Measuring Bulk Material Flow using Commercially-Available LIDAR Sensors

Master's Thesis in Electrical Engineering & Information Technology
(Automation)
presented at the

Faculty of Electrical Engineering & Information Technology of the
University of Applied Sciences, Düsseldorf

by

Nareshkumar Rao

1. Examiner: Prof. Dr.-Ing. M. Protogerakis

2. Examiner: Prof. Dr.-Ing. R. Beck

Düsseldorf
February 2022

Abstract

The availability of powerful commercial hardware in recent years has enabled not only the potential to reduce costs, but also allowed for the integration of the conventional software development methods of Industrial IoT. To test the viability of these products and methods, this project concerns itself with developing a cheaper and easier-to-install alternative to the conventional belt scales used for measuring the volume of bulk material on industrial conveyor belts.

Previous research has shown that optical and laser-based methods for measuring the volume of bulk material are indeed possible, but only through the use of research-grade or industrial-grade equipment. This work demonstrates—through the development of a prototype with the accompanying software—that a system using the Intel RealSense L515 and a Raspberry Pi has the required performance to run the required analysis, and deliver those results over Industrial Ethernet to conventional industrial PLCs.

Although a fully-functional product was not realized due to the unsuitable optical properties of the tested conveyor belt, the system is capable enough to deliver results in a laboratory setting. More work is required to further fine-tune the signal pre-processing issues in the field.

Acknowledgments

I would like to give special thanks to Prof. Dr.-Ing. Michael Protogerakis for his attentive supervision of my Master's Project and this thesis.

N. Stuhrmann and M. Meilchen too have my gratitude for their tireless assistance in the laboratory.

Thank you to Prof. Dr.-Ing. Ralf Beck for offering to be the second examiner of this thesis.

M. Viehöfer from Siep Kieswerk GmbH & Co. KG has my deepest appreciation for their generous support, allowing me to test my prototypes on their site.

Finally, I would like to thank S. Nagappan and D. Rashid for their proofreading of this work, and my partner I. Gvazdaityte whose everlasting support made this work possible.

Document produced with LaTeX
12pt Computer Modern
Last updated: January 1, 1980

Declaration of Academic Integrity — Eidesstattliche Erklärung

I have written this thesis independently, and I have not used any other sources or aids other than the ones stated.

Düsseldorf, February 2022

Nareshkumar Rao

This thesis was supervised by:
1. Examiner: Prof. Dr.-Ing. M. Protogerakis
2. Examiner: Prof. Dr.-Ing. R. Beck

Diese Arbeit ist von mir selbstständig angefertigt und verfasst worden. Es sind keine anderen als die angegebenen Quellen und Hilfsmittel benutzt worden.

Düsseldorf, Februar 2022

Nareshkumar Rao

Diese Arbeit wurde betreut von:
1. Prüfer: Prof. Dr.-Ing. M. Protogerakis
2. Prüfer: Prof. Dr.-Ing. R. Beck

Contents

Abstract **iii**

Acknowledgments **v**

Declaration of Academic Integrity — Eidesstattliche Erklärung **vii**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Aims of this Work . . . 2
1.3 Approach . . . 3

2 State of the Art **5**

3 Design and Implementation **7**
3.1 Principles of Operation . . . 7
3.2 Phases of Development . . . 12
3.3 Components . . . 14
3.4 Process Overview . . . 16
3.5 Software Architecture . . . 18
3.6 Housing . . . 21

4 Validation **23**
4.1 Sandbox Stage . . . 23
4.2 Laboratory Stage . . . 24
4.3 Field-Testing Stage . . . 26

5 Conclusion and Outlook **29**
5.1 Project Status and Feasibility . . . 30
5.2 Work Remaining . . . 30

Bibliography **33**
Academic References . . . 33
Online References . . . 34

1 Introduction

This chapter lays out an overview of this project and thesis. The reasoning and motivation for exploring the topic will be elaborated, followed by the parameters of the project. Finally, the chapter closes with an overview of the various stages of development of this project, from conception to completion.

1.1 Motivation

Transportation of Bulk Material

It is necessary in several industries [1], including those of mining and manufacturing, to transport bulk material from one location to another. In mining, it may be sand or gravel. In manufacturing, it may be powdered chemicals.

The transportation of this bulk material typically involves the use of a conveyor belt. These conveyors are specifically designed for the efficient transport of bulk material.

Measuring Bulk Material Flow

This transportation of bulk material flow introduces the need to accurately measure the rate at which the material is flowing. This is essential for various tasks such as keeping track of inventory or for control systems. Knowing when a belt is overloaded, running empty or broken is also an important safety concern. This work deals with the specific challenge of measuring bulk material flow on a conveyor.

Conventional Belt Scales

The conventional method of measuring bulk material flow in use in the industry today is the electronic belt scale—as shown in Figure 1.1. These scales use load cells to translate compression and tension into electrical signals. These signals representing weight may then be converted into measurements of volume.

These electronic belt scales are robust and proven in the field. However, there are also downsides with this approach.

1. High unit costs as well as higher retrofitting costs

2. Humidity and moisture content of the material may introduce significant errors which may not be easily compensated

3. Vibration from transport and loading introduces noise into the measurements [8]

Figure 1.1: *A conventional electronic belt scale.*

1.2 Aims of this Work

The central **research question** that is investigated in this work is:

How can a cheaper and easier-to-install measurement system for bulk material flow on a conveyor belt be designed?

Use of Commercially-Available Products

As given by the research question above, one of the central parameters is the question of cost. Since the cost of industrial equipment can be far greater than the cost of commercially available products, studying alternatives becomes attractive.

As an example, the cost price of the Intel RealSense L515 used in this project was € 380[1], whereas the SICK LM400 used by Fojtik [3] can cost upwards of € 4000[2].

Use of the LIDAR Sensor

The usage of the LIDAR sensor was implemented in order to fulfill the second requirement of the research question, namely that the solution must be easier to install than other conventional solutions.

[1]Due to Intel announcing that they are discontinuing their LIDAR sensor series, the price of this particular product has risen up to € 570 as of January 2022.

[2]This price is an aggregate estimate based on multiple online merchants as of January 2022

As will be discussed in the following section on design, the LIDAR sensor was selected primarily because it is a contactless sensor. This means that installation can be carried out with little to no adjustments to the existing conveyor belt system. The LIDAR sensor must simply be suitably positioned in order to gather and deliver data.

Requirements & Restrictions

Besides fulfilling the research question, the design solution should meet the following criteria as well.

- **Industrial Robustness** - The final product should be able to withstand the harsh environments that it would likely be installed in, i.e. in a gravel quarry. This means the product must be adequately housed and protected from the environment, against vibrations and shocks.

- **Industrial Connectivity** - The product should be able to interface with existing industrial networks, i.e. using Industrial Ethernet.

- **Real-Time Ability** - The product should ideally deliver values in Real-Time through the required interface. This means not only a high enough data resolution but also high determinism.

- **Remote Control** - The product should be able to be configured and diagnosed remotely, in order to prioritize simplicity of installation and maintenance.

Conception of the Design

Analysis of the research question as well as the other requirements have led to the use of the following components to build the final product.

- **Raspberry Pi 4 Model B** - Provides a low-cost platform with a Linux kernel and OS for processing data

- **netHAT** - A HAT format extension module for the Raspberry Pi that provides Industrial Ethernet capabilities

- **Intel RealSense L515 LIDAR Sensor** - Commercially available LIDAR sensor unit, compatible with the Pi

More details on each of these components are provided in section 3.3.

1.3 Approach

The following is a layout of the steps taken in order to realize the final product of this project. For more details on the specifics of the steps, see chapter 3.

Interfacing with the LIDAR sensor

The initial step of this project was naturally to establish an interface with the sensor itself. This includes:

- Preparing the development environment
- Building the necessary libraries and drivers
- Developing a basic test software to manipulate sensor data

Proof-of-Concept Software

After being able to successfully interface with the sensor, a proof-of-concept software was designed and developed. Later, a GUI interface was also added to the software to improve ease-of-use. The software was designed to be able to do the following things:

- Display the raw sensor data in a meaningful way
- Display, calibrate and use the sensor data as a line scanner, with the **cross-sectional area** as an output

Laboratory Prototype

Once the proof-of-concept software was stable, the setup was moved into a laboratory environment in order to further develop the main functionalities of the prototype. Among the functionalities that were developed were:

- Remote acquisition the raw sensor data over the network
- Image preparation (offset, rotation, skew)
- Cross-correlation methods to determine belt velocity
- Profinet interface to deliver processed data

Field Testing and Refinement

Eventually, a stage was reached where development on the prototype in a small-scale laboratory setting was no longer adequate. Development and testing were then continued on-site at a gravel quarry in order to validate laboratory results and further refine the software.

