systems and management

Master Thesis

Development of a hardware and software framework for the automated characterization of permanent magnets for low-field MRI systems

Submitted by

Marcel Werner Heinrich Friedrich Ochsendorf

Matriculation number: **3120232**

Examiner:

Prof. Dr.-Ing. Thomas Dey

Examiner:

Prof. Dr.-Ing. Volkmar Schulz

External examiner:

extbetreuer

Date:

01.01.2024

dank

Erklärung

I hereby declare that I have prepared this thesis independently and without outside assistance. Text passages, which are based literally or in the sense on publications or lectures of other authors, are marked as such. The work has not yet been submitted to any other examination authority and has not yet been published.

Aachen, _____, _____

Abstract

In the construction of low-field MRI devices based on permanent magnets, a large number of magnets are used. In order to realize a homogeneous B0 field with these magnets, which is necessary for many setups, the magnetic properties of these magnets have to be as similar to a certain degree. Due to the complex manufacturing process of neodymium magnets, the different properties, the direction of magnetization, can deviate from each other, which affects the homogeneity of the field. To adjust the field afterwards, a passive shimming process is typically performed, which is complex and time-consuming and requires manual corrections to the magnets used. To avoid this process, magnets can be systematically measured in advance. In this methodology, the recording, data storage and subsequent evaluation of the data play an important role. Various existing open-source solutions implement individual parts, but do not provide a complete data processing pipeline from aquation to analysis and the data storage formats of these are not compatible to each other. For this use case, the MagneticReadoutProcessing library was created, which implements all major aspects of acquisition, storage, analysis, and each intermediate step can be customized by the user without having to create everything from scratch, favoring an exchange between different user groups. Complete documentation, tutorials and tests enable users to use and adapt the FRamework as quickly as possible. The framework for the characterisation of different N45 neodymium magnets, which requires the integration of magnetic field sensors, was used for the evaluation.

Inhalt

1	Introduction	1
1.1	Background and Motivation	1
1.1.1	Low-Field MRI	1
1.1.2	Shimming procedure	1
1.2	Aim of this Thesis	1
1.3	Structure	1
2	State of the art	2
2.1	Opensource projects	2
2.2	Conceptual design	2
3	Unified Sensor	3
3.1	Sensor selection	3
3.2	Mechanical Structure	3
3.3	Electrical Interface	4
3.4	Firmware	4
3.4.1	CLI Interface	4
3.5	Example Sensors	4
3.5.1	1D: Single Sensor	6
3.5.2	1D: Dual Sensor	6
3.5.3	Full-Sphere	6
3.5.4	Integration of an Professional Teslameter	6
4	Software readout framework	7
4.1	Library requirements	7
4.1.1	Concepts	7
4.1.2	User interaction points	7
4.1.3	Export	7
4.1.4	Multi-Sensor setup	8
4.1.5	Examples	11

5 Usability improvements	12
5.1 Commandline interface	12
5.2 Programmable data processing pipeline	12
5.2.1 Web base pipeline configuration	12
5.3 Package management	12
5.3.1 Documentation	13
6 Evaluation	14
6.1 Prequesites for evaluation	14
6.2 Evaluation confiugration	14
6.2.1 Sensor readout	14
6.2.2 Processing pipeline	14
6.3 Test scenarios	14
6.4 Results	14
7 Conclusion and dicussion	15
7.1 Conclusion	15
7.2 Problems	15
7.3 Outlook	15
Literaturverzeichnis	16
Abbildungsverzeichnis	17
Tabellenverzeichnis	18

1 Introduction

1.1 Background and Motivation

1.1.1 Low-Field MRI

1.1.2 Shimming procedure

1.2 Aim of this Thesis

1.3 Structure

2 State of the art

2.1 Opensource projects

2.2 Conceptual design

- Entwicklung eines hardware und software framework zur einfachen Aquirierung von Magnetfelddaten
- Analysetools und Funktionen

