

Timon Rupelt

Project Report
**Investigation of using Reed
Sensors for Monitoring
Flood Protection Gates**

Faculty of Engineering and Computer Science
Department of Computer Science

Supervising examiner: PROF. DR. THOMAS C. SCHMIDT

**HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG**
Hamburg University of Applied Sciences

Contents

1 Introduction	2
2 Requirements	2
3 Implementation	3
3.1 Developed Hardware	3
3.2 Data Collection Software	4
4 First Field Test	5
5 Sensor Testing	7
5.1 Test Results	9
6 Conclusion and Outlook	13
References	14

Abstract In the context of the RESCUE-MATE research project, which aims to improve situational awareness in storm flood scenarios, it is necessary to equip flood gates with sensors that enable remote monitoring of their state. This report showcases the evaluation of reed sensors for this application. A field test gives insights in the optimal sensor placement and the design of a permanent mounting solution for long-term evaluation at a swing gate. Laboratory tests are conducted to evaluate the reliability of the sensors and quantify their errors in a high-usage scenario. The results show the sensors producing consistent position readings across many activation cycles. Observed deviations at the switching point are minor and acceptable given the coarse movement resolution required for this application.

Keywords: reed sensor, reed switches, proximity sensors, flood gate monitoring, riot, rescue-mate

1 Introduction

The RESCUE-MATE research project [1] aims to investigate the creation of a platform for dynamic situation overview for emergency situations, such as flooding. Various data sources from water level sensors to social media, are integrated to improve the understanding of the situation for emergency services and other personnel. This should enable better identification of hazards and more organized deployment of personnel. One of the aims of the “Resilient Sensor Networks for Dike Monitoring and Emergency Communication” subproject is to increase the availability of information on dikes and floodgates. Floodgates of varying sizes and closing mechanisms would need to be equipped with sensors, to give reliable information about their state. In this context, the previous [2] and this work focus on the investigation of the use of reed sensors for this monitoring task. Reed sensors or reed switches are electromagnetic switches [3]. Applying or removing an external magnetic field forces two internal contacts to open or close. This mechanism can be used as a simple proximity sensor. In the previous work, six research questions were formulated to guide the evaluation of reed sensors for this applications. This work is focused on discussing the following two research questions

RQ1 How can reed sensors be used to monitor the various locking mechanisms?

RQ2 How to quantify errors and the reliability with significant confidence?

Parts of the work for this project were done in cooperation with a fellow student evaluating inductive sensors for the same use case.

The remainder of this report is structured as follows. Section 2 gives an overview of the requirements and the approach used to evaluate the research questions. Section 3 discusses the developed hardware and software needed to evaluate and use the sensors in an application. The first field tests conducted for the installation of the sensors are presented in Section 4. Section 5 describes the test setup and procedure for evaluating the reliability and for quantifying the errors of the sensors, and discusses the results. Finally, a conclusion is drawn in Section 6 with regard to the results of the field test, and the in-house testing and closes with an outlook for further investigation is given.

2 Requirements

In order to deploy the sensors to monitor floodgates, the development of hard- and software components is necessary. The requirements for the hardware are having a

microcontroller to read the sensor data as well as support connection with a remote endpoint to upload the acquired data. It should also support the connection of multiple sensors to enable simultaneous evaluation of different sensors.

The software needs to bridge the gap between the sensor and the remote endpoint. It has to reliably read the sensor values, encode and transmit them to the remote endpoint. Encased in a housing, these components form a sensor box that can be deployed at a floodgate. A fixed power supply is used for the test phase, whereby the box should run on battery power in the long term.

To answer RQ2 (reliability and error quantization), i.e. whether a sensor deviates over time, it is necessary to test the sensors in such a scenario. For this purpose, a test setup is required to test the sensors during a large number of cycles needs a repeatable and deterministic way. This setup has to allow precise movement in small steps between two defined points. These points depend on the operational range of the used sensors. At each step between these points, the sensor values are recorded. This is observed for many cycles to see whether these distances change over time. Higher accuracy, i.e. smaller steps, enables a more precise determination of the activation and deactivation points.