2 State of the Art

The conventional methods of measuring the mass or volume flow of bulk materials [1] are using so-called *belt scales* or *belt weighers*. These typically either employ the gravimetric method or nuclear method in order to determine the mass or volume flow of bulk materials.

As already detailed in chapter 1, gravimetric belt scales use load cells to transform the compression due to the weight of the belt, into electrical signals.

Nuclear belt scales [2] function principally by measuring gamma ray attenuation through the bulk material. While these type of scales have their advantages over the gravimetric conventional method, such as ease-of-installation and calibration, there are also other severe disadvantages. Most importantly, the handling of radioactive products must be carried out by certified personnel. Secondly, the chemical composition of the bulk material must also be homogeneous.

The interest in the implementation of **optical methods** for the purposes of measuring bulk material is not novel. The reasoning is clear: conventional methods are intrusive and costly. A non-contact, non-intrusive approach makes any sort of optical solution to the measurement problem very desirable.

As early as 1997, Green et al. [4] were already experimenting with non-contact methods to calculate mass flow rates. In that time, they resorted to using electrodynamic sensors. These electrodynamic sensors were used to estimate both velocity and concentration, which in turn were used to derive mass flow rates. They also used a cross-correlation method to determine material velocity. Although a far cry from the resolution afforded by contemporary sensors, Green et al. and their electrodynamic sensors demonstrated the potential of non-contact sensing for bulk materials.

In 2014, Fojtik [3] released his paper on using laser scanning to measure the volume of bulk material on a conveyor belt. Fojtik focused on the measurement of wood chips, which required special consideration to the volume fluctuations due to humidity.

Independently, Zeng et al. [9] too released their paper on the use of laser scanning for measuring the volume flow of bulk material. The focus of their paper was using these technologies to increase energy efficiency. In that paper, they claim that non-contact methods of measuring the volume flow of bulk materials increased energy efficiency by up to 30 % and reduced maintenance costs by up to 20 %.

Although they differed slightly in their precise approaches, both Fojtik and Zeng et al. used the same fundamental principle to determine volume flow, namely the derivation of the cross-sectional area of material based on the difference between an empty and laden belt. Both of them also are similar in their use of SICK LMS industrial laser scanners.

Both Min et al. in 2020 [5], and Qiao et al. in 2021 [7] too have published their analyses and results on solving this problem. They both take novel approaches, however, using not only laser scanning but a hybrid solution involving regular optical imaging to supplement the analysis of the material surface. They both also attempt to implement more advanced mathematical models, using 3D reconstruction and neural networks.

As shown by the papers above, this work is not novel in its use of optical methods to solve the problem of measuring bulk material volume flow. This project does set itself apart, however, by a) focusing on the use of commercially-available hardware b) being an all-in-one solution and not requiring any additional sensory information, such as belt velocity.

3 Design and Implementation

This chapter outlines and further describes in detail the design decisions behind the end-product of this project, as well as the specifics on the implementation of the project.

3.1 Principles of Operation

The analysis of volume flow can be broken down into two fundamental operations that must be carried out:

- A calculation of the **cross-sectional area** of a slice (or slices) of material
- A calculation of the **velocity** of the material flow

Cross-Sectional Area

The methodology used in order to analyze the cross-sectional area of the material flow is **geometric analysis**. Simply put, the geometry of a laden belt is compared with that of an empty belt. The resulting difference in area is that of the material itself.

As shown in Figure 3.1, the LIDAR sensor returns a 2-dimensional image with the value of each pixel representing depth data. This 2-dimensional image can then be separated into slices. A slice represents the depth data of a single dimension, in this case, the crosswise dimension of the belt.

During calibration, the empty belt is used to fit the polynomial belt curve $f(x)$. The fitting of this nth-degree polynomial is done with the least-squares method.

After calibration, the current slice curve $g(x)$ can be used to obtain the Cross-Sectional Area A_C as shown in Equation 3.1.1 and Figure 3.2.

$$A_C = \int_{x_a}^{x_b} [g(x) - f(x)]\, dx \tag{3.1.1}$$

Further Considerations

The accuracy of the computed cross-sectional area depends primarily on the accuracy of the depth data as well the frame rate of the sensor.

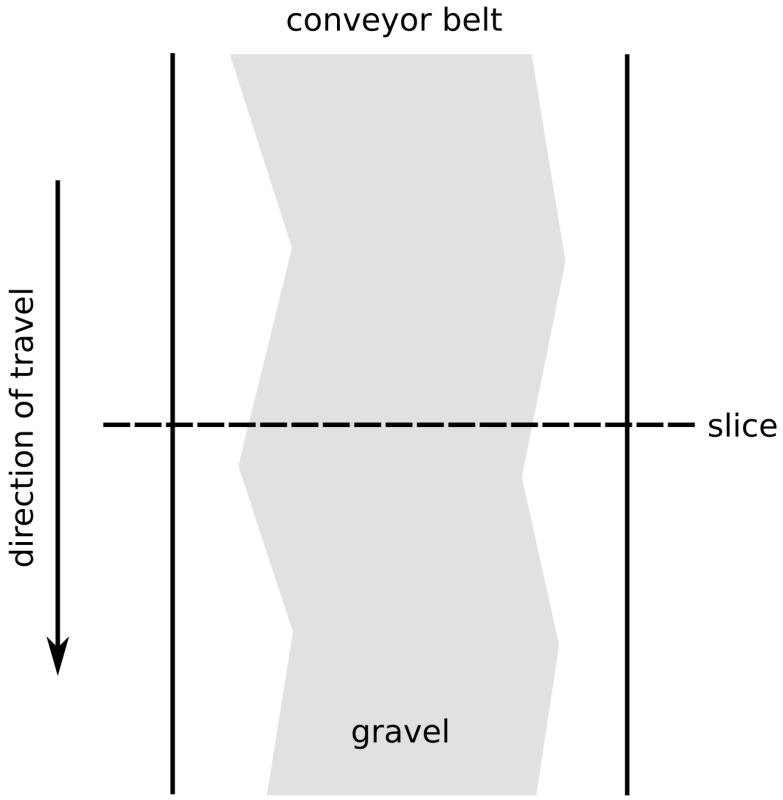


Figure 3.1: *Graphical depiction of the LIDAR sensor image. The slice is a one-dimensional extract of the sensor image crosswise over the belt.*

However, further operations may be implemented in order to increase accuracy, such as:

- Computing the cross-sectional area from multiple slices of each frame and averaging these
- Computing the average cross-sectional area between frames, in order to create a smoother—and possibly more accurate—estimation of the volume flow

It is important to note though, that the implementation of further operations may exhaust the processing capabilities of the platform. Therefore, a crucial balance must be struck between performance and accuracy.

Furthermore, this method of estimating the cross-sectional area does not take into account the warping of the belt when it is laden with material. This algorithm operates under the assumption that the error introduced by warping is negligible. This error can further be reduced by placing the sensor strategically over sections of the belt which are supported by struts. The ability to re-calibrate the belt curve $f(x)$ regularly will also help reducing this error.

Belt Velocity

Conventional belt scales use some form of a rotary encoder in order to measure the belt velocity. This is—however accurate—only an approximation of the velocity of the

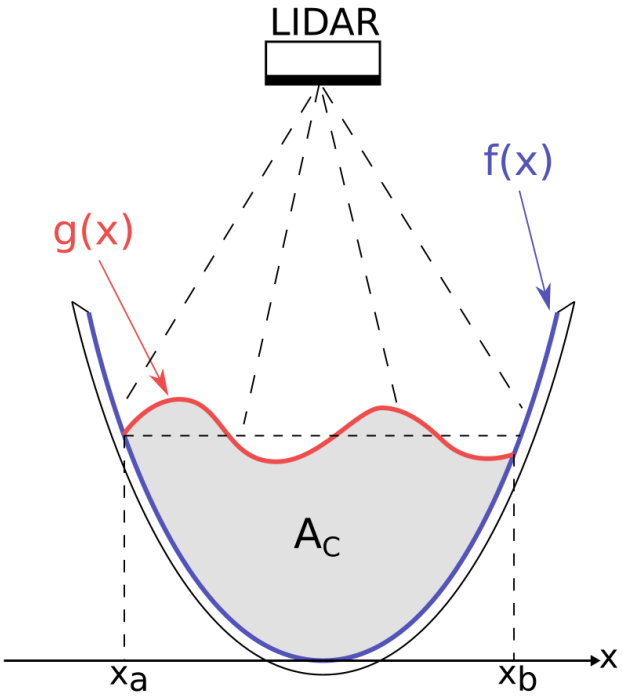


Figure 3.2: *Cross-section of material flow on a conveyor belt.*

material flow itself, since material velocity may deviate from belt velocity depending on environmental or material conditions.

