FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Analysis of a ToF Sensor for Applications in Touchless Interfaces

Maria Inês Cirne Saldanha Durão

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof. Armando Jorge Miranda de Sousa

Second Supervisor: Engenheiro Carlos Rebelo (3Decide)

August 21, 2022

Resumo

Os sensores de tempo de vôo (*Time-of-Flight*, ToF) tornaram-se parte integral em aplicações modernas de deteção de objetos e prevenção de colisão. A sua capacidade de fornecer informação de distâncias precisas impactou fortemente os sistemas de deteção de objetos. Este novo tipo de sensor é um grande avanço em relação aos sensores de distância já disponíveis no mercado, devido ao seu baixo custo e alta precisão de medição da distância ao objeto. A 3Decide (empresa que propôs o desafio), já possui um produto funcional utilizando o sensor VL53L0X - um quiosque interativo com múltiplos sensores, destinado a ser utilizado em espaços públicos, onde a transmissão do COVID-19 é mais provável, como museus. Entretanto, uma nova versão do sensor (VL53L1X) foi lançada no mercado. Serão apresentadas as características de ambos os sensores (pois o objetivo da dissertação é estudar a nova versão do sensor e ver se é possível adicionar novas funcionalidades ou melhorias ao seu produto), analisadas (teoricamente e na prática) e comparadas para retirar conclusões. Para isso, foi desenvolvido um software para manusear os sensores em JavaScript e Typescript. Concluída esta etapa, ambas versões foram testadas em diferentes condições para estudar o seu comportamento e tentar encontrar uma solução para melhorar das medições a precisão no quiosque interativo da 3Decide, onde se utiliza acrílico ou vidro em frente ao sensor. Anteriormente, o VL53L0X só podia medir até cerca de 30 cm com um vidro ou acrílico (superfície translúcida) em frente. No final, verificou-se que a melhor solução para o VL53L0X é retirar o seu *cover glass*, atingindo os 1,4m. Quanto ao VL53L1X, a melhor solução é retirar o seu *cover glass* e cortar a superfície translúcida, colocando a divisão do corte entre o receptor e o transmissor do sensor. Desta forma, o sensor é capaz de medir até 2m. Embora a nova versão possa medir até uma distância maior, a versão antiga mostrou ser mais precisa e menos sensível na presença de uma superfície translúcida. É, então, recomendado o uso do sensor VL53L0X quando o objetivo é realizar medições de curta distância (até 1,4m) com uma superfície translúcida em frente. O sensor VL53L1X apenas deve ser usado quando a distância de medição pretendida for superior, podendo apenas medir com precisão até aos 2m.

Abstract

Time-of-Flight (ToF) sensors have become an integral part of modern-day object detection and collision prevention. Their ability to provide accurate distance information has greatly impacted the field of object detection. This new type of sensor is a great advancement compared to the distance sensors already available in the market, due to its low cost and high precision of measurement of the distance to the object. 3Decide (the company that proposed the challenge), has already a functional product using the VL53L0X sensor - an interactive kiosk with multiple sensors, meant to be used in public spaces, where the transmission of COVID-19 is more likely to happen, such as museums. In the meantime, a new version of the sensor (VL53L1X) was launched into the market. The characteristics of both of the sensors will be presented (as the point of the dissertation is to study the new sensor version and see if it can possibly add new functionalities or improvements to their product), analyzed (both theoretically and in practice), and compared in order to withdraw the conclusions. In order to do this, software to manage the sensors in JavaScript and Typescript was developed. After this step was complete, both versions were tested in different conditions to study their behavior and to try and find a solution to improve the measurement accuracy in 3Decide's interactive kiosk, where acrylic or glass is used in front of the sensor. Previously, VL53L0X could only measure up to around 30cm with a glass or acrylic (translucent surface) in front. In the end, it was found that the best solution for VL53L0X is to remove its cover glass, reaching at least 1.4m. As for VL53L1X, the best solution is to remove its cover glass and slice the translucent surface, placing the division of the cut in between the receiver and transmitter of the sensor. This way, the sensor is able to measure up to 2m. Even though the new version can measure until a higher distance, the old version has shown to be more accurate and less sensitive in the presence of a translucent surface. It is, therefore, recommended to use the VL53L0X sensor when the aim is to carry out short distance measurements (up to 1.4m) with a translucent surface in front. The VL53L1X sensor should only be used when the desired measurement distance is greater and can only measure accurately up to 2m.

Agradecimentos

Antes de mais, agradeço ao meu orientador, Armando Jorge Miranda de Sousa, e ao proponente e CEO da 3Decide, Carlos Rebelo, por terem acreditado em mim desde o início, por me guiarem no desenvolvimento deste processo e pela disponibilidade.

Um especial agradecimento à minha família pelo apoio e paciência ao longo da realização desta dissertação, em especial aos meus irmãos: ao Luis, pela partilha de experiência e conhecimento ao longo destes 5 anos de formação e pela sua prontidão em ajudar sempre que necessário; e ao José Pedro, pela ajuda na realização dos testes de performance do sensor.

Aos meus colegas e amigos, em particular Margarida Pias, Pedro Martins, Diogo Remião e Tiago Colaço, estou eternamente agradecida por me acompanharem e encorajarem sempre, principalmente nos momentos mais difíceis.

Esta dissertação foi muito desafiante e a sua realização não seria possível sem o apoio incondicional que recebi. A todos que direta ou indiretamente me ajudaram nesta etapa, o meu grande obrigada.

Maria Inês Cirne Saldanha Durão

"Everything is theoretically impossible
until it's done."

Robert A. Heinlein

Contents

List of Figures

List of Tables

Abbreviations and Symbols

API	Application Programming Interface
ASIC	Application-Specific Integrated Circuit
BSI	Backside Illuminated Pixel
CAPD	Current Assisted Photonic Demodulator
CLI	Command-line Interface
cps	Counts Per Second
CSV	Comma-Separated Values
FoV	Field of View
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input Output
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
IMP	Inter-Measurement Period
IR	Infrared Radiation
JS	JavaScript
LED	Light Emitting Diode
NVM	Non Volatile Memory
MCU	MicroController Unit
ROI	Region of Interest
SBC	Single-board Computer
SBS	System Breakdown Structure
SPAD	Single-Photon Avalanche Diode
TB	Timing Budget
ToF	Time-of-Flight
TS	Translucent Surface
UI	User Interface
VCSEL	Vertical-Cavity Surface-Emitting Laser
xcd	Crosstalk calibration distance
xtalk	Crosstalk

Chapter 1

Introduction

1.1 Context and Motivation

Nowadays, touch screens are used daily, mainly on cell phones. Due to the pandemic, touch screen devices became a public health hazard, especially when used by many users, like interactive screens in museums or hospitals. In order to continue to use these devices, they must be cleaned and disinfected more regularly, which can damage the equipment. Therefore, the company that proposed this dissertation - 3Decide - had the idea to design touchless interfaces that allow the users to interact with the application without physical contact. This is one of the many projects of 3Decide, a company that is always trying to find intelligent solutions to the daily life problems that keep coming up as the world evolves.

The technology implemented by them was based on computer vision and object recognition. It consists of positioning Time-of-Flight (ToF) sensors behind a glass or acrylic screen so that they can detect movement without needing to touch the device directly. The sensor used was the VL53L0X (version 1), but in the meantime, a new version has been released - the VL53L1X (version 2).

1.2 About the Company

3Decide is a business-to-business company with ten years of existence and defines itself as a unique visual software house. It has the experience and strong know-how in technologies and visual media, like 360º photos or videos, interactive 3D models, animated infographics, and several maps and diagrams. It provides a wide range of features that helps companies sell better, promote more, and have better workflows.

The company works with three major organizational lines: marketing, management, and sales. The kind of solutions that it has proposed are advanced virtual tours where people can walk through spaces, aiming to sell or even get to know them. 3Decide achieves this in feasible ways, using daily platforms such as tablets, smartphones, and computers so that our solutions can be reused worldwide and reach an accurate global scale.

It works with plenty of markets, from tourism, heritage, and culture to companies like NOS, where it provides a series of dashboards to visualize corporate information. The company is now in a new phase, supported by more than 100 well-succeeded projects. Therefore, it aspires to go to other countries, particularly in the European space. For that, it includes some of its partners and companies with which it shares exciting projects. They are also research partners like the University of Porto, namely the Faculty of Engineering, the Laboratory of Artificial Intelligence and Computer Science, and several other knowledge partners.

"By fusing advanced visual content with modern information systems and social networks, 3Decide takes Technology one step closer to Humanity!"

1.3 Company's Project

1.3.1 Initial state of the project

3Decide has developed an interactive kiosk using multiple VL53L0X sensors. The kiosk aimed to provide a solution to interact with an application without the need to touch any surface. Figure 1.1 shows a photo of a kiosk.

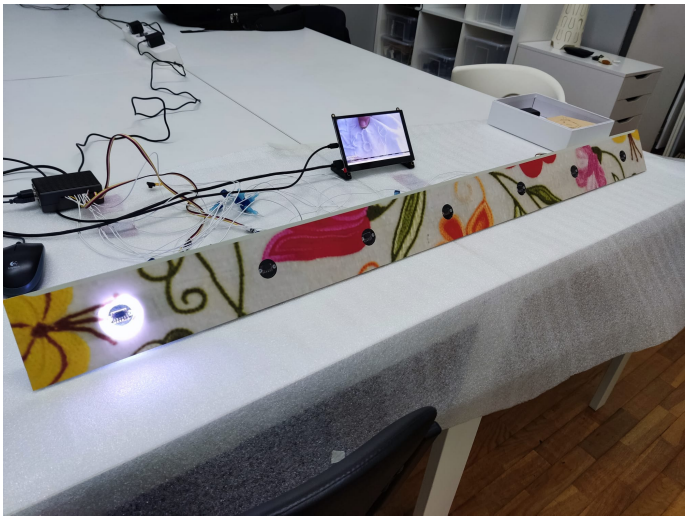

Figure 1.1: Interactive Kiosk by 3Decide

This specific kiosk was sold to a museum, and its purpose was to allow the visitors to interact with a video being exposed. This video contained various stages of the production of a product, and for each step of the process, there was a touchless button (using a VL53L0X sensor, as shown in Figure 1.2). By selecting the respective button, the museum visitors could skip front or back in the video.

In Figure 1.1 seven buttons can be seen, i.e., VL53L0X sensors covered with an acrylic surface placed in front of the sensors, a Raspberry Pi 4, and a (provisional) LCD screen to display the video.

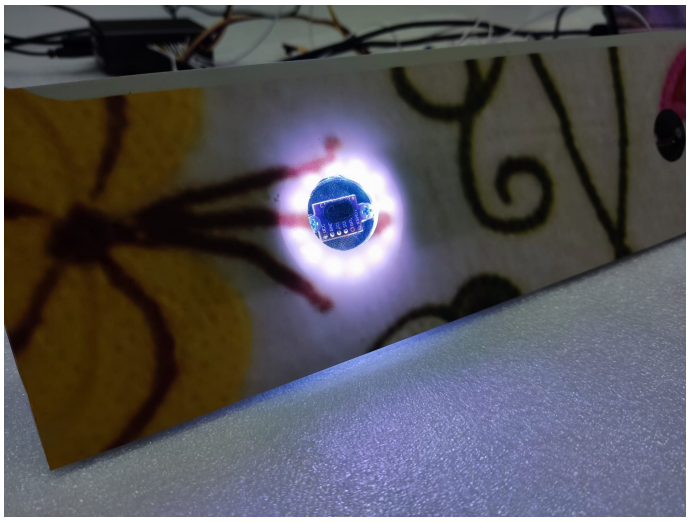

Figure 1.2: VL53L0X sensor used

Around each button is an LED strip, and when the sensor detects a target (in this case, the hand of the user) in front of it, the LEDs start to light up one at a time. The button is selected when all the LEDs are on.

The product hindrance is that, because there is an **acrylic surface placed in front of the sensors** to prevent damage to the equipment, the maximum distance detected is limited to a few centimeters (less than 10cm), and the accuracy of those measurements is not very good.

1.3.2 End application needs & company's tools

The project system is composed of sensors and an SBC communicating via I^2C.

The information flow starts in the **sensors** (information input), collecting the measurements and passing them from the user (target) into the system.

To process the information, the SBC chosen by 3Decide was **Raspberry Pi 4**, as it has the most advantages for this application among the various microprocessors available in the market. One of the most valuable advantages is that it has I2C buses, which allow multiple devices to be connected to the Raspberry Pi.

As for the programming language used to control the sensors, **JavaScript** and **Typescript** were elected by 3Decide running on Node.js. In the Interactive Kiosk developed by 3Decide, the approach to interact with the system was made via a JSON file, where the user could change the necessary parameters without deep programming knowledge. The parameters were then passed into the JavaScript program to be processed. Using these programming languages, the company can easily adapt any project with these sensors into a Web-controlled application, for example. The code implemented by 3Decide did not include all the calibration processes, so the rest of these functions are to be implemented.

Finally, the information flow ends on the **display screen** where the video is playing (information output).

1.4 Objectives and Goals

The company has developed an *Interactive Kiosk* to be used in public places, such as museums, to make the visitor's experience more interactive and safer. For that, they used the VL53L0X sensor due to its low cost and high measurement accuracy when used without a translucent surface (TS) in front of it.

3Decide had some problems in the Kiosk development due to the lack of deep knowledge of the VL53L0X operation dynamic and because they added an acrylic surface in front of the sensors. Additionally, the code used to use the sensor did not include calibrating all the parameters necessary to reach the sensor's best performance. The result was an application that could only detect movements in a range of some centimeters. The company's temporary solution was to pierce the acrylic to uncover the sensor. This way, the sensor could detect the user's movements more precisely and increase its range of detection. Therefore another challenge was added: try to manipulate the sensor parameters or the TS, i.e., glass or acrylic, itself to allow the sensor to detect movements with precision when a TS is placed in front of it.

The **dissertation's primary goal** is to understand how the hardware of the **new version** of the sensor – the **VL53L1X** sensor – works, develop software to calibrate and operate the sensor and test its performance in different conditions. After that, some tests are to be done on the old version too, to compare both versions and understand the new version's advantages compared to the old one.

Recapping, the **client's (3Decide)** specifications are:

- Study the sensors' hardware
- Develop new software for the new version (preferably in JavaScript) for **single** and for **multiple** sensors simultaneously
- Find a solution to let the sensors operate behind a glass or acrylic surface
- Compare their performance

Solving all the problems listed above will provide the company with solutions and tools to efficiently work with the new version and also understand the old version's operation more deeply. With this information, 3Decide will be able to adapt both sensors to future projects.

1.5 Document Structure

Other than this introductory chapter, this document contains four more chapters.

In chapter 2, firstly there will be presented some other options of ToF sensors available in the market and a comparison with the sensors VL53L0X and VL53L1X. Additionally, the main theoretical knowledge about ToF sensors, VL53L0X and VL53L1X, important to understand the procedures and the engineering area of this dissertation, is described.

Chapter 3 consists of the study of the possible sources of error in the measurements when a glass or acrylic is added in front of the sensor. This chapter predicts the errors that will be verified in the next chapter.

Chapter 4 contains the project development, which includes the code implementation and some explanatory notes.

In Chapter 5 there are the results and a discussion of the results.

Lastly, chapter 6 is the conclusion of this document, which there will be discussed the major results, findings until this point, and future work.

Chapter 2

Fundamental Issues

2.1 Time of Flight Principle

As the name suggests, distance sensors are used to determine the distance to an object. It can be measured by the time the signal returns to the sensor, the intensity of the received signal, and its phase change. Various types of distance sensors are available in the market, such as ultrasonic, infrared, laser, and ToF. This dissertation will be focused on the last one.

ToF sensors use the time-of-flight principle, i.e., they measure the elapsed time from the emission to the reception of a wave pulse and calculate the distance to the object. Essentially, the ToF sensor emits infrared (IR) waves from a Vertical-Cavity Surface-Emitting Laser (VCSEL) towards an object. This wave is then reflected when it reaches the target and returns to the sensor (figure 2.1), where it is received by a Single-Photon Avalanche Diode (SPAD). The SPAD array generates a pulse for each photon of the signal detected. The total time of flight (ToF) is the time between sending and receiving the signal. Finally, the distance is calculated using the approximate speed of light in the air and the total ToF.

Figure 2.1: ToF principle

There are two ways to measure the ToF - the direct and the indirect methods. The direct method measures the time difference between the emitted and the received pulses (digital analysis). Indirectly, the ToF is measured by the phase difference between the sinusoidal wave of light

emitted and the incoming signal (analog analysis). The most used is the direct method because its processing is faster.

There are some problems with this type of sensor too. One is that the object may be considered out of the Field of View (FoV) because of a dead zone between the VCSEL and the SPAD [11]. This rarely happens because the distance between them is small, but there is still this possibility. Also, if two objects are in the FoV, the sensor measures the weighted average of the distances, leading to measurement errors.

Compared to the other types, some of the advantages of these sensors are that they have a more comprehensive range, faster readings, greater accuracy, and can produce 3D images. Also, they measure distance independently of the target size, color, or reflectance. That is why they can be used in many areas, such as robotics and industrial applications. The major disadvantage is that they are usually more expensive. Still, the sensor that will be explored in this dissertation is a low-cost ToF sensor - the VL53L1X sensor, an updated version of the VL53L0X sensor.

2.2 Market Survey on ToF Sensors

The use of market research data helps businesses make significant adjustments that lower the level of risk associated with making critical business choices. Therefore, some of the ToF sensors available in the market were analyzed, and some exciting alternatives to VL53L1X were found.

2.2.1 TeraRanger Evo 3m sensor

Terabee sensor modules are compact, lightweight, and incredibly versatile. Thanks to practical USB or I2C/UART interfaces and easy-to-use data streams, they may be easily integrated into sensing applications.

These Time-of-Flight distance sensors allow us to measure a single point and quickly provide millimeter-level distance data. They are simple to use as single sensors or in multi-sensor arrays, eye-safe, and plug-and-play. Some typical applications include robotics, drone proximity sensing, level monitoring, and object counting.

The TeraRanger Evo 3m sensor (figure 2.2), developed by TeraBee [12], is ideal for close-range distance sensing applications at higher speed.

Figure 2.2: The TeraRanger Evo 3m sensor (reproduced from [1])

According to the comparison information between VL53L1X and TeraRanger Evo 3m, displayed in Table 2.1, this sensor is significantly bigger than VL53L1X, and its price is much greater

(around 50€ [1]). Even though this is a high-speed measuring sensor, overall, the VL53L1X is more advantageous.

Table 2.1: Comparison between VL53L1X and TeraRanger Evo 3m

	VL53L1X	TeraRanger Evo 3m
Distance measurement	Ranging up to 4m	Ranging from 10cm to 3m
Ranging frequency	33 Hz	100 Hz
Size	4.9 x 2.5 x 1.56 mm	29 x 29 x 22 mm
Eye Safety	Yes	Yes
FoV	15 to 27 degrees	approx. 2 degrees
Accuracy	Not specified	±2cm

2.2.2 Sony DepthSense ToF sensor (IMX556)

The IMX556 (figure 2.3) is a fully integrated optical ToF camera sensor, as well known as the Helios Time of Flight (ToF) 3D Camera. Thanks to its Current Assisted Photonic Demodulator (CAPD) combined with its Backside Illuminated Pixel (BSI) technology, it has an excellent 3D imaging performance [13].

Figure 2.3: The IMX556 sensor (reproduced from [2])

Bellow, on table 2.2, there is a comparison of some of both sensors' features in order to measure the pros and cons of each.

Table 2.2: Comparison between VL53L1X and IMX556

	VL53L1X	IMX556
Distance measurement	Ranging up to 4m	Ranging up to 6m
Size	4.9 x 2.5 x 1.56 mm	50 x 50 x 50 mm
Operating Voltage	2.6 to 3.5V	18 to 24 V
Operating temperature	-20 to 85 °C	-10° to 60°C
Eye Safety	Yes	Yes
FoV	15 to 27 degrees	59 x 45 degrees
IR Laser emitter	940nm	850nm
Accuracy	Not specified	Near Mode: ±5mm; Far Mode: ±10mm
Low power consumption	20mW	10W

The significant advantages of this sensor, compared to VL53L1X, are that it can capture 3D images, while multiple VL53L1X sensors are needed to obtain the same result, and it can measure up to 6m. There is a new version of the sensor, Helio2+, which can reach 8m.

The biggest disadvantages are the price, which is quite expensive (around 1500€ [2]), the sensor's size, and the power consumption.

The IM556 sensor can be handy for 3D applications but is very expensive. For 3D applications, the client can use several VL53L1X sensors, which will return good results for significantly less price.

2.3 Sensors

In order to ease the comprehension of the following chapters, below is a descriptive list of essential concepts associated with both sensors, according to the information available in VL53L1X User Manual [3]:

- *Cover glass* is the protective window placed on top of the sensor receiver and emitter.
- *Timing budget (TB)* is defined as the sensor's programmed time for performing and reporting ranging measurement data, i.e. the time the sensor takes to complete one range measurement. There is a trade-off between improved measurement accuracy and increased power usage when the TB is increased.
- *Inter-Measurement Period (IMP)* is the programmable interval between two successive measurements. If the IMP is less than or equal to the TB, the real IMP will be twice as high as projected. Figure 2.4 shows these last two concepts to ease their comprehension.

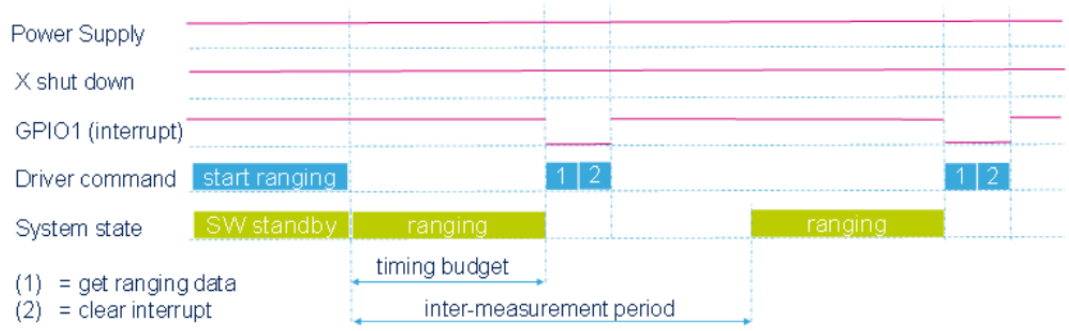

Figure 2.4: Ranging Sequence and Timings (reproduced from [3])

- *Offset* is a measurement deviation to the real distance to the object in **millimeters**. It can be due to the soldering of the device on the costume board and the presence of cover glass in front of the device.
- *Cross-talk* is the amount of return signal that the sensor receives as a result of VCSEL light reflection inside the cover glass that is added to the module's top for aesthetic and protective reasons. The amount of return signal may be large and impact the sensor's performance,

depending on the quality of the cover glass, and it is measured in **cps** (counts per second). The amount of correction required to account for the impact of the cover glass is estimated using crosstalk calibration.

2.3.1 VL53L0X

Since VL53L0X's launch in 2016, it has been considered the most miniature ToF sensor in the world (Figure 2.5), thanks to the leading-edge "in-house" packing technology that integrates a VCSEL and its driver, a SPAD receiver, and the rest of the circuit in an all-in-one module. Another interesting fact about this sensor is that it is considered "Eye Safe", which means that the laser doesn't damage the eyes of the user in any way.

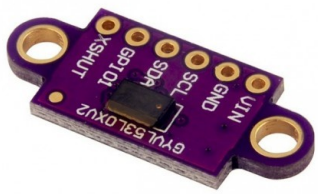

Figure 2.5: VL53L0X sensor [4]

VL53L0X detects objects in the FoV over a maximum range of 2 meters. The manufacturer (STMicroelectronics) provides an API for data transfer and device control. This API consists of a set of C functions that permit the user to prevent the sensor, such as initializing and setting the system's accuracy.

Even though the laser used in this sensor is IR, VL53L0X doesn't use the same method to measure the distance to the object as an IR distance sensor. This type of sensor measures the length by the amount of light returned to the sensor. On the other hand, VL53L0X uses the time the signal takes to return to the sensor, so it is more precise than IR distance sensors.

This sensor has an FoV of 25º, which means it almost exclusively detects what is in front of it, leading to small measurement error values. This is an advantage compared to sonars, for example, because they can see surrounding objects by mistake due to the multiple reflections of the ultrasonic waves.

Considering the information in Table 2.3, VL53L0X is a very efficient choice for many applications because of its small dimensions, precise object detection, and low power consumption. Additionally, it has an interestingly low cost (less than 10$).

Table 2.3: VL53L0X's technical specification[6]

Distance measurement	Ranging up to 2 meters
Ranging frequency	33 Hz
Size	4.4 x 2.4 x 1.00 mm
Operating Voltage	2.6 to 3.5V
Operating temperature	-20 to 70°C
FoV	25 degrees
IR Laser emitter	940nm
Accuracy	+/-3 %
Programmable modes	3 modes (short, medium, long)
Low power consumption	20mW

2.3.1.1 Calibration

ToF systematic errors are typically fixed, and under normal conditions, calibration can be used to compensate for their effects. It is more challenging to develop a model to describe and rectify an unsystematic error since it is caused by the external environment and has an unfixed form.

An initial and once-only calibration step is required at the customer lever for accurate measures. This process is different for both versions. A comparison is shown below based on the sensors' user manuals and the respective API function names.

For **version 1** [5], the calibration flow (shown in Figure 2.6) is the following:

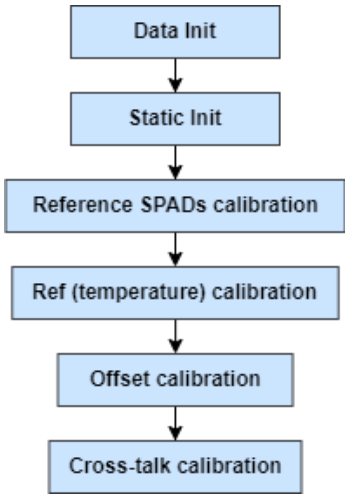

Figure 2.6: Calibration Flow VL53L0X (adapted from [5])

- **Data Init** (*DataInit()*) - Performs the device initialization and is only called once after it is brought out of reset.

- **Static Init** (*StaticInit()*) - Allows to load device settings specific for a given use case.
- **Reference SPADs calibration** (*PerformRefSPADManagement()*) - This step is performed on the bare modules during the Final Module Test at STMicroelectronics. The calibration data (SPAD numbers and type) are stored in the device Non Volatile Memory (NVM). The customer must recalibrate the reference SPADs if a cover glass is used on top of VL53L0X. This needs to be done only once during the initial manufacturing calibration; calibration data should then be stored on the Host. The function returns the number and type of reference SPADs to be used and loads this information into the device. This calibration doesn't require any particular conditions, such as a specific target or lighting conditions.
- **Ref (temperature) calibration** (*PerformRefCalibration()*) - Includes the calibration of two parameters - VHV and phase cal - which are temperature sensitive. They are used to set the device sensitivity. It should be performed during the manufacturing calibration and whenever the sensor's temperature varies more than 8ºC (compared to the initial calibration temperature).
- **Offset calibration** (*PerformOffsetCalibration()*) - Is performed during Final Module Test at STMicroelectronics, and the offset value is stored in the device NVM. Using a cover in front of the sensor can affect the ranging, so the customer should perform a new offset calibration before starting ranging. For calibration, it is recommended to use a **white target at 100mm from the sensor in a dark environment**, but the customer can change the target distance, depending on his constraints. The only restriction is that this value has to be chosen in the linear part of the ranging curve (represented in blue in Figure 2.7). The return value of *PerformOffsetCalibration()* is the offset calibration value in micrometers and has to be stored in Host memory through the function *SetOffsetCalibrationDataMicroMeter()*.

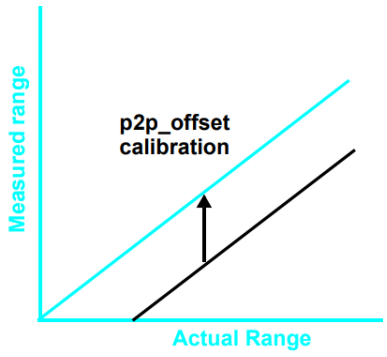

Figure 2.7: Range offset calibration [6]

- **Cross-talk calibration** (*PerformXTalkCalibration()*) - Is a function that compensates the effect of having a cover glass in front of the sensor. It needs an input value, that corresponds to the calibration distance in millimeters. This distance depends on the quality of the cover. Without the cover, the result is a linear function between the distance measured and the

actual target distance also called the ideal curve. With a glass, what happens is that the sensor only measures until a certain value and then the measured distance starts to decrease (Figure 2.8). With this in mind, there is a valid window of values that can be used as inputs to this calibration function. The first is when the actual signal starts to deviate from the ideal curve and the last is when the measured distance starts to decrease, compared to the real distance (Figure 2.9).

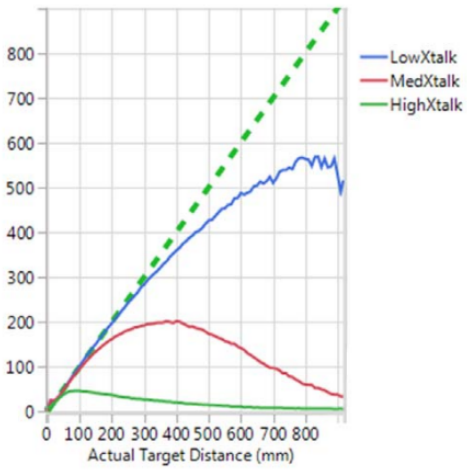

Figure 2.8: Cover window impact on ranging [5]

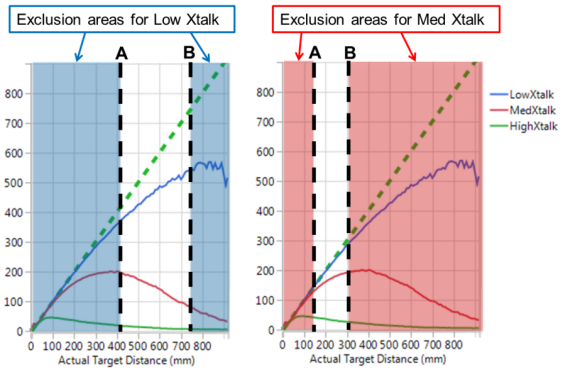

Figure 2.9: Crosstalk calibration's window of valid distances [5]

2.3.1.2 Ranging Flow

After the initialization (Data Init and Static Init steps) and calibration comes the sensor ranging flow, which is represented concisely summarized in Figure 2.10.

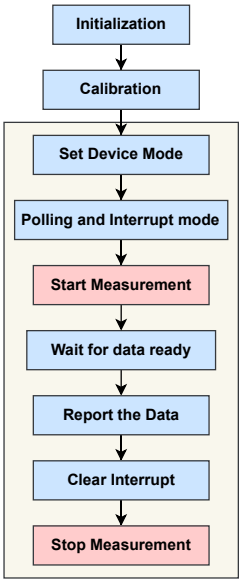

Figure 2.10: VL53L0X Ranging flow

- **Set Device Mode** (*SetDeviceMode()*) - This function aims to select one of three modes of operation: Single, Continuous or Continuous Timed Ranging.
- **Polling and Interrupt mode** (*SetGPIOConfig()*) - Once a measurement is ready, the Host receives an interrupt. This function configures the system interrupt mode: either an interrupt or a poll.
- **Start Measurement** (*StartMeasurement()*) - To start a measurement, this function needs to be called.
- **Wait for data ready** (*GetMeasurementDataReady()*) - After starting the measurement, the sensor will wait for data ready using this function, which polls on the ranging or on the interrupt status.
- **Report the Data** (*GetRangingMeasurementData()*) - The ranging data is reported through this function.
- **Clear Interrupt** (*ClearInterruptMask()*) - After the completing the measurement, the interrupt is cleared with this function.
- **Stop Measurement** (*StopMeasurement()*) - At the end of the measurements, they must be stopped with this function. If a range measurement is in progress when the stop request is made, the measurement is finished before stopping. The stop command is immediately effective if it happens during the interval between measurements.