3 Implementation

Both of the subtasks described above require interaction with the sensors. This requires both suitable hardware and software to read and process the sensor data. The hardware components used and the software developed are described in the following.

3.1 Developed Hardware

The core of the system is the *Adafruit Feather nRF52840 Sense*¹. The nRF52840 SoC is built around a 32-bit ARM Cortex-M4 CPU running at 64 MHz with 1MB Flash Memory and 256 KB RAM. Equipped with 21 GPIO pins, six of which are 12-bit ADC, as well as SPI and I2C support for all pins, it is well suited for this purpose of reading sensors in a low-power application. Using an expansion board - a so called shield - a LoRa radio can be added to the microcontroller.

A breakout PCB was designed with a focus on allowing easier switching of the sensors in the current test phase. Later revisions will see more permanent connection solutions.

¹<https://www.adafruit.com/product/4516#technical-details>

The schematics are shown in Figure 1. Screw terminal blocks are used to allow access to 3.3V, ground, 4 digital pins, and 4 ADC pins.

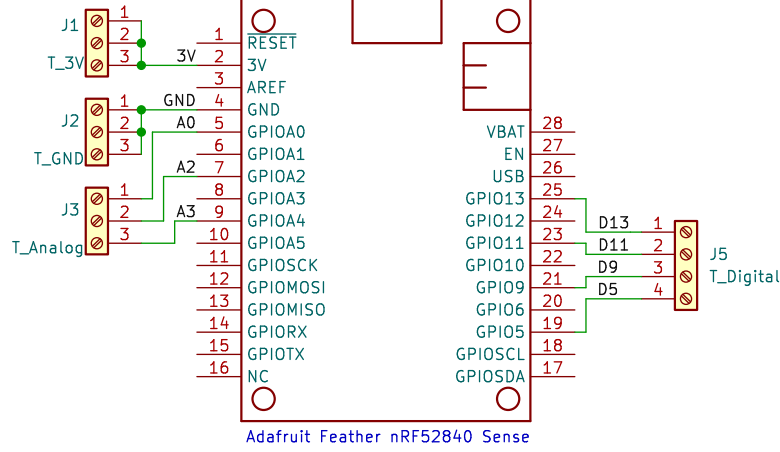


Figure 1: The schematics of the first revision of the main PCB for the sensor box.

The housing for the main board is a 120x120x70 mm box. It has IP ratings IP66, IP67 and IP68, meaning it is dust-tight and protected against continuous immersion in water [4]. It was extended with three connectors featuring similar IP ratings allowing protected access to pins and the board from the outside.

The sensors used in this work are two types of reed switches, the MS-324-4-3-0500² by PIC and the MK11/M8-1C90C-5002³ by Standex-Meder. The abbreviated names *MS-324* and *MK11* are used in the following sections.

3.2 Data Collection Software

The main requirements for the software were to support the readout of the sensor types under investigation. The acquired data has to be made available for a remote endpoint to see current states and to gather the data to monitor behavior and trends. In the context of this work, the data was retrieved via direct serial connection. In the next phase, when a deployment will take place for a longer period of time, LoRa will be used.

²<https://www.pic-gmbh.com/en/products/ms-324-4>

³<https://standexelectronics.com/products/mk11-plastic-series-reed-sensor/>

The developed application will run on the device via the real-time operating system RIOT [5], which is designed for resource-constrained embedded devices. It provides implementations for a variety of technologies and standards and supports a wide range of microcontrollers, sensors, and actuators.

The software works as a basic state machine. The main thread is waiting for IPC messages to arrive, which trigger the reading of the triggering sensor. Depending on the type of sensor, an interrupt is registered for either a rising edge, a falling edge or both. Each triggers the sending of a message to the main thread. This is the case with the reed sensors used. Alternatively, a timer can be configured to trigger the same process when it expires. This is the case for sensors that do not switch between states.