For the purposes of this project, the term *belt velocity* will be used loosely to describe an indeterminate aggregate of belt, volume and surface velocities. This is because the techniques used in order to determine the velocity—as shown below—are not designed to isolate a single velocity type.

The fundamental operation being used in the following methods in order to determine the belt velocity is **cross-correlation**. In essence, a signal is compared to an older version of itself in order to determine a displacement—or in signal terms, a *delay*.

In the case of this project, given the known interval between two consecutive signals—i.e. the frame rate—it is possible to express this delay in the form of a physical displacement, in meters.

The various analytic methods used in this project differ only by which data is selected to represent the signal during cross-correlation. The algorithm of calculating the cross-correlation itself remains the same.

Equation 3.1.2 shows how the cross-correlation r between two signals A and B of lengths n may be calculated by multiplying each element i together. This is done for each possible delay d value. The maximum value of the correlation r corresponds to the most likely value of d.

$$r(d) = \sum_{i=1}^{n} A_i \cdot B_{i+d} \tag{3.1.2}$$

Chosen Method - Statistical Method

This method was developed as an aggregate of the previously attempted methods, improving on and solving issues earlier iterations had. It is therefore simply the most successful iteration.

The statistical method carries out the following operations:

1. A user-provided area of interest is cropped out of the entire sensor frame. This is done to isolate only the most relevant and data-dense regions, as well as to eliminate error from static elements as much as possible.

2. This subset of the frame is then divided into one-dimensional vertical strips.

3. For each of the strips, the cross-correlation displacement is calculated.

4. With the of displacement values for each strip, statistical outlier values are removed and a mean displacement is calculated.

5. This mean displacement in pixels, together with camera frame geometry, is used to calculate the physical displacement in meters.

The use of this statistical approach using multiple vertical strips—see Figure 3.3—is very similar to directly using 2-dimensional correlation, however it attempts to solve a significant problem with the 2-dimensional cross-correlation method.

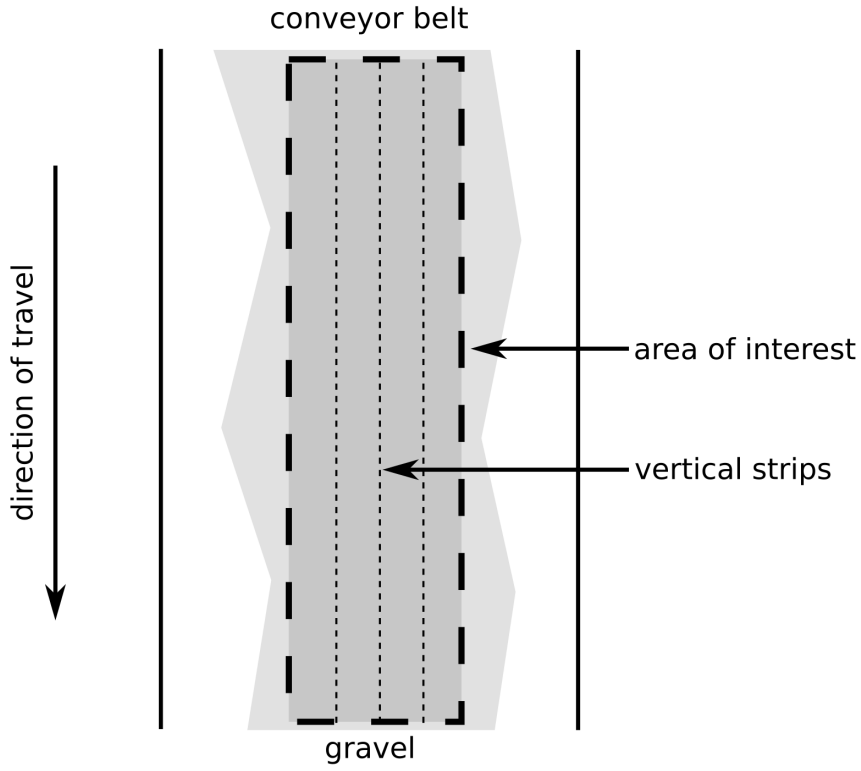


Figure 3.3: *Representation of the area of interest and vertical strips used for the cross-correlation operations.*

Since a 2-dimension cross-correlation would simultaneously consider the entire area of interest, any static elements in the frame would highly influence the results of the correlation, causing it to always be close to zero. This introduces a high error and variability in the result.

This statistical approach allows us to discard outlier values—such as values close to zero—and retain only those slices which do not contain any static elements.

Alternative Methods

Before arriving at the statistical method described above, multiple iterations of possible methods were tested. In this subsection, two of the most significant alternative methods are described.

Firstly, as already mentioned above, the **2-dimensional cross-correlation**. This method produces robust values and is less computationally complex than the statistical approach, however it is significantly more sensitive to static elements. This introduces many challenges since static elements may not be entirely avoided, either on the belt, or on the sensor itself.

The other alternative method, called the **topographical method**, is much less computationally expensive, since it only runs one cycle of the cross-correlation algorithm per frame.

The topographical approach works in the following manner. The values within each **crosswise** slice are summed. This reduces the 2-dimensional sensor data into a 1-dimensional representation which is called the *topography*. This topography can be used as the signal for cross-correlation.

Although this method was robust to signal noise, the uncertainty of the values was due to information being lost in the summation process.

Volume Flow

Once the cross-sectional area and velocity of the material have been determined, the only operation that remains is to derive from them the volume flow.

The volume of material that has passed the sensor per frame V_F, can be calculated simply as described in Equation 3.1.3. A_C is the cross-sectional area measured in a particular frame, v_m is the velocity of the material flow and f is simply the framerate of the sensor.

$$V_F = A_C \cdot v_m \cdot \frac{1}{f} \tag{3.1.3}$$

Accuracy

The accuracy of the system is limited primarily by the framerate of the sensor.

Figure 3.1.4 shows the theoretical maximum accuracy for a simple single line-scanner method of determining volume. Thus, for the targeted framerate of 30 FPS, the theoretical maximum accuracy of this method is limited to 3,3 %.

$$\text{Accuracy Upper Limit} = \frac{1}{f} \cdot 100\,\% \tag{3.1.4}$$

Introducing multiple line-scans per frame would proportionally reduce this upper limit, at the cost of increased computational complexity.

The various manufacturers of conventional belt scales have claims of accuracy between 0,5 % and 2 %.

3.2 Phases of Development

The following phases of development are not grouped chronologically over the span of the project schedule, rather into conceptual groups.

Preparing Development and Build Environment

Setting up the development environment for this project was a non-trivial task, due to a lack of pre-compiled binaries for the intended architecture, the ARM chipset of the Raspberry Pi.

The Raspberry Pi 4 Model B used in this project was delivered with a Quad Core Cortex-A72 64-bit SOC. However, a 32-bit kernel OS was used in this project, due to the netHAT drivers being delivered as 32-bit compiled binaries.

Intel RealSense SDK

The Intel RealSense SDK, or librealsense, is a cross-platform library provided by Intel for use with their RealSense devices. Pre-compiled binaries for 32-bit ARM were not provided, and therefore must be compiled [11].

ZeroMQ

ZeroMQ is a lightweight asynchronous messaging library that extends the traditional socket interfaces [16]. In this project, ZeroMQ was used in order to:

- Broadcast raw sensor data to remote controllers using a publish/subscribe model
- Exchange configuration parameters between the remote controller and the local processor using a request/reply model

In this project, TCP sockets were used for communications.

ZeroMQ was chosen to provide the aforementioned functionalities for the following reasons:

- Simple to implement
- Data can be transferred as binary data, instead of requiring serialization
- Availability on numerous platforms, as well as pre-compiled binaries

Real-Time Kernel Patch

Traditionally, the Linux kernel only allows one process to preempt another process in specific circumstances. This means, that even a high-priority thread may not preempt kernel code until the kernel explicitly yields control.

This is particularly disadvantageous for any operations requiring real-time performance. In order to circumvent this, the `CONFIG_PREEMPT_RT` Kernel Patch is used in order to allow kernel code to be preempted [15].

In the case of this project, this means that the local processor can process and deliver data in a more deterministic fashion.