2.3.1.3 Research

Based on [7], some precision tests have been made on the sensor with different materials and light conditions. This research was arranged to understand if the VL53L0X's performance was adequate for independent robotic movement.

The materials used for these tests were plywood (figure 2.11), polished metal (figure 2.12) and black fiber (figure 2.13). In order to obtain a good amount of data, the whole sensor range (in the default mode) was used - 50 to 1200mm - with steps of 50mm.

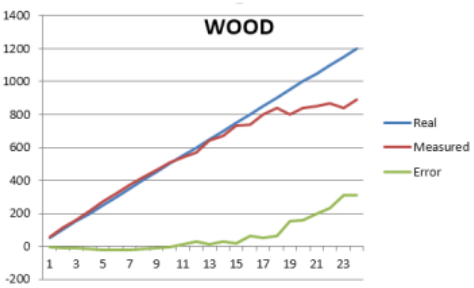

Figure 2.11: Results with wood surface (reproduced from[7])

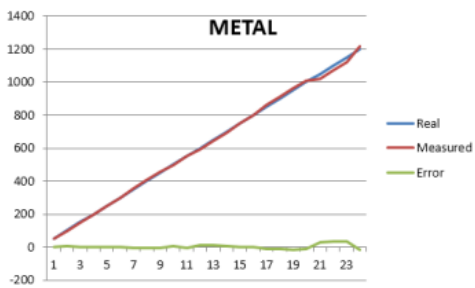

Figure 2.12: Results with metal surface (reproduced from[7])

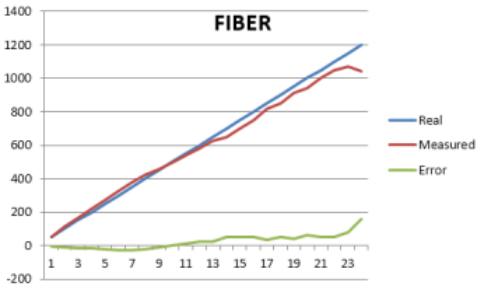

Figure 2.13: Results with fiber surface (reproduced from[7])

The results are interesting because, theoretically, this sensor is independent of the reflectivity parameter of the surfaces. It's noticeable that the error is more significant at longer distances, and the surface that results in the most critical mistake is the wood surface.

As for the error induced by ambient light, tests were performed with two different light sources, a 45W incandescent lamp, and a 40W LED lamp. The samples were taken at different heights and are presented in figures 2.15 and 2.14 [7].

Lamp Height[cm]	Measurement[mm]	Error[%]
10	512	2.4
20	513	2.6
30	510	2
40	512	2.4
50	514	2.8
60	511	2.2
70	512	2.4
80	514	2.8
90	513	2.6

Figure 2.14: Results with LED lamp (reproduced from[7])

Lamp Height[cm]	Measurement[mm]	Error[%]
10	580	16
20	563	12.6
30	553	10.6
40	548	9.6
50	535	7
60	525	5
70	515	3
80	514	2.8
90	513	2.6

Figure 2.15: Results with incandescent lamp (reproduced from[7])

The results demonstrate that the LED lamp doesn't interfere significantly with the measurements, with error values rounding 2.5%. On the other hand, the errors of the measurements with the incandescent lamp have higher values, especially when the lamp is lower. This is expected as incandescent lights produce more light outside the visible wavelength band and the IRs emitted interfere with the sensor's SPADs, which are sensitive to IR light.

2.3.2 VL53L1X

In 2017, VL53L1X (Figure 2.16) was launched. It was a success because it contains all of the previous version features, builds on them to theoretically drastically increase the performance, and has an exciting cost (less than 10€ [14])and a standard interface. As version 1 (VL53L0X), version 2 (VL53L1X) is "Eye Safe".

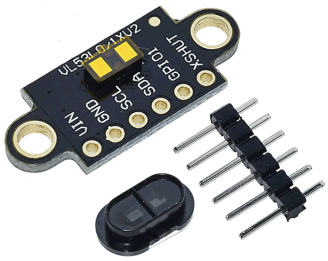

Figure 2.16: VL53L1X sensor [8]

VL53L1X is one of the fastest miniature ToF sensors available on the market and can measure up to 4m, with a ranging frequency up to 50Hz. It measures absolute distance independently of the target color and reflectance and can change the FoV programmatically within 15 and 27 degrees. The VL53L1X's features are listed in Table 2.4.

Table 2.4: VL53L1X's technical specification[10]

Distance measurement	Ranging up to 4 meters
Ranging frequency	Up to 50 Hz
Size	4.9 x 2.5 x 1.56 mm
Operating Voltage	2.6 to 3.5V
Operating temperature	-20 to 85 °C
FoV	Programmable (15-27 degrees)
IR Laser emitter	940nm
Accuracy	Not specified
Programmable modes	3 modes (short, medium, long)
Low power consumption	20mW

The default VL53L1X configuration is called the *Autonomous Ranging Mode*, where the ranging is continuous with a programmable inter-measurement period and raging duration. In this mode, some distance thresholds can be set. For example, if the upper threshold is set to 1m and the distance exceeds that value, an interrupt is raised. The region-of-interest (ROI) size and position can also be customized, starting from 4x4 SPADs to 16x16 SPADs.

The sensor's software driver includes some functions to read the output of the measurements. The read values are the ranging distance (in mm) and status (status examples: booting, ranging, not ranging).

There are three distance measurement modes: short, medium, and long. The short mode is immune to ambient light, and its maximum ranging distance is around 130cm, while the ambient light limits the medium and long modes. Theoretically, it can measure up to 4m, but in a dark environment, the maximum distance is 3.60m.

2.3.2.1 Field of View

One of the new features added in this version is the programmable FoV, between 15 and 27 degrees. It is a great advantage because the sensor becomes more adaptable to different situations, allowing the user to use the same sensor for other applications.

The FoV can be adjusted by setting different ROIs, which are determined by the number of SPADs used. In other words, fewer SPADs mean a smaller ROI and a more limited FoV. The minimum size of the ROI is 4x4 SPADs (FoV of 15º), and the maximum size is 16x16 SPADs (FoV of 27º), which corresponds to the default size (Figure 2.17).

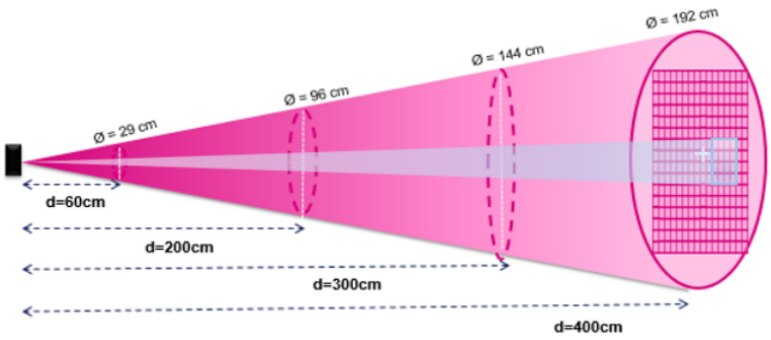

Figure 2.17: VL53L1X Field of View cone [9]

The ROI can be square or rectangular and is defined by its center, width, and height. If the user specifies a value of width or height lesser than 4 or greater than 16, that value is automatically capped to 4 and 16, respectively.

The fact that the FoV is a cone can be a problem because the bigger the distance to the target, the greater the probability of the sensor detecting the surroundings. That is why it is necessary to reduce the angle of the FoV in some situations. The only inconvenience about reducing the FoV is that the maximum ranging distance reduces too because when the number of SPADs is reduced, the number of reflected photons that hit a SPAD is also reduced.

2.3.2.2 Calibration

To benefit from the sensor's full performance, according to the client specifications, the calibration flow needs to be performed before the measurements are taken.

The calibration flow of VL53L1X (shown in Figure 2.18) is the following [3]:

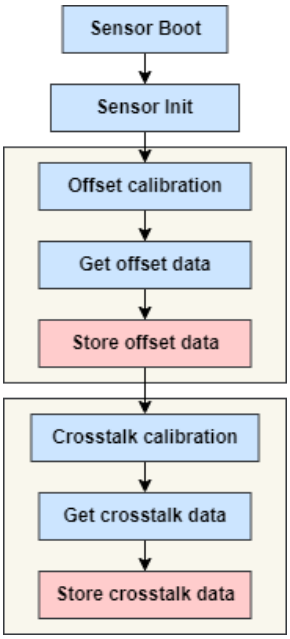

Figure 2.18: Calibration Flow VL53L1X (adapted from [3])

- **Sensor Boot** (*BootState()*) - It is used to check if the sensor has booted. It is mandatory to perform this step before the first I^2C access.
- **Sensor Init** (*SensorInit()*) - It is called once to initialize the sensor with a default configuration. With the default configuration, the sensor ranges at 10 Hz in long distance mode.
- **Offset Calibration** - Must be always used to compensate for some environmental effects on the measurement. This step includes calculating the offset value to apply, getting it, and setting/storing it into the sensor. To calculate the offset, the recommended measurement conditions are a **dark environment, a 17% gray target at 140mm** from the sensor. The process of calibration is done by the *CalibrateOffset(val)* function, which realizes 50 measurements, calculates the mean value of those measurements, and then subtracts it from the actual distance to the target (val), which is the input of this function. After finding the offset value, the function applies and returns the offset correction value. The value must be stored in the host system and written into the sensor. This is done through the SetOffset() function.
- **Crosstalk Calibration** - Crosstalk is defined as the amount of return signal received on the sensing array due to VCSEL light reflection inside the protective window (cover glass) placed on top of the module for aesthetic and protective purposes and any other glass surfaces added in front of the sensor. This may affect the sensor's performance, depending on

whether the return signal amount is significant. The VL53L1X provides crosstalk calibration to compensate for this crosstalk effect through the *CalibrateXtalk(xcd)* function. Before this function, the maximum distance measured by the sensor needs to be measured to send as input to the function because the calculation formula needs it to find the crosstalk correction value. This value is named "crosstalk calibration distance" (xcd) and is represented in Figure 2.19. The calibration conditions are a **dark environment with a 17% gray target at the crosstalk calibration distance**. The function applies the correction and returns this value. After this, the crosstalk compensation value needs to be stored in the host system using the *SetXtalk()* function.

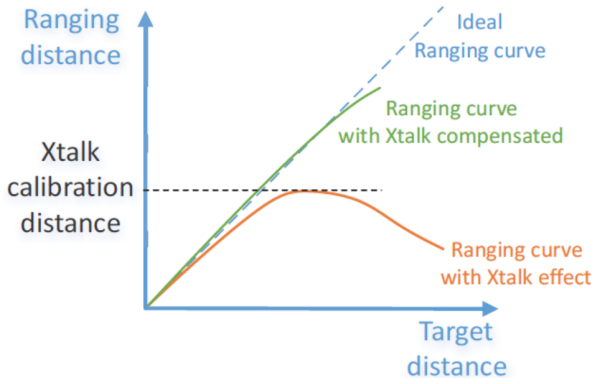

Figure 2.19: Crosstalk calibration distance definition [3]

2.3.2.3 Ranging Flow

After the initialization (Data Init and Static Init steps) and calibration processes, the ranging flow takes place. This flow is represented in Figure 2.20.

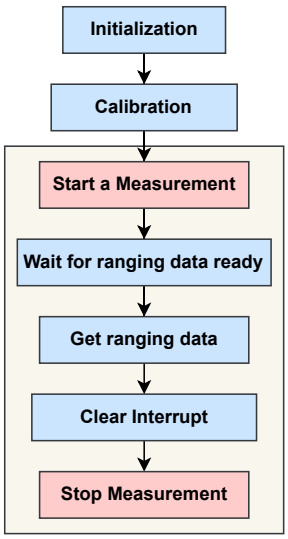

Figure 2.20: VL53L1X Ranging Flow (adapted from [3])

- **Start a Measurement** (*StartRaging()*) - This function starts the measurements.
- **Wait for the Ranging Data to be ready** (*CheckForDataReady()*) - When new ranging data is ready, the value "1" is returned.
- **Get Ranging Data** (*GetRangeStatus* and *GetDistance()*) - These are the two main functions used to get ranging data. The first returns a value (0, 1, 2, 4, or 7), on which the 0 represents that there is no error. The second function returns the measurement in millimeters.
- **Clear the interrupt** (*ClearInterrupt()*) - This step's aim is to clear the interrupt, which should be done to enable the next interrupt event when new ranging data is ready.
- **Stop the measurement** (*Stop()*) - Because the ranging is done continuously, when the user wants to stop taking measurements, he must use this function. If the stop command is issued while the ranging operation is in progress, the sensor completes the current ranging operation before stopping.

2.3.2.4 Applications

As previously mentioned, ToF sensors can be used in many engineering applications. In the following paragraphs, some examples will be presented.

The VL53L1X was included in research about distance measurement in a vineyard, along with two ultrasonic sensors - HC-SR04 and JSN-SR04T-2.0 [15]. The study aimed to find the best option for this specific application - measure the growth density of the vine leaves. The measurements were performed at 1m of height in a vineyard row section of length 3m, placed 1m-1.5m in front of it.

During the tests, it was concluded that VL53L1X long and medium measurement modes reduced the sensing range, considering that it's an outdoor test with high ambient light. Due to this fact, the distance of the tests was reduced to 1m, while the other two sensors were tested at 1.5m. VL53L1X has shown that most of its collected data was inside the tolerance limits (around 83%) but was limited in range (only up to 1.2m). After applying a Kalman filter, 100% of the results were inside the tolerance limits, which shows that this sensor measures are reliable in distances below 1.2m. In the end, the JSN-SR04 sensor had the best results for this application. This study demonstrates that VL53L1X is unsuitable for outdoor environments, especially when the distance to the target is greater than 1.2m [15].

Contrariwise to the previous study, an indoor application will now be analyzed to understand the functional performance of the sensor without strong ambient light. This example consists of a hybrid localization system, where Bluetooth Low Energy (BLE) is used alongside laser proximity sensors [16]. VL53L1X is one of those sensors due to its small package which can easily be integrated into localization system circuits. Some basic features, such as FoV, ranging bias, and ranging standard deviation, were tested under various conditions. The sensor was set with long mode, ranging rate ten times/sec, and the default FoV of 27 degrees was used.

The experiment measured the distance between a person dressed in a black t-shirt, then in a white t-shirt, and finally bare-chested. Additionally, the measurements were made in an office with

blinds open and then closed to evaluate the ambient light impact on its accuracy. The maximum distance where it was possible to detect the person's presence was 3.5m.

The ranging bias is negligible in short distances but rises when the space is greater than 2m, and for 3.5m, it's over 30cm. This evidence shows that the sensor's performance is not dependent on the clothing's colors nor the lightning conditions [16].

The ranging standard deviation is not larger than 5cm for distances below 2.5m. It rises for more considerable distances, reaching near 20cm at the maximum sensor's range. The ranging standard deviation appears sensible to the surface reflectance: its value is significantly bigger for the black t-shirt than for the white t-shirt. On the other hand, this value does not depend on lighting conditions.

It was also detected that the actual FoV of the sensor is narrower than declared in the application note. VL53L1X has been demonstrated to be a viable solution to improve the system's accuracy for this application.

From these studies, it can be concluded that VL53L1X's performance is very satisfactory when used in indoor environments. However, its range is limited when used in high ambient light environments (outdoors conditions).

2.3.3 Comparison between VL53L0X and VL53L1X

Table 2.5 shows the summary of the differences between the two versions.

Table 2.5: Comparison between the two sensors

	VL53L0X	VL53L1X
Distance measurement	Ranging up to 2 meters	Ranging up to 4 meters
Ranging frequency	33 Hz	Up to 50 Hz
Size	4.4 x 2.4 x 1.00 mm	4.9 x 2.5 x 1.56 mm
Operating Voltage	2.6 to 3.5V	2.6 to 3.5V
Operating temperature	-20 to 70°C	20 to 85 °C
FoV	25 degrees	Programmable (15-27 degrees)
IR Laser emitter	940nm	940nm
Accuracy	+/-3 %	Not specified
Programmable modes	3 modes	3 modes
Low power consumption	20mW	20mW

To minimize measurement errors, some parameters must be calibrated [17], as already mentioned. Both versions of the sensor can be calibrated regarding various parameters, which can be very useful to adapt to different applications.

VL53L1X has a programmable FoV, which is a great advantage compared to VL53L0X, but the fundamental difference between the two versions is their range: version 1 reaches up to 2m and

version 2 up to 4m, in ideal conditions [18]. The new version also has a Programmable Region of Interest (ROI). It offers the ability to limit the regions of the receiver that will transmit the information they capture, thus restricting the FoV to a specific region of interest. This feature avoids wasting resources, power, and computational throughput because the sensor doesn't have to capture the whole FoV [18].

Whereas VL53L1X has a more extended range, it is slightly larger and more expensive than the VL53L0X. The client can consider using the oldest version in some situations where the product doesn't require a long distance measurement. On the other hand, if the developer needs better performance or a more personalized configuration, he should choose the newest version, as it has a faster ranging frequency and more programmable features. Regarding the programmable ROI, the VL53L1X API provides some commands to program the size and position of the FoV and allows the creation of multiple regions of interest. The VL53L1X can also be entirely energy-autonomous, because it has the ability to set the rest of the system to sleep, saving a large amount of power. It wakes itself up automatically when detecting movements or by a specified time interval (that can be defined using the *VL53L1_SetInterMeasurementPeriodMilliSeconds()* function).

In terms of software, there were also some improvements over the VL53L0X. For instance, a great part of the VL53L0X's API is Windows executable files, which doesn't allow to see and study the code itself. As for the new version, the code is available for consultation, as it is a set of C files that permits the user to learn more pieces of information about the sensor's behavior.

Some base features are standard because one version is inspired by the other. They have the same pin-outs (Figure 2.21) and tolerate a voltage range from 2.6 to 3.5V.

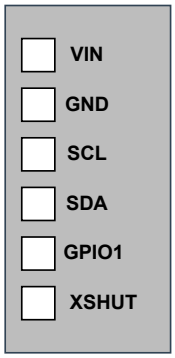

Figure 2.21: Sensors' Pinout

The communication is via I^2C, and, to control the sensors, the manufacturer (STMicroelectronics) provided APIs. They consist of pseudo-code C functions that allow the user to interact with the sensor, such as initializing and setting its accuracy. The driver is not ready to compile and contains syntax errors as it's only supposed to be used as a guideline to implement the I^2C interface. Creating a generalized driver would be impractical because any device with I^2C can communicate with and use this sensor, hence a driver would have to be written for MCUs, SBCs, desktop computers, ASICs, FPGAs, etc, as well as different operating systems and processor architectures.

2.4 Programming Language - JavaScript and Typescript

The programming language used in this project is JavaScript (JS), as required by 3Decide. It is a scripting language not only used in Web pages but also in non-browser environments, such as Node.js - the one to be used in this dissertation.

Scalable network applications can be created using Node.js, an open-source, cross-platform JavaScript runtime environment that uses the V8 engine and executes JavaScript code outside a web browser. Developers can use JavaScript to create command-line tools and for server-side scripting, which produces dynamic web page content on the server before the page is transmitted to the user's web browser. Node.js, which unifies web application development around a single programming language rather than separate languages for server-side and client-side scripts, reflects a "JavaScript everywhere" paradigm.

JS is a dynamically-typed scripting language and its syntax can be quickly learned because it is similar to Java and C++.

One drawback of this language is that variables in JS are not directly associated with any particular data type, and any variable can be assigned (and reassigned) values of all types, but this can be mitigated by using TypeScript, which is a superset of JavaScript, adds type safety, and transpiles into regular JavaScript.

Another drawback is that JavaScript is single-thread, which could have been a problem for this dissertation, as multiple sensors will be controlled at the same time but, fortunately, V8 has a solution for this: the event loop. The event loop allows the developer to create the illusion of multi-threading by having non-blocking I/O operations which run callbacks once they are finished and allow the script to keep being executed until the I/O operations finish and their output needs to be handled. This problem which was classically solved using callbacks got two syntax-sugar improvements over time. First came Promises, which wraps asynchronous code in an object that saves its output once it's done so then it can be obtained by calling the method *then*. This improved the readability but kept callbacks in the *then*, *catch* and *finally* methods. Secondly came the *async/await* syntax which completely removed the need for callbacks when using Promises and async code. Due to the considerable complexity of this and other aspects of the language, prior to the code implementation for this dissertation, studies and research were done to understand its dynamics. To explain this *async/await* syntax, the simple example shown below (reproduced from [19]) will be analysed.

```
1 function wait(ms) {
2   return new Promise(r => setTimeout(r, ms));
3 }
4
5 async function hello() {
6   await wait(500);
7   return 'world';
8 }
9
```

```
10 async function foo() {
11   await wait(500);
12   throw Error('bar');
13 }
```

Listing 2.1: Example of the await/async syntax

The ***async*** functions always return a Promise, regardless whether *await* is used or not. Promises resolve with the function's *return* value, like *'world'* in the *hello()* function, or rejects with the value of its *throw*, as it does in the *foo()* function with *Error('bar')*. The keyword ***await*** only works when used inside *async* functions. It pauses, i.e., suspends the execution of the function until the promise settles and returns its result, and resumes it afterward. In the Listing 2.1's example, when the function *hello()* is called, its execution will be paused for 500ms, while the promise in function *wait(ms)* is resolved. After that, its return value will be the return value of *wait(ms)*, which will be passed into the *hello()* function.

TypeScript was used in 3Decide's VL53L0X sensor's code implementation. This language adds optional static typing to JS to reduce variable types-related errors. As TypeScript is a superset of JavaScript, existing JavaScript programs are also valid TypeScript programs.

2.5 Raspberry Pi

Raspberry Pi is a series of SBCs that are very useful for many applications in engineering, such as robotics and automation. It is widely used mainly because of its low cost, high versatility, and performance [20]. In this dissertation, the model Raspberry Pi 4 will be used. This is the latest version available and provides a high processor speed, memory, multimedia performance and connectivity, and a desktop interface for the user to interact with. Another advantage of Raspberry Pi is that it supports I^2C and has a set of General Purpose Input Output (GPIO) pins, which allows the sensors to be connected to the system. Additionally, I^2C allows several devices to be connected in a daisy chain but this also means that only one sensor can be read at a time because they share the same bus and can't communicate simultaneously.

The Raspberry Pi will give the sensors a programming interface, in this case, through Visual Studio Code, connected via SSH. This connection was made so that the code can be directly implemented in the Raspberry Pi, even though the code is written on the local computer.

2.5.1 GPIO pinout

The most fundamental yet approachable feature of the Raspberry Pi is the GPIO. Since GPIO pins are digital, they can exist in either an *off* or an *on* state. Programming languages like Python, JavaScript, and Node-RED can be used to control the state and direction of the pins, which can have an input or output direction.

Figure 2.22 shows the Raspberry Pi 4 GPIO Pinout along with the connections used to connect the sensors via I^2C. Figure 2.21 shows the sensors' pinout, as they are the same for both versions.

The **red** cable is connected to the sensor's VIN (supply) pin, the **yellow** cable to the SDA (data) sensor pin, the **gray** cable is connected to the SCL (clock) sensor pin and the **black** cable is connected to the GND (ground) pin.

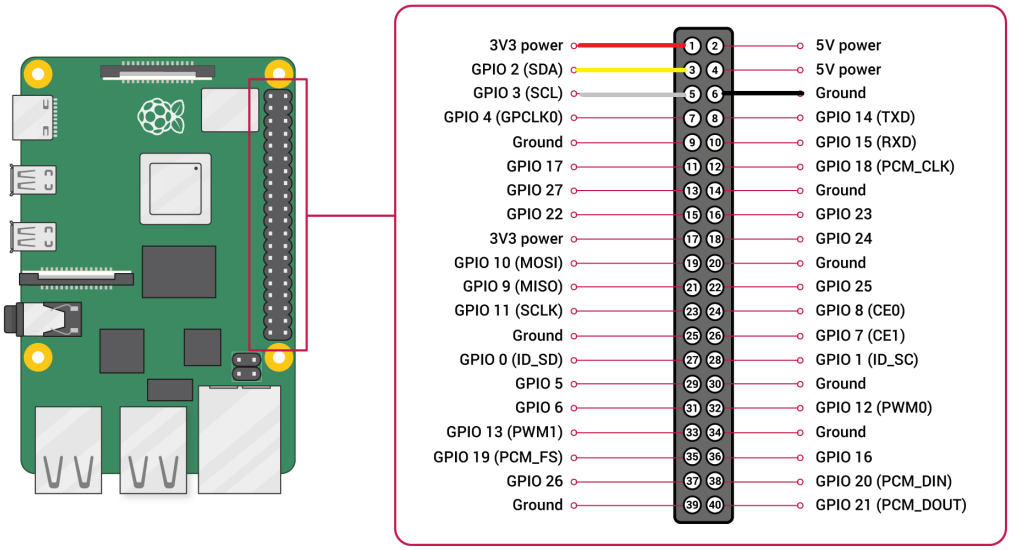

Figure 2.22: Raspberry Pi 4 GPIO Pinout

2.5.2 I^{2}C

I^{2}C is a serial communication bus widely used for attaching lower-speed peripheral ICs (Integrated Circuits) to processors and microcontrollers in short-distance and intra-board communication.

The connection with the sensor is made via pins SDA (Serial Data Line) and SCL (Serial Clock Line) signals, (consult figure 2.21). Every device connected to the bus has the same address. The bus has a maximum speed of 400 kbits/s and uses a default device address (for sensor VL53L1X this address is 0x29).

Chapter 3

Theoretical analysis of the effects created by a TS in front of a ToF sensor

The challenge proposed by 3Decide is that the sensor should be able to work behind an additional cover glass in front of it.

In the project developed by 3Decide, the calibration was done with heuristics obtained by trial and error without deep knowledge of the sensor's behavior. It was sufficient for the company's project to succeed, but 3Decide wants to continue working with these sensors in future projects and so deeper knowledge of their operation is an asset.

Therefore, before testing the sensors with glass and acrylic (i.e. TS) in front, it is necessary to study the theoretical effects of such surfaces on the signal emitted by the sensor. There are three phenomena happening to the light when a TS is in front of the sensor: reflection, refraction, and absorption.

There are at least three sources of error in the measurements when a TS is placed in front of the sensor: the distance offset (refraction), the reduction of the density of light rays that reach the target (specular reflection and light absorption), and the delay caused by the loss of light velocity in the glass (refraction).

3.1 Light Reflection

When a ray of light hits a TS, a part or the entirety of it is reflected, depending on the incidence angle. When the surface is smooth and reflective, like glass, the light is reflected with the same angle of incidence. This type of reflection is called specular reflection and is represented in Figure 3.1. This type of reflection will be assumed in the next calculations.

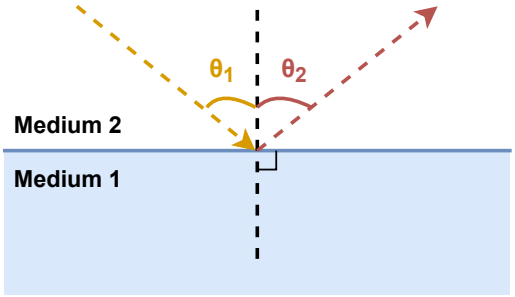

Figure 3.1: Specular Reflection

The moment the light reaches the TS, the refractive index (n) between the two mediums changes instantaneously, causing the specular reflection phenomenon. It is known that for a normal incident angle, i.e., 0º, the amount of light reflected in percentage can be calculated by the simplified Fresnel's Equation 3.1 (when the incident angle is 0º).

$$\%R = 100 \times \left(\frac{n_{air} - n_{glass}}{n_{air} + n_{glass}} \right)^2 \tag{3.1}$$

Considering the refractive index of the air (n_{air}) as 1.0 and the glass's (n_{glass}) as 1.5, the amount of reflected light (with a normal incident angle) is 4% [21]. In view of this project, the sensor will be tested in contact with the glass, with and without its cover glass. In the first case, there will be a gap between the sensor and the glass of a few millimeters, corresponding to the cover glass width. Figure 3.2 represents this scenario, where λ is the gap, i.e., the width of the cover glass of the sensor.

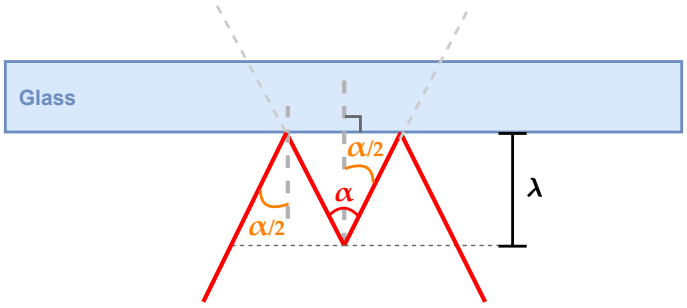

Figure 3.2: Influence of light reflection in the sensor's performance

The normal angle of incidence is the ideal scenario. The bigger the angle of incidence, the bigger will be the reflected angle, as they are the same, and different amounts of light would be reflected because the full Fresnel reflectance equations would be in effect, not just the simplified version for $\alpha = 0^{\circ}$.

3.2 Light Refraction

The fraction of light that is neither reflected nor absorbed by the TS, crosses the material and suffers the refraction phenomenon.

3.2.1 The Snell-Descartes Law

When discussing light passing through a border between two different isotropic media, such as water, glass, or air, a formula is employed to express the relationship between the angles of incidence and refraction. This formula is the Snell-Descartes Law.

This law is used in ray tracing to calculate the angles of incidence or refraction and in experimental optics to determine a material's refractive index.

According to Snell-Descartes's law, for a pair of mediums, 1 and 2, the ratio between the sines of the angle of incidence (θ_1) and angle of refraction (θ_2) is equal to the ratio between the light velocities in the two mediums (v_1 and v_2), which is equivalent to the ratio of their refractive indices (n_2 and n_1), as shown in Equation 3.2.

$$\frac{sin(\theta_1)}{sin(\theta_2)} = \frac{v_1}{v_2} = \frac{n_2}{n_1} \tag{3.2}$$

The medium's indices of refraction (n_1 and n_2) represent the factor by which a photon's speed drops when traveling through a refractive medium, like glass, as compared to its velocity in a vacuum ($n = \frac{c}{v}$, where c is the light speed in vacuum). Depending on the refractive indices of the two mediums, the light will either be refracted to a smaller or larger angle as it crosses the boundary. These angles are relative to the boundary's normal line (Figure 3.3).

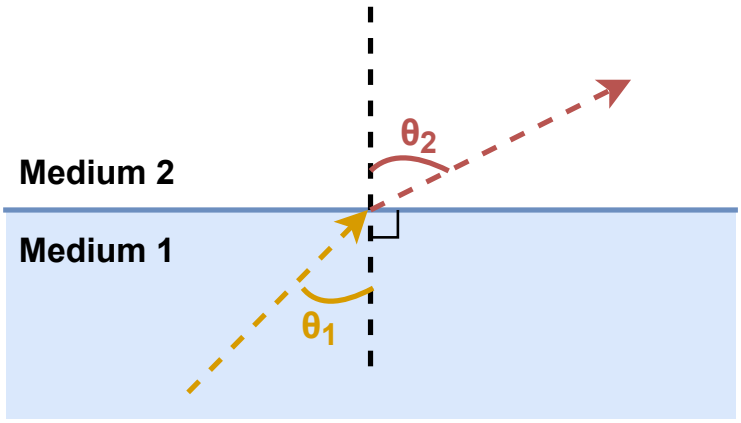

Figure 3.3: Refraction of light

3.2.2 Light refraction on a glass window

This law can predict the light's behavior when crossing a glass window (Figure 3.4).

Considering medium 1 as air and medium 2 as glass and that glass has a higher optical density than the air, the refraction index of the glass (n_2) will be greater than the air's (n_1), which means that their ratio (n_2/n_1) is greater than 1. When the ratio is greater than 1, based on Equation 3.2, the $sin(\theta_1)$ is greater than $sin(\theta_2)$, the refracted angle (θ_2) will be smaller than the incident one (θ_1).