The sensor data is encoded in the Concise Binary Object Representation (CBOR) data format defined in the RFC 8949 Internet Standard [6]. It is based on the JSON data model and allows for extremely small code size, fairly small message size, and extensibility. The sent data consists of an array of raw sensor values *d*. The number of items and the size of the values depend on the number and types of the sensors. An event counter *c* represents the number of events which have occurred up to that point. The counter can be used to identify missed events. The third value is a sequence number *s* to keep track of the messages sent. An example of a message payload with three sensor values is shown in Listing 1.

```
{  
  "d": [0,1,430], "c":3, "s":1  
}
```

Listing 1: Example message payload.

In the scenario of the field test discussed in Section 4, it is necessary to gather all sensor values at the same time for a consistent comparison between the sensors. Therefore, a secondary mode is used, in which all sensor values are collected when a message arrives. Also, only one sensor is configured to trigger an interrupt and therefore the sensor reading.

4 First Field Test

The aim of the field test was to explore the mounting options on-site for the sensors under consideration, which will help to answer RQ1. Possible locations for the sensor box with power and network availability were also examined.

In this field test only the MS-324-4-3-0500 reed sensor was selected for testing. A 3D-printed bracket was used for easier mounting and to position the sensor forwards. Flat brackets and adhesive putty were used to temporarily fix the sensor and magnet to the gate and ‘frame’.

A simple manual swing gate, shown in Figure 2, was chosen as the first test subject. The sensor box and reed sensor were placed on the wall on the door edge. The sensor was placed on the wall close to the edge, where the door is positioned when closed. The counter magnet has been placed on the gate itself, hanging over the edge, so that the sensor is triggered when the gate is closed. This setup is shown in Figure 3.



Figure 2: The ‘test subject’ manual swing gate.

Positioning the sensor and magnet at the appropriate distance, opening and closing the gate triggers the sensor as expected. Operating the latches on the gate increases the contact pressure of the gate to the frame. This additional change in distance is very small, a few millimeters at most. Using a reed sensor to trigger at this distance would require very accurate and robust positioning. Any change in the position or closing distance of the gate could result in erroneous readings.

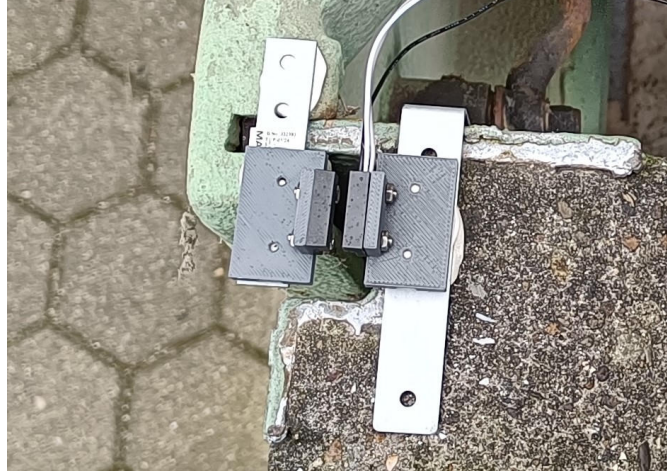


Figure 3: The mounted sensor and counter magnet on the gate and ‘frame’.

Furthermore, measurements and images of the gate were taken. These will help in the design of a more permanent mount to be used in the next phase. This mount needs to allow for adjustments of the sensors towards the gate. Due to the characteristics of the gate, the counter magnet must be positioned hanging over the end of the gate to align with the sensor.

5 Sensor Testing

In order to evaluate the reliability of the sensors, they must be operated and their outputs recorded in a controlled and reproducible environment. To this end, a test setup has been constructed that operates the sensors in a deterministic manner over a large number of cycles. Instead of building a fully custom setup, a 3D printer can be repurposed. It already supports precise motion on different axes in steps of up to 0.01 mm. G-code can be used to easily control a 3D printer to move back and forth between defined points in a given number of steps. G-code is a commonly used computer numerical control (CNC) language used to control machine tools and 3D printers, standardized by the International Standards Organization (ISO) 6983 [7].