GUI with Qt

The Qt GUI framework [14] was used in order to create a GUI for the remote controller. This allowed for the sensor data to be more easily calibrated and aligned, as well as providing a consistent interface for end-user configuration. Qt was chosen for its ease of use, as well as its ability to be compiled cross-platform.

Development of Main Functionality

At this stage of the design process, the functionality that is fundamental to the principle operation described earlier was developed. These functions include:

- Transmission of raw sensor data
- Calibration of sensor data
- Configuration for processing
- Transmission of configuration parameters

Testing and Validation

The project was largely developed in an iterative process with multiple cycles of testing and validation. This allowed more fundamental elements to be developed and debugged first, before building more complex elements on top.

The iterative cycles can be broken down into:

1. **Sandbox Testing:** At this stage, the most fundamental features were developed and tested, such as:

    a) Connection to LIDAR sensor and retrieval of data

    b) Representing the data visually

    c) Basic manipulation of the sensor data

2. **Laboratory Testing:** In the lab, the main features and functionalities of the program could be developed and tested on a more controlled, smaller scale. Among the main functionalities:

    a) Development of the processing algorithms, i.e. for cross-correlation, determining the cross-sectional area

    b) Processing of output data

    c) Establishing and transmitting output data over Profinet

    d) Tuning performance

3. **Field Testing:** At a certain point, development in the laboratory setting could no longer proceed without taking into consideration the environmental and full-scale factors of deployment in the field. Thus, the final iteration of development must be carried out on-site.

See chapter 4 for elaboration and results of each of the testing cycles.

3.3 Components

As already touched upon in section 1.2, the components used in this project were chosen mainly for their commercial availability and low cost. This section will elaborate more on the decision to select these specific components.

The cost for each of these components are listed at the end of this section in Table 3.4.

Raspberry Pi 4 Model B

The Raspberry Pi was chosen as the computation platform primarily for its widespread use in IoT and IIoT, low cost and commercial availability. It also supports the Linux

kernel and operating system which greatly eases the software development and deployment process.

As shown in Table 3.1, the Quad-Core ARM processor as well as 2 GB memory capacity provide ample performance for the intended computation. The wireless networking capability of the Raspberry Pi makes it an ideal candidate for an IoT product.

Processor	Quad Core Cortex-A72 (ARM v8) 64-Bit
Memory	2GB
Networking	2GHz and 5GHz 802.11ac Wireless Bluetooth 5.0 Gigabit Ethernet
Connectivity	2x USB 3.0 2x USB 2.0 40 pin GPIO header
Storage	Micro-SD card slot
Operational Temperature	0 °C to 50 °C

Table 3.1: *The relevant technical specifications of the Raspberry Pi 4 Model B used in this project [12].*

Intel RealSense L515

The Intel Realsense L515 was also chosen primarily for its low cost. However, the open-source and Linux-friendly nature of Intel's RealSense SDK also make it an ideal choice to pair with the Raspberry Pi. The small form-factor of the sensor would also allow a final product size that would be compact and easy to install.

Table 3.2 provides an overview of the specifications of the RealSense L515 sensor.

	Depth	Color
Resolution	Up to 1024x768	Up to 1920x1080
Field-of-View	70°x55°	69°x42°
Frame Rate	30 Frames per Second	
Depth Range	Up to 3,9 m at 15% Reflectivity Up to 9 m at 95% Reflectivity	
Depth Accuracy	5 mm at 1 m 14 mm at 9 m	
Laser Wavelength	860 nm Infrared	
Power Consumption	Up to 3,3 W	
Operational Temperature	0 °C to 30 °C	
Mounting Options	ISO1222 Tripod Mounting Point 2x M3 Mounting Point	

Table 3.2: *The relevant technical specifications of the Intel RealSense L515 used in this project [10].*

netHAT

The netHAT by Hilscher provides a simple-to-use Industrial Ethernet interface for the Raspberry Pi. Through the Raspberry Pi HAT standard, the netHAT is easily installed on the GPIO pins of the Raspberry Pi.

The drivers—called CIFX—and API library provided–called libCIFX—provide a simple way to interface with Industrial Ethernet networks from software. Section 3.5 shows how CIFX was integrated into the rest of the software architecture.

Table 3.3 gives an overview of the capabilities and specifications of the netHAT.

Processor	Hilscher netX 52
Memory	4MB Quad SPI Flash
Interface	SPI up to 125MHz
Network	2x Ethernet 100 BASE-TX

Table 3.3: *The relevant technical specifications of the Hilscher netHat [13].*

Cost Breakdown

Table 3.4 lists the individual costs of each of the components, and their total. This total is not reflective of the final cost of the completed product as it does not yet include costing for the housing, wiring and other installation costs.

Intel RealSense L515	€ 380[1]
Raspberry Pi Model 4 B	€ 84
netHAT	€ 69
Total	**€ 553[2]**

Table 3.4: *Breakdown of the costs used in the development of this project.*

3.4 Process Overview

With the objective of creating a marketable commercial product in mind, the process flow was designed for ease-of-use and ease-of-configuration for the end-user. This is the justification for implementing a remote controller that allows the setup to be remotely configured once installed.

While Figure 3.4 gives a brief overview of the interrelationship of the remote and local sides in the complete process, it is here further elaborated:

1. **Transmission of Raw Data:** Upon the first startup, the local processing software immediately begins to broadcast the raw sensor data. This data can then be subscribed to by a remote controller.

[1]The cost has since increased to € 570 as of January 2022.
[2]Due to a malfunctioning unit, a new sensor was purchased, bringing the total cost of development to around € 1120.

2. **Remote Configuration:** Once the remote devices have subscribed to the raw data broadcast, it will be previewed on the FlowRemote GUI interface. See Figure 3.7 for an overview of the FlowRemote interface.

    a) **Image Pre-Processing:** The engineer then pre-processes the image—rotating, skewing and cropping—until the conveyor belt appears vertically aligned and the perspective has been corrected.

    b) **Calibration and Fitting:** Now the engineer may select a single layer or slice of the sensor image which will be used to fit for the conveyor belt curve. The fitting parameters are selected, and the calibration is saved. This corresponds to the curve $f(x)$.

    c) **Belt Parameters:** In order to correctly determine the belt velocity and volume flow, the conveyor belt parameters such as visible length and width need to be provided.

3. **Transmission of Configuration Parameters:** The parameters that were configured in the previous step are then transmitted back to the local processor, FlowPi, and local processing mode is then engaged.

4. **Local Processing:** In local processing mode, raw data is no longer transmitted for performance purposes. The sensor data is directly processed on the Raspberry Pi, and the outputs are delivered over Profinet to the PLC.

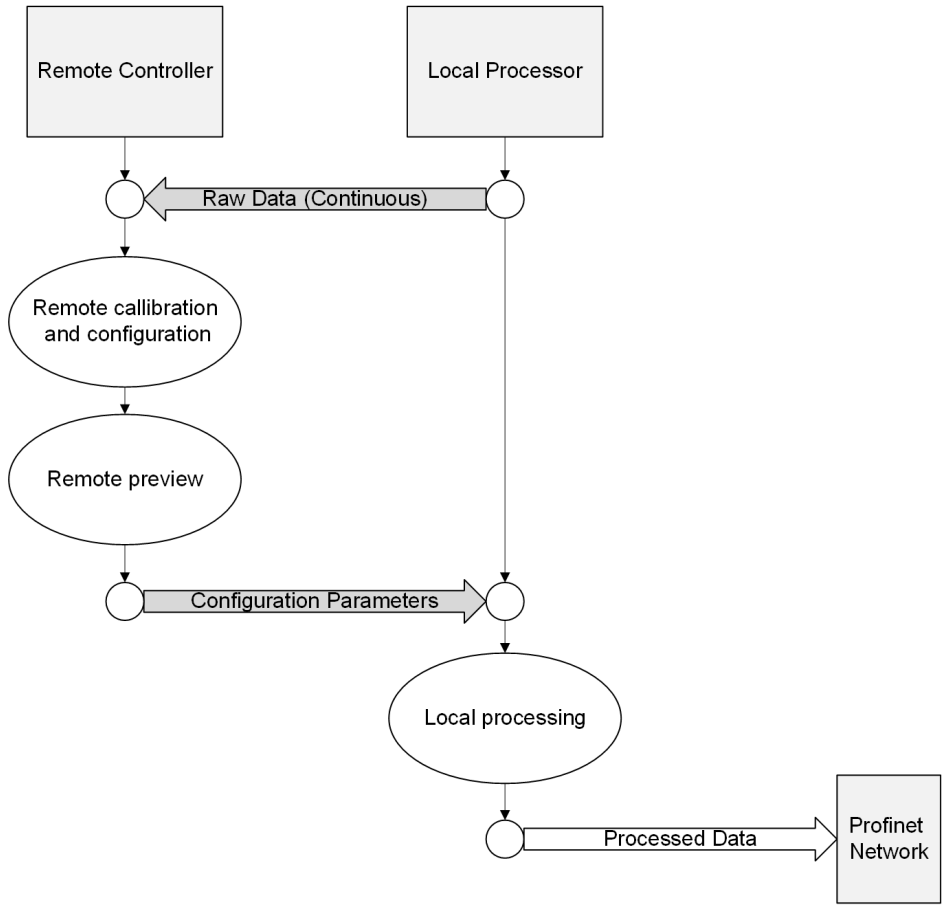


Figure 3.4: *Overview of the communication and processing process between the remote controller and the local processor.*