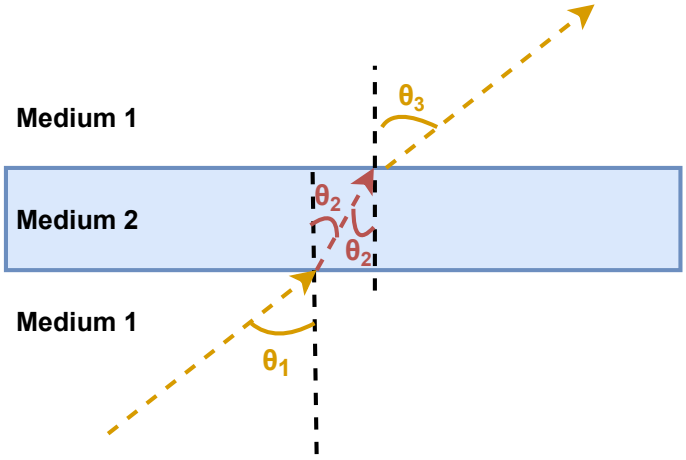

Figure 3.4: Snell-Descartes Law with a cover glass surface

The Snell-Descartes law allows finding the angle of the outgoing light (θ_3) based on the angle of the incident (θ_1) and the refracted (θ_2) rays. Below is the calculation of this angle, based on Equation 3.2.

$$\frac{\sin(\theta_1)}{\sin(\theta_2)} = \frac{n_1}{n_2} \quad \Leftrightarrow \quad \sin(\theta_2) = \frac{n_1}{n_2}\sin(\theta_1) \quad \Leftrightarrow \quad \theta_2 = \arcsin(\frac{n_1}{n_2}\sin(\theta_1))$$

Using the same logic between the incident and refracted rays, but with the refracted and outgoing rays, and the expression of θ_2 calculated above, it can be deduced that the incoming and outgoing light rays are parallel ($\theta_1 = \theta_3$).

$$\frac{sin(\theta_2)}{\sin(\theta_3)} = \frac{n_2}{n_1} \quad \Leftrightarrow \quad \sin(\theta_3) = \frac{n_2}{n_1}\sin(\theta_2) \quad \Leftrightarrow$$

$$\Leftrightarrow \quad \sin(\theta_3) = \frac{n_2}{n_1}\sin(\arcsin(\frac{n_1}{n_2}\sin(\theta_1))) \quad \Leftrightarrow \quad \theta_3 = \arcsin(\frac{n_2}{n_1}\sin(\arcsin(\frac{n_1}{n_2}\sin(\theta_1)))) \quad \Leftrightarrow$$

$$\Leftrightarrow \quad \theta_3 = \arcsin(\sin(\theta_1)) \quad \Leftrightarrow \quad \theta_3 = \theta_1$$

Thus, it is deduced that the incident rays coming from the sensor will be narrowed closer to the center with the same outgoing angle, effectively "moving forward" the cone of vision, i.e., the FoV, which in turn creates an offset in the measurements.

3.2.3 Study of light refraction influence in the sensor's measurements

In order to understand the effect of placing a glass in front of the sensor, a more detailed analysis will be performed based on the points explained in the previous sections.

The FoV angle of the VL53L1X sensor can be any value between 15 and 27 degrees, which will be represented by the α angle in this analysis. In this case, the incident ray angle is β, and the refracted angle is θ.

The λ variable will be used in the following calculations to depict the air gap between the sensor and the glass. The d_1, d_2 and d_3 represent three different distances in relation to the target: the distance to the sensor, the distance at which the sensor would need to be placed to have the same opening without a glass surface and the distance from the glass surface to the target, respectively. The μ_2 variable expresses the horizontal distance traveled by the light in the glass, and μ_1 is the horizontal distance traveled by the light after leaving the glass medium. The t variable corresponds to the glass thickness. The $d^{'}$ variable is double the distance from the normal line and the place where the photon hits the target. Lastly, Δ represents the offset that the measurement will have due to the presence of the TS **each time the photon passes through it** (the actual measurement offset is double this value because the light passes the TS twice). All of these variables can be visualised in Figure 3.5.

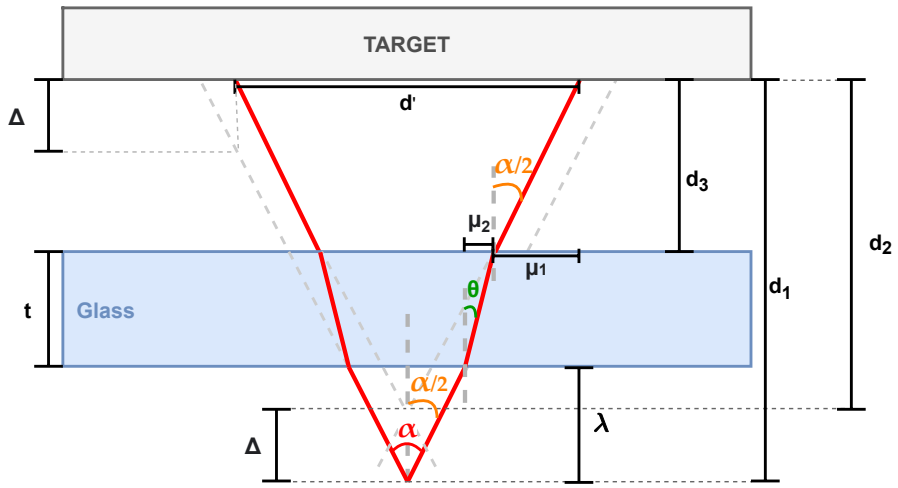

Figure 3.5: Behavior of the sensor's rays with a glass in front of it

For brevity, $\beta = \frac{\alpha}{2}$, the incidence angle, will be used from hereon.

3.2.3.1 Offset caused by light's refraction

The first thing to be evaluated is the **influence of** Δ in the distance measurement. Note that Δ is half of the offset because the light will be distorted both when going towards the target and when going back towards the sensor and the distortion is the same both ways because the refraction indices are the same. Hence, the offset is $2 \cdot \Delta$.

The Δ value is obtained by subtracting d_2 from d_1. These distances may be represented using other variables, as exhibited in the following demonstration:

$$\Delta = d_1 - d_2$$

$$\begin{cases} d_1 = \lambda + t + d_3 \\ d_3 = \frac{\mu_1}{\tan(\beta)} \\ \mu_1 = \frac{d'}{2} - \mu_2 - \lambda \cdot \tan(\beta) \\ \mu_2 = t \cdot \tan(\theta) \\ d' = 2 \cdot d_2 \cdot \tan(\beta) \end{cases} \Leftrightarrow \begin{cases} \mu_1 = \tan(\beta) \cdot (d_2 - \lambda) - t \cdot \tan(\theta) \\ d_3 = (d_2 - \lambda) - \left(\frac{t \cdot tan(\theta)}{tan(\beta)}\right) \\ d_1 = d_2 + t \cdot (1 - \frac{\tan(\theta)}{\tan(\beta)}) \end{cases}$$

$$\Delta = d_1 - [d_1 - t \cdot (1 - \tfrac{\tan(\theta)}{\tan(\beta)})] \Leftrightarrow \Delta = t \cdot (1 - \tfrac{\tan(\theta)}{\tan(\beta)})) \Leftrightarrow \Delta = t \cdot \left(1 - \frac{\cos(\beta)}{n \cdot \cos(\arcsin(\frac{\sin(\beta)}{n}))}\right)$$

$$offset = 2 \cdot \Delta = 2 \cdot t \cdot \left(1 - \frac{\cos(\beta)}{n \cdot \cos(\arcsin(\frac{\sin(\beta)}{n}))}\right) \tag{3.3}$$

This analysis shows that the distance offset only depends on the glass' thickness, the incident angle, and the TS's refraction index. The offset is not the same for every photon emitted because it depends on each photon's incidence angle. Figure 3.6 shows how the offset varies with the angle.

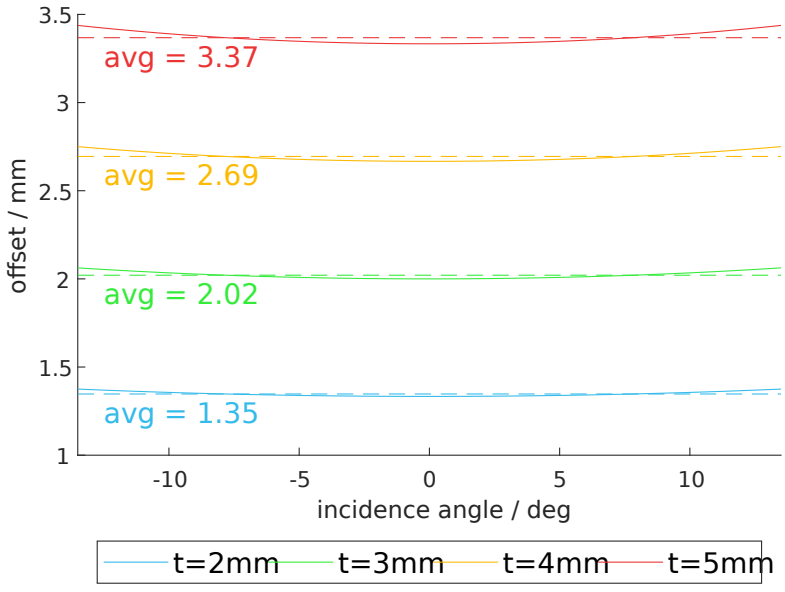

Figure 3.6: Offset caused by refraction in the TS, using n=1.5

As it can be seen in Figure 3.6, the offset is not constant for every ray being emitted, but it increases with the incidence angle. The offset's average (from 0º to 27º) is presented as a dashed

line, and it can be used as a constant approximation.

3.2.3.2 Offset caused by loss of velocity

Because the light's velocity inside the TS is not c but the sensor calculates the distance by assuming that the velocity is constant and equal to c, there is an additional measurement error.

The time that light takes to travel through the TS will be represented by the variable t_{TS}, v is the light velocity inside the TS, the distance that would have been traveled by the light in t_{TS} time will be represented by d_4. t_{TS} is deduced in Equation 3.4 and d_4 in Equation 3.5.

$$t = v \cdot t_{TS} \cdot \cos\theta = \frac{c}{n} \cdot t_{TS} \cdot \cos\theta \Leftrightarrow t_{TS} = \frac{t \cdot n}{c \cdot \cos\theta} \tag{3.4}$$

$$d_4 = c \cdot t_{TS} \cdot \cos\beta = t \cdot n \cdot \frac{\cos\beta}{\cos\theta} = t \cdot n \cdot \frac{\cos\beta}{\cos\left(\arcsin\frac{\sin\beta}{n}\right)} \tag{3.5}$$

Knowing d_4, now we can deduce the offset created by it. The offset is deduced in Equation 3.6.

$$offset = 2 \cdot \left(d_4 - \frac{t}{\cos\theta}\right) = 2 \cdot \frac{t}{\cos\theta} \cdot (n \cdot \cos\beta - 1) = 2 \cdot t \cdot \frac{n \cdot \cos\beta - 1}{\cos\left(\arcsin\frac{\sin\beta}{n})\right)} \tag{3.6}$$

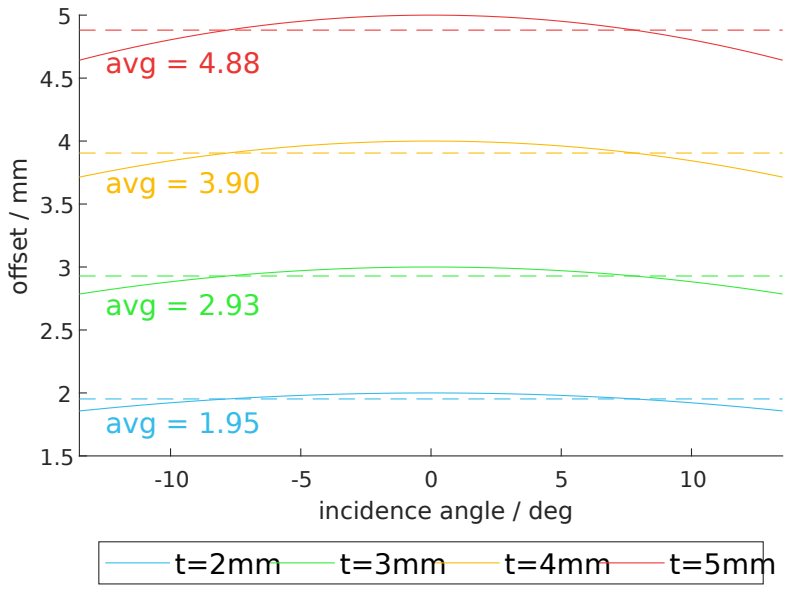

Figure 3.7: Offset caused by loss of velocity in TS, using n=1.5

3.3 Light Absorption

The amount of absorbed light depends on many conditions such as the material's spectrum (each element's isotope absorbs different wavelengths of light), atomic morphology (shape of the atoms), its thickness (amount of matter that the light traverses), and the light's intensity.

3.3.1 Influence of the reduction of light rays density

The distance offset adds another error to the sensor's measurement: the **reduction of the number of rays per distance that reach the target**, i.e., the reduction of light rays density.

Reducing the density of rays that hit the object reduces the sensor's accuracy because the probability of the emitted photons being captured by the sensor decreases with the reduction of the number of rays that hit the target. However, this type of interference does not create an offset but instead increases the chance of a measurement not sensing the target.

Also, because this probability depends on so many conditions and it is unknown whether or not the probability of missing a target is stochastic, no measurable conclusions can be taken from light absorption.

3.4 Summary of the TS influences on the sensor's performance

In summary, the theoretically measurable errors of the sensor due to the presence of a TS in front of it are the ones created by **refraction** (Figure 3.6) and **light speed** inside the TS (Figure 3.7).

Hence, the total expected offset is the sum of equation 3.3 and 3.6. The plot of that total is shown in Figure 3.8.

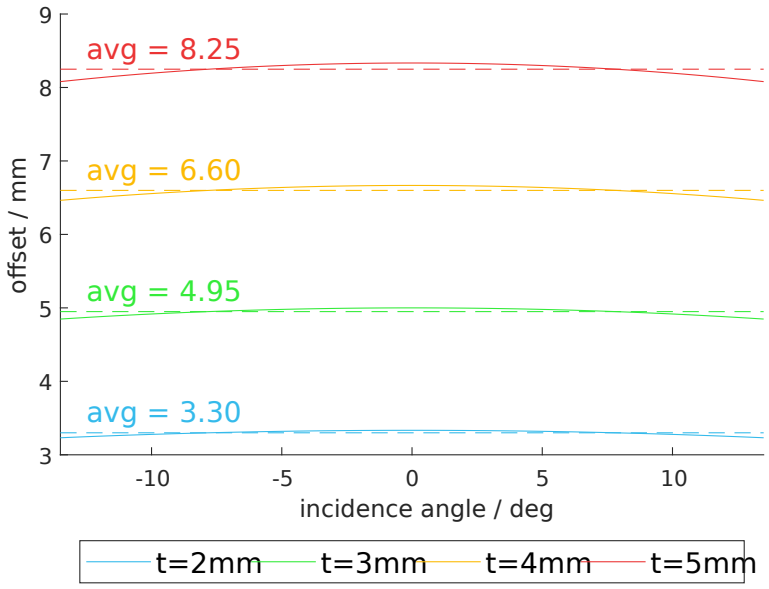

Figure 3.8: Total offset caused by TS, using n=1.5

Another interesting conclusion is that the offset is proportional to the TS's thickness (t) and, for n=1.5, can be approximated to a constant $offset \cong 1.65 \cdot t$.

Also, even though these expressions were deduced, the real offset can still be different due to other factors like the sensor's default offset (which is determined after manufacture), the object's characteristics (for example, its shape, rotation), and the environmental conditions (for example, air humidity - which increases the air's refraction index, ambient lighting). These expressions consider that the air's refraction index is 1 and they are only concerning the influence of the TS, which is why it's imperative that the calibration is always performed before being used (or at least every time the environment or the target change) in order to get more accurate measurements.

Chapter 4

Project Development

The project's development was divided into two parts: the VL53L1X performance tests and then VL53L0X's. Software development had to be done to manage the hardware and be able to do the tests.

4.1 VL53L1X

4.1.1 Project Design

The project will follow the System Breakdown Structure (SBS) displayed in Figure 4.1, and with more detail in Figure 4.2.

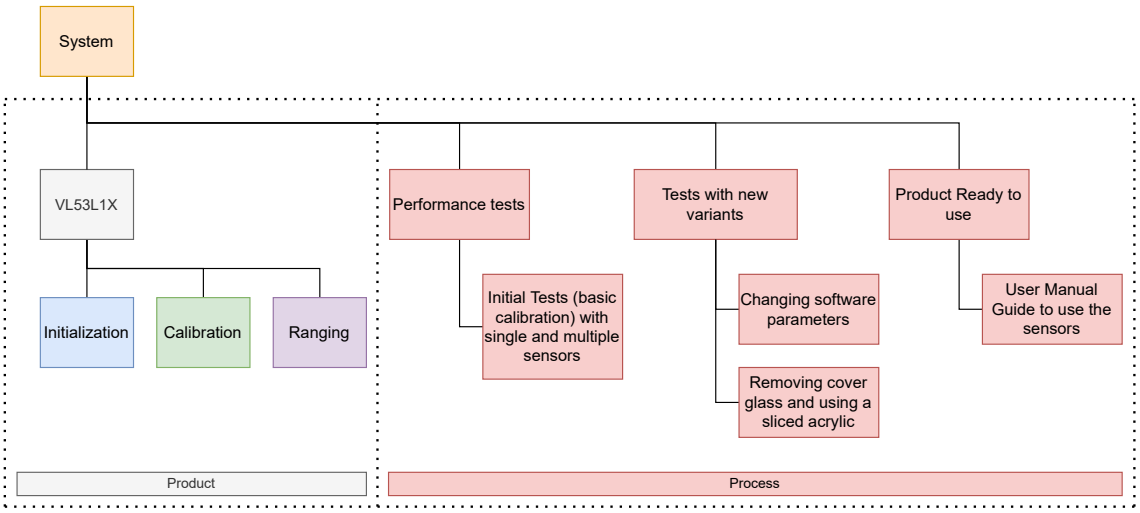

Figure 4.1: First level of System Breakdown Structure of Project VL53L1X

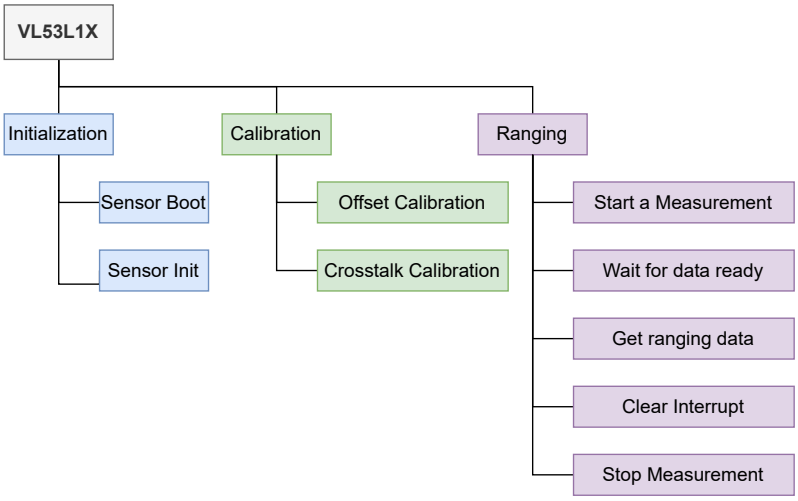

Figure 4.2: Second level of System Breakdown Structure of Project VL53L1X

4.1.2 Code Implementation

The first part of the project consisted of code implementation. Initially, the plan was to use the API in C provided by STMicroelectronics to do some preliminary tests, to understand deeply the way that the sensor works, as it includes detailed information that is not included either in the datasheet or in the user manual. Since the code provided had implementation errors and was not complete, this plan was discarded.

Even though the C API was not used in practice to operate the sensors, a lot of information was obtained from it, especially from the *VL53L1X_api.c* and *VL53L1X_calibration.c* source files. The first one includes the main functions, such as the initialization and ranging functions, as well as the setter and getter functions for the many parameters. The second file is where the functions related to the sensor's calibration are implemented, both offset and crosstalk calibration.

The *VL53L1X_api.h* has the list of I^2C's register interface where the parameters information is stored, which is essential to implement the code. It is listed in Figure 4.3:

```
#define SOFT_RESET                                          0x0000
#define VL53L1_I2C_SLAVE__DEVICE_ADDRESS                    0x0001
#define VL53L1_VHV_CONFIG__TIMEOUT_MACROP_LOOP_BOUND        0x0008
#define ALGO__CROSSTALK_COMPENSATION_PLANE_OFFSET_KCPS      0x0016
#define ALGO__CROSSTALK_COMPENSATION_X_PLANE_GRADIENT_KCPS  0x0018
#define ALGO__CROSSTALK_COMPENSATION_Y_PLANE_GRADIENT_KCPS  0x001A
#define ALGO__PART_TO_PART_RANGE_OFFSET_MM                  0x001E
#define MM_CONFIG__INNER_OFFSET_MM                          0x0020
#define MM_CONFIG__OUTER_OFFSET_MM                          0x0022
#define GPIO_HV_MUX__CTRL                                   0x0030
#define GPIO__TIO_HV_STATUS                                 0x0031
#define SYSTEM__INTERRUPT_CONFIG_GPIO                       0x0046
#define PHASECAL_CONFIG__TIMEOUT_MACROP                     0x004B
#define RANGE_CONFIG__TIMEOUT_MACROP_A_HI                   0x005E
#define RANGE_CONFIG__VCSEL_PERIOD_A                        0x0060
#define RANGE_CONFIG__VCSEL_PERIOD_B                        0x0063
#define RANGE_CONFIG__TIMEOUT_MACROP_B_HI                   0x0061
#define RANGE_CONFIG__TIMEOUT_MACROP_B_LO                   0x0062
#define RANGE_CONFIG__SIGMA_THRESH                          0x0064
#define RANGE_CONFIG__MIN_COUNT_RATE_RTN_LIMIT_MCPS         0x0066
#define RANGE_CONFIG__VALID_PHASE_HIGH                      0x0069
#define VL53L1_SYSTEM__INTERMEASUREMENT_PERIOD              0x006C
#define SYSTEM__THRESH_HIGH                                 0x0072
#define SYSTEM__THRESH_LOW                                  0x0074
#define SD_CONFIG__WOI_SD0                                  0x0078
#define SD_CONFIG__INITIAL_PHASE_SD0                        0x007A
#define ROI_CONFIG__USER_ROI_CENTRE_SPAD                    0x007F
#define ROI_CONFIG__USER_ROI_REQUESTED_GLOBAL_XY_SIZE       0x0080
#define SYSTEM__SEQUENCE_CONFIG                             0x0081
#define VL53L1_SYSTEM__GROUPED_PARAMETER_HOLD               0x0082
#define SYSTEM__INTERRUPT_CLEAR                             0x0086
#define SYSTEM__MODE_START                                  0x0087
#define VL53L1_RESULT__RANGE_STATUS                         0x0089
#define VL53L1_RESULT__DSS_ACTUAL_EFFECTIVE_SPADS_SD0       0x008C
#define RESULT__AMBIENT_COUNT_RATE_MCPS_SD                  0x0090
#define VL53L1_RESULT__FINAL_CROSSTALK_CORRECTED_RANGE_MM_SD0               0x0096
#define VL53L1_RESULT__PEAK_SIGNAL_COUNT_RATE_CROSSTALK_CORRECTED_MCPS_SD0  0x0098
#define VL53L1_RESULT__OSC_CALIBRATE_VAL                    0x00DE
#define VL53L1_FIRMWARE__SYSTEM_STATUS                      0x00E5
#define VL53L1_IDENTIFICATION__MODEL_ID                     0x010F
#define VL53L1_ROI_CONFIG__MODE_ROI_CENTRE_SPAD             0x013E
```

Figure 4.3: List of addresses VL53L1X

As aforementioned, one of the client's requirements was to use JavaScript as the programming language, since the code they developed for version 1 was in Typescript and JavaScript. The library which was used was found on GitHub [22], is written in JavaScript, and is compatible with Raspberry Pi, but it only included the basic functions for sensor's initialization and helper functions, like all the other libraries available in different languages too. The other more advanced but necessary methods (namely the calibration methods) were implemented based on the VL53L1X sensor's C API functions.

The functions initially available in this VL53L1X JavaScript library were:

writeByte(index, byte) (8-bit)
writeWord(index, data) (16-bit)
writeDoubleWord(index, data) (32-bit)
readByte(index) (8-bit)
readWord(index) (16-bit)
readDoubleWord(index) (32-bit)
sleep(ms)
sensorInit()
getSensorId()
startRanging()
getInterruptPolarity()
checkForDataReady()
waitForDataReady()
clearInterrupt()
stopRanging()
getDistance()
bootState()
getTimingBudgetInMs()
getDistanceMode()
setTimingBudgetInMs(timingBudgetInMs)
setDistanceMode()
getInterMeasurementInMs()
setInterMeasurementInMs()
getRangeStatus()

There are no calibration methods implemented, so these and other necessary methods were implemented. The final code is available for consultation on Github [23], as it is a fork from the initial library. Some of the main methods will be shown below.

The first challenge of this project was to find a way to connect multiple sensors to the microcontroller. As the connection between the sensors and microcontroller is made by only one pin via I^2C, all the devices are connected in series. This means that all sensors will be connected to that same pin but will initially have the same addresses (the default address on reset is 0x29), which does not allow to get the measurements of each sensor separately. For this, two possible solutions were thought of:

1. **Set the sensor XSHUT pin value to 0 (active low) for each sensor.**
   (a) Pros
      i. Fast
      ii. Can be an automatic start-up sequence
      iii. Does not require user intervention
   (b) Cons
      i. One more wire per each sensor
      ii. All of the sensor's XSHUT pins would have to be connected to a pin on the controller, making them parallelized instead of serialized, which would defeat the purpose of I^2C. A multiplexer could also be used, but that adds even more complexity to the system.

2. **Incrementally add more sensors to the bus and have the controller change the addresses one by one as they are added**

   (a) Pros

      i. The sensors are all connected serially

      ii. No extra wires are needed

      iii. Once initialized, the sensors are very easy to use and physically operate as they are all connected to the same bus

   (b) Cons

      i. Slow

      ii. Cannot be fully automated

      iii. Needs user intervention

The first option was used in the 3Decide project with VL53L0X. The major reason for this choice is that no user intervention is required, to make it easier for the clients to initiate the sensors.

As for this dissertation, the **second solution was chosen**. It allows the user to add as many sensors as possible (there is a limit of addresses, of course) because no more wires are needed, which makes the working environment simpler and cleaner.

The procedure to implement this in code was to change the sensor's address of each sensor and wait for it to respond to the new address. The method *changeAddress()*, displayed in Listing 4.1, performs this steps.

```
async changeAddress(newAddress) {
        await this.writeByte(VL53L1_I2C_SLAVE__DEVICE_ADDRESS, newAddress & 0xFF);
        this.address = newAddress & 0xFF;
        this.i2cWrite = util.promisify(this.i2c.write.bind(this.i2c,this.address));
        this.i2cRead = util.promisify(this.i2c.read.bind(this.i2c, this.address));
        const timeoutMs = 2000;
        const startTime = getCurrentEpochMs();
        while(true) {
            const currentMs = getCurrentEpochMs();
            if(currentMs - startTime >= timeoutMs) { break; }
            try {
                await this.waitForBooted(startTime + timeoutMs - currentMs)
                return;
            } catch (e) {
                if (e.code !== 'EREMOTEIO') { throw e; }
            }
            await sleep(1);
        }
        throw new Error('timed out while changing address');
    }
```

Listing 4.1: Change Address method

This method is called through an auxiliary method (*getSensors()*), which receives as its first argument the addresses in an array and returns an array containing all sensor's objects and their addresses. This is the method that sets the interaction with the user, using the *waitForEnter()* helper method (available in the appendix B), that waits to receive an ENTER and optionally prints an instruction in the terminal.

```
async function getSensors(addresses, changeAddresses = false) {
    const sensors = new Array(addresses.length)
        .fill(null)
        .map((_, i) => new VL53L1X({
            i2c,
            address: changeAddresses ? 0x29 : addresses[i]
        }));
    if (changeAddresses) {
        console.log(`Setting address${addresses.length > 1 ? 'es' : ''} to ${
            addresses.map(numberToHex).join(', ')}`);
        for (const i in sensors) {
            const sensor = sensors[i];
            await waitForEnter(`Press enter to set address of sensor ${i} to ${
                numberToHex(addresses[i])}...`);
            await sensor.waitForBooted();
            await sensor.sensorInit();
            await sensor.setDistanceMode(VL53L1X.DISTANCE_MODE_LONG);
            await sensor.changeAddress(addresses[i]);
        }
    } else {
        for (const sensor of sensors) {
            await sensor.waitForBooted();
            await sensor.sensorInit();
            await sensor.setDistanceMode(VL53L1X.DISTANCE_MODE_LONG);
        }
    }
    return sensors;
}
```

Listing 4.2: getSensors method

Therefore, to initialize the sensors, the user needs to add one sensor at a time, pressing ENTER after each, following the instructions shown in the terminal, as explained in Listing 4.2 and in the Usage Protocol in the appendix A. After this initialization, the sensors are correctly identified with the corresponding addresses and ready to be used.

Before starting the measurements, the sensor must be calibrated. The **calibration methods** are now going to be explained, as well as the corresponding *get* and *set* methods to read and write the values to and from the sensor's addresses, respectively.

The offset calibration method, *calculateOffsetCalibration(distance)*, is shown in (Listing 4.3).

```
async calculateOffsetCalibration(dist) {
        await this.writeWord(ALGO__PART_TO_PART_RANGE_OFFSET_MM, 0x0*4);
        await this.writeWord(MM_CONFIG__INNER_OFFSET_MM, 0x0);
        await this.writeWord(MM_CONFIG__OUTER_OFFSET_MM, 0x0);
        await this.startRanging();
        let avgDist = 0;
        for (let i = 0; i < 50; i++) {
            let tmp = false;
            while (!tmp) {
                tmp = await this.checkForDataReady();
                await sleep(1);
            }
            let distance = await this.getDistance();
            await this.clearInterrupt();
            avgDist = avgDist + distance;
        }
        await this.stopRanging();
        avgDist = avgDist / 50;
        return (dist - avgDist);
    }
```

Listing 4.3: Offset calibration method

Following the example in the sensor's API, the offset calibration method performs 50 measurements and calculates their mean value, but before starting the process, the offset value is set to zero to avoid performing a calibration on top of a previous calibration. In the sensor's datasheet, it is recommended that the *dist* value is used as 140mm, but it is kept as a variable in case the user wants to calibrate with another value. The result of the **offset compensation value** corresponds to subtracting the mean value of the 50 measurements to the *dist* value.

For the crosstalk calibration, a helper method is needed to obtain the *xcd* value. To do this, the adopted procedure was to get measurements during 10 seconds and vary the distance to the target, so that it is possible to obtain the maximum distance measured (*xcd*). After finding this value, the crosstalk compensation value is calculated through the *calculateXTalkCalibration(xcd)* method (Listing 4.4).

```
async calculateXTalkCalibration(dist) {
        await this.startRanging();
        let avgDist = 0;
        let avgSPADnum = 0;
        let avg_sr = 0;
        for (let i = 0; i < 50; i++) {
            let tmp = 0;
            while (tmp === 0) {
                tmp = await this.checkForDataReady();
            }
            let sr = await this.getSignalRate();
            let distance = await this.getDistance();
```

```
13              await this.clearInterrupt();
14              let spad_num = await this.getSpadNb();
15              avgDist = avgDist + distance;
16              avgSPADnum += spad_num;
17              avg_sr += sr;
18          }
19          await this.stopRanging();
20          avgDist = avgDist / 50;
21          avgSPADnum = avgSPADnum / 50;
22          avg_sr = avg_sr / 50;
23          const calcXtalk = 512 * (avg_sr * (1 - (avgDist/dist))) / avgSPADnum;
24          return calcXtalk;
25      }
```

Listing 4.4: Crosstalk calibration method

Both methods use basically the same logic, which is to perform fifty measurements, calculate their average, and the difference between the average and the distance passed as an argument to the method. The difference between them is that the crosstalk compensation also includes the mean value of the signal rate and the number of SPADs used. This formula is not explained either in the sensor's datasheet or User Manual.