The criteria for the usability of a 3D printer model for this application is an available API to send custom G-code to the printer without using a slicer software. The functionality needed is to send a G-code command to move to position $\mathbf{x}$, by using the G1 G-code (e.g., on axis Z: `G1 Z 10.0`) and report when the command has been executed so the sensor values can be collected. The issue with this command arises when a printer reports back

after it has scheduled a command, such as moving to position x , but before the motion is complete. This issue can be resolved by using a specific G-code command (**M400**), which instructs the printer to wait until the motors have finished moving before continuing. With this, the printer only reports back when the motors have stopped moving. This sequence is illustrated in Figure 4. Once the motors have stopped, sensor readings can be collected to log the current position and sensor readings.

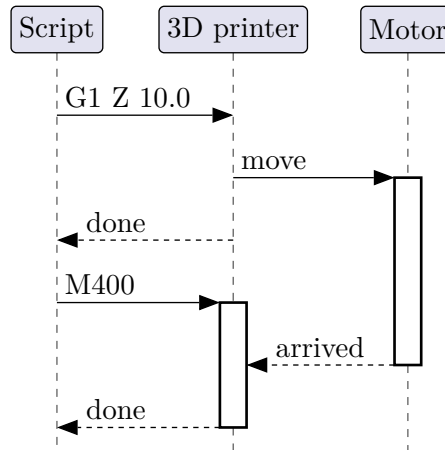


Figure 4: Abstract sequence diagram of the communication between the test routine script and a 3d printer.

The 3D printer used for this is the Sovol SV08⁴. It is running Linux with the Klipper 3D printer firmware.

A Python script generates the test routine of G-codes based on the given parameters of activation and deactivation points, as well as the number of steps and cycles. More steps result in more intermediate positions at which the printer stops but also a longer execution time. The G-codes are sent to the printer via the API of Moonraker⁵, which is a *Web API Server* for Klipper. It allows sending G-codes via HTTP requests as shown in Listing 2.

```

POST /printer/gcode/script
Content-Type: application/json
{
    "script": "G1 Z 10.0" // move to 10.0 on the Z axis
}
  
```

Listing 2: HTTP Request to send a G-code to the printer.

⁴<https://www.sovol3d.com/products/sovol-sv08-3d-printer>

⁵<https://github.com/Arksine/moonraker>

The Python script that coordinates the test routine by sending commands to the printer and recording sensor values runs on the printer itself. The printer is connected to a feather board for reading the sensors via USB. It runs a modified version of the software described in Section 3.2, which allows sensor values to be read using custom shell commands sent to the board via a serial connection. An example of this is shown in Listing 3.

```
> get_data  
36164831901...
```

Listing 3: Shell command to fetch the CBOR encoded data and its output.

To attach the sensors and a corresponding counter magnet to the printer, special mounts were designed and 3D printed. The mount for a sensor shown in Figure 5 consists of multiple parts with horizontal slots to allow for sliding adjustments of the relative position. Together this allows the sensor to be positioned on a 2D plane of 28x28 mm. The upper part can be exchanged for other mounting variants specific to a different sensor.

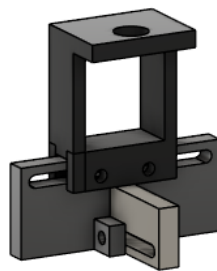


Figure 5: 3D model of the 3-part adjustable sensor mount.

5.1 Test Results

The test routine was run using the parameters specified in Table 1. The two sensors were tested in separate runs as only one sensor could be fitted on the printer at a time. The counter magnet supplied with the MS-324 sensor was used for both sensors. Optimal results will show no deviations, meaning the sensor values are consistent throughout every cycle. When deviations occur, it is important to determine if a pattern can be identified. In addition to the total number of deviations, the timing and the relationship to other deviations need to be considered. This makes it possible to determine whether these deviations are isolated occurrences or sequential. Additionally, it is important to distinguish between occurrences that are spread over all cycles and those that only occur

Cycles	Steps per Cycles	Start Position (mm)	End Position (mm)	Step Distance (mm)
1000	60	15	0	0.25

Table 1: The test routine parameters.

towards the end of the routine. It is important to note that no comparison can be made between the two sensors in terms of their absolute positions, as no consideration was given to positioning them with millimeter precision in the same place. Furthermore, the housing of the sensors complicates the determination of the precise position and orientation of the reed switches, further hindering the process.

