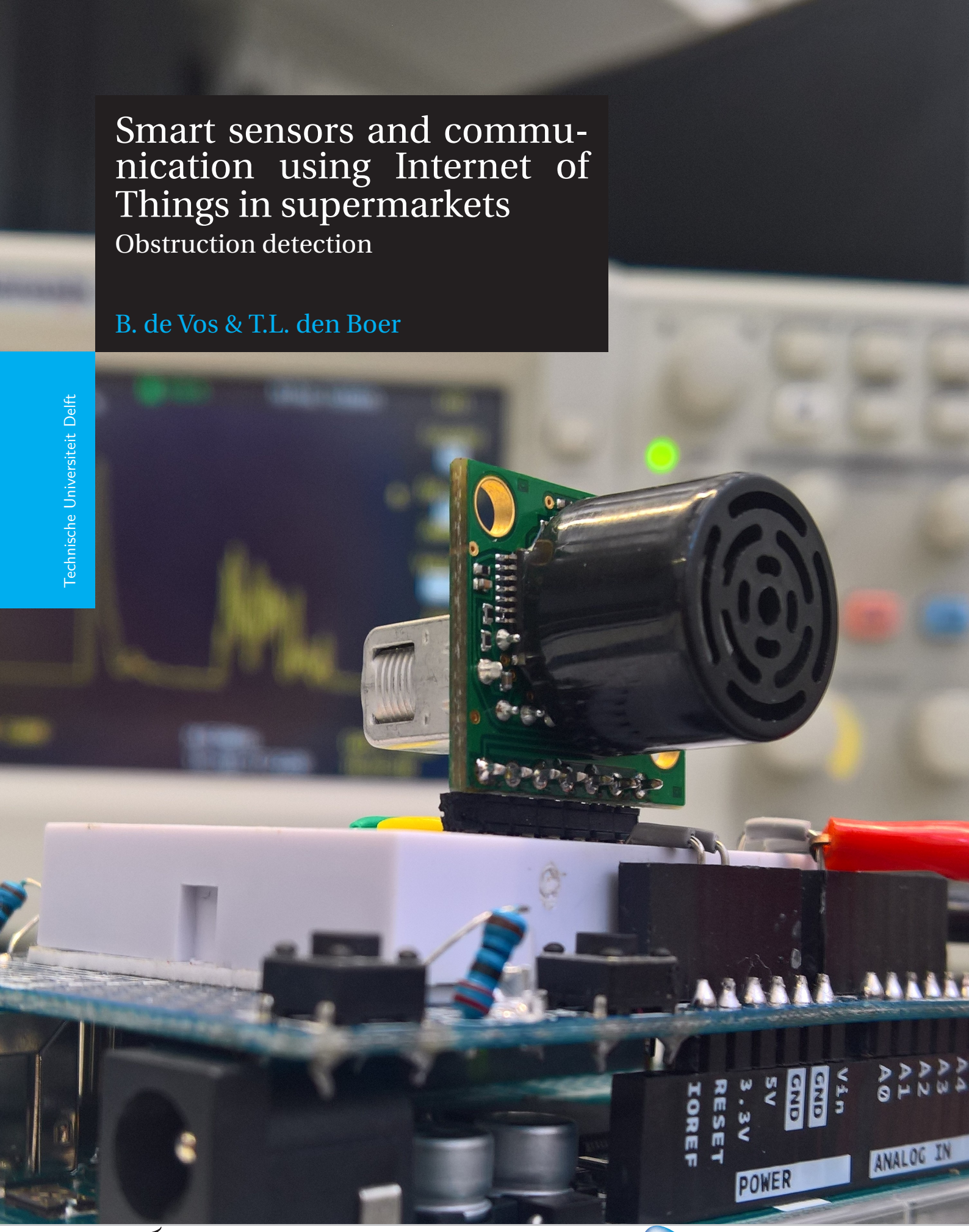
Smart sensors and communication using Internet of Things in supermarkets