The compensation values are then sent as arguments to their corresponding *set* methods (*setOffset(val)* and *setXTalk(val)*), completing the calibration process.

4.2 VL53L0X

4.2.1 Project Design

The project consists of creating a JavaScript file to get the measurements in a personalized way, based on the VL53L0X library implemented by 3Decide [24].

After getting the measurements, data analysis was done to understand this sensor's performance. Figures 4.4 and 4.5 show the SBS of the VL53L0X project.

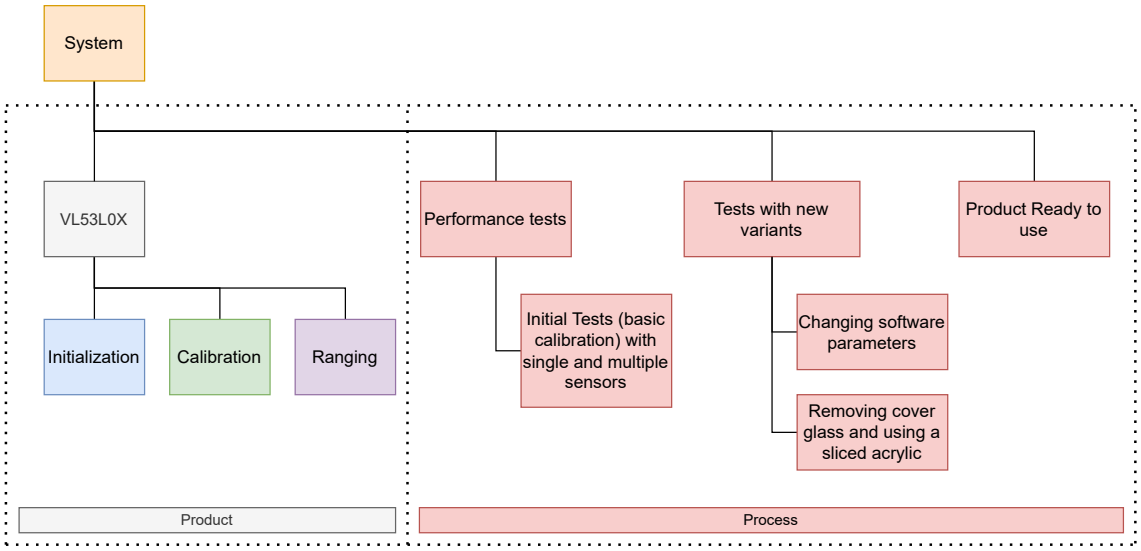

Figure 4.4: First level of System Breakdown Structure of Project VL53L0X

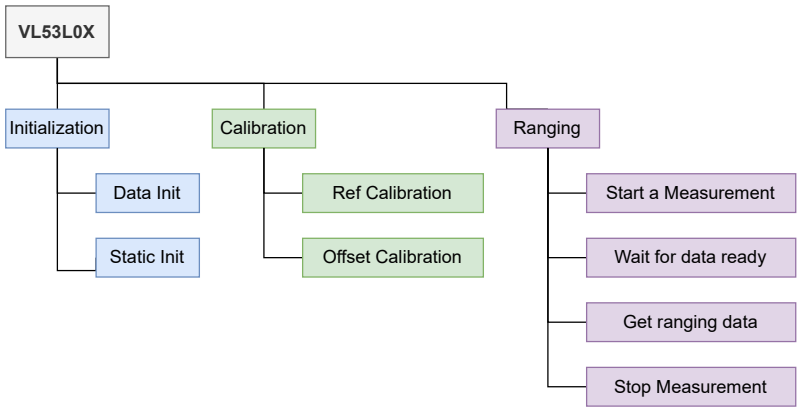

Figure 4.5: Second level of System Breakdown Structure of Project VL53L0X

4.2.2 Code Implementation

For this sensor, the calibration process is not as clear as it is for VL53L1X. The datasheet and the API user manual mention four calibration parameters (Reference SPADs, Ref (Temperature), Offset, and Crosstalk), but in the library used by 3Decide, only the Ref calibration is implemented.

After spotting this problem, some libraries in different languages (such as GoLang and Python) were searched and found, but none had implemented the calibration of the Reference SPADs, offset or crosstalk, just like in the library used by 3Decide in its previous project. This may be because of the poor information available in the sensor's API code. Therefore, it was decided to use the 3Decide's library as a base to start the code implementation.

Even though there isn't much information in the API, it is clear that the process to calibrate the offset is the same: to perform fifty measurements, calculate the mean of those measurements and subtract the calibration distance. This process was then reproduced in JavaScript without directly using the sensor's addresses and simply adding the offset value to the output measurements (as presented in Listing 4.5).

```
1  const calculateOffsetCalibration = async (sensor, sensorId, refDistance = 100) => {
2    await waitForEnter(`Press enter after placing a white obstacle at ${refDistance}
         mm to start calibrating offset...`);
3    const N = 50;
4    let sumTemp = 0;
5    for (let i = 0 ; i<N ; i++) {
6      const m = (await sensor.api.measure(sensorId))?.[sensorId];
7      if (!m) {
8        i--;
9        continue;
10     }
11     sumTemp += m;
12     console.error(m, sumTemp, sumTemp / (i+1));
13     await sleep(10);
14   }
15   const average = sumTemp / N;
16   console.error(`Uncalibrated distance is ${average}mm, which gives ${refDistance -
         average}mm offset`)
17   return refDistance - average;
18 }
```

Listing 4.5: Change Address method

The method *waitForEnter()* is the same as used in the VL53L1X (Listing B.1).

Another problem spotted about this code is that the first measurement values are always 8191, and only after some seconds do the actual measurements start appearing. This was interpreted as measurements performed while the sensor was not yet initialized.

Therefore, this problem was solved by filtering these initial values only because this value appears to represent an error in the measurement and happens, for example, when the ranging distance limit is reached. The process was to split the code into different and sequential methods, as shown in Listing B.2.

The Poller is an object created by a closure. It contains:

- *startPolling* function to start polling the sensor
- *stopPolling* function to stop polling the sensor
- *getLatestValue* function to get the most recent value polled from the sensor (or get an immediate read from the sensor if the poller isn't ranging. if there isn't a polled value yet or if this function is called before the poller gets a new value from the sensor)
- *offsetCal* property to get and set the offset calibration
- *sensor* property to be able to get the sensor from the poller's object

Whenever it is necessary to calibrate the sensor, the offset calibration method should be called as follows: *poller.offsetCal = await calculateOffsetCalibration(vl53l0x, sensorId);*.

4.3 Arquitecture

For the implementation of this project, it is needed both hardware and software development. Introducing the procedures required to solve this dissertation's problem: firstly, the connections between the sensor and a Raspberry Pi using I2C communication will be assembled. After that, it will be developed code in JavaScript to control the Raspberry Pi.

In Figure 1.1 is a photo of one of the partner company's projects. In this project, each "button" corresponds to a part of a video displayed on a tablet. For example, when we select button 2, the video skips/returns to part 2 of the video. To use the button, we must place a hand in front of the sensor for a certain number of seconds. When we do this, the LEDs positioned around the sensor in a circle light up one by one until they all are on, which means the time to select the button has passed, and the option was chosen successfully. In Figure 1.2, there is a close-up photo of one of the buttons, where the VL53L0X sensor can be seen in more detail.

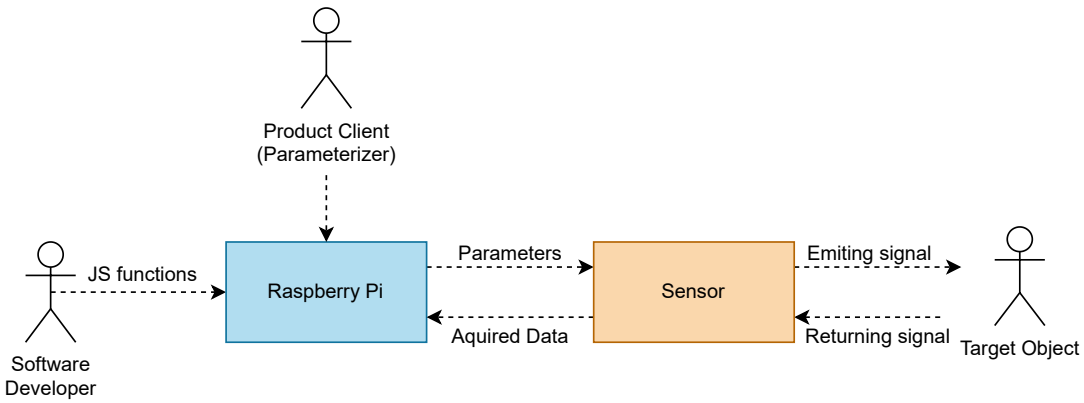

Figure 4.6: Project Arquiteture

Chapter 5

Tests and Results

Different test conditions were used to describe the influence of different lightning intensities, the thickness of the TS, and the surface colors of planar objects. In some tests, acrylic with a cut will be used. To clarify and identify the TSs that are not sliced, they will be referred to as "unsliced TS". All of the tests use the long distance mode.

The offset and crosstalk parameters were calibrated as recommended by ST in a dark environment with a gray target for VL53L1X and a white target for VL53L0X for each TS thickness. All the tests followed this procedure.

The JavaScript test scripts captured and stored the measurement data in CSV files. After that, the data was organized as needed to plot the charts with Matlab scripts.

Figure 5.1 shows how the connections with the Raspberry Pi were made, as shown in Figure 2.22.

Figure 5.1: Connections to Raspberry Pi

The final tests setup is displayed in Figures 5.2 and 5.3.

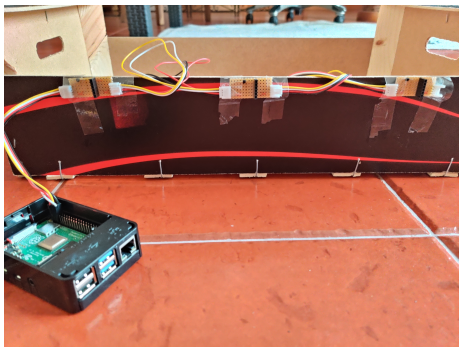

Figure 5.2: Tests setup (side view)

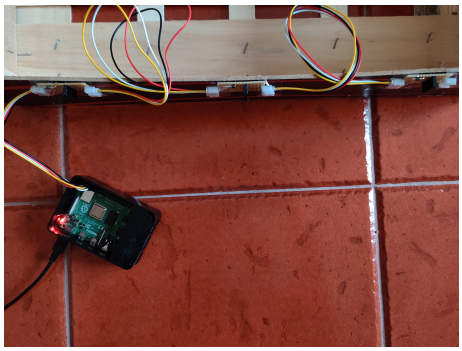

Figure 5.3: Tests setup (top view)

5.1 VL53L1X

The test conditions are the following:

- Using default conditions
  - With three sensors in the dark with three target colors (white, gray and black)
  - With one sensor in the dark with three target colors (white, gray and black), to understand the influence of using multiple sensors in the measurements
  - With one sensor in an "office" environment with a gray target, to understand the influence of the ambient light conditions on the measurements
- Changing the sensor's parameters
  - Reducing the FoV
  - Increasing the Timing Budget with three target colors (white, gray and black)
- Changing the hardware conditions using a gray target
  - Removing the cover glass only
  - Using a sliced acrylic only
  - Removing the cover glass and using a sliced acrylic
- Using a hand as a target
  - Without TS
  - Using the best solution found to add a TS

Before starting the performance tests, the variance of the offset calibration values depending on the sensor's position in the setup had to be analyzed. The three sensors were placed on the setup supports, and their positions were rotated, meaning that the sensors got tested in three dispositions: 1-2-3, 3-1-2, and 2-3-1. Then, the sensors were initialized and calibrated in each position without any TS in front. In absolute value, the crosstalk calibration values were always smaller than 0.03

mm, so this calibration type does not vary with the position. Hence, only the offset calibration values are used in this analysis. Those values are presented in Table 5.1.

Table 5.1: Offet values (in mm) of the three sensors in different setup positions

Sensor ID \ Setup Pos	1	2	3
1	-13.4	-12.9	-13.2
2	-8.9	-9.1	-8.7
3	7.5	7.9	7.4

From the small variations in the offset values depending on the position, an assumption was made that **a sensor's offset and crosstalk calibration values were independent of its position in the setup**. This assumption was maintained for the rest of the tests done in the context of this thesis.

5.1.1 Three sensors with default configurations in a dark environment

The default configurations are presented in Table 5.2:

Table 5.2: Default configurations of the VL53L1X sensor

Distance Mode	Timing Budget	FoV
Long	100 ms	Maximum (27°)

The first tests were performed with **three sensors** at the same time, all aligned and placed at the same distance to the target, in **dark conditions** for white, gray, and black targets. The sensors were placed on a surface perpendicular to the floor, separated 8cm from each other. These will allow an understanding of the influence of different material thicknesses and target colors.

The calibration's compensation values, calculated by the sensors with the calibration methods, for the four thicknesses of glass and acrylic are presented in Table 5.3 below.

Table 5.3: Calibration parameters for the three VL53L1X sensors with unsliced acrylics and glasses

Material	Thickness (mm)	Offset (mm)			Crosstalk (cps)		
		Sensor 1	Sensor 2	Sensor 3	Sensor 1	Sensor 2	Sensor 3
None	**-**	-8.840	-9.879	-13.159	0.03177	0.015848	0.008124
Acrylic	**2**	120.08	58.540	54.879	76.654	56.141	49.359
	3	121.62	87.34	66.760	15.47150	14.08751	10.55782
	4	122.40	90.46	80.680	27.44925	26.20048	46.36162
	5	140	130.3	90.14	45.10615	30.85732	31.91615
Glass	**2**	127.78	67.98	52.44	14.69328	0.313766	0.543167
	3	129.74	98.66	67.7	32.59392	8.824483	7.626554
	4	108.42	75.22	54.379	44.56362	4.418694	7.876633
	5	109.46	90.22	83.22	31.03078	10.37374	14.74506

Through the information in Table 5.3, some aspects of the sensor's performance can be predicted: globally, the thicker the TS, the greater both offset and crosstalk compensation values; the offset values are similar in both materials, which suggests that the offset depends mostly on the thickness of the TS and that both materials have very close refraction indices, even though the offset values of the glass are slightly lower, suggesting that the glass' refraction index is lower than the acrylic's. The differences between sensors could be due to slightly different inclinations or structural differences in the devices (the real reason is presented in the paragraph right after Figure 5.5).

Understanding the difference between the calibration values with and without a TS in front is essential. The offset calibration is done at 140mm of the target. The values without a TS are around $\pm$10mm, but with a TS, the values increase almost ten times to values closer to 140mm. This means that in order to measure 140mm, the sensor has to add those values to its original measurements, suggesting that it only "sees" a few millimeters in front.

As for the crosstalk values, when there is no TS in front, the values are approximately zero. These values rise considerably when a TS is added, showing that the TSs significantly impact the sensor's measurement capacity.

These values prove that VL53L1X **can not function behind a TS** in these conditions. This conclusion will also be clear in Figures 5.4 and 5.5. The results showed that the crosstalk effect caused mainly by the reflection in the TSs was too high to be calibrated by the basic crosstalk calibration provided in the sensor's API. This is because it was implemented to compensate only the crosstalk caused by the small cover glass placed on top of the sensor, which is also cut in the middle to separate the emitters from the receptors. **The black line in all scatter plots represents the ideal curve of measurements.**

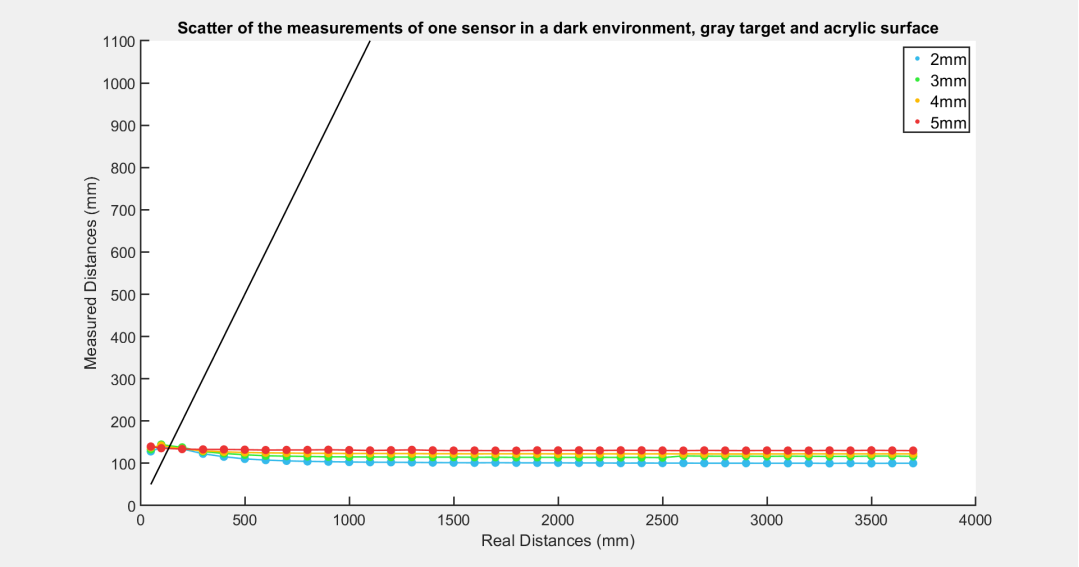

Figure 5.4: Scatter of the data (in mm) of 3 VL53L1X sensors in a dark environment, gray target and unsliced acrylic surface

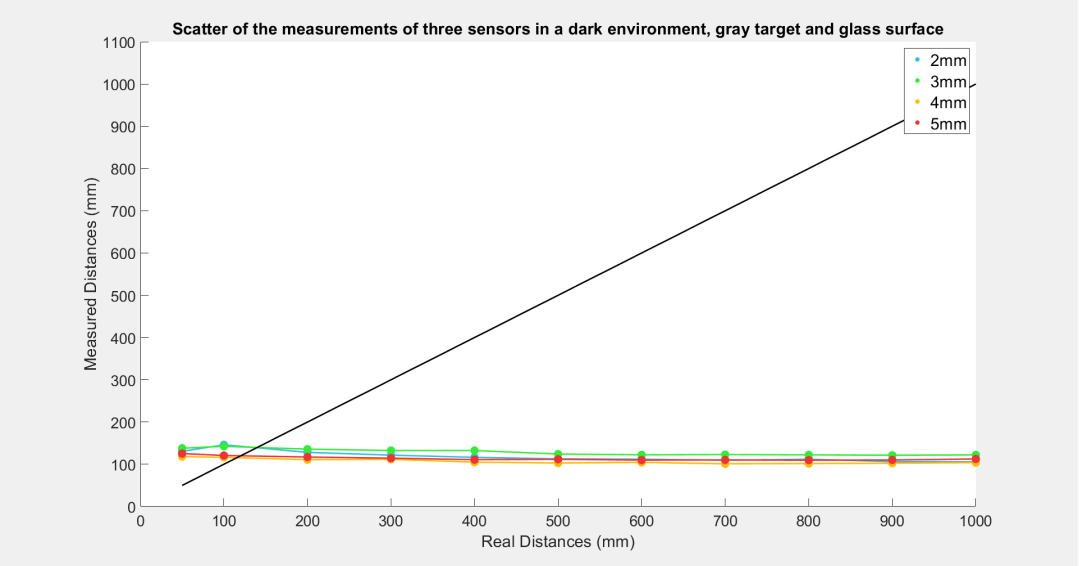

Figure 5.5: Scatter of the data of 3 VL53L1X sensors in a dark environment, gray target and unsliced glass surface

From these figures, it is clear that the sensor does not work correctly behind a TS due to the reflection on the TS. Also, due to these non-viable measurements, all the other tests made with TS in these conditions were discarded from the analysis. The calibration values in Table 5.3 are so discrepant because the measurements' data is invalid, which shows that the sensor is unusable.

5.1.1.1 Without TS

The following graphics compare the measurements when the color is changed. This comparison is composed of three different graphics: a scatter plot with the measurements of the three sensors separately (Figures 5.6, 5.9 and 5.12), and a boxplot of the measurement accuracy (Figures 5.7, 5.10 and 5.14) and a histogram of the probability of the relative error (Figures 5.8, 5.11 and 5.13) of all data of the three sensors.

Gray target

Figure 5.6 shows that, for all sensors, the measured distances are very close to the real distances until their maximum values (around 2.5 meters). After this point, the measurements are no longer accurate, because in this zone, for the same measured distance there is more than one real distance value. For example, approximately at 2400mm of ranging distance in sensor 1, the real distance may either correspond to 2400mm or 3500mm. This conclusion can also be reached through the information in Figure 5.7, where the measurement accuracy is represented. It is clear that above roughly 2500mm, the measurement accuracy starts to decrease significantly, which explains the relative error histogram bars below zero (Figure 5.8).

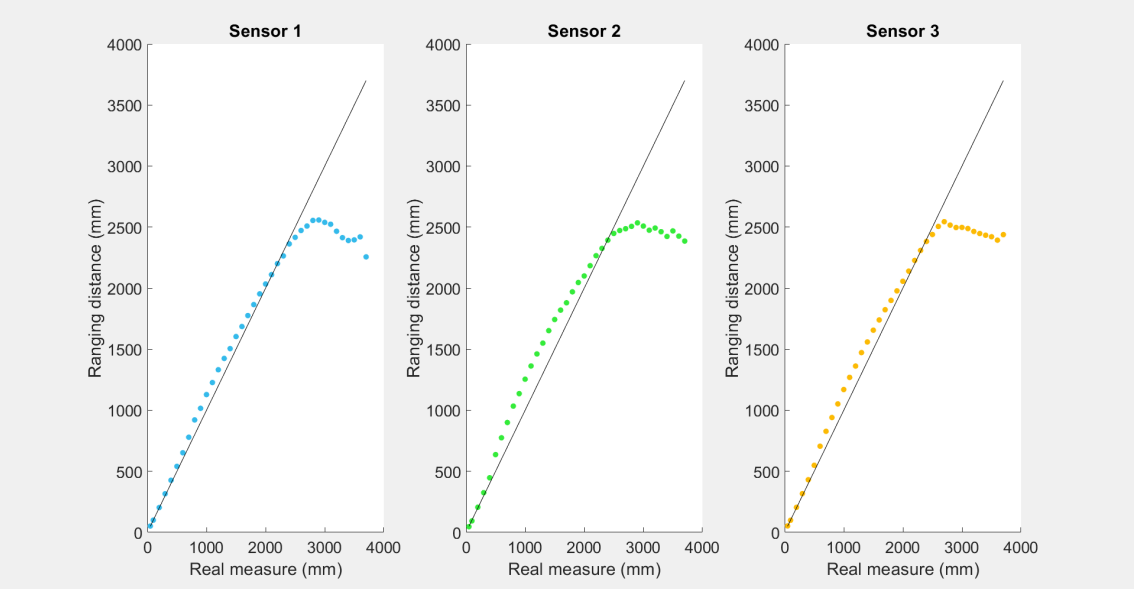

Figure 5.6: Scatter of the data (in mm) of 3 VL53L1X sensors in a dark environment, gray target, without TS

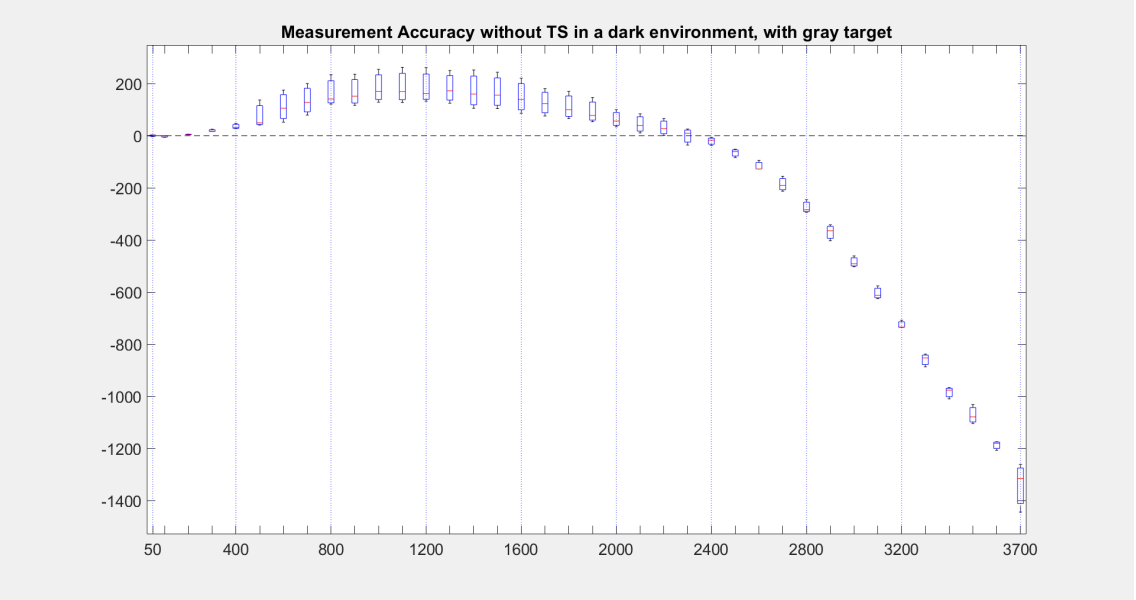

Figure 5.7: Boxplot of the measurement accuracy (in mm) of the data of 3 VL53L1X sensors in a dark environment, gray target, without TS

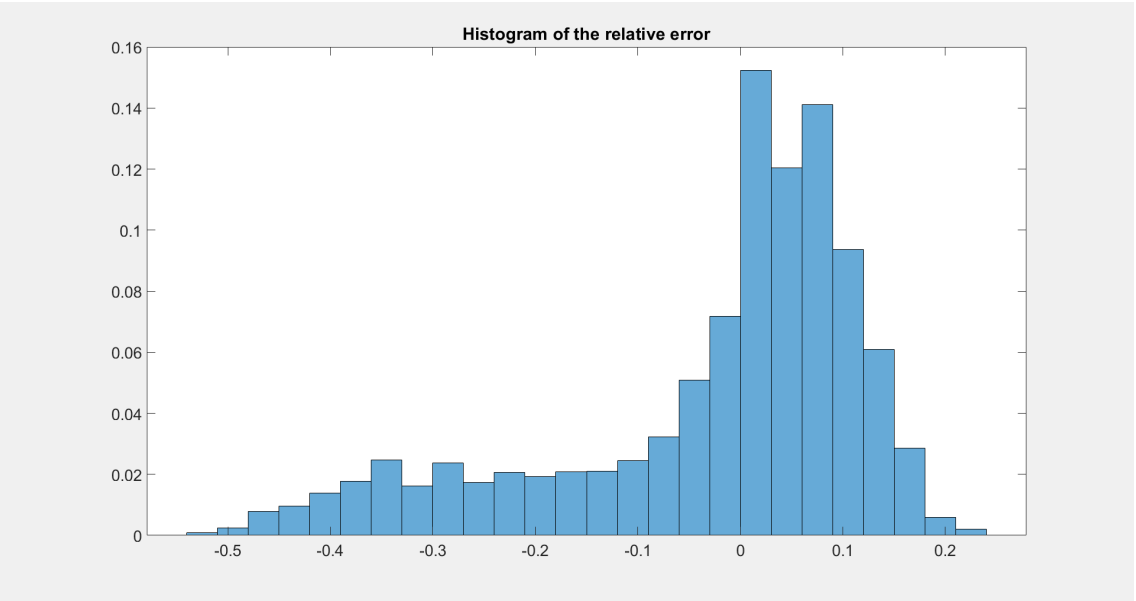

Figure 5.8: Histogram of the relative error of 3 VL53L1X sensors in a dark environment, gray target, without TS

To clarify, as described on the sensors' datasheets, the accuracy corresponds to the mean value of the measurements minus the real value. The boxplot has a box per real distance, from 5cm to 3.7m. In each box, there is a central mark, that represents the median, and a top and a bottom edge, that represents the 25th and 75th percentiles, respectively. The outliers are plotted separately using the '+' marker symbol, but in this case, they were not plotted by choice. The whiskers extend to the most extreme non-outliers data points.

Black target

Comparing Figure 5.9 with Figure 5.6 it is clear that changing the color of the target influenced the measurements a lot, specially for smaller distances, as presented in the accuracy boxplot (Figure 5.10). It is to point out that it reduced the maximum distance measured in all sensors. This variation is also evident in the histogram of the relative error (Figure 5.11), where the error above zero increased compared to the error with a gray target. This behavior is unfortunate, but also predictable because darker objects absorb more light, so the probability of a photon returning to the sensor is lower. Thus, the measurements for each real distance are much less precise, which can be seen in the size of each box in Figure 5.10 when compared to the boxes' sizes in Figure 5.7.

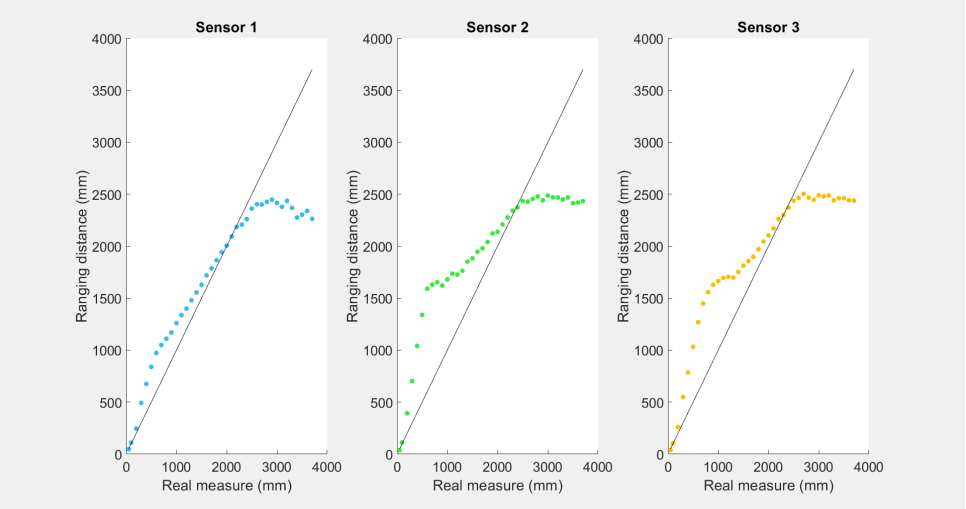

Figure 5.9: Scatter of the data (in mm) of 3 sensors in a dark environment, black target, without TS

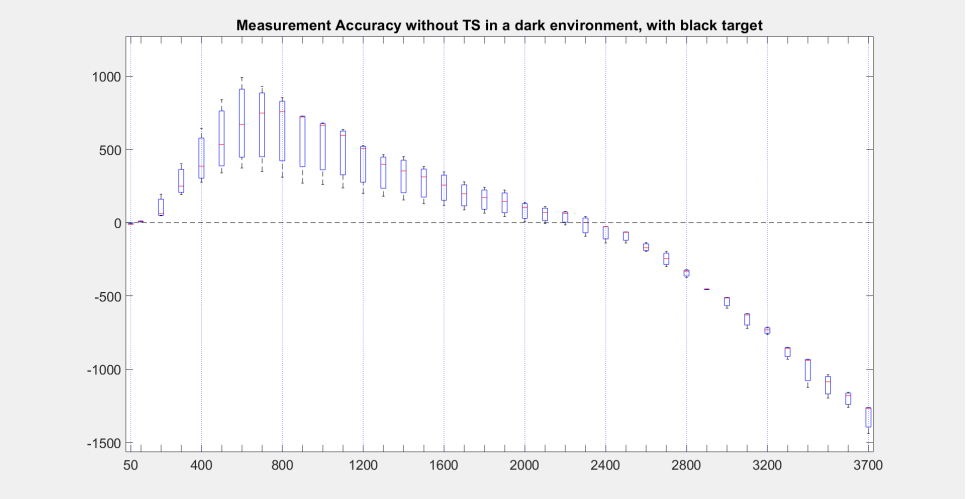

Figure 5.10: Boxplot of the measurement accuracy (in mm) of the data of 3 VL53L1X sensors in a dark environment, black target, without TS

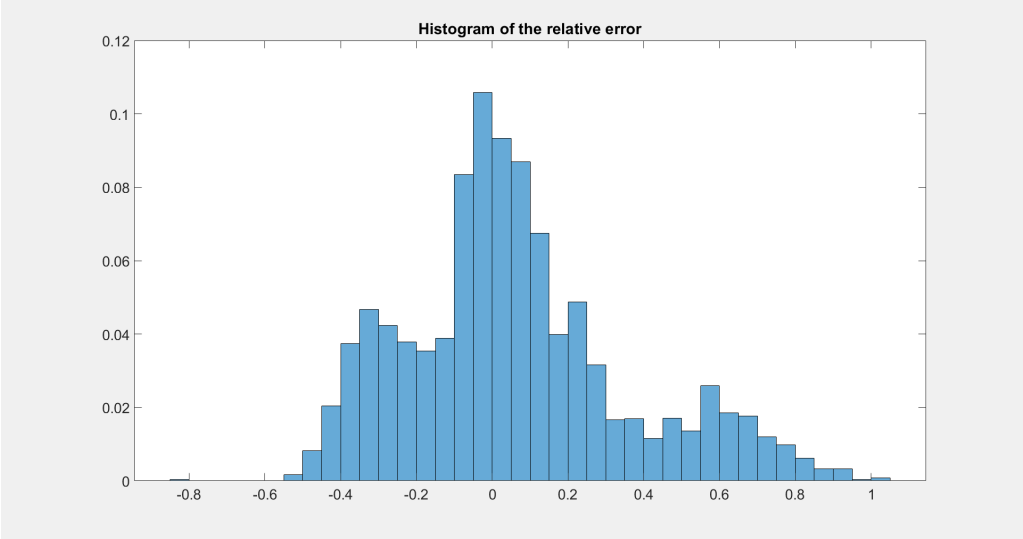

Figure 5.11: Histogram of the relative error of 3 VL53L1X sensors in a dark environment, black target, without TS

White target

White objects are more reflective, so the measurements with a white target (visible in the scatter plot in Figure 5.12) are expected to be the most accurate. The sensors' performance with a white target is very similar to the one with a gray target, as proved both by the relative error histogram in Figure 5.13 and by the boxplot of the accuracy in Figure 5.14, which are very similar to the ones in Figures 5.8 and 5.7, respectively. The maximum distance measured is very close to the one measured with the gray target. Even though white objects are more reflective than gray, the sensor's calibration was performed with the gray target, so the measurements with a gray target are expected to be more accurate.