The results of the test runs are shown in Table 2. The measurements are divided into two phases: *Activation* and *Deactivation*. Activation is moving towards the sensor, and deactivation is moving away from it. The *Switching Point* is the point at which the sensor is triggered or switches back to its original state respectively. This is listed as a pair of positions between which the value change occurs. The total *Deviations* describe how often a value has deviated from the ‘normal’ state. This is the value that was measured most often at that position. The *Accuracy* describes the total deviations in relation to the total steps, while the accuracy per position shows the likelihood of a deviations occurring at a position. This value is higher for many deviations at a single position than it is when they are spread over several positions, with only a few deviations at each position.

Both sensors have a normally-open (NO) and a normally-closed (NC) pin. Because these pin values should always be the inverse of the other, they are simplified here to one state describing if the sensor is *Activated* (NO=1, NC=0) or *Deactivated* (NO=0, NC=1). Should these values not be the inverse of each other, the sensor would be broken.

Sensor	Phase	Switching Point (mm)	Deviations	Accuracy	Accuracy per Position
MS-324	Activation	9.5/9.75	0	1.0	1.0
	Deactivation	11.75/12.0	11	0.91	0.998
MK11	Activation	11.5/11.75	5	0.96	0.998
	Deactivation	14.5/14.75	0	1.0	1.0

Table 2: Test Routine Results

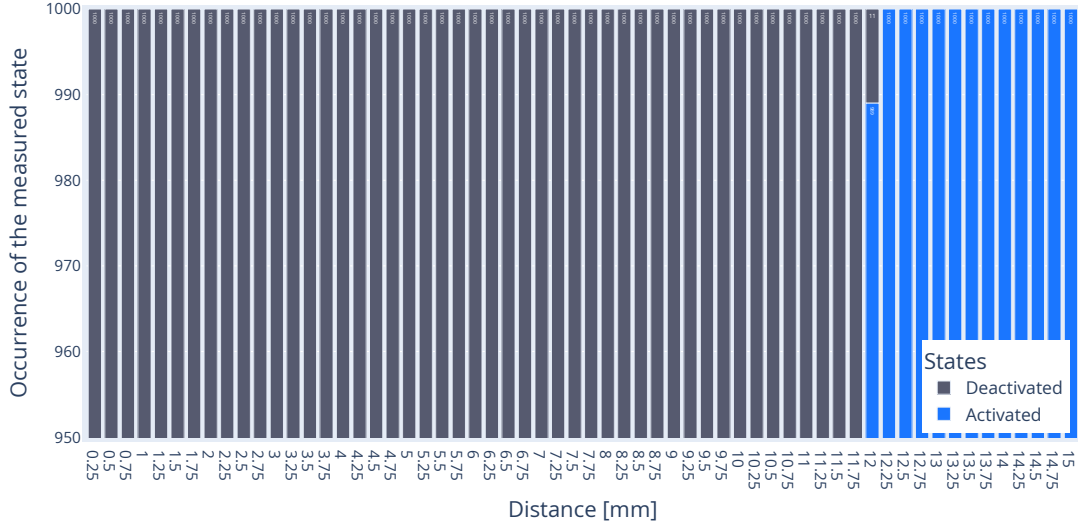


Figure 6: Measured occurrence of states per position of the **MS-324-4-3-0500** during the *Deactivation* sequence (Y-axis limited for better readability). 11 deviations were measured at 12 mm.