3 Unified Sensor

- ziel ist es einen low cost hallsensor-interface zu entwickeln welcher möglichst universell
- verschiedene sensoren abbilden kann
- mit verschiedenen magneten typen und formen nutzbar
- reproduzierbar
- 1d, 2d, 3d
- integration

3.1 Sensor selection

- list of typical low cost hall sensors
- => alle i2c in der regel
-

3.2 Mechanical Structure

- 3d druck toleranztest
- magnet halterung mit kraftloser arretierung
- motoren und andere unter umstaänden magnetische teile in der nähe des sensors
- nylon schrauben, 3d druck, 3d gedruckte klemmverbindungen

- später rausrechnen durch kalibierung

3.3 Electrical Interface

- usb, ethernet
- pps input output
- multiplexer for i2c sensors already implemented

3.4 Firmware

- automatic sensor detection
- serial cli support for manual mode
- sync impulse => 1 mastersensor als taktquelle
- interne mittelung und speichern der werte im buffer
- was durch den user implementiert werden muss klasse

3.4.1 CLI Interface

- einfache bedienung durch nutzer auch ohne weitere software
- configuration
- debugging

3.5 Example Sensors

anbei werden drei erschienene sensoren für unterschiedliche anwendungsfälle tablle statisch dynamisch

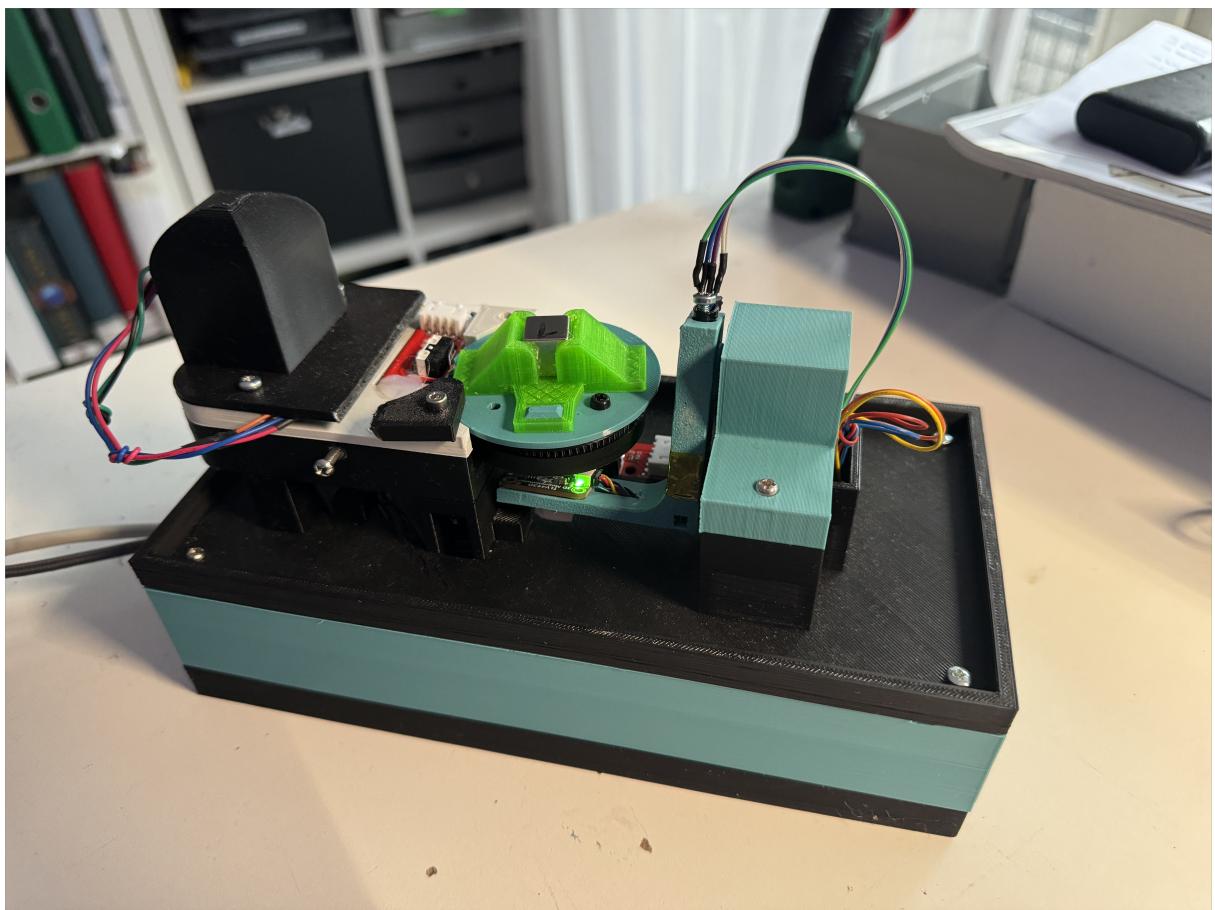


Bild 3-1: fullsphere sensor

3.5.1 1D: Single Sensor

- einfachster aufbau rp pico + sensor

3.5.2 1D: Dual Sensor

- gleicher abstand zwei gleicher sensoren
- schnelle erkennung der plarisationsebene ggf offset vom mittelpunkt dieser

3.5.3 Full-Sphere

- komplexester aufbau sensor + mechanik
- polar mechanisches system
- full sphere sensor

3.5.4 Integration of an Professional Teslameter

- einfache anbindung professioneller teslameter
- Voltkraft

4 Software readout framework

4.1 Library requirements

4.1.1 Concepts

- beispiele für projekte welche nur einzelne schnritte implementieren
- so kann man sich auf die implementierung

4.1.2 User interaction points

- grafik zeigen
- einzelne module erläutern

HAL

- aufbau hal im grunde wird nur ein die commandos an das sensor cli weitergegeben
- alle sensoren implementieren mehr oder weniger die gleichen befehle
- hal gibt nur weiter und ist “dumm”