Obstruction detection

B. de Vos & T.L. den Boer

Technische Universiteit Delft



TU Delft Delft University of Technology

kpn

SMART SENSORS AND COMMUNICATION USING INTERNET OF THINGS IN SUPERMARKETS

OBSTRUCTION DETECTION

by

B. de Vos & T.L. den Boer

in partial fulfillment of the requirements for the degree of

Bachelor of Sciences

in Electrical Engineering

at the Delft University of Technology,

Students:	B. de Vos, 4180038
	T.L. den Boer, 4210514
Project duration:	March 24, 2017 – July 7, 2017
Supervisors:	Dr. J. Hoekstra (TU Delft)
	B. Frens (KPN)
	Ir. P. Marcelis (KPN)

TU Delft Delft University of Technology

kpn

EXECUTIVE SUMMARY

Internet of things (IoT) applications become more and more prominent in our modern society. IoT has the potential to improve business processes and change the ways we live. KPN New Business has a high affinity with IoT projects and supplied the problem definition that this project builds on. The project has been done in close collaboration with KPN.

This document describes the design process for a stand-alone obstruction detection system for supermarket environments. The focus lies on obstructions near emergency exits, where they pose safety threats and obstructions in shopping aisle, where they hinder customers. The system should detect obstructions and communicate the presence of an obstruction to a central server. The communication is however not within the scope of this design report.

Five sensing techniques were considered for the detection system, ultrasonic ranging sensor were deemed the most suitable for this application. Ultrasonic sensors lend themselves well for IoT applications as they are easy to implement, cheap and operate on low power.

A MaxBotix MB1300 sensor was chosen for the implementation of the prototype as it offers an Analog Envelope (AE) output of the measurements. This allows for multiple objects to be detected in a single measurement. Furthermore, the MB1300 sensor has the largest beam width in the AE product line and can thus cover the most surface area.

The processing of the measurements in the prototype is done using an Arduino Mega 2560. The Arduino board allows for easy prototyping and has sufficient memory on-board to store and process the measurements. The measurements are sampled using the built-in ADC of the Arduino at a sampling frequency of approximately 8.93 kHz. The sampling frequency is a trade-off between the spatial resolution of the system and the memory required to store the measurements.

An algorithm is developed for the detection of objects from the sampled measurements. The process involves the windowing of the measurements to only look at relevant samples. A background subtraction is performed to avoid the detection of scenic objects, and the objects are detected using a threshold.

A data structure was created in order to store the detected objects. An algorithm was developed to check whether detected objects remain static. If an object is repeatedly detected it is marked as an obstruction. If an object goes unseen for a certain period, the object is removed from the memory. Thresholds for when an object is marked as an obstructions, the deletion of objects and measurement intervals are suggested for emergency exits and shopping aisle systems. These should however be verified in further development.

The performance of the system was evaluated in testing environments for the emergency exits and shopping aisles. Testing results showed that the system performs well on most design criteria. The area covered by a single sensor was however below expectations. This could be problematic for the implementation of the system in shopping aisles, as lots of sensors would be required to cover entire shopping aisles. Also some false positives occurred in the detection of obstructions. This could however easily be solved with minor changes to the detection algorithm.

Based on the testing results the conclusion is drawn that the design goals are partially met. The system is able to detect obstructions, but does not yet operate as a stand-alone unit. This design goal was however considered in every design choice. Power consumption of the prototype remains to be tested, this however is also dependant on the communication system.

Recommended is to further look into the power consumption of the system, the implementation of a multi-sensor system to cover more floor area in shopping aisles, and to reconsider CCTV images for the detection of obstructions in shopping aisles.

Contents

1

INTRODUCTION

This chapter holds the introduction to this thesis. First an introduction and background to the origin of the project is given in section 1.1. This is followed by the problem definition in section 1.2. In the subsequent section 1.3 the technical review is conducted. The technical review consists of a state of the art analysis in collaboration with an overview of different technical solutions that can be used for the thesis problem. This section is the basis for the solution validation. The chapter is concluded with section 1.4 which consists of an outline of the thesis.

1.1. BACKGROUND

The Internet of Things is no longer a new concept, and probably doesn't need an extensive explanation. The idea that soon the most basic electronic devices will be connected to the internet has become widely accepted. This bachelor graduation project has been conducted in close collaboration with KPN New Business. Part of what KPN New Business does is the development of new services within the Internet of Things. A client of KPN has identified different problems within a supermarket that can be solved with Internet of Things applications. (As part of a non-disclosure agreement, no further details on the client can be given.)