For the MS-324 sensor, no deviations were measured during the activation sequence. However, during the deactivation phase 11 deviations occurred. Comparing the accuracy and the accuracy per position shows that the later is higher, suggesting localized deviations at a single position. This is confirmed by the plotted occurrence of sensor values shown in Figure 6. The bar plot shows the occurrence for each step of the cycles. The distribution is almost completely uniform. Deviations occur only at position 12.0. The occurrence of the deviations for position 12.0 over all cycles is plotted in Figure 8 as the gray line. There are two deviations around cycle 550 and two groups with five and four consecutive deviations around cycle 820, respectively.

The test run results of the MK11 sensor show 5 deviations occurring during the activation phase. As can be seen in Figure 7, the deviations only occur at position 11.5 and between cycle 412 and 426 as shown in Figure 8 as the blue line.

There is no clear pattern for either sensor, indicating that the deviations are not caused by a sensor fault. Because of this, and the fact that the positions of the deviations 12.0 mm and 11.5 mm are close to the respective switching points, it is more likely to be caused by movement in the test bench setup. These movements could have been caused by irregular motion due to varying speeds of interfaces used, or by an external force.

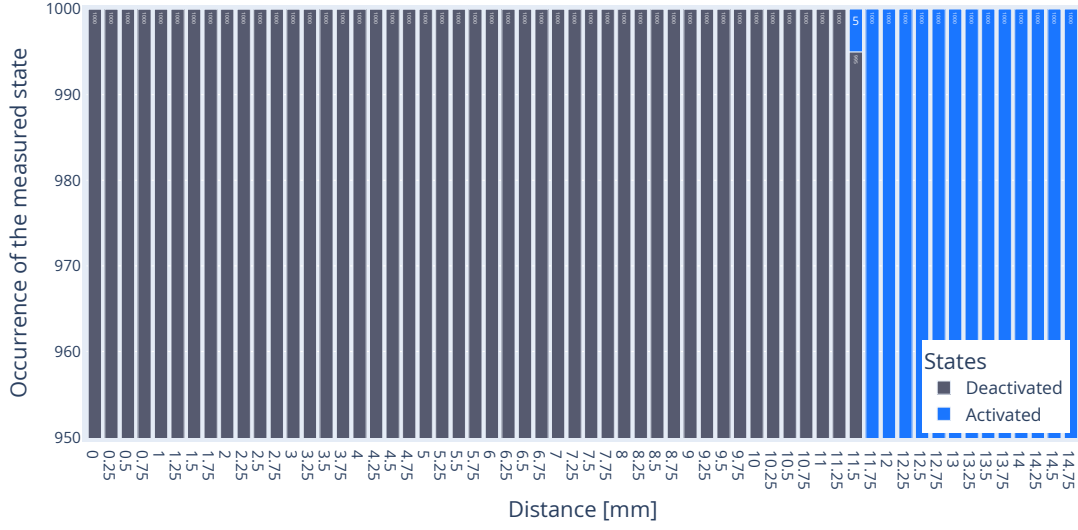


Figure 7: Measured occurrence of states per position of the **MK11/M8-1C90C-5002** during the *Activation* sequence (Y-axis limited for better readability). Five deviations were measured at 11.5 mm.

These deviations are not to be seen as errors but as a result of the mounting and the number of steps used in the test run. At some point the switching, has to happen, and if a measurement is taken at that position, there is a high probability of deviations. More importantly, there is no deviation at any other position, indicating accuracy around the switching point.

Therefore, it can be concluded that there are “safe areas” before and after activation where it can be guaranteed that the state is correct. However, the measured unsafe range is very small, with only one position. Even if the surrounding positions at ± 0.25 mm are added to the unsafe area as a buffer zone, the sensors remain usable.