Visualisation

4.1.3 Export

- format import export

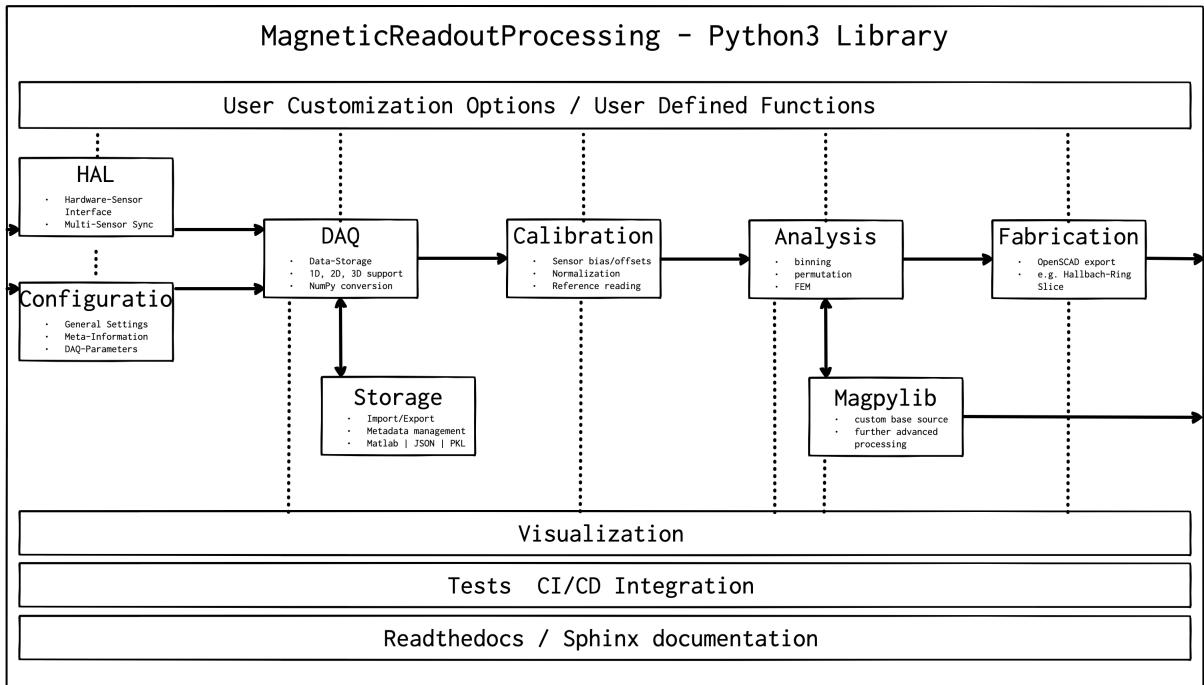


Bild 4-1: MRPlib COMPLETE FLOW

- matlab

Meta-Data

4.1.4 Multi-Sensor setup

At the moment, it is only possible to detect and use sensors that are directly connected to the PC with the library. This has the disadvantage that there must always be a physical connection. This can make it difficult to install multiple sensors in measurement setups where space or cable routing options are limited. To make sensors connected to a small remote PC available on the network, the ‘Proxy’ module has been developed. This can be a single board computer (e.g. a Raspberry Pi). The small footprint and low power consumption make it a good choice. It can also be used in a temperature chamber. The approach of implementing this via a REST interface also offers the advantage that several measurements or experiments can be recorded at the same time with the sensors.

Another application example is when sensors are physically separated or there are long distances between them. By connecting several sensors via the proxy module, it is possible to link several instances and all sensors available in the network are available