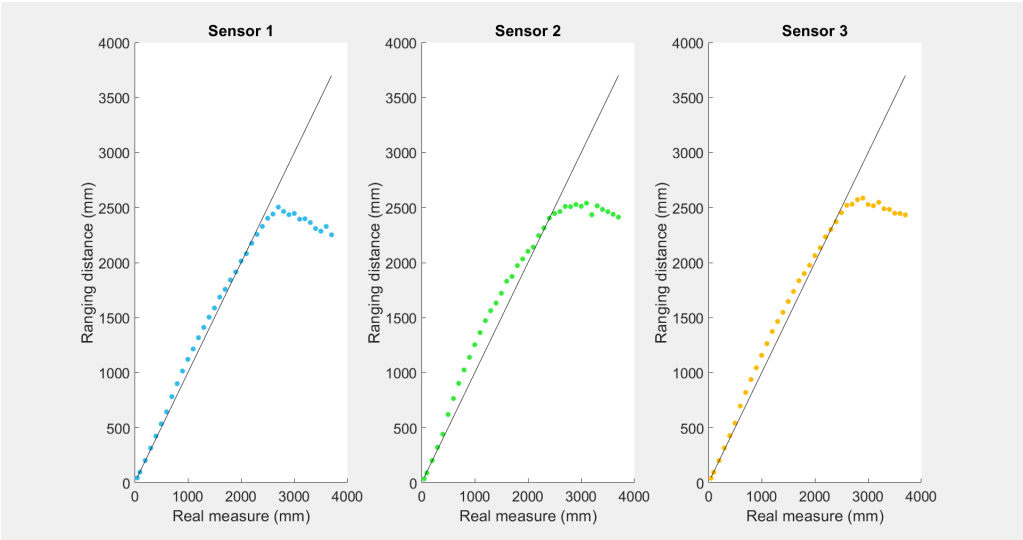

Figure 5.12: Scatter of the data (in mm) of 3 VL53L1X sensors in a dark environment, white target, without TS

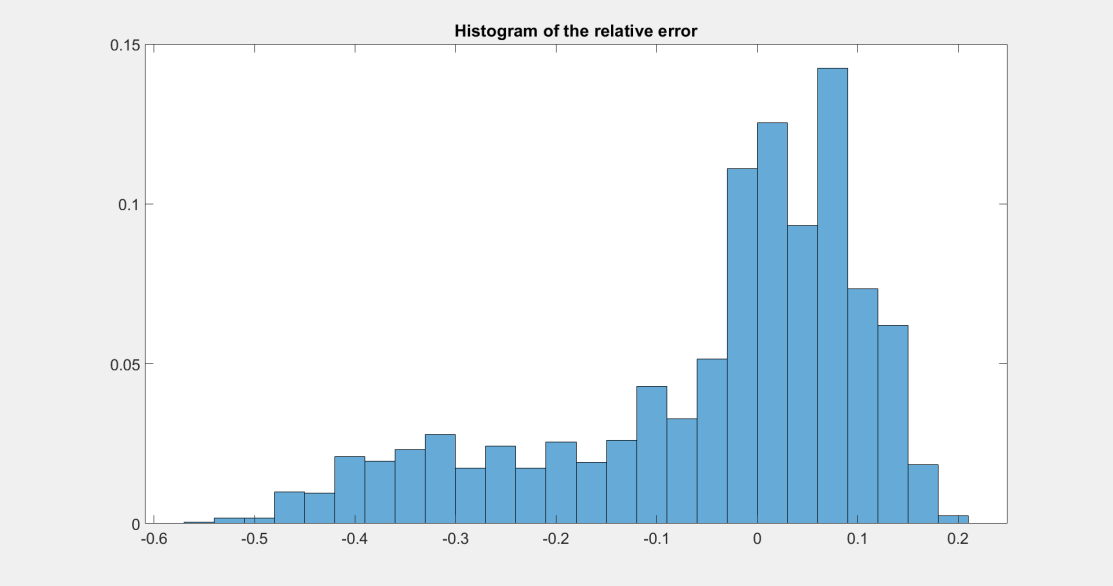

Figure 5.13: Histogram of the relative error of 3 VL53L1X sensors in a dark environment, white target, without TS

These results demonstrate that, in these conditions, the sensor should not be used in a range above 2.5 meters, which is also visible through the measurement accuracy (Figure 5.14).

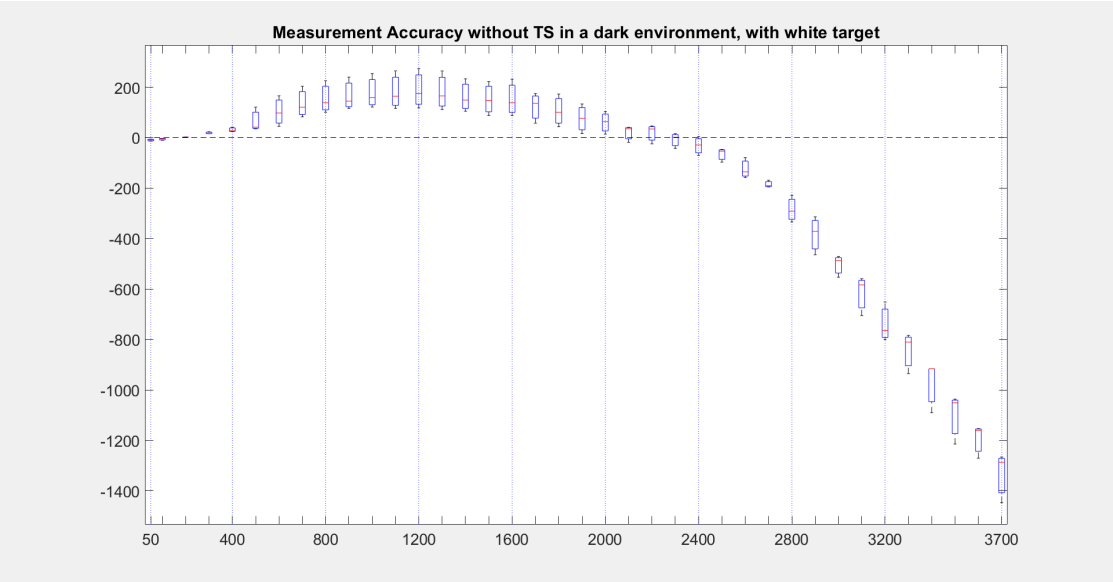

Figure 5.14: Boxplot of the measurement accuracy of the data (in mm) of 3 VL53L1X sensors in a dark environment, white target, without TS

5.1.2 Single sensor with default configurations in a dark environment

5.1.2.1 With TS

To prove that the interference in the measurements of the three sensors being used simultaneously with a TS (Figure 5.5, for example) is indeed caused by reflection and not by crosstalk from other sensors, **the same tests were done but for a single sensor**. The sensor used was sensor 3.

The calibration values for a single sensor are presented in Table 5.4.

Table 5.4: Calibration values for 1 sensor with unsliced acrylics and glasses in dark conditions

Material	Thickness (mm)	Offset (mm)	Crosstalk (cps)
None	-	-18.5	0.00459
Acrylic	**2**	41.1200	3.3549
	3	48.2399	-0.5751
	4	83	21.182
	5	81.2	17.5094
Glass	**2**	45.3199	3.07643
	3	50.9800	8.62817
	4	102.28	9.472145
	5	102.2	35.26364

Comparing these values with the ones in Table 5.3 it can be seen that especially the crosstalk values decreased considerably, which suggests that there was some interference between the sensors. However, Figures 5.15 and 5.16 show that the performance is similar to the measurements with three sensors. The 4mm and 5mm results are almost constant and have higher values, due to their higher offset. The other two thicknesses are more variant but still are not viable.

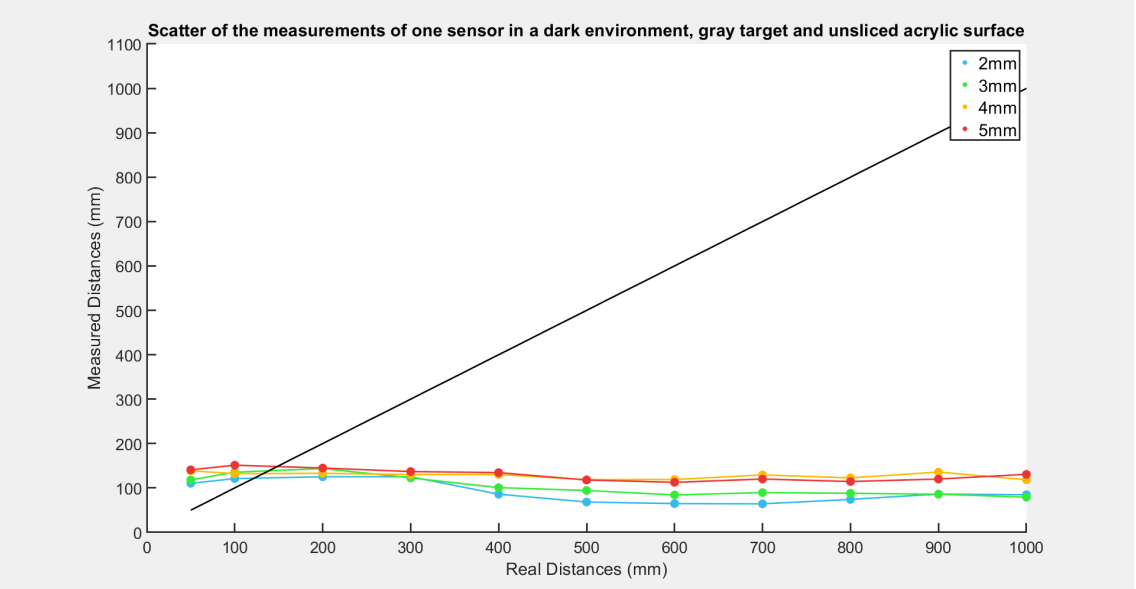

Figure 5.15: Scatter of the data (in mm) of one VL53L1X sensor in a dark environment, gray target and unsliced acrylic surface

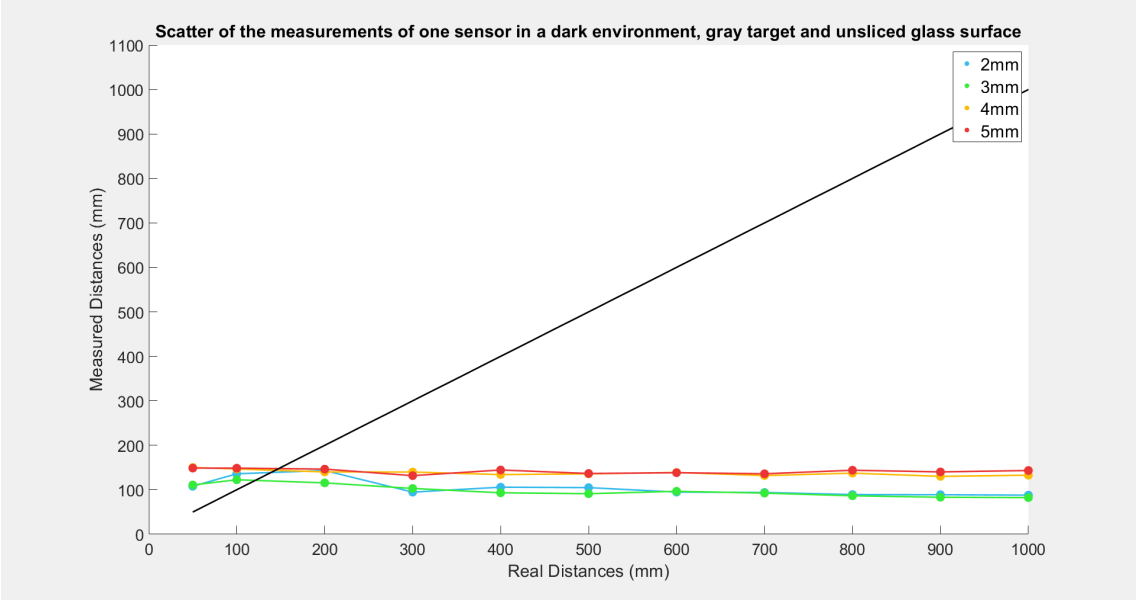

Figure 5.16: Scatter of the data (in mm) of one VL53L1X sensor in a dark environment, gray target and unsliced glass surface

5.1.2.2 Without TS

Gray target

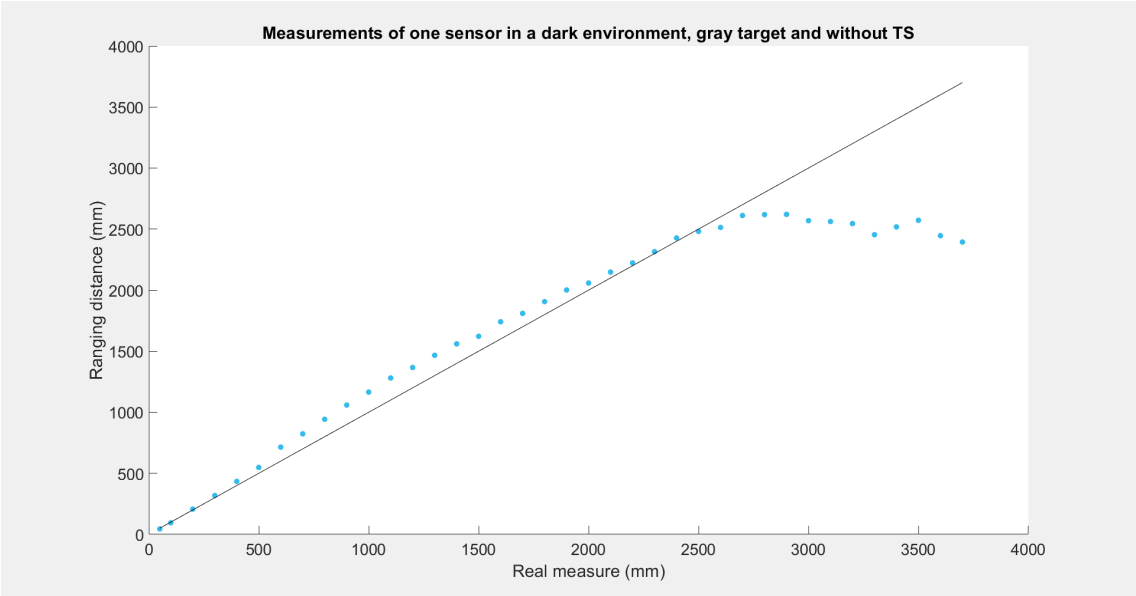

Figure 5.17: Scatter of the data (in mm) of one VL53L1X sensor in a dark environment, gray target and without TS

Black target

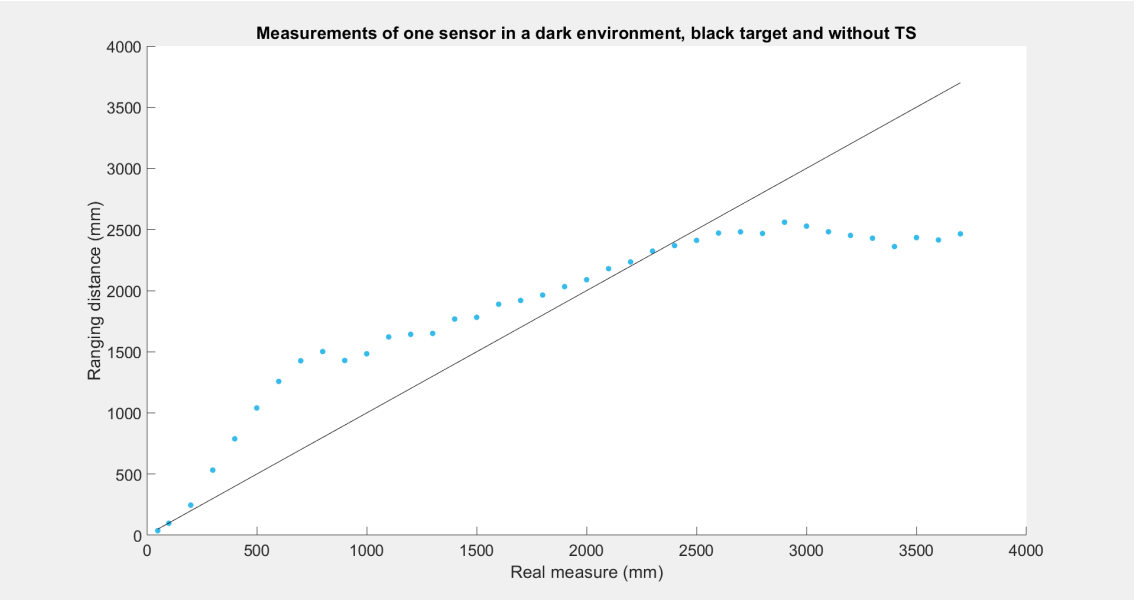

Figure 5.18: Scatter of the data (in mm) of one VL53L1X sensor in a dark environment, black target and without TS

White target

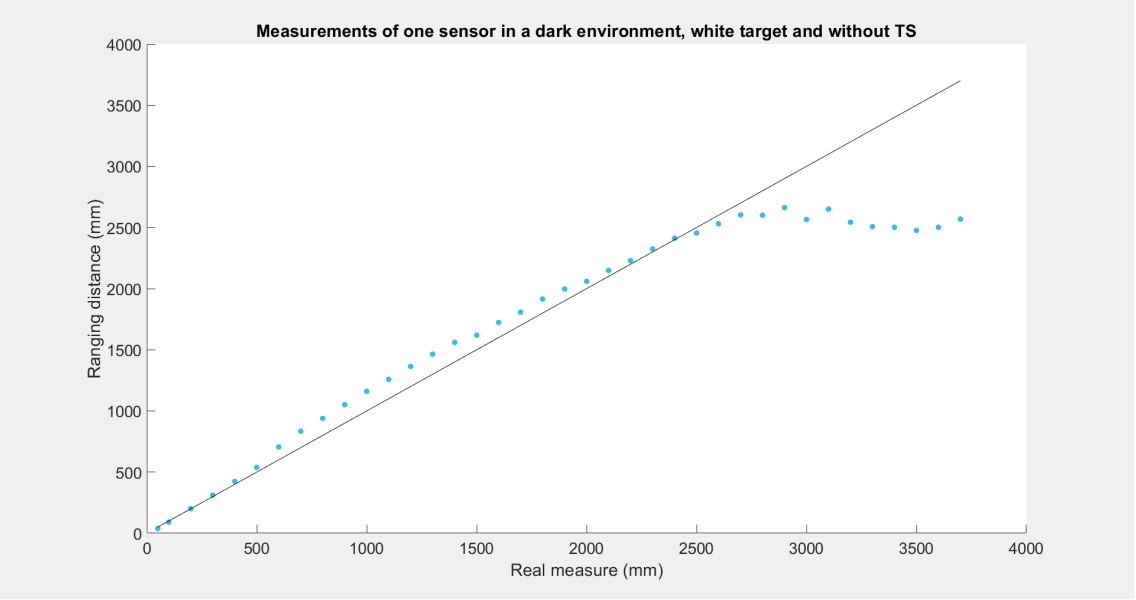

Figure 5.19: Scatter of the data (in mm) of one VL53L1X sensor in a dark environment, white target and without TS

Figures 5.17, 5.18 and 5.19 show that the sensor can measure with good accuracy until 2.60m, 2.55m and 2.66m, respectively, proving, once more, that the color has influence also in the maximum ranging distance. The scatters of the gray and white targets show the sensor has got almost

linear behavior until its maximum measure. The scatter of the black target shows greater randomness, which lines up with the conclusions taken previously about this target color.

Comparing these scatter plots (Figure 5.17, Figure 5.18 and Figure 5.19) with the previous tests (Figure 5.6, Figure 5.9 and Figure 5.12) with three sensors, it can be seen that the graphics are similar, proving that the influence between sensors in the measurements is low and that multiple sensors can be used simultaneously without costing accuracy to the measurements.

5.1.3 Tests with default configurations and medium light conditions

The subsequent tests were performed with a **single sensor with "office light"**, where the ambient light is higher, to understand the influence of the ambient light in the measurements. Because the calibration is performed in the dark, as per the documentation's suggestion, its values are the same as the previous ones (Table 5.4).

The test conditions are one sensor, medium ambient light, i.e., "office with LED lights", and no TS in front of the sensor. The previous tests proved that having a TS in front of the sensor breaks its correct functionality.

5.1.3.1 Without glass or acrylic

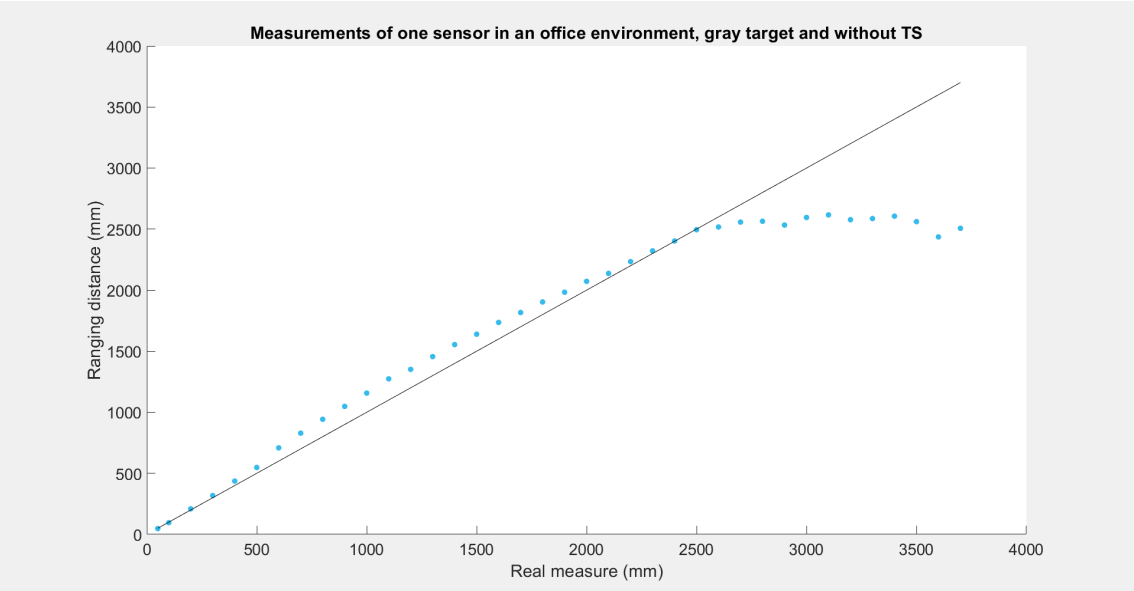

Figure 5.20: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target, without TS

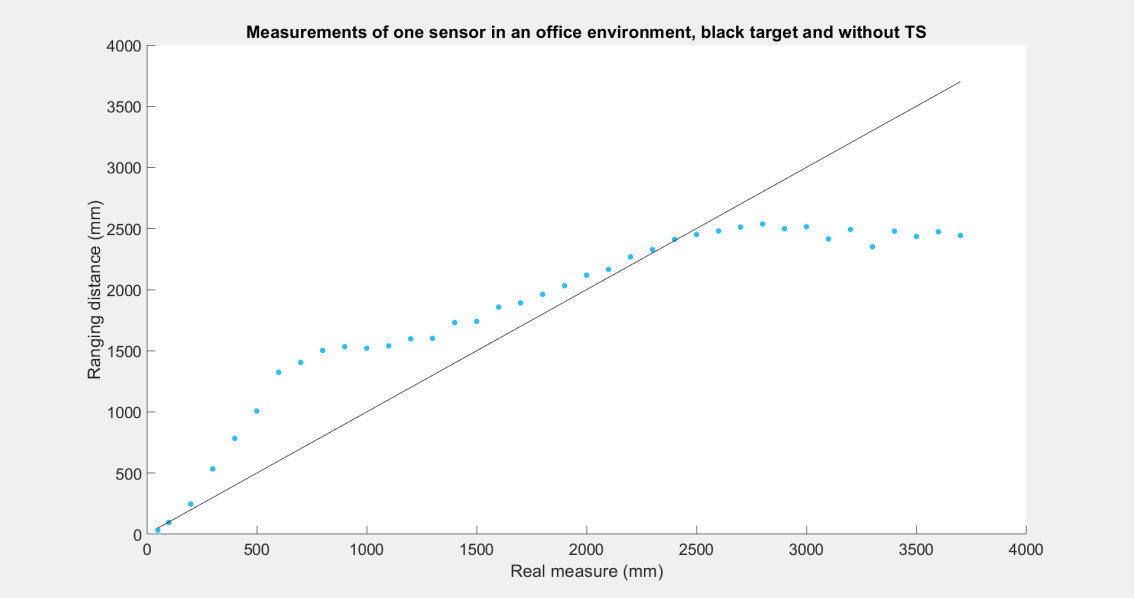

Figure 5.21: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, black target, without TS

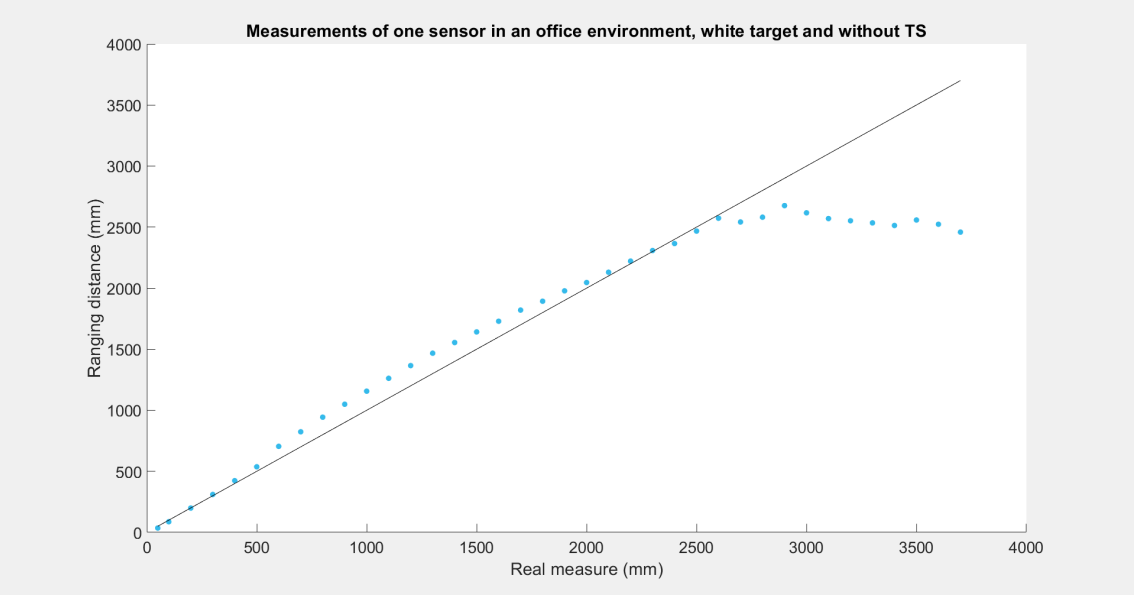

Figure 5.22: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, white target, without TS

The results shown in Figures 5.20, 5.21, and 5.22 are practically the same as the results shown previously for a dark environment, which proves that the sensor is not very sensible to the visible light conditions. The 3Decide interactive kiosk is to be used in this type of environment (indoors, with "office" light). Because it has now been proven that the visible light conditions do not influence the measurement accuracy, and medium light will be the lighting conditions where the client's project will be used, the following tests will always be performed with this ambient light.

5.1.4 Changing sensor parameters

The tests done previously showed the performance of the sensor with the default configurations and they demonstrate that even with the calibration process, the sensor cannot measure behind a TS.

In this subsection, the influence of varying some internal parameters of the sensor (FoV and timing budget) will be studied, to understand if any of these changes can improve the sensor's accuracy when used behind a TS.

5.1.4.1 Minimum FoV

As it is predictable and proven by preliminary tests, changing the FoV requires new calibration, but even after this process, the measurements were not accurate nor satisfactory, even when there was no TS placed in front of the sensor.

As aforementioned, the maximum FoV of 27º is reached with a ROI of 16x16 SPAD array. At first, the minimum value was tested, changing the ROI size to 4x4, which was shown to not have accurate measurements at all. For short distances, the values measured were close to, or even zero, and for longer distances, there was a huge error of 50cm to 1m. The ROI size 8x8 was also used and the results were better, but still not viable, so this parameter was kept as its maximum size, and the tests with TS were not performed as they would have yielded the same results.

5.1.4.2 Increased timing budget

It is known from the datasheet that a higher timing budget theoretically increases the range and the precision of the measurements, so this parameter value was increased to understand if it could improve the measurements with glass and acrylic surfaces. The possible values of the timing budget go from 20ms to 1000ms, and the ones used in these tests were 200ms and 500ms. The higher value of the TB was not used because, not only would it significantly increase the power consumption in production, but also because this significantly increases the time that each measurement takes, which would make the calibration process and the tests (that take many samples) much slower by several orders of magnitude.

The default TB value is 100ms, so measurements with this TB have already been done in the previous subsections.

Timing Budget of 200ms

First, the Timing Budget value was increased to 200ms and the results proved that the accuracy without TS improved, so this parameter will be tested with glass and acrylic in front of the sensor. A new calibration was performed, to make sure that the measurements with a different TB were well calibrated. The calibration values are presented in Table 5.5.