3.5 Software Architecture

The software architecture developed in this project—see Figure 3.5—consists of two separate but tightly interconnected parts, namely:

- **FlowPi:** The local processing software that runs on the Raspberry Pi, and
- **FlowRemote:** The remote control software that is meant to run on an external PC for configuration purposes

Development Language Choice

The software is written in `C++` for compatibility and performance reasons. All the device drivers provide libraries in either `C` or `C++`, while some drivers such as the library for the netHAT—called CIFX—are only provided in `C`.

The topic of performance between languages and systems is one of heated debate, however `C++` was chosen for this project due to the ability to program comfortably in a higher-level language, while having the ability to *"drop down"* into `C`. The `C` Programming Language is often the benchmark for higher-level programming languages

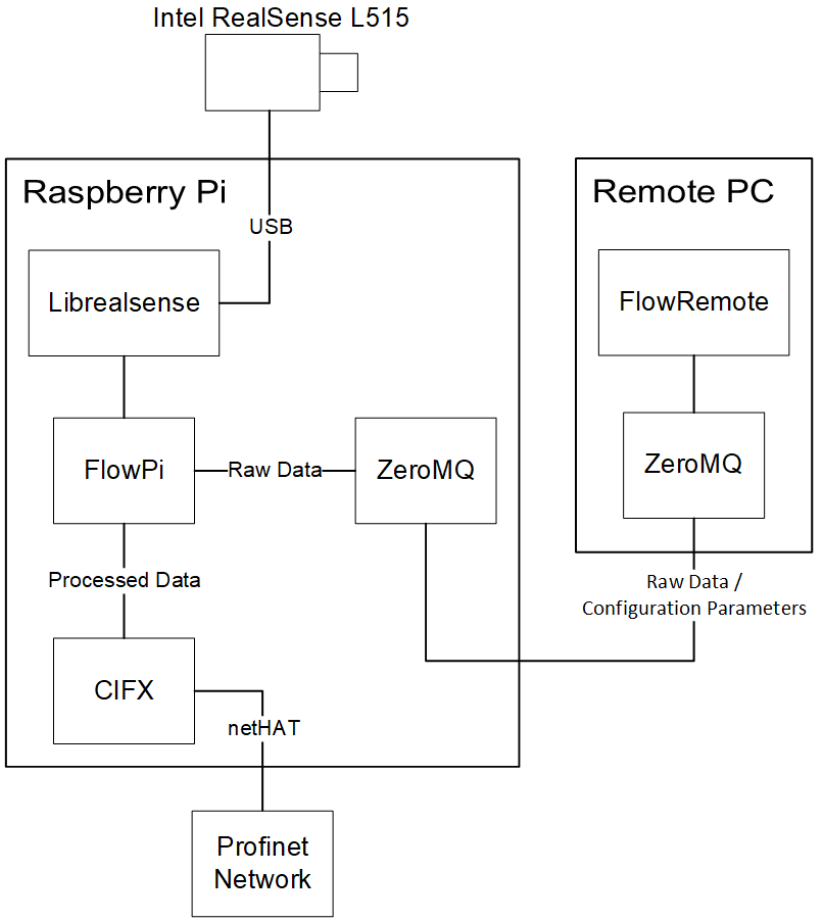


Figure 3.5: *Overview of the interactions between the various software components and their communication.*

when programming for Real-Time Systems due to its predictability and the ability to run operations with few layers of abstraction on memory directly [6].

Furthermore, since the scale of the processing unit of the program is relatively small, the benefits that come from using a higher-level programming language—such as increased productivity, organization, and re-usability [6]—are not strictly necessary.

As shown in Figure 3.6, the main functionality of the processing unit includes:

- Image Pre-Processing i.e. rotation, skew, cropping
- Curve-Fitting
- Cross-Correlation

These are implemented in `C` as much as possible. This is then encapsulated by a `C++` wrapper. This provides ease-of-use on the remote side, where processing is not real-time critical, while still allowing the local side to directly call the `C` processing functions.

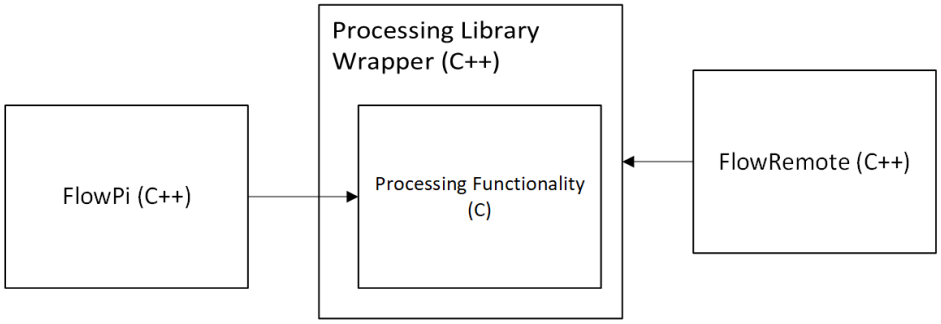


Figure 3.6: *Representation of the how the processing unit is called by different components of the program.*

FlowRemote – Remote Control GUI

FlowRemote is designed in order to allow for easier configuration and calibration of the setup, as well as enabling the engineer to do so remotely. The idea being that—once the Raspberry Pi and LIDAR sensor have been installed over a conveyor system and a network connection—the engineer no longer requires a direct physical connection to the Raspberry Pi in order to configure and calibrate the system. Figure 3.8 shows the design of the GUI.

As described in Figure 3.7, FlowRemote allows the engineer to remotely preview the raw sensor data, run pre-processing on it, configure the processing parameters and deliver those back to the local processor running on the Raspberry Pi.