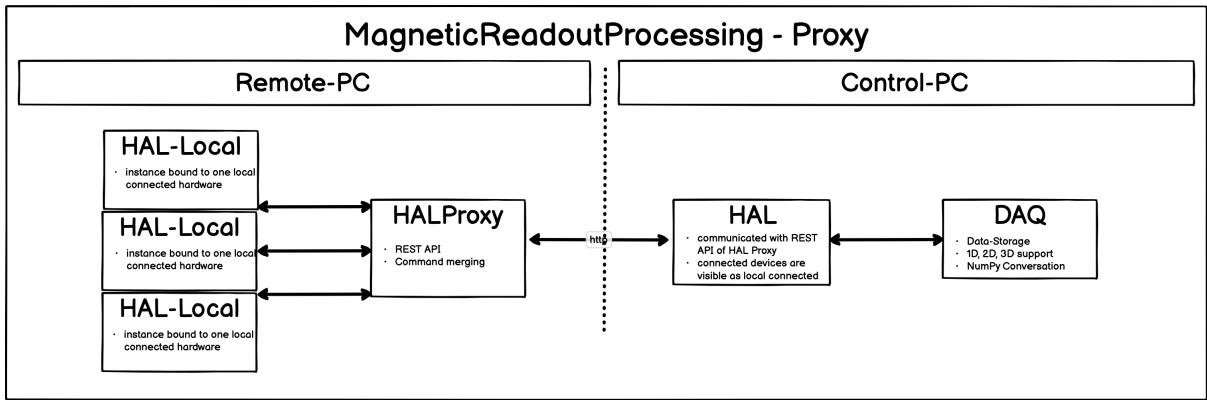


Bild 4-2: MRPlib Proxy Module

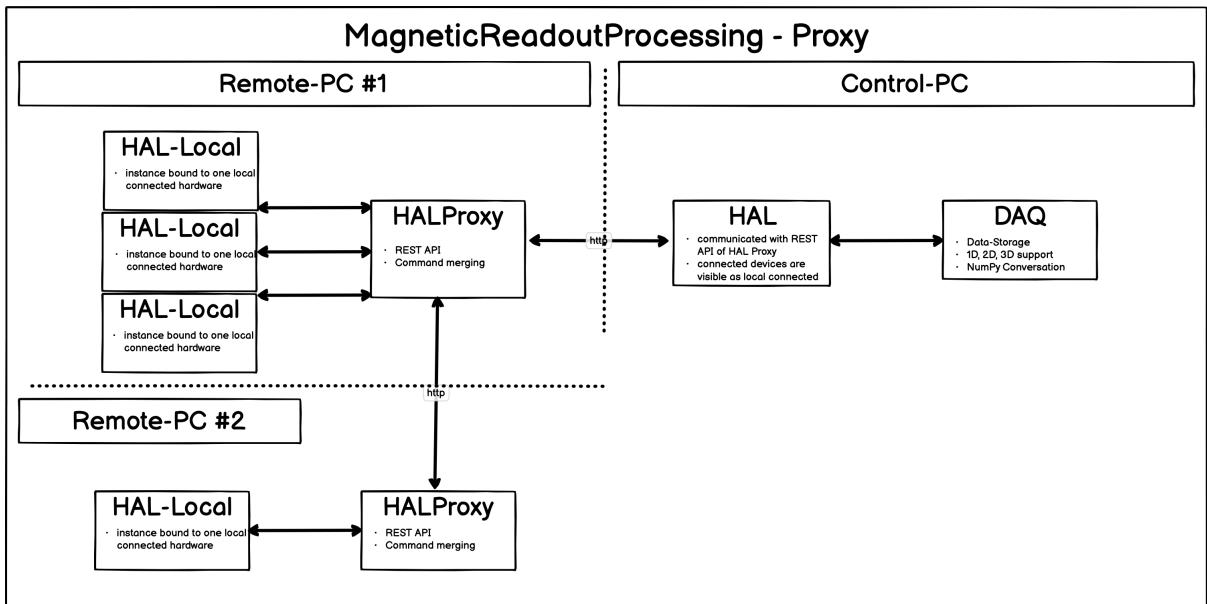


Bild 4-3: mrp proxy multi

to the control PC.

The graphic 4-3 shows the modified multi-proxy - multi-sensor topology. Here, both proxy instances do not communicate directly with the control PC, but remote PC #2 is connected to remote PC #1. This is then visible as a sensor opposite the Control PC, even if there are several proxy instances behind it.

Network-Proxy

The graphic 4-2 shows the separation of the various HAL instances, which communicate with the physically connected sensors on the remote PC and the control PC side, which

communicates with the remote side via the network. For the user, nothing changes in the procedure for setting up a measurement. The proxy application must always be started on the remote PC side.

```
1 # START PROXY INSTNACE WITH TWO LOCALLY CONNECTED SENSORS
2 $ python3 mrppproxy.py proxy launch /dev/ttySENSOR_A /dev/
   ttySENSOR_B # add another proxy instance http://
   proxyinstance_2.local for multi-sensor, multi-proxy
   chain
3 Proxy started. http://0.0.0.0:5556/
4 PRECHECK: SENSOR_HAL: 1337 # SENSOR A FOUND
5 PRECHECK: SENSOR_HAL: 4242 # SENSOR B FOUND
6 Terminate [Y/n] [y]:
```

After the proxy instance has been successfully started, it is optionally possible to check the status via the REST interface:

```
1 # GET PROXY STATUS
2 $ wget http://proxyinstance.local:5556/proxy/status
3 {
4     "capabilities": [
5         "static",
6         "axis_b",
7         "axis_x",
8         "axis_y",
9         "axis_z",
10        "axis_temp",
11        "axis_stimestamp"
12    ],
13    "commands": [
14        "status",
15        "initialize",
16        "disconnect",
17        "combinedsensorcnt",
18        "sensorcnt",
19        "readsensor",
20        "temp"
21    ]
22    # RUN A SENSOR COMMAND AND GET THE TOTAL SENSOR COUNT
23    $ wget http://proxyinstance.local:5556/proxy/command?cmd=
      combinedsensorcnt
24    {
25        "output": [
26            "2"
27        ]
28    }
29 }
```