Table 5.5: Calibration parameters values for 1 sensor with unsliced acrylics and glasses of 2 and 3mm, with TB 200ms

Material	Thickness (mm)	Offset (mm)	Crosstalk (cps)
None	-	-16.560	0.00143
Acrylic	**2**	42.58	4.40626
	3	87.22	20.9150
Glass	**2**	56.379	11.3138
	3	68.96	33.0958

Gray target with TS

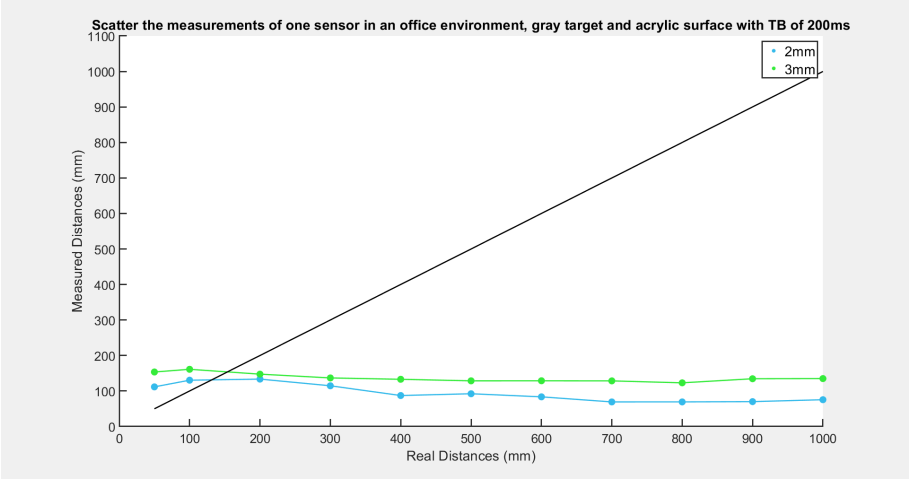

Figure 5.23: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target and unsliced acrylic surface, with TB of 200ms

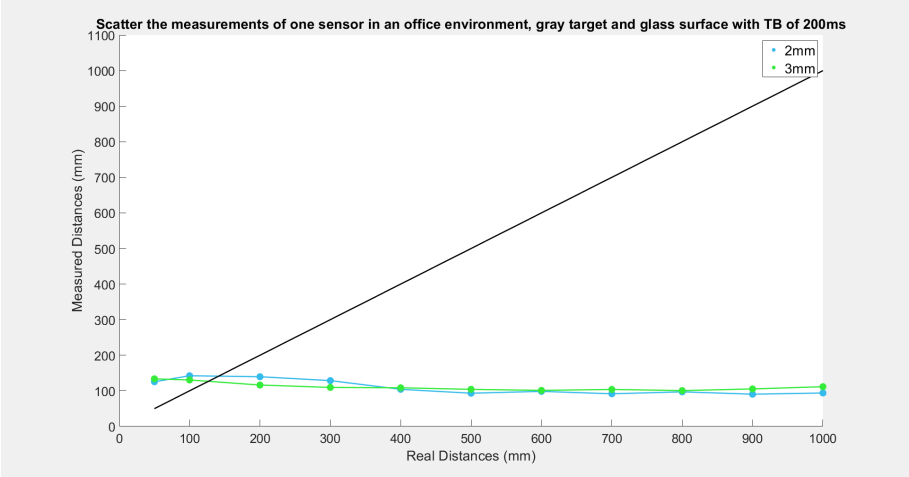

Figure 5.24: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target and unsliced glass surface with TB of 200ms

Unfortunately, as shown in Figures 5.23 and 5.24, changing the TB to this value didn't improve much the sensor's performance behind a TS. Thus, **no TS will be used in the tests for the other target colors**.

Gray target without TS

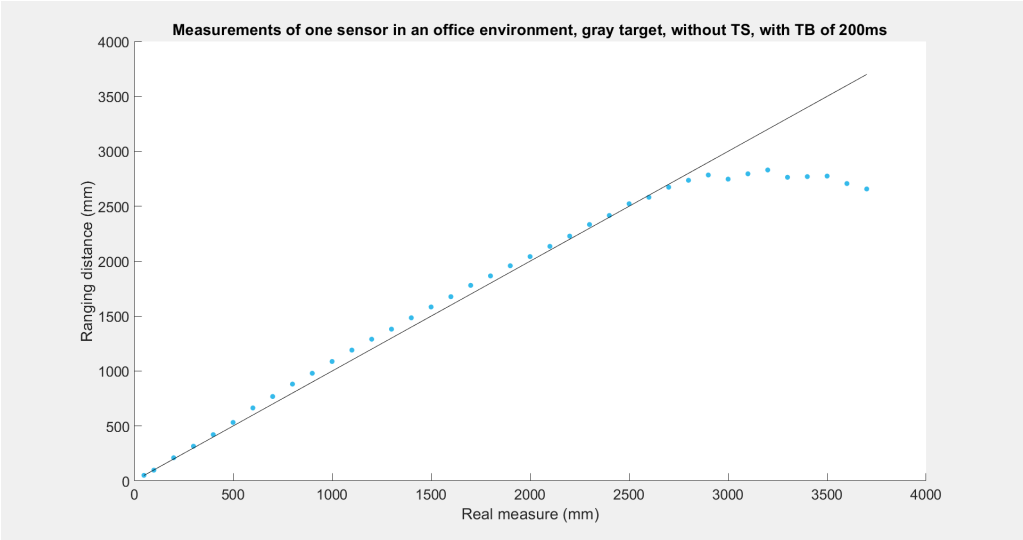

Figure 5.25: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target, without TS, with TB of 200ms

Black target without TS

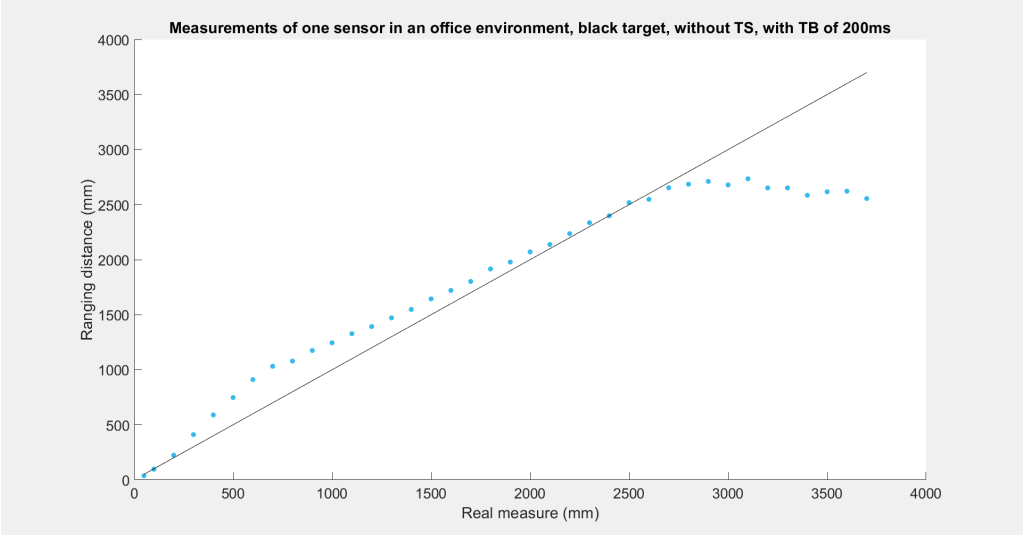

Figure 5.26: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, black target, without TS, with TB of 200ms

White target without TS

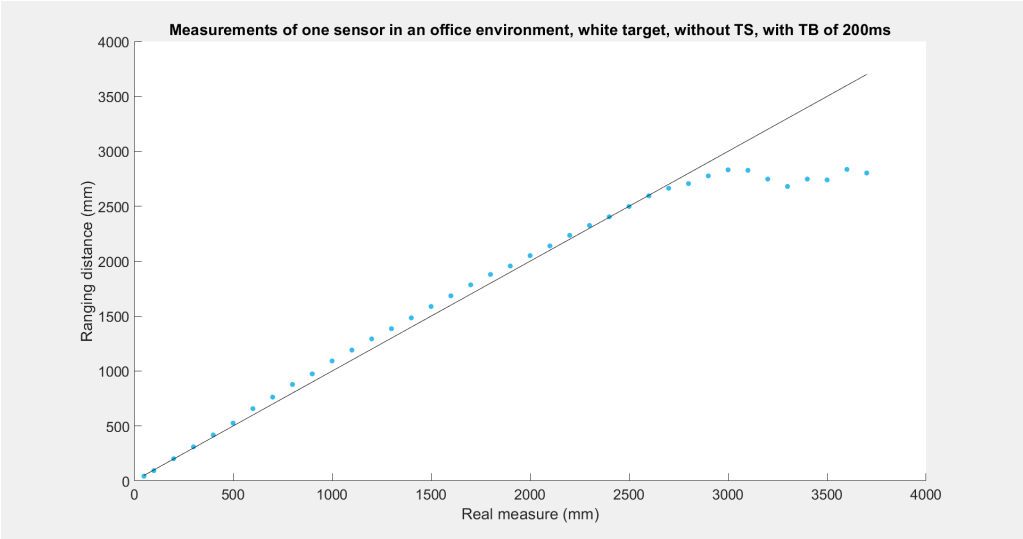

Figure 5.27: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, white target, without TS, with TB of 200ms

The results prove the theoretical knowledge that increasing the TB value increases the maximum ranging distance, but also shows that it improved the accuracy of the sensor, especially for a black target, approximating the graphic to the ideal line of measurements. It also showed that this TB value is not sufficient to allow the sensor to work with a TS.

Timing Budget 500ms

The 500ms value was then tested. The aim was to understand if this value was worth using, comparing the results with the measurements done with TB 200ms.

The sensor was calibrated in the same conditions again, except now with TB set to 500ms (Table 5.6).

Table 5.6: Calibration parameters values for one sensor with unsliced acrylics and glasses of 2mm and 3mm, with TB 500ms

Material	Thickness (mm)	Offset (mm)	Crosstalk (cps)
None	-	-17.639	-0.00007
Acrylic	**2**	76.68	138.3802
	3	73.54	12.19014
Glass	**2**	59.540	5.614572
	3	67.8	2.559076

Gray target with TS

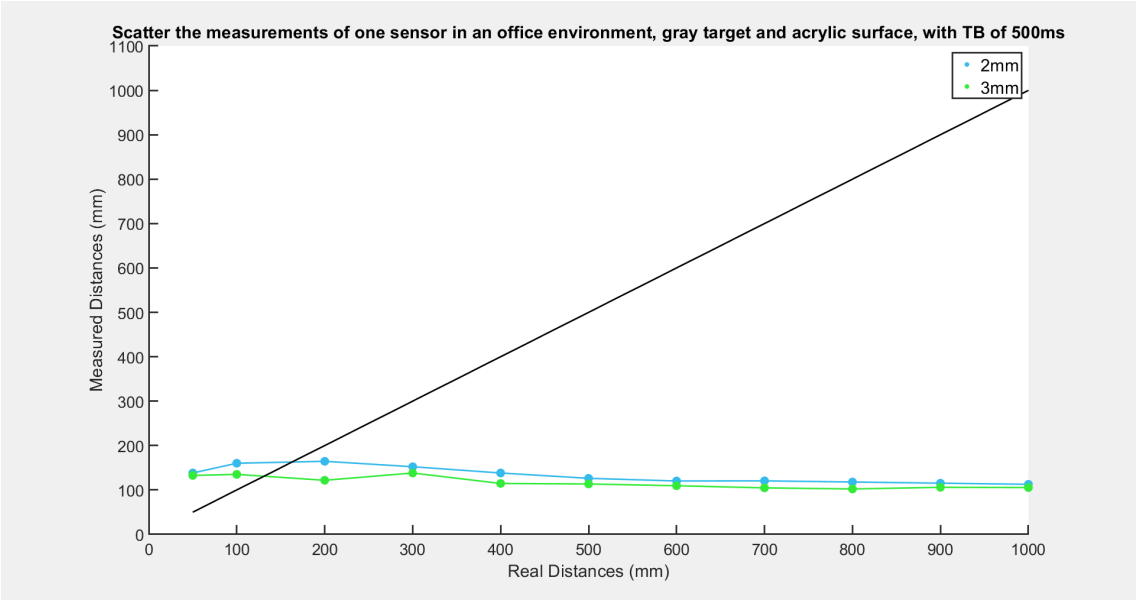

Figure 5.28: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target and unsliced acrylic surface, with TB of 500ms

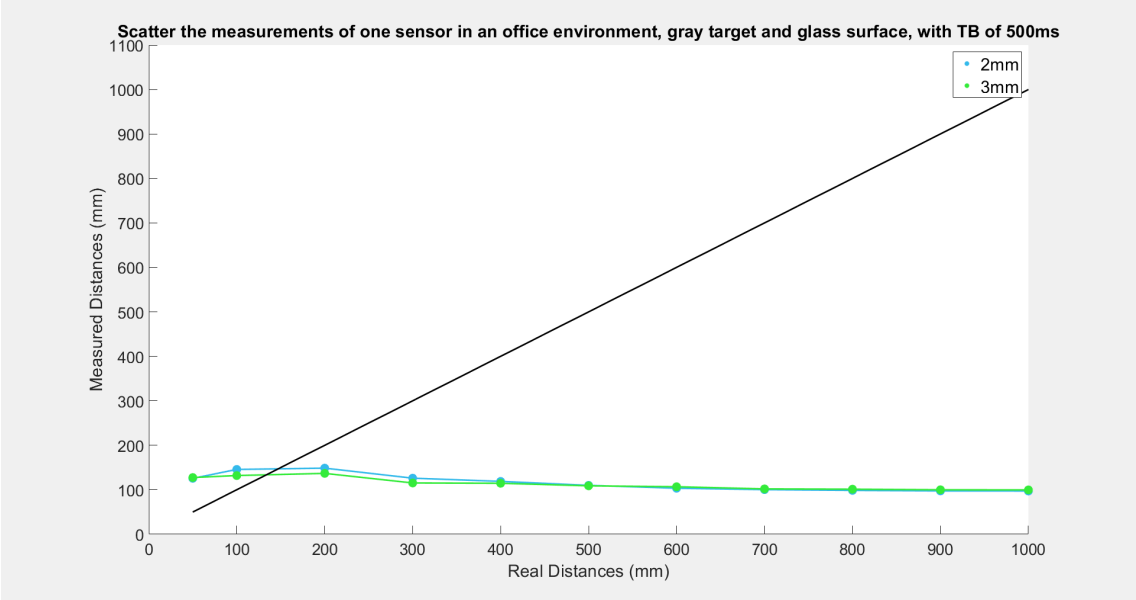

Figure 5.29: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target and unsliced glass surface with TB of 500ms

Gray target without TS

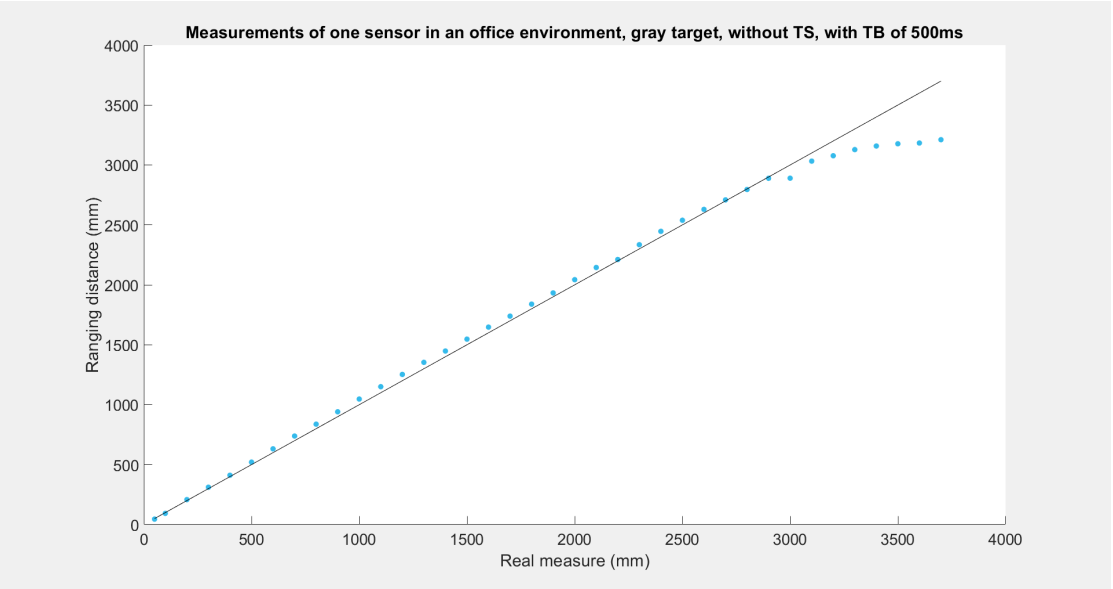

Figure 5.30: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target, without TS, with TB of 500ms

Unfortunately again, as shown in Figures 5.29 and 5.30, changing the TB to this value didn't improve the sensor's performance behind a TS. Thus, **no TS will be used in the tests for the other target colors**.

Black target without TS

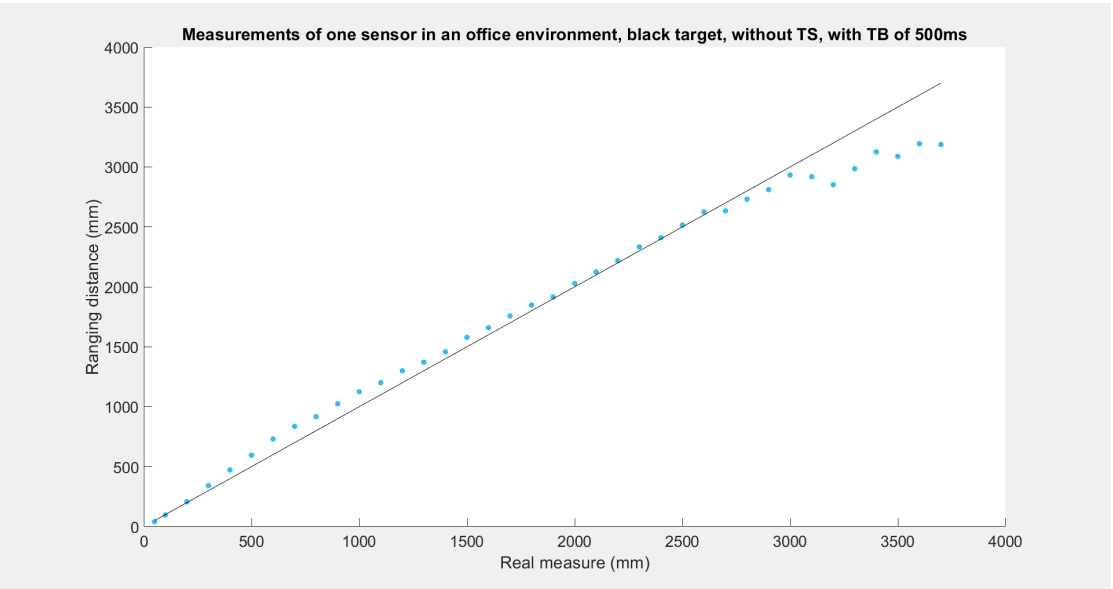

Figure 5.31: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, black target, without TS, with TB of 500ms

White target without TS

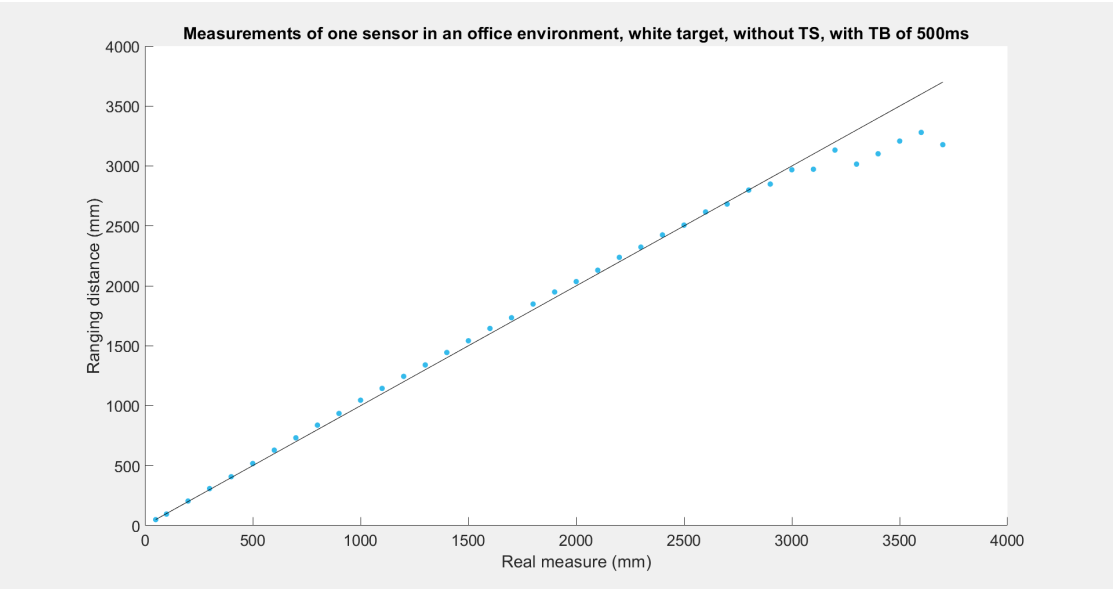

Figure 5.32: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, white target, without TS, with TB of 500ms

Using the TB value of 500ms further increases the sensor's accuracy, and the maximum ranging distance without TS can now reach above 3m. However, it still can't fix the 3Decide's problem of using the sensor behind a TS.

5.1.5 Changing Hardware Conditions

The previous tests have proven that the sensor can't operate behind a TS by manipulating its software only. The subsequent tests will change the physical conditions of the sensor (removing the cover glass) and the TS (using a sliced acrylic). For this group of tests, only a gray target was used because the influence of the target color was already analyzed in the previous tests, and the TB used was 200ms because it showed to improve the sensor's accuracy without increasing the sampling time and power consumption as much as 500ms did.

5.1.5.1 Remove the sensor's cover glass

The sensor includes the protective cover glass in front, which means that all the theoretical problems about TSs discussed in Chapter 3 apply to the cover glass, albeit on a smaller scale. Therefore, tests without the cover glass were performed to check if this could solve the problem and allow the sensor to calibrate the crosstalk related to the TS alone, measuring with a greater accuracy behind a TS. The sensor was again calibrated, especially because some differences are expected in the crosstalk compensation value. The calibration values are presented in Table 5.7.

Table 5.7: Calibration parameter values for one sensor with unsliced acrylics and glasses of 2mm and 3mm, without cover glass

Material	Thickness (mm)	Offset (mm)	Crosstalk (cps)
None	-	-4.060	0.00422
Acrylic	**2**	-11.40	0.38384
	3	21.840	3.73047
Glass	**2**	-17.560	0.57752
	3	21.14	2.23392

The crosstalk compensation values decreased notably compared to all its previous values. This may predict a good performance of the sensor with a TS, especially for the 2mm thickness in both materials because their values are close to zero, as the value with no TS. The same train of thought can be done to the offset compensation value: for 2mm, in both materials, the values are negative, like when there is no TS. This also proves that the measurements will be more accurate for this thickness.

With TS

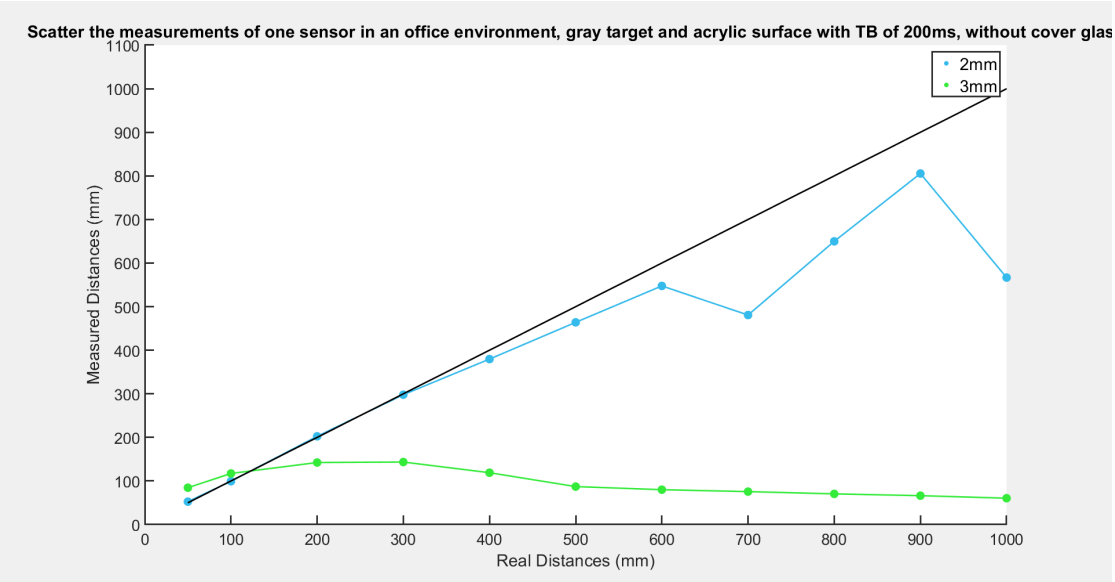

Figure 5.33: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target and unsliced acrylic surface, with TB of 200ms, without cover glass

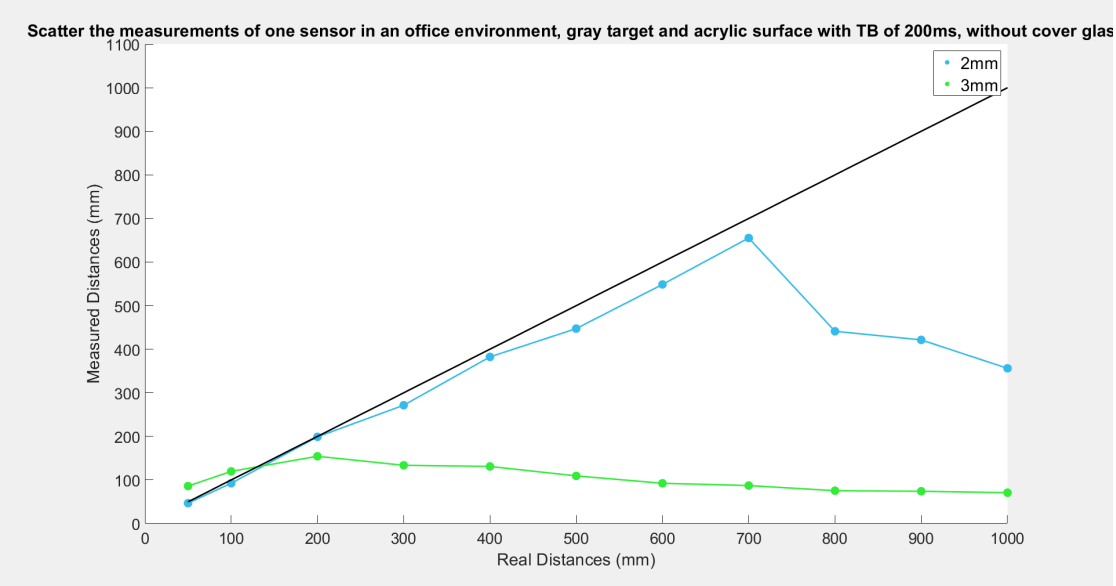

Figure 5.34: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target and unsliced glass surface with TB of 200ms, without cover glass

As shown in Figures 5.33 and 5.34, the measurement accuracy increased considerably. It is also visible that this solution is only viable for a 2mm thickness in both materials, as it was predicted through the values on Table 5.7. When acrylic is used, the measurements are almost ideal until 60cm and after this distance, they are not very viable. As for the glass, the measurements are ideal until 70cm. Because of this observation, the tests were only performed until 1m.

Without TS

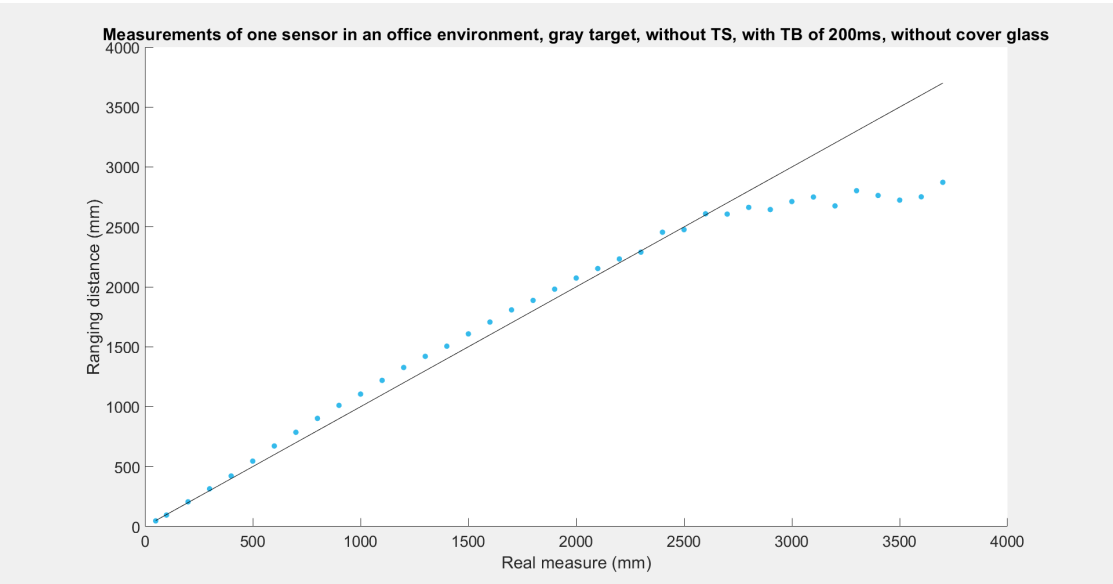

Figure 5.35: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target, without TS, with TB of 200ms, without cover glass

The results in Figure 5.35 are practically the same as the ones in Figure 5.25, which shows that removing the cover glass not only allows the sensor to work behind a 2mm TS, but also keeps the measurements without TS unshaken.

The only problem with removing the cover glass is that the sensor is more exposed and it can be easily damaged. Therefore, the cover glass should only be removed when a TS is placed in front of it, because it will cover the sensor, offering protection.

5.1.5.2 Using a sliced acrylic

Following the manufacturing way of reducing the crosstalk effect inside the cover glass (by dividing the VCSEL and the SPAD with a "wall", represented in Figure 5.38), a cut was made on a 3mm acrylic. Ideally, a TS of 2mm should be used, because it showed to have better accuracy than a 3mm TS, but it was the only piece available and provided by 3Decide. In the middle of this cut, a black paper strip was placed to make sure that the signal does not cross this division, as presented in Figures 5.36 and 5.37.

Figure 5.36: Sliced acrylic (side view)

Figure 5.37: Sliced acrylic (top view)

The cut in the acrylic needs to be placed in between the transmitter (VCSEL) and the receiver of the sensor (SPAD), as shown in Figure 5.39.

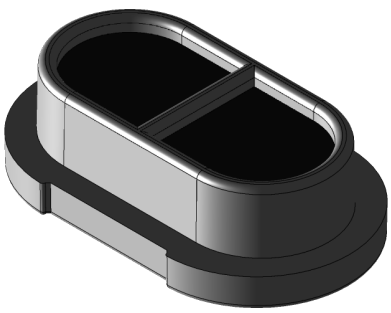

Figure 5.38: Cover glass with a wall in the middle

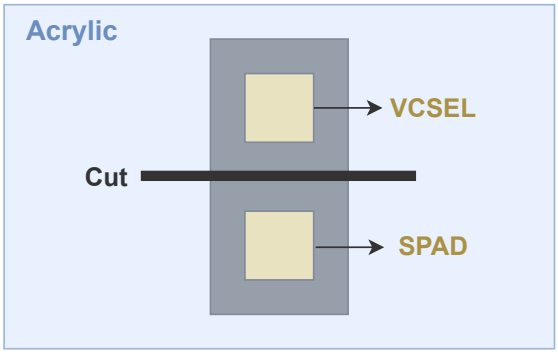

Figure 5.39: Scheme of the use of the sliced acrylic in front of the sensor

Because the acrylic is sliced, the test conditions are different and the sensor has to be calibrated. The calibration values presented in Table 5.8 show that the crosstalk is close to zero and that the offset is below zero. As seen in the previous tests without cover glass, these values mean that the measurements will be accurate with a TS.