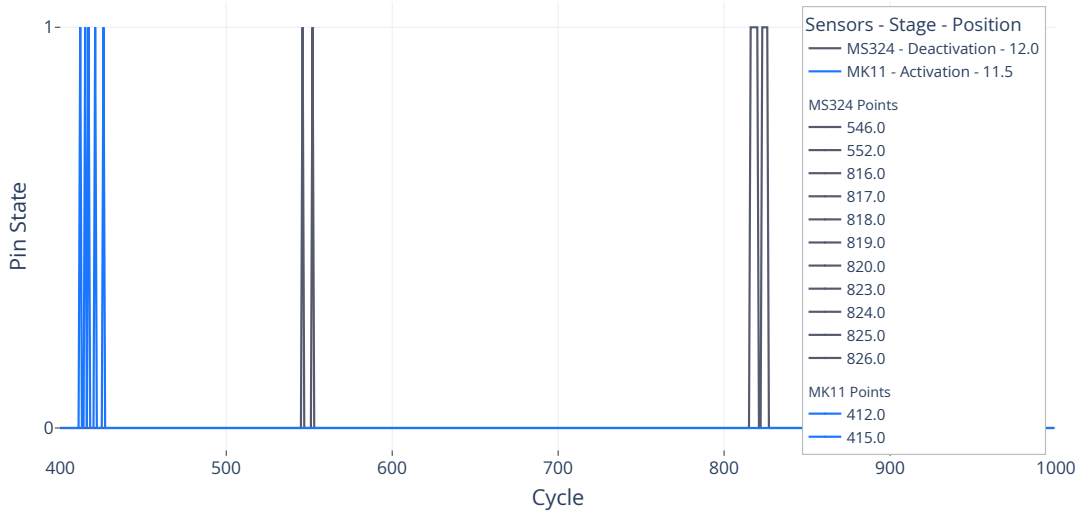


Figure 8: Occurrences of the sensor deviations over all cycles (Limited to the range from cycle 400 to 1000 for better readability).

6 Conclusion and Outlook

The main objective of this project was to gather information in terms of flood gate installation options and sensor reliability. The first field test granted insights in how and where to position a sensor to measure the state of a manual closable floodgate. With this knowledge, a more permanent fastening of the sensors can be designed for that gate to allow for long-term tests. The lab tests result showed no errors in the sensors. The sensors provide constant values for positions around the switching point over a higher number of activation cycles. Since only a relatively rough recording of the movement is needed when using the sensors, deviations at the switching point are insignificant. The next phase of the project will be to design a prototype for long-term deployment and data collection in the field. In addition, the laboratory test could be extended to account for possible interference from adjacent metal and temperature changes.

References

- [1] RESCUE-MATE, “RESCUE-MATE.” Accessed: Mar. 06, 2025. [Online]. Available: <https://www.rescue-mate.de/>
- [2] T. Rupelt, “Investigating Reed Sensors for Monitoring Closing Mechanisms of Flood Protection Gates,” 2024. [Online]. Available: https://inet.haw-hamburg.de/teaching/ss-2024/project-class/fw1_timon_rupelt.pdf/view
- [3] Standex-Meder Electronics, “Reed Technology.” Accessed: Mar. 06, 2025. [Online]. Available: <https://standexelectronics.com/resources/technical-library/technical-papers/technical-databook-reed-switch-technology/>
- [4] IEC Technical Committee 70, “Degrees of protection provided by enclosures (IP Code), IEC 60529,” International Electrotechnical Commission, 2013. [Online]. Available: <https://webstore.iec.ch/en/publication/2452>
- [5] E. Baccelli, C. Gündogan, O. Hahm, P. Kietzmann, M. Lenders, H. Petersen, K. Schleiser, T. C. Schmidt, and M. Wählich, “RIOT: an Open Source Operating System for Low-end Embedded Devices in the IoT,” *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4428–4440, Dec. 2018, [Online]. Available: <http://doi.org/10.1109/JIOT.2018.2815038>
- [6] C. Bormann and P. Hoffman, “Concise Binary Object Representation (CBOR),” IETF, RFC 8949, Dec. 2020. [Online]. Available: <https://doi.org/10.17487/RFC8949>
- [7] Technical Committee ISO/TC 184, “Automation systems and integration — Numerical control of machines — Program format and definitions of address words,” International Organization for Standardization, ISO 6983, Geneva, Switzerland, 2009. [Online]. Available: <https://www.iso.org/standard/34608.html>