The query result shows that the sensors are connected correctly and that their capabilities

have also been recognised correctly. To be able to configure a measurement on the other, only the IP address or host name of the remote PC is required:

```
1 # START PROXY INSTNACE WITH TWO LOCALLY CONNECTED SENSORS
2 $ python3 mrpproxy.py proxy launch /dev/ttySENSOR_A /dev/
   ttySENSOR_B
3 Proxy started. http://0.0.0.0:5556/
4 PRECHECK: SENSOR_HAL: 1337 # SENSOR A FOUND
5 PRECHECK: SENSOR_HAL: 4242 # SENSOR B FOUND
6 Terminate [Y/n] [y]:
```

Sensor Syncronisation

- alle clients über ptp verbunden
- zentraler reset zwischen zwei pps pulsen um den readout counter auf 0 zu setzen

Command-Router

- nummerierung zuerst lokale sensoren dann weitere proxy sensoren
- commando templating

4.1.5 Examples

5 Usability improvements

5.1 Commandline interface

- automatische sensor dedetion
- planung verschiedener messungen mit untersch. hardware
- zentrale abarbeitung

5.2 Programmable data processing pipeline

- datenanalyse für nicht programmieruer
- automatisierter aufbau der call-tree
- mit typcheck
- alle funktionen mit bestimmmer signatur werden automatisch aus globals geladen und stehen nutzer zur verföhung

5.2.1 Web base pipeline configuration

- drag& drop system
- automatische convertierung zum yaml system

5.3 Package management

- verteilung durch paketmanager

- docker cli Beispiel

5.3.1 Documentation

6 Evaluation

6.1 Prequesites for evaluation

6.2 Evaluation configuation

6.2.1 Sensor readout

6.2.2 Processing pipeline

6.3 Test scenarios

6.4 Results

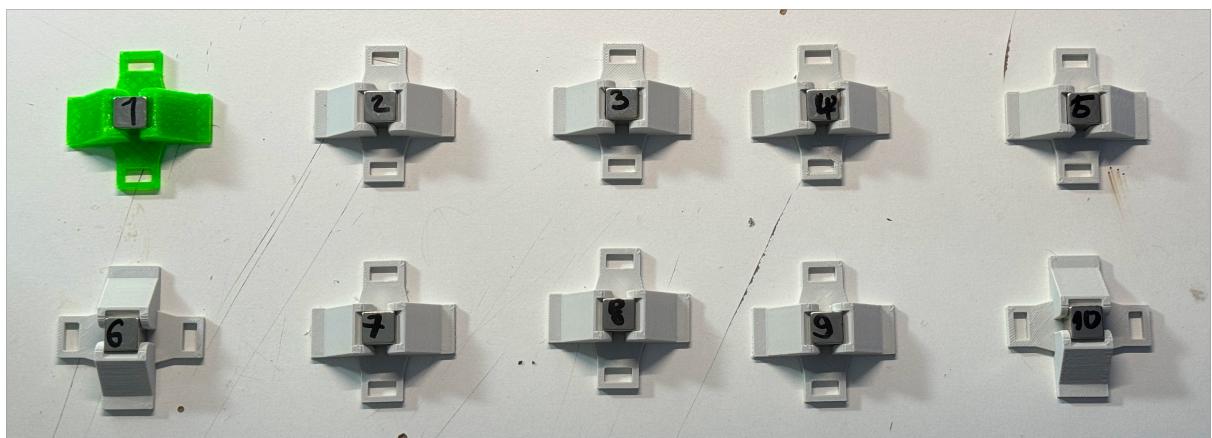


Bild 6-1: testmagnets in holder

7 Conclusion and dicussion

7.1 Conclusion

7.2 Problems

7.3 Outlook

- magfield camera

Literaturverzeichnis

Abbildungsverzeichnis

3-1	fullsphere sensor	5
4-1	MRPlib COMPLETE FLOW	8
4-2	MRPlib Proxy Module	9
4-3	mrp proxy multi	9
6-1	testmagnets in holder	14

Tabellenverzeichnis