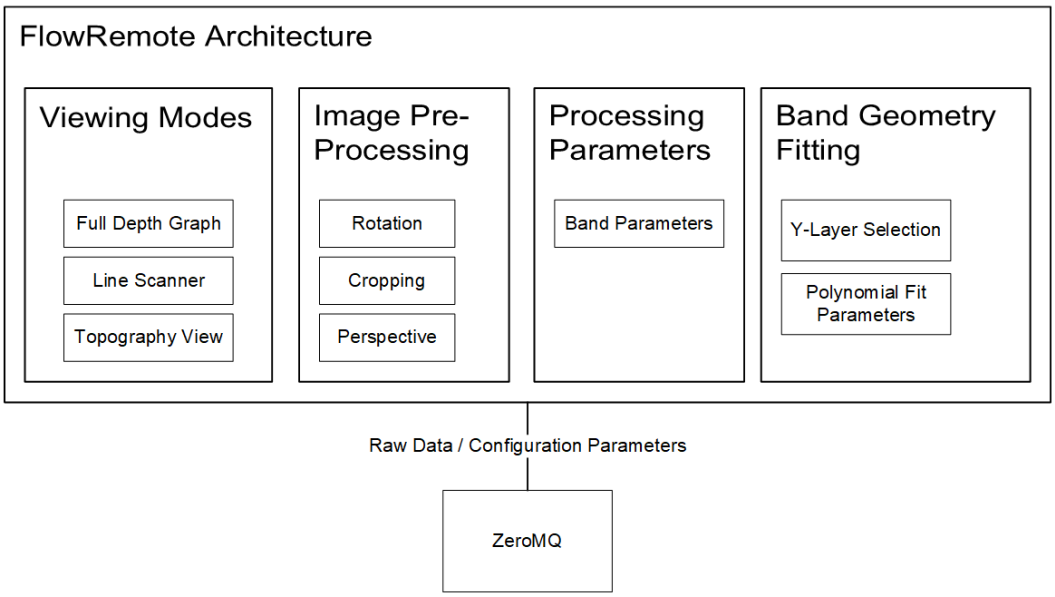


Figure 3.7: *Overview of the FlowRemote features and configurable parameters.*

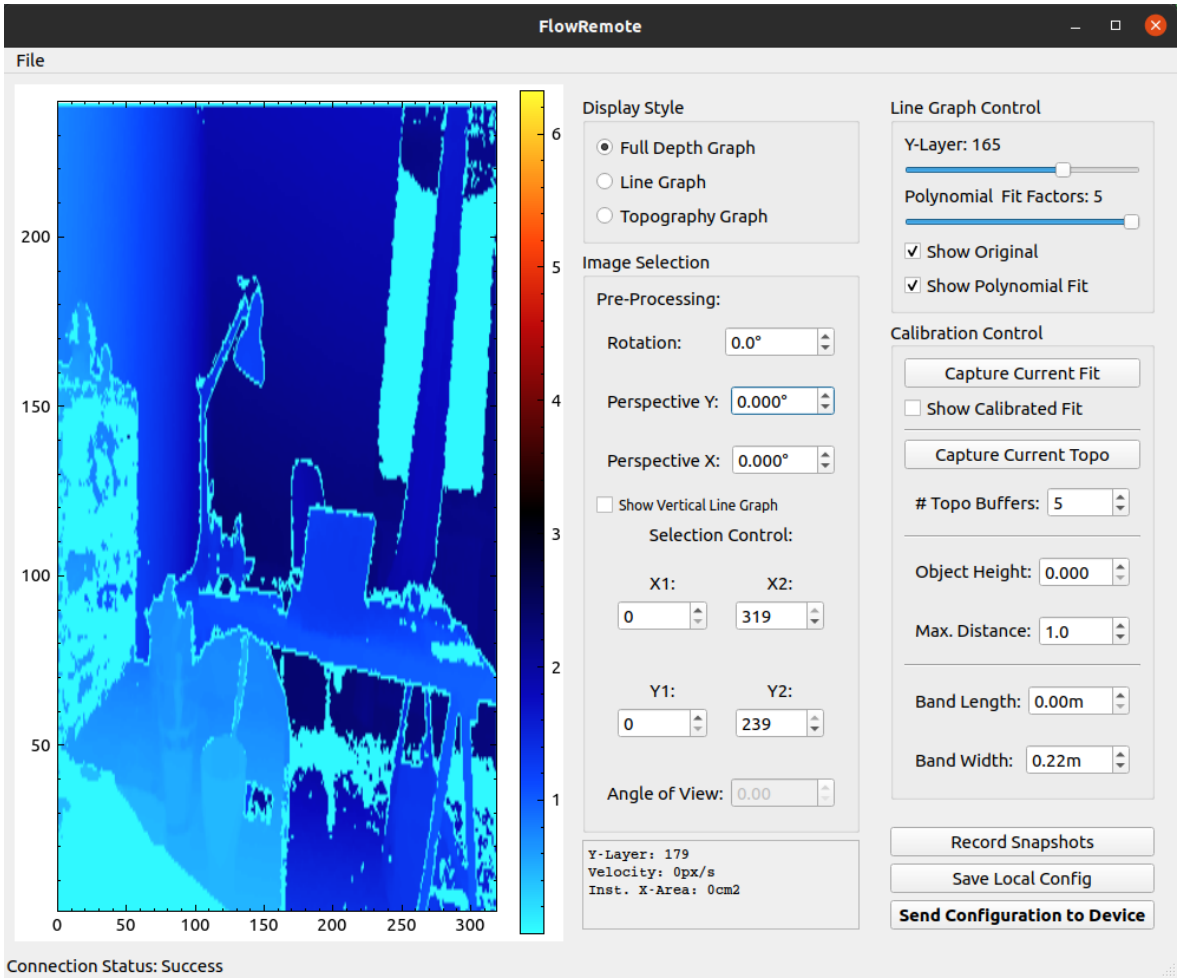


Figure 3.8: *Design of the FlowRemote GUI.*

3.6 Housing

For the purposes of field-testing the project, a rudimentary housing was designed in CAD—see Figure 3.9—and 3D printed. The housing provided a small amount of protection from the environment for the otherwise bare Raspberry Pi.

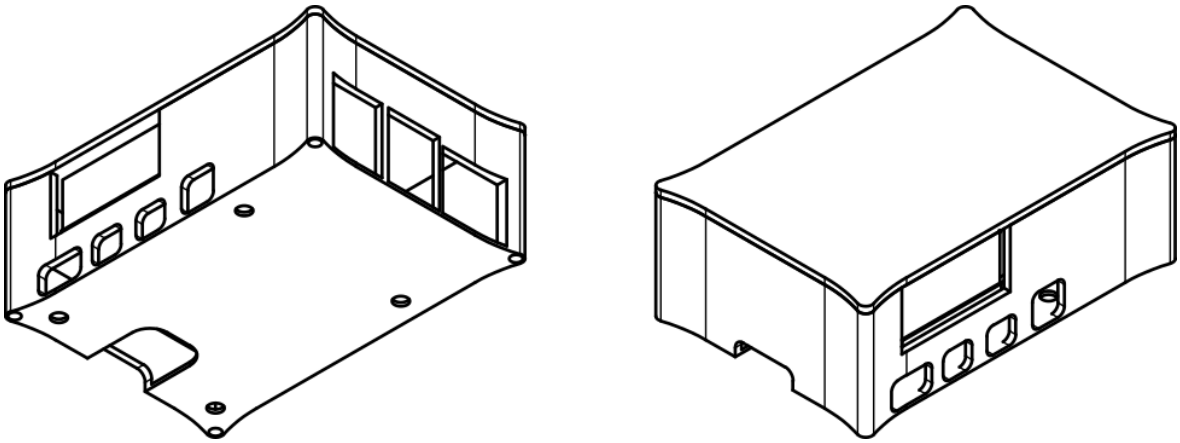


Figure 3.9: *Isometric view of the underside (left) and topside (right) of the prototype housing.*

The housing was constructed around the standard Raspberry Pi 4 Model B with the

netHAT modules attached, allowing for the extra ports to be accessible through the housing as well.

This housing is naturally unsuitable for production use, as it lacks adequate weather and shock-proofing.

4 Validation

As was briefly touched upon in section 3.2, the development of this project was carried out in iterative phases. After each iteration, the developed features were validated for their functionality and suitability. This chapter will go over these stages—sandbox, laboratory and field-testing—in depth.

4.1 Sandbox Stage

The fundamental objective of the sandbox stage of development was to investigate the suitability of the Intel RealSense L515 LIDAR sensor.

In this stage, the testing was mainly to determine

1. if a connection to the sensor can be established
2. if the data received can be visually represented and manipulated—see Figure 4.1
3. if the accuracy of the data was within a tolerable range

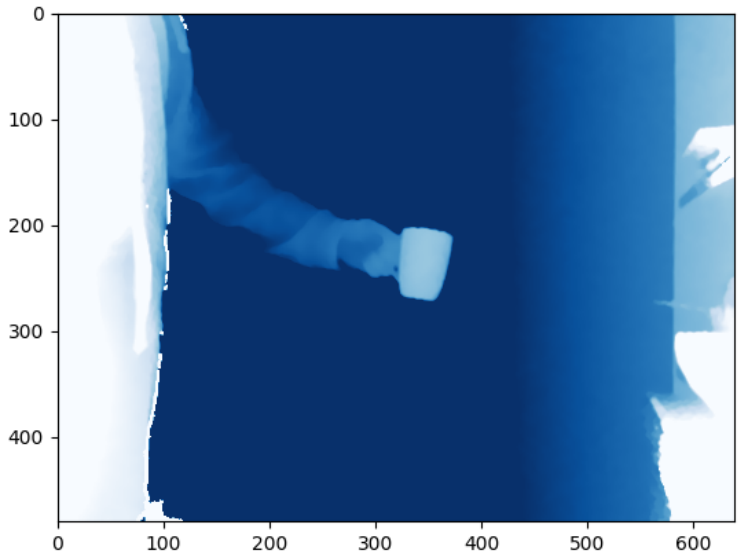


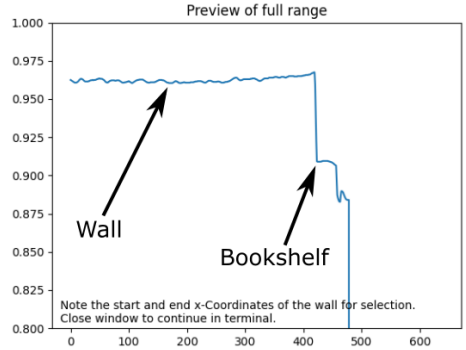
Figure 4.1: *Visual representation of the sensor data plotted as a color-graph.*

In order to accomplish this, a precursor to the FlowDAR software was developed that was connected to the sensor directly over USB. The aim of the software was to process

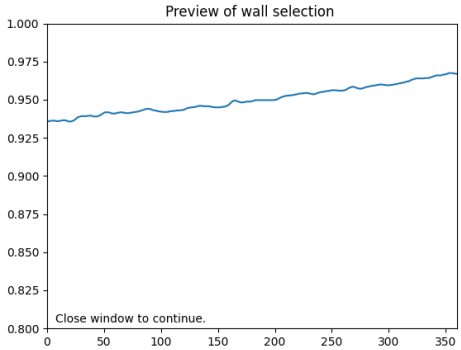
the raw sensor data in order to determine the cross-sectional area of an object upon a flat plane. In this case, a small cardboard box was placed against a wall.