KPN has requested to research the possibilities of solving three of the problems as were identified by the client. These are:

1. Detecting obstructions of emergency exits and shopping aisles
2. Monitoring the stocking level and quality of the banana shelves
3. Connection and communication of sensors in a supermarket environment

The project was done with a group of six students. Subgroups of two students have worked on one of the three problems. The solutions to these problems will be integrated into one system. Figure 1.1 shows an overview of the entire system. In the end the banana shelve and obstruction detection sensors together with all other supermarket sensors will be communicate their sensor data to a central gateway in the store. From here the information will be send to the central processing centre outside the store. This thesis focuses on the first problem: detecting obstructions of emergency exits and shopping aisles. The connection and communication of the obstruction detection system does not fall within the scope of this thesis.

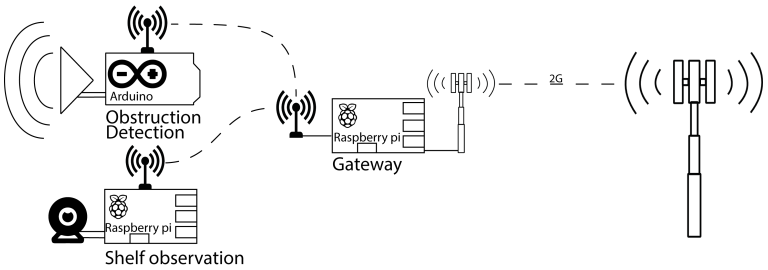


Figure 1.1: System overview of the integrated supermarket solution.

1.2. PROBLEM DEFINITION

The problem concerning the detection of obstructions within the supermarket can be subdivided into two components. The first is the detection of obstructions of emergency exits, and the second is the detection of obstructions in shopping aisles.

The emergency exits within a supermarket should always be free of obstructions. This is because an obstructed emergency exit poses a safety issue. Within a supermarket it may occur that objects are placed in front of an emergency exit. This can be caused by an inattentive shelf stocker that misplaced his stocking trolley or some empty boxes. It can also be caused by a customer abandoning his/her shopping cart. When an object is placed in front of an emergency exit, an employee should be notified in order to remove the object.

These obstructions due to abandoned restocking trolleys, shopping carts or simply boxes, can also occur in shopping aisles. This does not directly result in a safety issue. However it may look untidy, become a tripping hazard or create congestion. This directly impacts the customer satisfaction and therefore indirectly the supermarkets revenues.

1.2.1. GOAL OF THE PROJECT

Through KPN we were asked to look at possible solutions unifying these two problems. Based on the information offered by the client and in cooperation with KPN the following goal for the project was set:

Develop a proof of concept for a stand-alone obstruction detection system for supermarket environments.

1.2.2. OBJECTIVES OF THE PROJECT

Based on the goal stated above, six objectives were formulated:

1. Depict applicable sensing techniques and choose the best for the given problem.
2. Design a sensing system that can detect objects multiple static objects and neglect dynamic objects.
3. Design a sensing system that can detect obstructions near emergency exits.
4. Design a sensing system that can detect obstructions in shopping aisles.
5. Develop a functioning prototype.
6. Further develop the system to a stand-alone IoT system.

1.3. TECHNICAL REVIEW

This section contains a state-of-the-art analysis in order to determine which sensing techniques already exist for the detection of objects or obstructions. Concluding, a decision is made which sensing technique will be used for the supermarket obstruction detection system.

No specific studies were found for the detection of objects in a supermarket environment or near emergency exits. However, some studies were found concerning related problems. These include the detection of abandoned objects on surveillance cameras, and the detection of obstructions on railroad crossings.

1.3.1. SENSING TECHNIQUES

For the detection of objects within the supermarket five sensing techniques were considered: optical systems based on camera images, ultra wideband (UWB) radar systems, IR barrier systems, lidar based systems and sonar systems. Below, each of these systems are considered for the application in this project, based on scientific papers.

OPTICAL SYSTEMS

The detection of static or abandoned objects is something that has been extensively researched for applications in public spaces like airports or subway stations, in order to improve public security through the detection of suspicious objects. These detection systems are often based on CCTV (closed-circuit television) systems.