Table 5.8: Calibration parameters values for one sensor with a 3mm sliced acrylic

Offset (mm)	**Crosstalk (cps)**
-26.1399	0.0100172

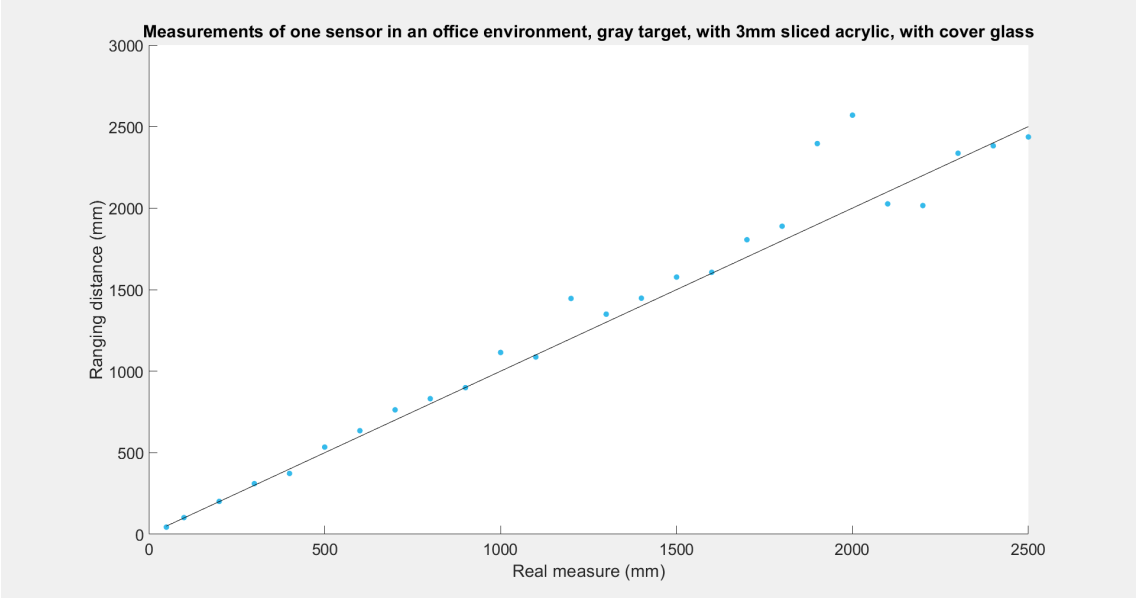

Figure 5.40: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target, with 3mm sliced acrylic, with cover glass

Figure 5.40 shows that the sensor has good accuracy, even though the measurements have some noise on higher distances. This may be explained by the fact that using this acrylic means placing the division precisely on top of the cover glass division, which can sometimes slightly move and cause these imperfections. Overall, these results show that this is a possible solution to the client's problem.

5.1.5.3 Using a sliced acrylic without cover glass

After these good results, both conditions were tested at the same time, in hope of finding the ideal conditions for the sensor to be used behind a TS. To improve the sensor's performance in this new condition, another calibration was performed (Table 5.9).

Table 5.9: Calibration parameters values for one sensor without cover glass and with a 3mm sliced acrylic

Offset (mm)	Crosstalk (cps)
12.280	0.004122

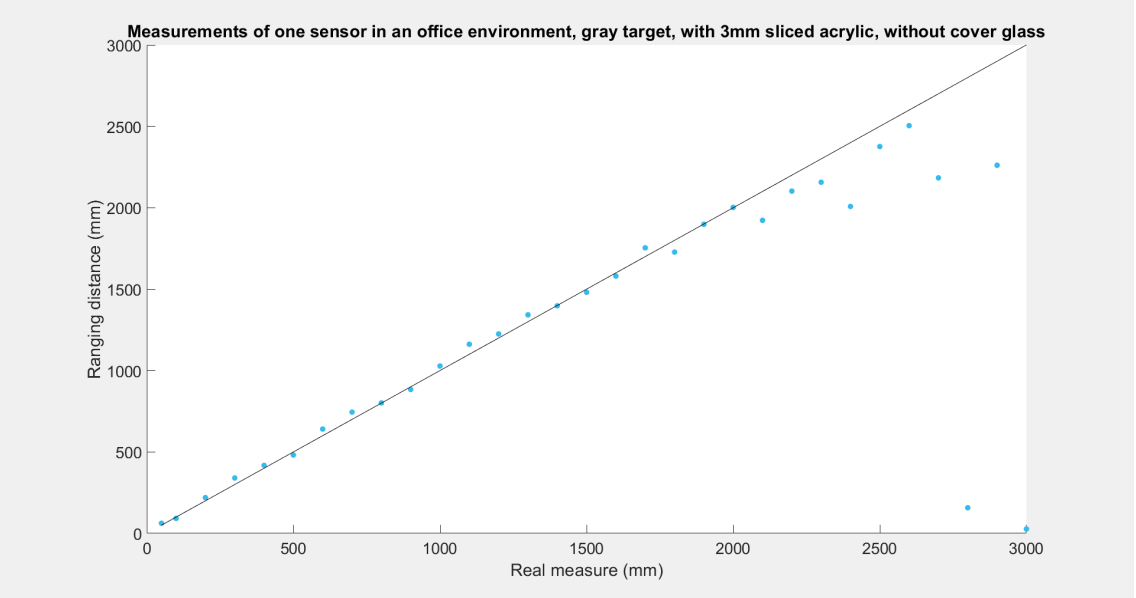

Figure 5.41: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, gray target, without cover glass with 3mm sliced acrylic

Comparing the values in Figures 5.41 and 5.40 with the ones in Figure 5.35, it can be concluded that when both conditions are simultaneous, the better accuracy of the tests removing the cover glass and the greater ranging distance of the tests with a sliced acrylic are combined. This combination has proven to be the best solution so far.

5.1.6 Detecting a hand

This was made as a bonus test to check how the sensor responds when the target is the user's hand, in this case without TS and with cover glass, i.e., in the sensor's "natural state". This is to predict the sensor's behavior in the 3Decide interactive kiosk, where the target is the user's hand. The calibration values are the same as in Table 5.3 because the test conditions are the same.

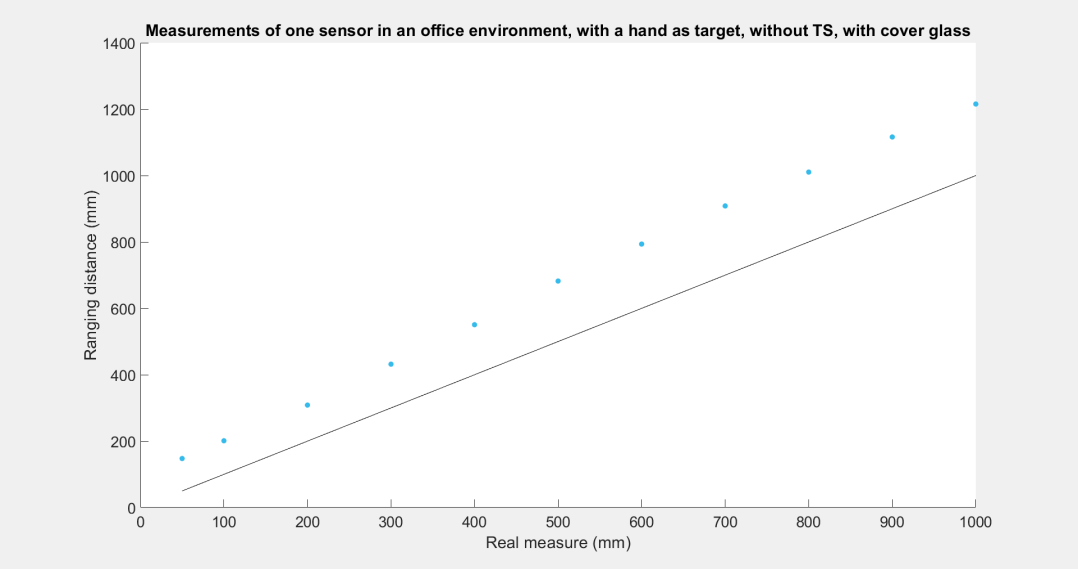

Figure 5.42: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, with user's hand, without TS

The results show an apparent offset, which can be easily calibrated, so the sensor should work with great accuracy after that. Therefore, 3Decide should calibrate the sensor with a hand instead of a gray target to reach higher accuracy in its measurements.

5.1.6.1 Using a Sliced Acrylic of 3mm without cover glass

After finding out that the best solution for VL53L1X to work behind a TS is to use a sliced TS and remove its cover glass, a final performance test was performed. The result is shown in Figure 5.43.

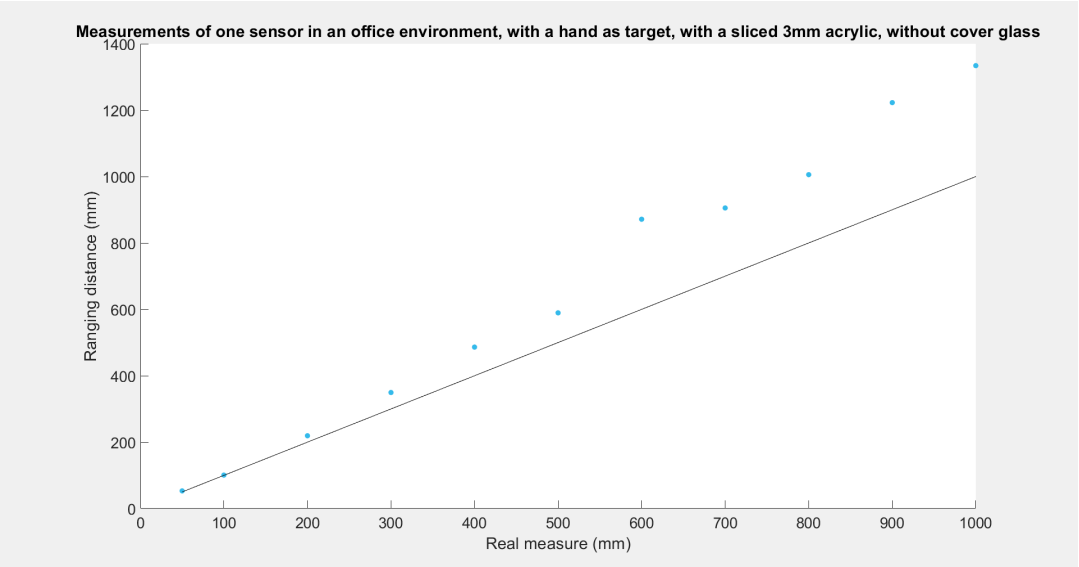

Figure 5.43: Scatter of the data (in mm) of one VL53L1X sensor in an office environment, with user's hand, with a sliced acrylic of 3mm, without cover glass

The results have shown that, even after a new calibration is done with a hand, the measurements have an "offset" from around 30cm compared to the ideal curve of measurements. The measurements still have a good accuracy until 50cm, therefore it is concluded that this sensor can be used with a TS in these conditions until this distance.

5.2 VL53L0X

The test conditions for VL53L0X were similar to the VL53L1X's, to allow their comparison. The major difference is that in these tests, only one sensor was used. With this said, the test conditions are the following:

- Using default conditions
  - With one sensor in an "office" environment with three target colors (white, gray and black), to understand the influence of the target color and
- Changing the hardware conditions using a gray target
  - Removing the cover glass only
  - Using a sliced acrylic only
- Using a hand as a target
  - Without TS
  - Using the best solution found to add a TS

5.2.1 Single Sensor in an office environment

The tests were done until 2m because that is this sensor's range limit. Following the discoveries from the VL53L1X test results, for VL53L0X only the 2mm and 3mm thicknesses will be used for acrylic and glass surfaces.

5.2.1.1 Gray target

With TS

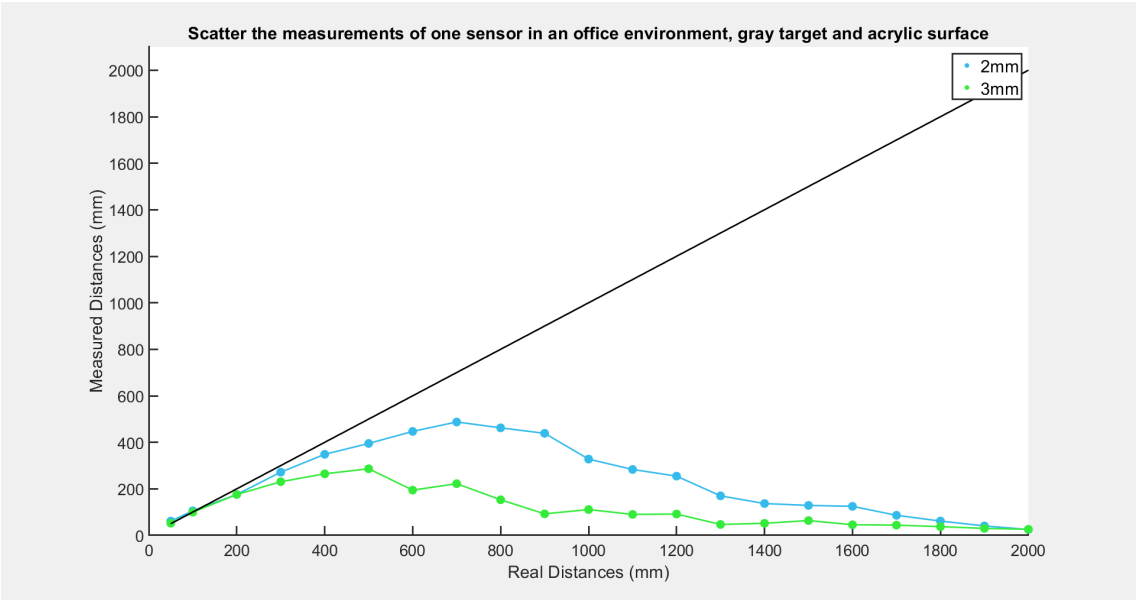

Figure 5.44: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, gray target and unsliced acrylic surface

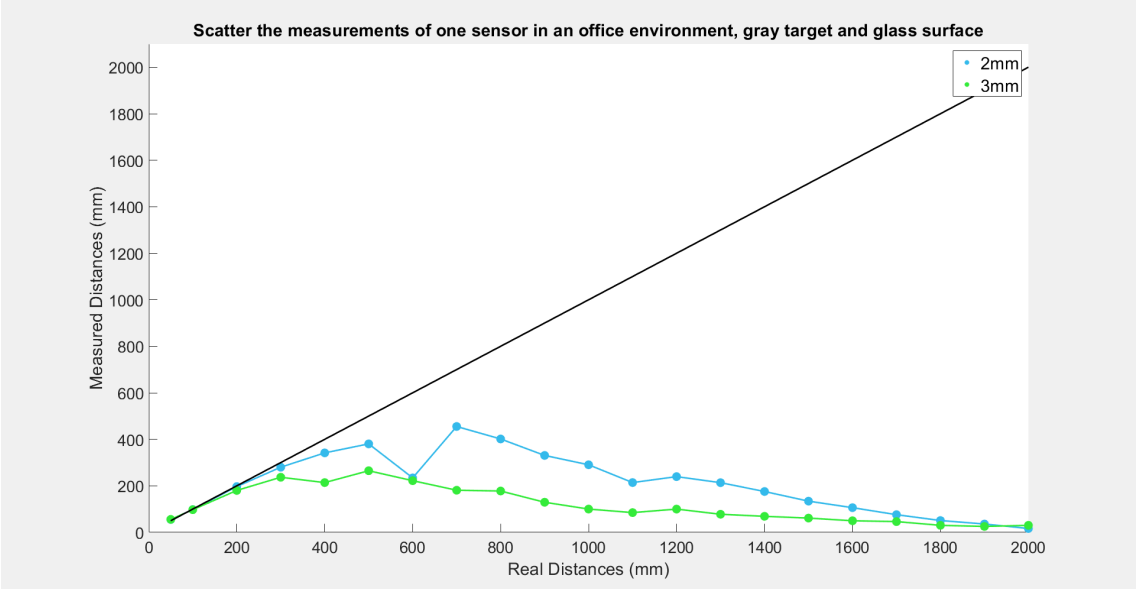

Figure 5.45: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, gray target and unsliced glass surface

These results show that VL53L0X is less sensitive to crosstalk effects from TSs than VL53L1X. With an acrylic surface, the sensor can measure reasonably up to around 70cm and 50cm with thicknesses of 2mm and 3mm, respectively. As for the glass surface, the sensor can measure reasonably up to around 50cm and 30cm with thicknesses of 2mm and 3mm, respectively.

Figures 5.44 and 5.45 also show that the curve of the measurements is different than the ones with TS with VL53L1X: in VL53L1X the values have a peak, and then the measurement value turns constant; with VL53L0X, the curve goes down, tending to zero, after the peak.

Without TS

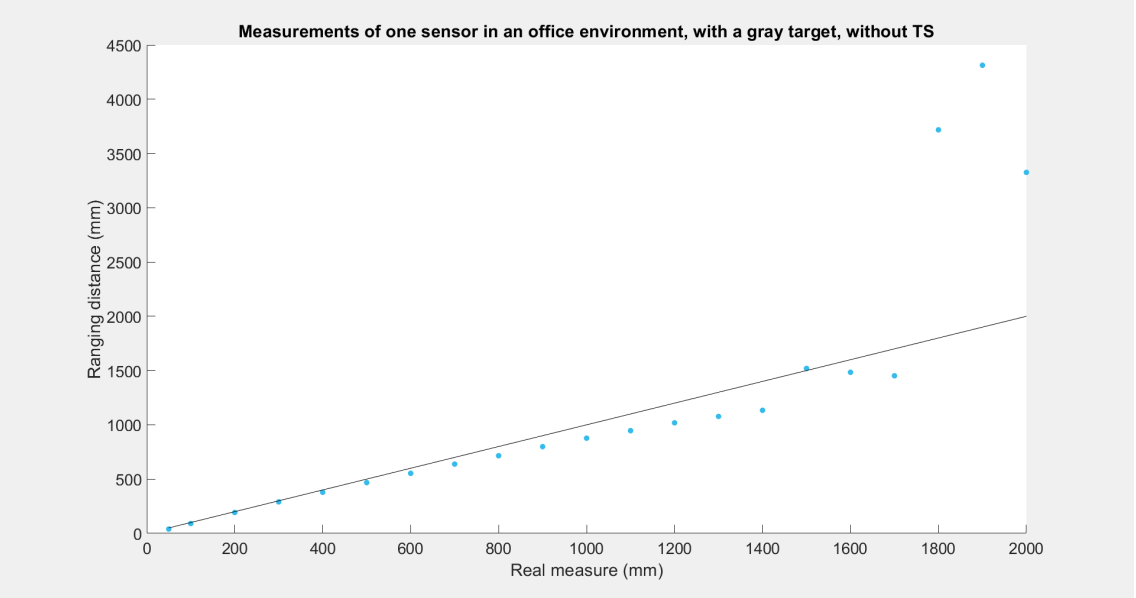

Figure 5.46: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, gray target, without TS

In this case, we can see that version 1 of the sensor isn't as accurate as version 2 as it even creates outliers when measuring closer to 2m.

5.2.1.2 Black target

With TS

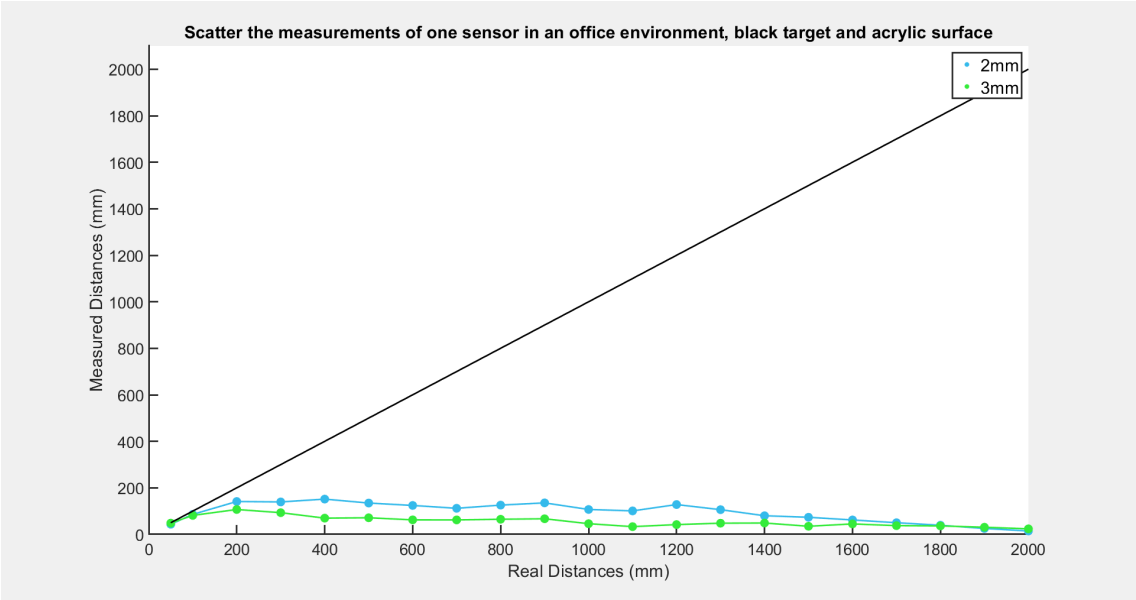

Figure 5.47: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, black target and unsliced acrylic surface

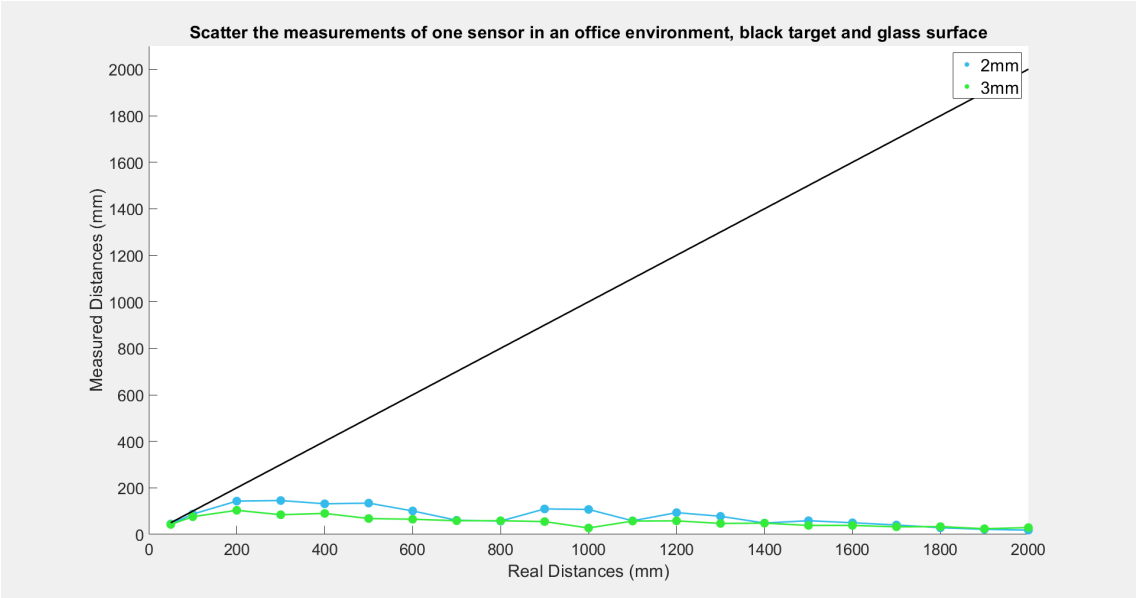

Figure 5.48: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, black target and unsliced glass surface

Without TS

In order to compare the sensors' performance without the TS, these tests were also performed. In Figure 5.49, it is possible to see that the sensor's performance is much worse with black targets than it was both in this version's test with a gray target and in version 2's test with a black target.

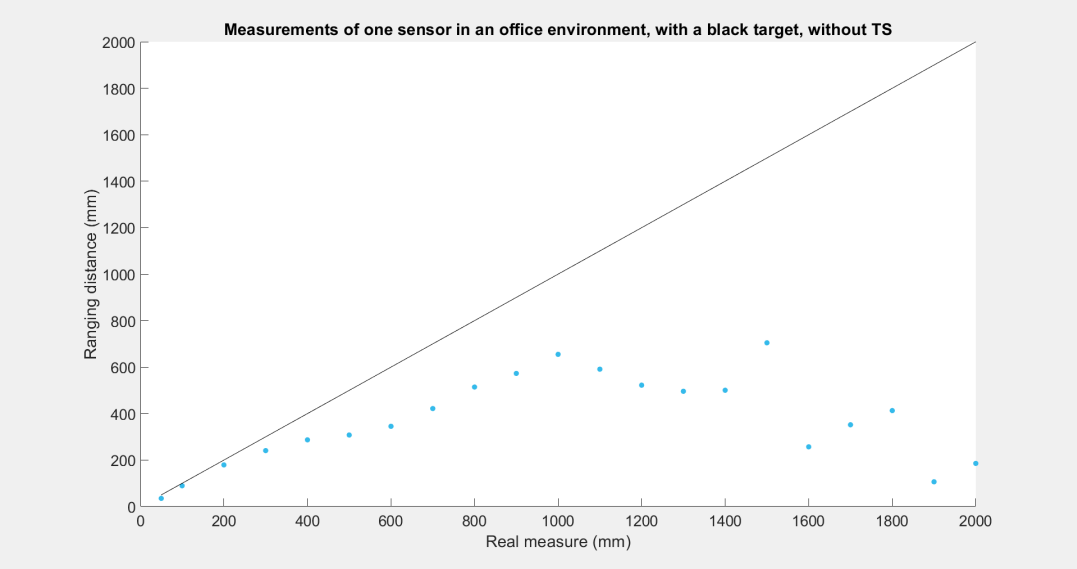

Figure 5.49: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, black target, without TS

5.2.1.3 White target

With TS

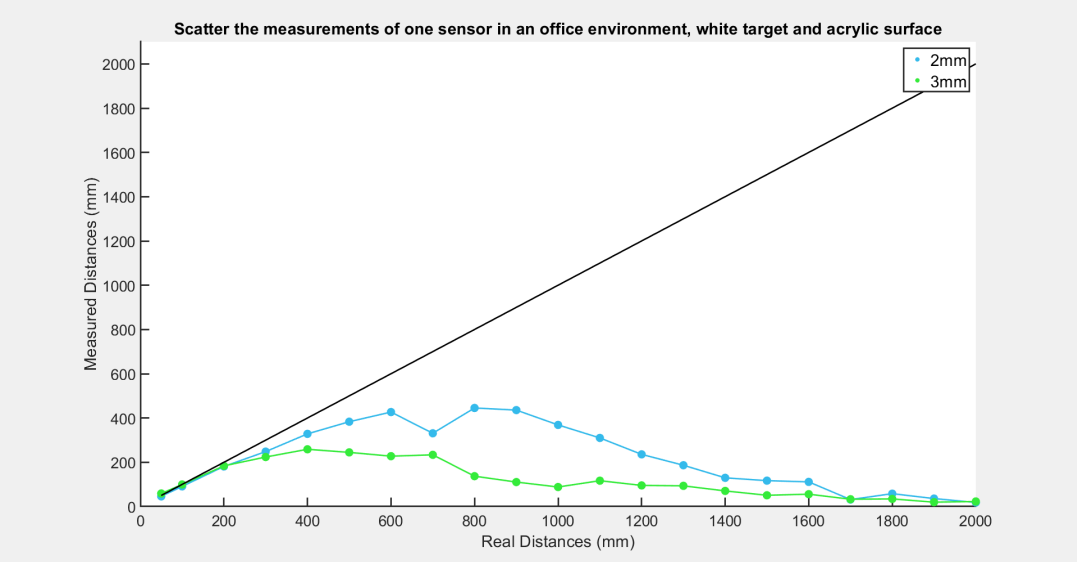

Figure 5.50: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, white target and unsliced acrylic surface

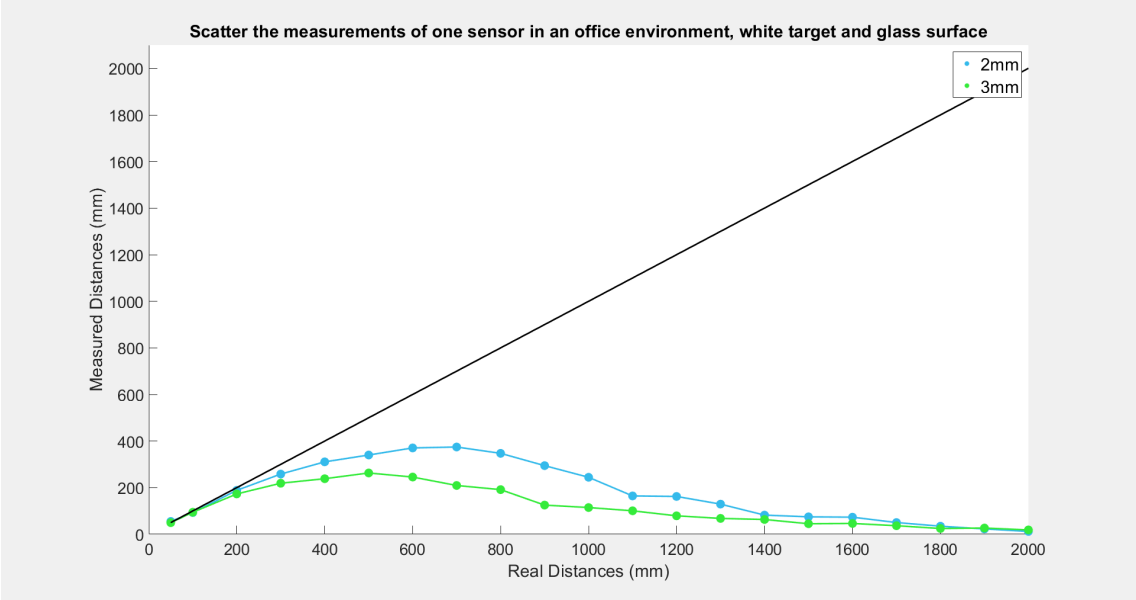

Figure 5.51: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, white target and unsliced glass surface

Without TS

These tests were performed to compare the sensors' performance without the TS.

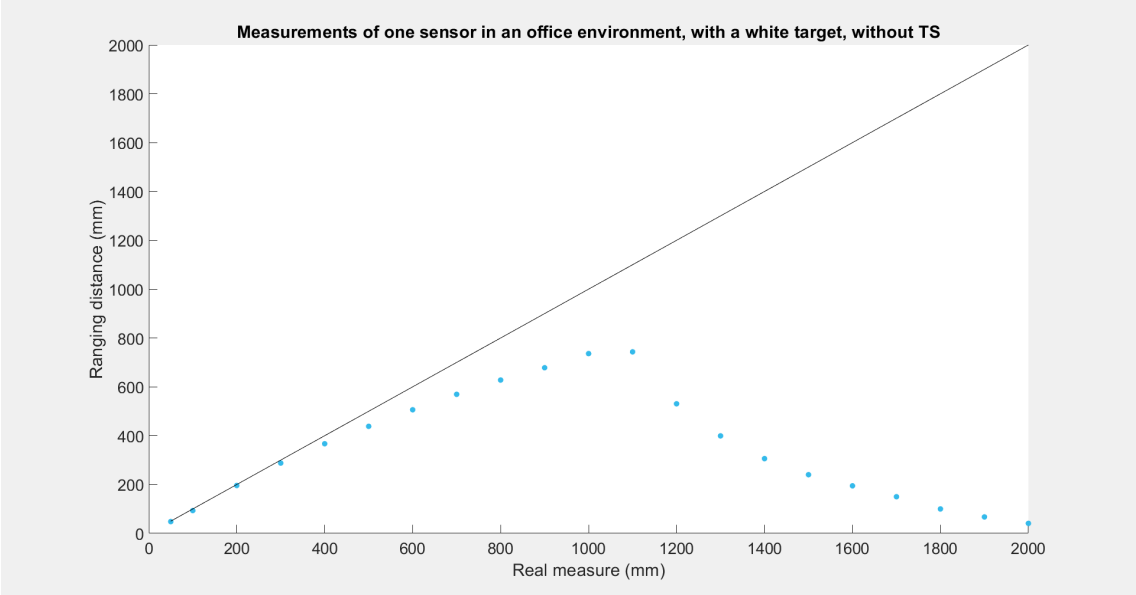

Figure 5.52: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, white target, without TS

The results with TS (Figures 5.44, 5.45, 5.47, 5.48, 5.50 and 5.51) show that for gray and white targets, the sensor can measure until around 40cm and 50cm (of measured distance) with

the 2mm thickness, and until 20cm to 30cm with the 3mm thickness. As always, the results with a black target are the worst among the colors of the targets and can only measure until 15cm.