As represented in Figure 4.2, the software first calibrates itself to the flat plane—the wall—using linear regression to generate a straight-line. Then upon placing the object on the plane, using the techniques discussed in chapter 3, the cross-sectional area of the object could be measured.

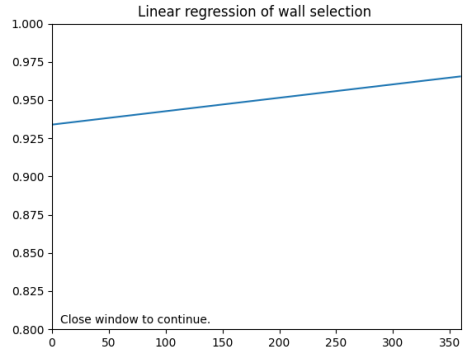
An object with the cross-sectional area of 3150 mm^2 was used in this validation. The software measured 3288 mm^2, yielding an error of 4 %.



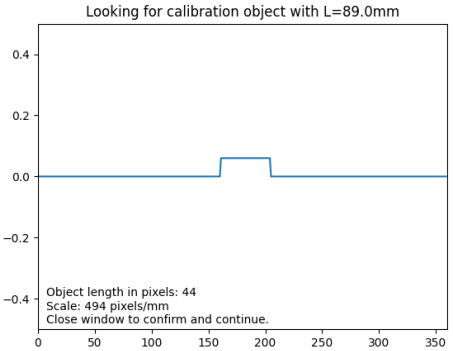
(a) *Full scanning range of the LIDAR sensor, recording the geometry of a wall and bookshelf.*



(b) *A zoomed in selection of just the straight section of wall.*



(c) *The selection is then used to run a regression algorithm, to produce a straight line that should represent the wall's surface.*



(d) *Calibrating the sensor using an object of known length.*

Figure 4.2: *Operation of the prototype software.*

4.2 Laboratory Stage

The development of the fundamental features of this project was done in an iterative process of rapid prototyping and testing. In order to accomplish this, a controlled environment that can be easily accessed and modified must be established. This setup was realized in the Telelaboratory[1].

[1]Laboratory for the development of remote systems at the Faculty of Electrical Engineering, University of Applied Sciences Düsseldorf

The setup consisted of a miniature looped conveyor belt system. The looped nature of this conveyor system was advantageous, as it could be loaded with material that would continuously circulate. This allowed development to be carry on uninterrupted and even remotely if necessary. The objects used to simulate material on the belt were miniature cars that were chosen simply for their availability and simple geometry.

The LIDAR sensor was mounted using its ISO 1222 tripod mounting point on a regular camera tripod and positioned over the conveyor belt. The Raspberry Pi was connected to the laboratory network which allowed for configuration and testing to be done over the network. Using a VPN tunnel, further configuration and testing could also be done remotely from outside the laboratory network.

Cross-Sectional Area

The development and testing of operations to measure Cross-Sectional Area were straightforward as described in chapter 3.

Object Cross-Sectional Area	Uncertainty
$20\,\mathrm{mm}^2$	$10\,\%$
$120\,\mathrm{mm}^2$	$4\,\%$

Table 4.1: *Uncertainty of Cross-Sectional Area measurement for different sized objects.*

As shown in Table 4.1, measurements of the miniature cars—with cross-sectional areas of $20\,\mathrm{mm}^2$—had a relatively high uncertainty of around $10\,\%$. The uncertainty was reduced to $4\,\%$ when using a cardboard box of a larger size. This however, is to be expected according to the specified uncertainty of the LIDAR sensor at $1\,\mathrm{m}$.

Belt Velocity

In order to determine the accuracy and reliability of the cross-correlation algorithms, they were compared to the known velocity of the belt. This was determined simply by recording the time taken for an object to travel a known length of the belt and was determined to be $11\,\mathrm{cm\,s^{-1}}$.

While the algorithm did return a value of $(11 \pm 1)\,\mathrm{cm\,s^{-1}}$ while the object was in the center of the area-of-interest, the error varied up $100\,\%$ while the object was near the edges of the area-of-interest. This rendered the results of the tests inconclusive. One reason could be due to the small size of the objects—height up to $5\,\mathrm{mm}$—causing strong deviations near the edges, where the sensor uncertainty is usually at its highest.

Since this error may be caused simply by the small-scale laboratory conditions, the algorithm was brought forward to be tested in field-conditions with little change.

PLC Test

Once the development of the Profinet interface was complete, a simple test project involving a PLC was designed in order to test the interface's functionality. The ex-

isting setup was connected via the netHAT's Ethernet interface to a PhoenixContact PLCNext controller. The PLCNext controller was wired to an I/O kit that provided various actuators and LED outputs for prototyping purposes.

The values sent to the controller by the FlowPi over the Profinet interface were the following:

- Cross-Sectional Area
- Band Velocity
- Volume Flow

A simple PLC program was written to activate an output—in this case, turning on an LED—whenever the Cross-Sectional Area was over a certain threshold value.

The FlowPi software as well as the Profinet interface were shown to be functioning as the LEDs lit up in a robust manner whenever a miniature car passed under the scanning area of the LIDAR sensor. A rigorous measurement of the latency was not carried out, however the latency was deemed to be under one second.

Linux RT-Patch

In order to test the effect of the Linux RT-Patch, a simple test comparing the jitter values of Profinet-IO communications was conducted.

The system was connected over Profinet to a Virtual PLC running on Codesys, and the maximum jitter values as reported by Codesys were recorded. During the 30 s duration of the test, the cross-section area was processed and delivered.

The results show that the RT-Patched kernel had a maximum jitter of 2166 µs, which was 26 % lower than the normal kernel. This lower jitter may be indicative of higher-determinism of the system.

4.3 Field-Testing Stage

The field-testing stage was carried out at the Siep Gravel Quarry[2]—see Figure 4.3. There, a bucket loader was being used to excavate gravel into a hopper. The hopper first filtered out larger rocks and boulders through a set of evenly spaced rods. Acting as a buffer, the hopper would continuously load a conveyor belt with gravel.

The quarry currently uses a conventional belt scale system that was placed under the conveyor belt. This simultaneously measured both belt velocity and the material mass flow—in tonnes per hour—delivering the values to a PLC in a nearby control box.

[2]Siep Kieswerk GmbH & Co. KG in Jülich. See Acknowledgments.

Figure 4.3: *Siep Kieswerk GmbH & Co. KG in Jülich where the field testing was carried out.*

Setup and Testing

The LIDAR sensor was attached to a walkway that went over the conveyor belt—see Figure 4.4. The sensor must be placed at a minimum distance of 0,5 m from the belt, in addition to clearance accounting for the height of the gravel on the belt as well. In this case, the sensor was placed at a height of 1 m from the belt. The housed Raspberry Pi and various connections were also attached to the walkway.

Figure 4.4: *The view of the conveyor belt and material as seen by the sensor.*

A standard home-grade wireless access point was used to provide a local network, through which the configuration of the system could take place.

Once all the devices were connected and turned on, the pre-configured Raspberry Pi connected itself to the wireless access point that was reachable by the engineering laptop. A connection between the FlowRemote configuration software and the FlowPi processing software could be established and configuration could begin.

Configuration took place in the regular manner that was refined during the laboratory testing.

Issues

Upon the establishment of the connection and the preview of the sensor's data, it was immediately apparent that parts of the sensor image that contained the distance of the conveyor belt was zero. Gravel and other objects placed on the belt had regular and non-zero depth values. This meant that the conveyor belt not being "seen" by the sensor. In other words, the reflectivity of the conveyor belt near the wavelength of the sensor—860 nm—was too low to consistently and accurately calculate depth.

Solutions

This issue meant that the fundamental principles in which the operation of the measurement software depended on, was unusable. A new approach must be investigated by finding a method to obtain the geometry of the conveyor belt, and using a modified algorithm to calculate its volume without continuous access to the conveyor belt's geometry. In essence, the gravel will appear to be "floating" in the sensor image. Such an algorithm may use more precise measurements of the conveyor belt during the configuration and calibration phase, in order to make estimates of the conveyor belt geometry during live processing.

Ultimately however, the implementation of a solution to this issue requires another iteration of development that is not within the scope of this project.

5 Conclusion and Outlook

The state of this project upon completion can be analyzed by recalling the research question and requirements set out in chapter 1. The research question being: *How can a cheaper and easier to install measurement system for bulk material flow on a conveyor belt be designed?*

A breakdown of the various factors that determine the suitability of the implementation presented in this project:

Sensor Suitability

The wavelength of the infrared laser used in this project of 860 nm was shown to be unsuitable for use with the conveyor belt during the on-site testing. This is most likely due to the absorption spectrum of the belt material that had very low reflectivity at this infrared wavelength. The similarly black colored belt used in laboratory testing however was visible to the LIDAR sensor. A further study of belt materials commonly deployed in the field is necessary.

The Intel RealSense L515 Sensor was designed for indoor use and therefore has no vibration certification or waterproofing certification. Either a housing must be designed to adequately protect sensor, or another sensor with appropriate ratings must be used instead.

Temperature Suitability

On the higher end of the temperature range, the LIDAR sensor used in this project is the limiting factor. The maximum temperature of 30 °C is easily exceeded in particularly hot weather or even in direct sunlight. Design of the housing must account for adequate cooling, as well as reflectivity, should the system be deployed in view of direct sunlight.

Hardware Suitability

The Raspberry Pi provided sufficient processing power in order to develop, test and deploy the prototype software. The flexibility of the Linux platform also grants sufficient flexibility in order to easily add further functionality—i.e. a web server or other interface—or modify existing functionality.

The netHAT was also shown to be performant and stable during testing. Combined with the Raspberry Pi, it provides a low cost platform to bring IoT to Industrial Networking.

Process Suitability

The process flow which was developed with ease-of-use in mind was shown to be beneficial during both laboratory and field-testing. Once the sensor and Pi units were set up on the belt, the engineer could continue working on the configuration

of the system remotely—or even externally through the use of a VPN.

Software Suitability
The software architecture as laid-out in chapter 3 was shown to fulfill the requirements of the process. The individual software elements and libraries such as ZeroMQ and Qt were shown to be robust enough to carry out the configuration process. The software was shown to successfully combine the separate process elements into a single workflow which simplified the configuration process.

Cost Suitability
At a development cost of just under € 600—even at a profit margin of 500 %—the system is still able to remain competitive with conventional systems in use in the industry today[1].

Housing Suitability
The housing designed for the field-testing stage of this project is only suitable as a prototype. A more robust housing must be developed out of more durable materials, and account for weather and vibration.

5.1 Project Status and Feasibility

Although a commercially viable product was not realized during the duration of this project, great strides were nevertheless made towards this goal.

This work has shown that commercially available products do have the necessary performance to carry out the operations required to deliver the intended results.

It has also shown, that the software development methods used, methods that are more common in commercial software development rather than industrial development, are highly flexible and powerful. This was naturally enabled by the availability of the Linux kernel on smaller and more performant platforms. This work shows the strong potential of merging the industrial world with that of more conventional software development, also known Industrial IoT.

Discounting the setbacks faced during field-testing due to sensor and material issues, laboratory testing has shown that a fully-functioning result of this project is in theory, feasible.

5.2 Work Remaining

One or two more iterations of development are required in order to fully realize this project. The fundamental operations have already been developed and tested, namely the analysis of the cross-sectional area and belt velocity.

The issues at this stage are only that of signal acquisition and signal pre-processing.

[1] See Table 3.4

The field-testing has shown that the expectation of the signal was slightly different from reality due to the optical properties of the conveyor belt. New methods and operations need to be developed to circumvent these issues.

Once these signal issues have been overcome, all that remains is testing the system for accuracy, stability and robustness. For example, stability of the measurements may be impacted by vibration. Therefore, suitable software additions must be made to filter out such vibrations, should they introduce significant error. Future work must deal with these questions of environment-proofing and housing.

In order to study the commercial viability of this product to its end, future work must also investigate the potential sourcing and supply chains of the hardware used. As mentioned earlier in this work, the RealSense L515 has been discontinued, and other suitable hardware must be sourced and integrated.

Bibliography

Academic References

[1] *A Discussion on the Opportunities and Implementation of LIDAR-based Volumetric Analysis for Industrial Applications*. In collab. with Michael Protogerakis. Wintersemester 2021/22.

[2] E. Elias, W. Pieters, and Z. Yom-tov. "Accuracy and Performance Analysis of a Nuclear Belt Weigher". In: *Nuclear Instruments and Methods* 178.1 (Dec. 1, 1980), pp. 109–115. ISSN: 0029-554X. DOI: 10.1016/0029-554X(80)90863-0.

[3] David Fojtík. "Measurement of the Volume of Material on the Conveyor Belt Measuring of the Volume of Wood Chips during Transport on the Conveyor Belt Using a Laser Scanning". In: *Proceedings of the 2014 15th International Carpathian Control Conference (ICCC)*. May 2014, pp. 121–124. DOI: 10.1109/CarpathianCC.2014.6843581.

[4] R. G. Green et al. "Velocity and Mass Flow Rate Profiles of Dry Powders in a Gravity Drop Conveyor Using an Electrodynamic Tomography System". In: *Measurement Science and Technology* 8.4 (Apr. 1997), pp. 429–436. ISSN: 0957-0233. DOI: 10.1088/0957-0233/8/4/010.

[5] Fusong Min, Andong Lou, and Qun Wei. "Design and Experiment of Dynamic Measurement Method for Bulk Material of Large Volume Belt Conveyor Based on Laser Triangulation Method". In: *IOP Conference Series: Materials Science and Engineering* 735.1 (Jan. 2020), p. 012029. ISSN: 1757-899X. DOI: 10.1088/1757-899X/735/1/012029.

[6] Filip Pizlo et al. "High-Level Programming of Embedded Hard Real-Time Devices". In: *Proceedings of the 5th European Conference on Computer Systems*. EuroSys '10. New York, NY, USA: Association for Computing Machinery, Apr. 13, 2010, pp. 69–82. ISBN: 978-1-60558-577-2. DOI: 10.1145/1755913.1755922.

[7] Wei Qiao et al. "Dual-Field Measurement System for Real-Time Material Flow on Conveyor Belt". In: *Flow Measurement and Instrumentation* 83 (Mar. 1, 2022), p. 102082. ISSN: 0955-5986. DOI: 10.1016/j.flowmeasinst.2021.102082.

[8] Yuki Tomobe et al. "Continuous Mass Measurement on Conveyor Belt". In: *IEEJ Transactions on Electronics, Information and Systems* 126 (Jan. 1, 2006), pp. 264–269. ISSN: 0385-4221. DOI: 10.1541/ieejeiss.126.264.

[9] Fei Zeng et al. "Measurement of Bulk Material Flow Based on Laser Scanning Technology for the Energy Efficiency Improvement of Belt Conveyors". In: *Measurement* 75 (Nov. 1, 2015), pp. 230–243. ISSN: 0263-2241. DOI: 10.1016/j.measurement.2015.05.041. URL: https://www.sciencedirect.com/science/article/pii/S0263224115003061 (visited on 01/27/2022).

Online References

[10] *Intel RealSense LiDAR Camera L515 Datasheet*. Jan. 2021.

[11] *IntelRealSense/Librealsense*. Intel® RealSense™, Jan. 6, 2022. URL: https://github.com/IntelRealSense/librealsense (visited on 01/06/2022).

[12] Raspberry Pi Ltd. *Raspberry Pi 4 Model B Specifications*. Raspberry Pi. URL: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (visited on 01/19/2022).

[13] *netHAT*. Hilscher Gesellschaft für Systemautomation mbH. Aug. 16, 2017. URL: https://www.hilscher.com (visited on 01/20/2022).

[14] *Qt | Cross-platform Software Development for Embedded & Desktop*. URL: https://www.qt.io (visited on 01/14/2022).

[15] *Real-Time Linux Wiki*. URL: https://rt.wiki.kernel.org/index.php/Main_Page (visited on 01/06/2022).

[16] *ZeroMQ*. The ZeroMQ project, Jan. 5, 2022. URL: https://github.com/zeromq/libzmq (visited on 01/06/2022).