5.2.2 Changing Hardware Conditions

For the VL53L1X sensor, changing the sensor's parameters didn't solve the problem of the client, but removing the cover glass, using a sliced acrylic, or both simultaneously improved its performance in the presence of a TS. Hence, this test won't change any parameters, only the hardware conditions. The solutions found for version 2 (removing the cover glass and using a 3mm sliced acrylic) will be tested for VL53L0X to check if it works on it too.

5.2.2.1 Remove the sensor's cover glass

With TS

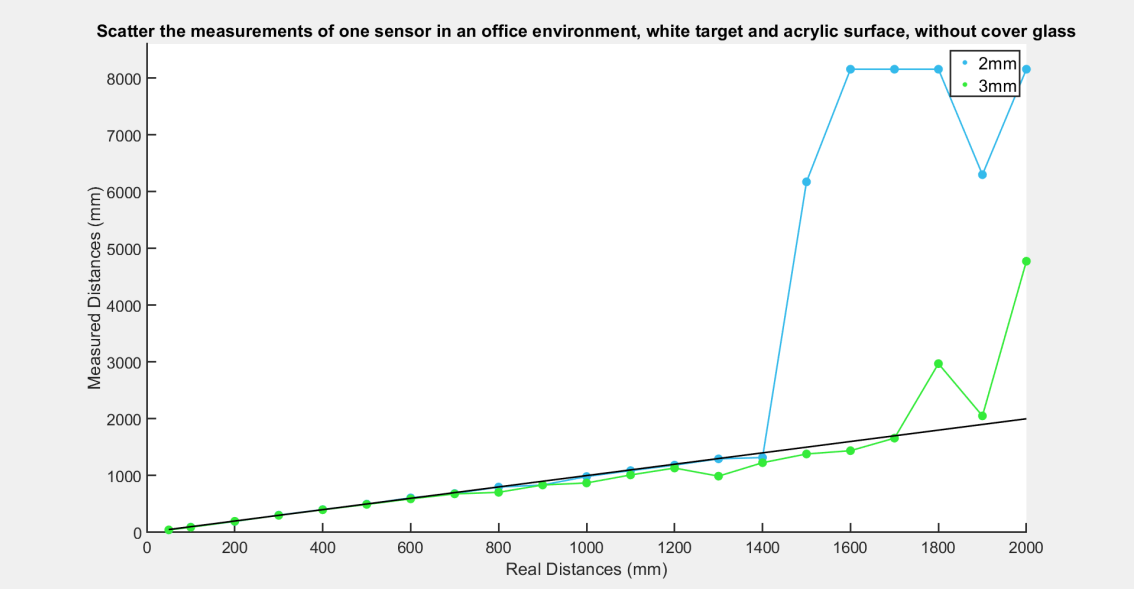

Figure 5.53: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, white target and unsliced acrylic surface, without cover glass

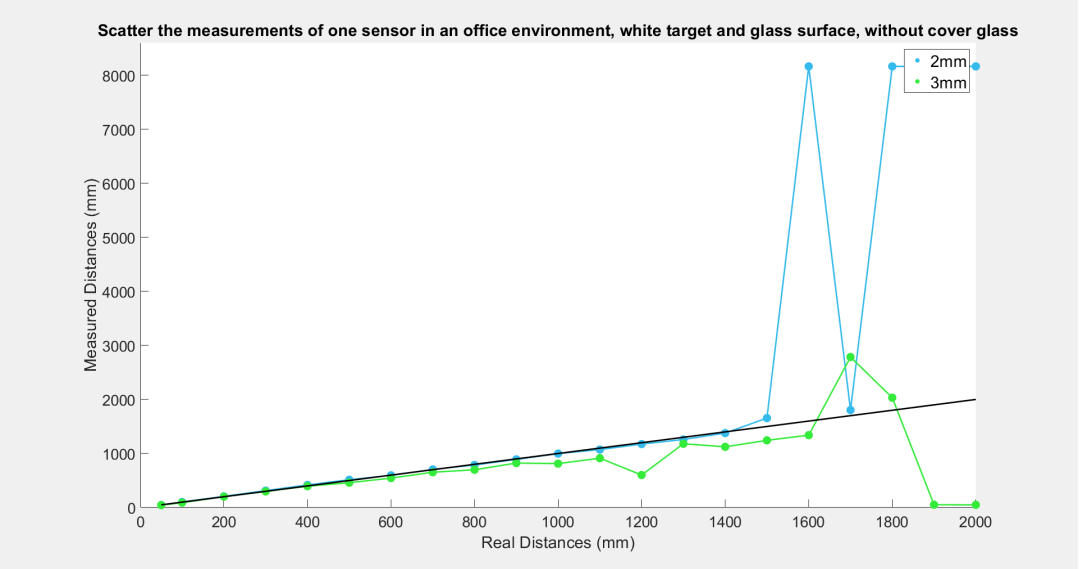

Figure 5.54: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, white target and unsliced glass surface, without cover glass

The results in Figure 5.53 and Figure 5.54 show that VL53L0X can measure with great accuracy until 1.4m and 1.5m, with 2mm acrylic and glass surfaces, respectively. Surprisingly, the measurements with a TS of 3mm have a higher range, going up to 1.7m. The outliers greater than 8000mm correspond to this sensor's "error value" - 8191mm. This means that the sensor cannot find any target at those real distances.

Without TS

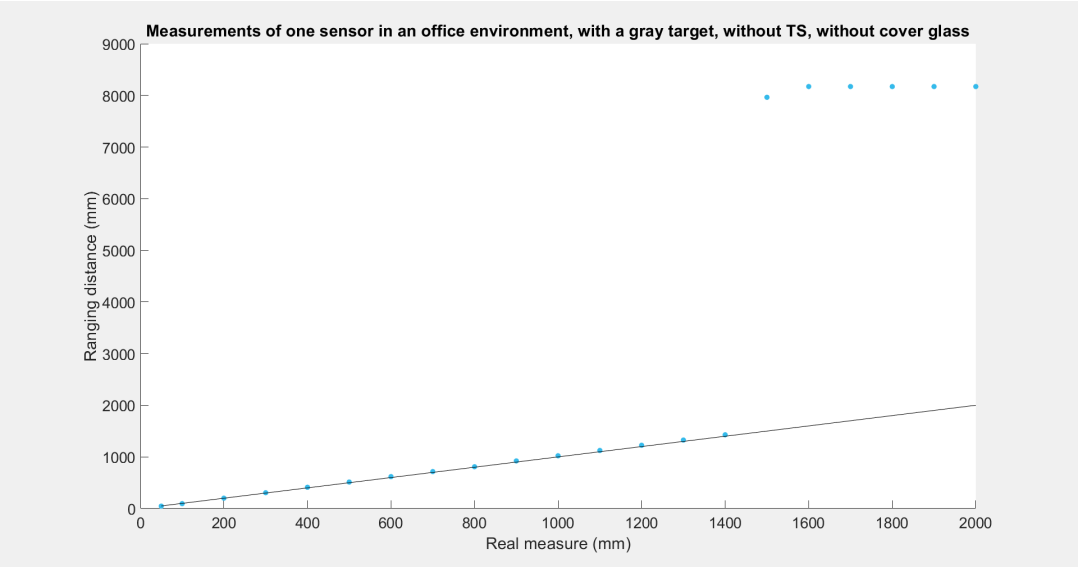

Figure 5.55: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, gray target, without TS, without cover glass

Figure 5.55 show those same outliers after 1.4m, proving that the sensor can only function up to that distance.

5.2.2.2 Using a sliced acrylic

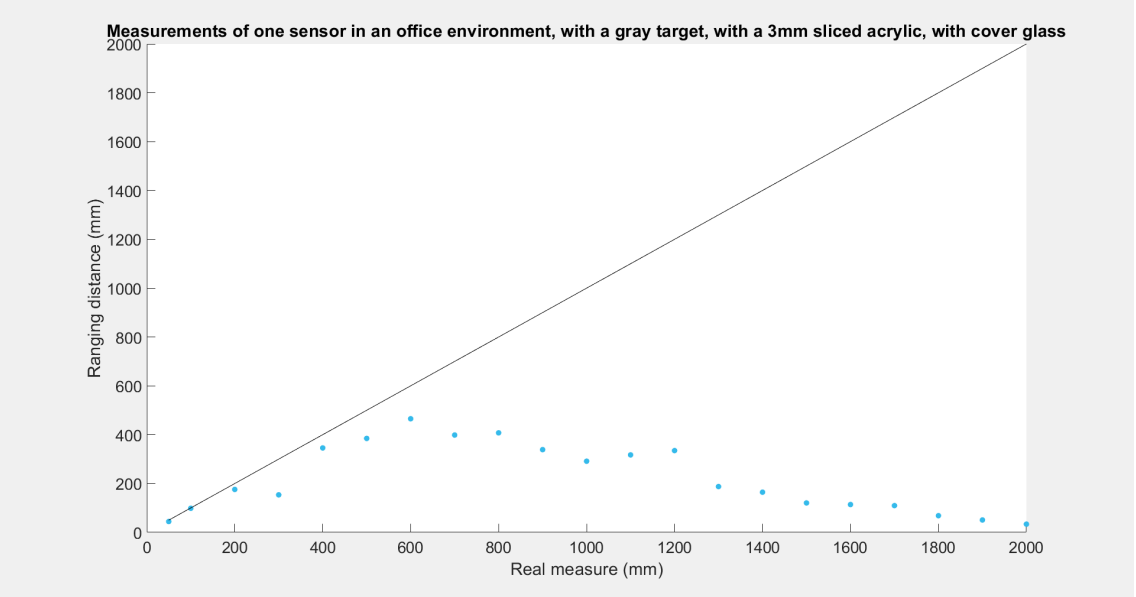

Figure 5.56: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, gray target, with 3mm sliced acrylic, with cover glass

As shown in Figure 5.56, this solution is not accurate at all for VL53L0X. This may be explained by the fact that in this version, the cover glass does not have the wall division between the VCSEL and the SPAD, so the sensor must be prepared for TSs on top by design but has worse performance than version 2 on normal scenarios.

5.2.3 Detecting a hand

The measurements using the natural state of this sensor with a hand as a target (Figure 5.57) have similar accuracy to the measurements with VL53L1X.

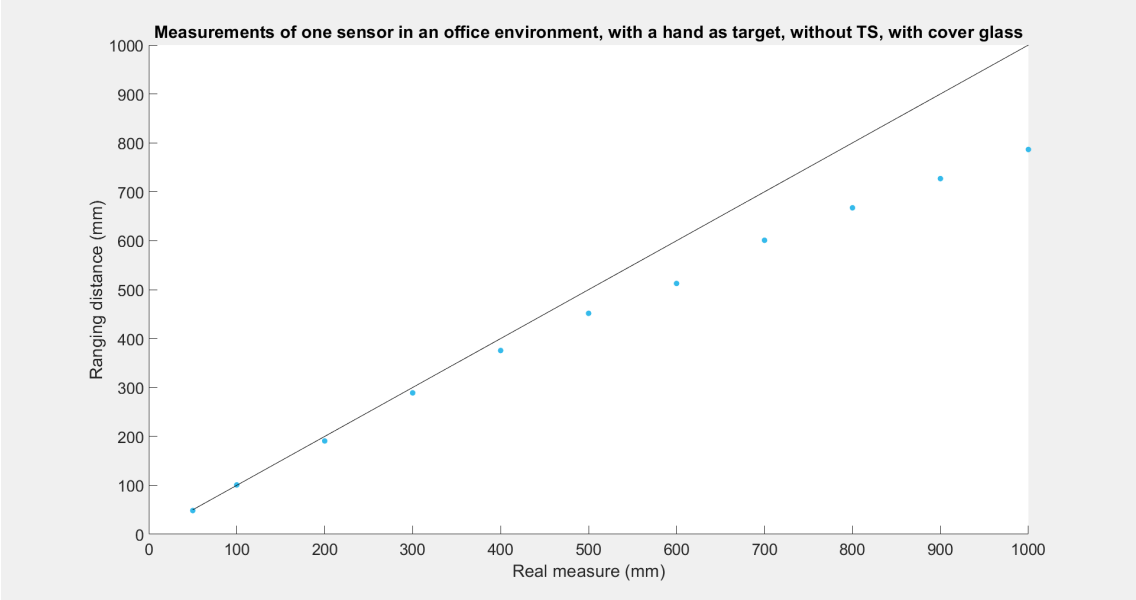

Figure 5.57: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, with user's hand, without TS, with cover glass

5.2.3.1 Removing the cover glass

The ideal solution for this sensor to work behind a TS is to remove its cover glass, as proven previously. Therefore, the following tests were performed to predict the sensor's behavior under the conditions used in the 3Decide projects. Figure 5.58 shows the results when using 2 and 3mm acrylic surfaces and Figure 5.59 with glass surfaces.

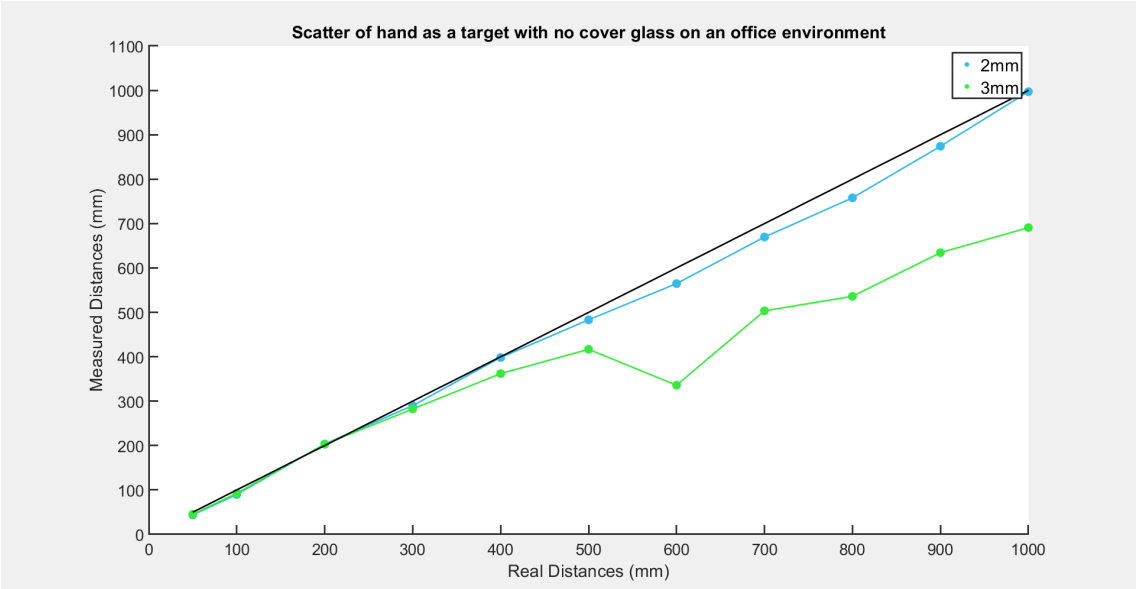

Figure 5.58: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, with user's hand, without cover glass, with unsliced acrylic surfaces

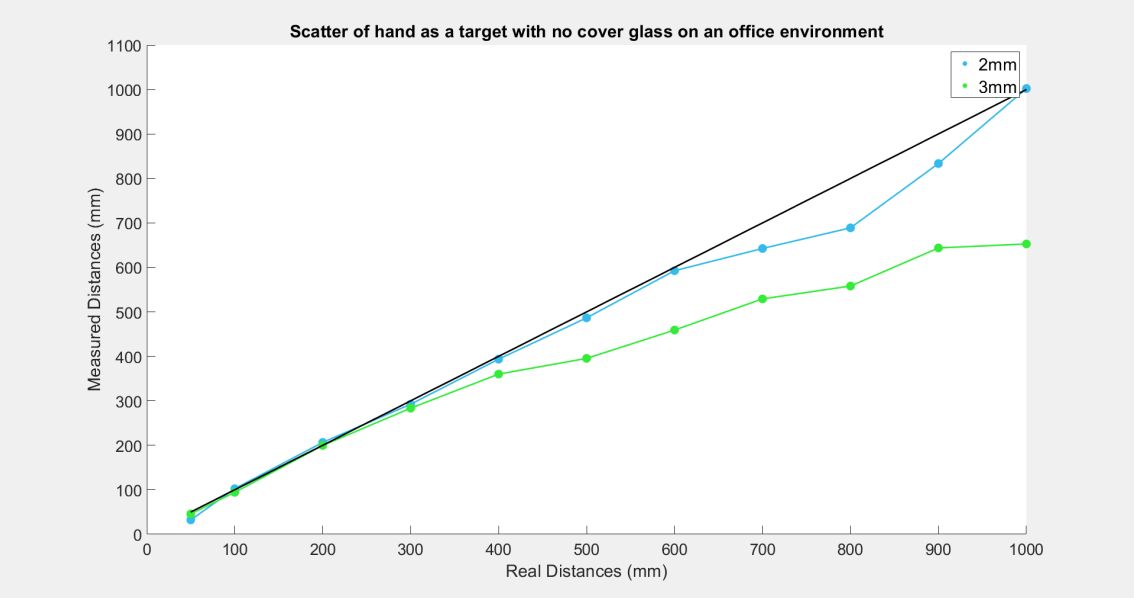

Figure 5.59: Scatter of the data (in mm) of one VL53L0X sensor in an office environment, with user's hand, without cover glass, with unsliced glass surfaces

The results above show that using a 2mm TS is the best solution for both materials. The recommended material to use is acrylic, as it has the best measurement accuracy until the full test distance range. Comparing these results with the ones found for version 2 it is clear that VL53L0X offers more accuracy in the presence of a TS.

Chapter 6

Conclusion and Future work

6.1 Conclusion

This dissertation's main goal was to study the new version of the sensor and add a new functionality, that is, to have measurements with accuracy in the presence of a TS so that 3Decide can improve its existing products and apply this knowledge to future ones.

The client's major problem was that the VL53L0X sensor, used in their previous project, could not work accurately behind a TS for distances greater than a few centimeters. This created a necessity to look for new solutions, such as exploring the next version (version 2) of the same sensor - VL53L1X - and checking if it was more accurate and had a better range of valid measurements than the first version with a TS in front of it.

With this in mind, both sensors were tested, and the results showed that version 1 - VL53L0X - is less sensitive to the presence of a TS and offers better accuracy. Still, its ranging distance with accurate measurements is minimal. Version 2 shows that it is more sensitive in the presence of a TS in front of it, being unable to measure. However, this limitation can be overcome by creating an incision over the boundary between the sensor's emitters and receptors and inserting an opaque material in it to block the light and prevent crosstalk, which is what the protective cover glass of this version already has.

When the cover glass is removed, version 1 can measure up to around 1.5m with great accuracy. In contrast, version 2 can only measure up to approximately 70cm, which is a significant improvement when compared to the results with a TS and the default parameters and hardware settings, which could only reach a few centimeters in both versions. When a sliced TS is used, the measurements with version 2 go up to 2m of actual distance. Using a sliced acrylic with version 1 proved disadvantageous, worsening the results' accuracy. When the cover glass is removed and a sliced TS is used, the measurements with version 2 have a higher maximum distance (above 2m) and better accuracy. This is the best solution for version 2, but for version 1, the only scenario that improves the measurement accuracy in the presence of a TS is when its cover glass is removed and said TS is not sliced.

Based on this information, it is concluded that the VL53L0X sensor should be used for distances up to 1,4m because it has higher measurement accuracy in the presence of a TS. However, if the client's application requires measurements greater than 1,4m, the VL53L1X should be used instead, as it can measure distances up to 2m in its ideal conditions.

Overall, all the specifications of 3Decide were fulfilled. With the information gathered during this dissertation, some important contributions were done, such as the theoretical study of the influence of the presence of TSs in front of ToF sensors, the development of JavaScript software to manipulate both versions of the sensor, and determining the ideal conditions for each sensor when placed behind a TS.

6.2 Future work

Now that the behavior of VL53L1X behind a TS is known, one interesting next step would be to add gesture recognition. This can be reached in this version because there can be various ROIs with different positions and sizes. By manipulating these parameters, the regions inside the FoV can be isolated, i.e., in the north, east, south, and west region. Whenever the target changes the region, a gesture can be detected. For instance, if the hand traveled from the north to the south region, the motion was an up-to-down gesture.

Appendix A

Usage Protocol

This usage protocol aims to explain how the VL53L1X tests were performed and how to use the sensors for measurements in general.

A.1 Setup the hardware

A.1.1 Material Needed

Three sensors were used for the experimental tests, three cables to connect them in series through the same I2C pin of the Raspberry Pi and three supports where the sensors are connected. Four 30x10cm glass and acrylic surfaces of 2mm, 3mm, 4mm, and 5mm were also used.

A.2 Setup the Software

To make it easier for the user to manage the CLI to handle the sensor without having to change the source code, it interprets command flags that configure how the program runs, as shown in Figure A.1.

```
// -n
const nSensors = (() => {···
})();
// -i, --init
const shouldInitAddresses = argv.init ?? argv.i ?? false;
// --co, --calibrate-offsets
const shouldCalibrateOffsets = argv["calibrate-offsets"] ?? argv.co ?? false;
// --of, --offsets-file
const offsetsFile = argv["offsets-file"] ?? argv.of ?? DEFAULT_OFFSETS_FILE;
// --cx, --calibrate-xtalks
const shouldCalibrateXTalks = argv["calibrate-xtalks"] ?? argv.cx ?? false;
// --xf, --xtalks-file
const xTalksFile = argv["xtalks-file"] ?? argv.xf ?? DEFAULT_XTALKS_FILE;
// -v
const verbose = argv.v ?? false;
// -x
const addresses = (() => {···
})();
```

Figure A.1: Shortcuts to handle the sensor

A.2.1 Initialization

The first step is to **set the number of sensors** to be used. It is done using ***-n*** followed by the number of sensors.

After setting the number of sensors, it is time to **set their addresses using the *-x* flag** followed by the address and repeat for the number of sensors. If the number of addresses inserted does not correspond to the number of sensors, an error message will be displayed in the terminal. If no *-x* flag is passed, the default values are 0x2A, 0x2B, 0x2C, and so on when the number of sensors is greater than 1 and is 0x29 when there is only one sensor.

If the sensors are not initialized yet, i.e., if it is a cold start, the initialization process needs to be performed in order to set the I^2C addresses for each sensor. This step only needs to be done once after the sensors have lost power because that's when they return to the 0x29 address. **To initialize, use the shortcut *-i***. In the begging, no sensors should be connected, and then, according to the instructions on the terminal (Figure A.2), they should be inserted one by one. Repeat the process for the rest of the sensors.

```
Setting address to 0x2A
Press enter to set address of sensor 0 to 0x2A...
```

Figure A.2: Initialization instructions

A.2.2 Calibration

As it was described in point 2.3.2.2, the calibration of both parameters is done under the same conditions: a dark environment and a 17% gray target. For the offset calibration, it is only neces-

sary to leave the sensors at 140mm to the target, as for the crosstalk calibration, first it is necessary to find the xcd, place the target at that distance and then wait for the compensation value calculation. These values need to be stored to be used in the future when the sensor is used under other environmental conditions.

The tests include the use of glass and acrylic surfaces with different thicknesses in front of the sensor, to understand the sensor's performance in these circumstances, and also to control tests without any TS. The calibration process was done in the indicated conditions for each material and its respective thickness. All these values were stored in JSON files, two for each situation: one file for the offset and another one for the crosstalk compensation values.

Only after this process, the tests can be performed.

The process of both calibration parameters is similar. First, it is indicated if the calibrations are to be performed at the beginning of the ranging or not, using the shortcuts ***–co true*** and ***–cx true***, to calibrate the offset and crosstalk, respectively. If this step is not included, the calibrations won't be performed, as the default value is **false**, except if no calibration values files were passed (as explained in the next paragraph), in which case the calibration is forced.

After that you can indicate the JSON file names where the calibration values will be stored in or loaded from, using the ***–of [JSON file path]*** and ***–xf [JSON file path]***, for "offset file" and "xtalk file", respectively. If these files are not specified, default file names will be used in the current directory. If the *–co* or *–cx* is *false*, these files are used to indicate where the calibration values should be loaded from.

A.2.3 Ranging

After initializing and calibrating the sensor, it is time to choose the mode of operation. There are two modes: the testing mode (1) and the measurements-only mode (2). **To set the mode of operation use the shortcut *-m*** followed by the mode number. The default mode is the test mode.

The testing mode presents instructions in the terminal to place the sensor at the correct distance to collect data, as shown in Figure A.3.

```
Set the distance from sensors to 5cm. Press ENTER to continue...
```

Figure A.3: Ranging in test mode instructions

The measurements-only mode only captures the data read by the sensor without any other instruction.

An example of the necessary terminal inputs to start one sensor with calibration of offset and crosstalk in a cold start is shown in Figure A.4.

```
node test_n.js -n 1 -m 1 --of ./offset_cal_file.json --xf ./xtalk_cal_file.json --co true --cx true -i
```

Figure A.4: Example of complete command to start the sensor

Appendix B

Helper functions

```
function waitForEnter(msg) {
    if (msg && typeof msg === 'string') {
       process.stdout.write(msg);
    }
    return new Promise(resolve => {
        process.stdin.once('data', () => resolve());
    });
}
}
```

Listing B.1: Wait for Enter

```
const newPoller = (vl53l0x, sensorId = '99', samplingPeriodMs = 20, offsetCal = 0)
    => {
  let shouldBeRunning = false;
  const oneShot = async () => (await vl53l0x.api.measure(sensorId))?.[sensorId];
  let latestValue = null;
  let timeout = null;
  const schedule = () => {
    if (!timeout) {
      timeout = setTimeout(tick, samplingPeriodMs);
    }
  }
  const unSchedule = () => {
    if (timeout) {
      clearTimeout(timeout);
      timeout = null;
    };
  }
  const tick = async () => {
    timeout = null;
    if (!shouldBeRunning) {
      latestValue = null;
    } else {
      latestValue = await oneShot();
```

```
23        schedule();
24      }
25    }
26    const getLatestValue = async () => {
27      let lv = latestValue;
28      latestValue = null;
29
30      if (!latestValue) {
31        unSchedule();
32        lv = await oneShot();
33      }
34
35      if (shouldBeRunning) {
36        schedule();
37      }
38
39      return lv + offsetCal;
40    }
41
42    const startPolling = () => {
43      shouldBeRunning = true;
44      schedule();
45    }
46
47    const stopPolling = () => {
48      shouldBeRunning = false;
49      unSchedule();
50      latestValue = null;
51    }
52
53    const ret = {
54      startPolling,
55      stopPolling,
56      getLatestValue,
57      offsetCal,
58      sensor: vl53l0x
59    }
60
61    delete ret.sensor;
62    Object.defineProperty(ret, 'sensor', {
63      get: () => vl53l0x,
64      enumerable: true
65    });
66    Object.freeze(ret);
67
68    return ret;
69  };
```

Listing B.2: New Poller method (filter initial values) method

References

[1] TeraRanger Evo 3m - Close-range ToF distance sensor, 3m, FIXED 100Hz, 12 grams. https://www.terabee.com/shop/lidar-tof-range-finders/teraranger-evo-3m/. Accessed: 2022-04-20.

[2] Helios Time of Flight (ToF) 3D Camera. https://thinklucid.com/product/helios-time-of-flight-imx556/. Accessed: 2022-04-20.

[3] Vl53l1x api user manual. page 27, 2021. URL: https://www.st.com/resource/en/user_manual/um2510-a-guide-to-using-the-vl53l1x-ultra-lite-driver-stmicroelectronics.pdf.

[4] Vl53l0x time of flight distance sensor. URL: https://www.botnroll.com/en/infrared/2531-vl53l0x-time-of-flight-distance-sensor.html.

[5] Vl53l0x api user manual. page 26, 2016. URL: https://www.st.com/resource/en/user_manual/um2039-world-smallest-timeofflight-ranging-and-gesture-detection-sensor-appl
pdf.

[6] World's smallest Time-of-Flight ranging and gesture detection sensor. page 40, 2021. URL: https://www.st.com/resource/en/datasheet/vl53l0x.pdf.

[7] Nikola Lakovic, Miodrag Brkic, Branislav Batinic, Jovan Bajic, Vladimir Rajs, and Nenad Kulundzic. Application of low-cost VL53L0X ToF sensor for robot environment detection. March 2019. URL: https://ieeexplore.ieee.org/document/8717779/.

[8] Sensor de distância vl53l1x de alta precisão 400cm. URL: https://www.marinostore.com/sensores/sensor-de-distancia-vl53l1x-de-alta-precisao-400cm.

[9] Using the programmable region of interest (roi) with the vl53l1x. 2018. URL: https://www.st.com/resource/en/application_note/an5191-using-the-programmable-region-of-interest-roi-with-the-vl53l1x-stmic
pdf.

[10] A new generation, long distance ranging Time-of-Flight sensor based on ST's FlightSense technology. page 35, 2021. URL: https://www.st.com/resource/en/datasheet/vl53l1x.pdf.

[11] John Kvam. Time of flight: Principles, challenges, and performance. 2017. URL: https://www.st.com/content/dam/technology-tour-2017/session-1_track-4_time-of-flight-technology.pdf.

[12] Teraranger evo 3m - close-range tof distance sensor, 3m, fixed 100hz, 12 grams. URL: https://terabee.b-cdn.net/wp-content/uploads/2021/02/Specification-Sheet-Evo-3m.pdf.

[13] Helios2 The next generation of Time of Flight. https://thinklucid.com/helios-time-of-flight-tof-camera/. Accessed: 2022-04-20.

[14] VL53L1CXV0FY/1. https://estore.st.com/en/vl53l1cxv0fy-1-cpn.html. Accessed: 2022-04-15.

[15] D.Hrubý D. Marko. Distance measuring in vineyard row using ultrasonic and optical sensors. 2020. URL: https://www.researchgate.net/profile/Dusan-Marko/publication/346611111_Distance_measuring_in_vineyard_row_using_ultrasonic_and_optical_sensors/links/5fc95fd8299bf188d4f1437c/Distance-measuring-in-vineyard-row-using-ultrasonic-and-optical-sensors.pdf.

[16] Marcin Kolakowski. Improving ble based localization accuracy using proximity sensors. 2018. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8611932.

[17] Marvin Lindner, Ingo Schiller, Andreas Kolb, and Reinhard Koch. Time-of-Flight sensor calibration for accurate range sensing. *Computer Vision and Image Understanding*, 114(12):1318–1328, December 2010. URL: https://linkinghub.elsevier.com/retrieve/pii/S1077314210001682, doi:10.1016/j.cviu.2009.11.002.

[18] Update: A new time-of-flight sensor builds on existing success stories. 2020. URL: https://blog.st.com/vl53l1x/.

[19] Async/await. https://web.dev/javascript-async-functions/.

[20] Raspberry pi 4b/4gb. URL: https://www.arrow.com/en/reference-designs/raspberry-pi-4b4gb-raspberry-pi-4-model-b-4gb-sdram-evaluation-board-based-on-bc b1deb068df5eda925714250e3776ad19.

[21] OPTICAL PROPERTIES OF GLASS: HOW LIGHT AND GLASS INTERACT. https://www.koppglass.com/blog/optical-properties-glass-how-light-and-glass-interact.

[22] JS library for the VL53L1X Laser Ranger. https://github.com/kr3l/vl53l1x-js. Accessed: 2022-05-06.

[23] Node.js library for a vl53l1x. https://github.com/upMICSD/vl53l1x-js/tree/second-attempt.

[24] A Node.js library for a vl53l0x proximity sensor. https://github.com/rip3rs/vl53l0x.git. Accessed: 2022-03-29.