Bouchafa, Aubert and Bouzar [1] researched crowd motion estimation and the detection of motionless objects in subway corridors using CCTV cameras and image processing. The motionless objects they're trying to detect are for example: a person not able to get up after falling, a dealer selling drugs, a beggar or unattended objects. Their system locates stationary objects using a dynamically updated reference image, to detect novelties in the images, and a module that locates all areas where motion is taking place. In an experimental set-up they were able to detect 14 out 15 stationary situations.

Tian, Feris and Hampapur [2] have presented a new framework to robustly and efficiently detect abandoned and removed objects in complex environments for real-time video surveillance. They again use image processing on camera footage to detect static objects. Their system includes three main components: background subtraction and static region detection, object type detection and abandoned and removed object alert detection. Their testing results based on different scenarios showed that their approach can handle occlusions in complex environments with crowds.

Zhang, Chu and Chen [3] have worked on the system for determining obstacles in corridors of buildings after an earthquake, to assess for example if escape routes might be obstructed. Their method includes images taken before and after the earthquake, and a comparison of the two using an image foreground technique. In an experiment using over 40 pairs of images they were able to reach an accuracy in obstacle assessment of 84%.

The image processing often relies on advanced computer hardware making it difficult to implement in a wireless sensing system. Rohilla et al [4] have looked at abandoned object detection using FPGA accelerated image processing, such that it might once be implemented in a dedicated integrated circuit. Still the use of image processing is not suitable for our application due to the complexity and the given time frame the project needs to be completed in.

UWB RADAR SYSTEMS

UWB radar systems make use of large bandwidth, short radio wave pulses and measured reflections. Pulse lengths are usually in the order of nanoseconds. Advantages of UWB radar includes relatively low operation power as compared to continuous wave radar, due to the short pulses that are transmitted. However, UWB radar often relies on complex signal processing. UWB radar has applications in anti collision systems for vehicles, person or object localisation and tracking, land mine detection, through wall imaging and environment mapping. [5] These systems often rely on multiple antennas, or moving antennas in the case of synthetic aperture radar (SAR).

Vitucci et al [6] conducted a preliminary study on the use of UWB (Ultra Wide Band) band multistatic radar for a railroad crossing surveillance system. They looked at a surveillance system comprising of 4 TX-RX UWB sensors located at a height of 3 meters at the vertices of the monitored area. Their simulations show that for accurate obstacle image reconstruction, the 4-receivers configurations probably should be enhanced with the introduction of additional receivers/transmitters.

Mroué et al [7] presented a system for detecting and identifying objects fallen of railway platforms onto the railway tracks. Their solution is based on UWB radar combined with slotted waveguide transmission line. A rectangular slotted waveguide all along the platform is used as a succession of monostatic radars. Their system provides an effective continuous detection barrier along the platform. Simulations showed that object identification through correlation with objects stored in a library was not feasible. However, the singularity expansion method showed more promise for the object identification.

Kämäräinen et al [8] did research on detection algorithms for slowly moving humans and other objects near moving machinery to prevent collision. In their research they used a commercially available off-the-shelf UWB radar system. They were able to reliably detect human and non-human objects up to 10 meters in diffi-

cult environments, like a machinery environment. Their system relies however on complex signal processing performed using MATLAB on a PC system.

UWB radar systems may offer high precision at large ranges, but are most suitable for the detection of moving and/or metallic objects. Object detection using UWB radar relies on complex signal processing, and was thus deemed unsuitable for this project, given the limited time frame and hardware constraints.

IR BARRIER SYSTEMS

Infrared (IR) barrier sensors are based on emitters and detectors of IR light, that are placed in an array to cover a two dimensional space in which an object can be detected. When an object is placed in the plane of detection, it blocks one or more of the IR beams. This can be detected by the detectors. IR barrier sensors are widely applied in security systems, creating an invisible wall and in elevators to determine if the doors can be closed.

Garcia et al [9] presented a method to again, detect obstacles in railways, based on an IR barrier. Their sensor system is based on a barrier of emitters and another of receivers, placed on either sides of the railway. Their system was able to detect the presence of obstacles and inform about their locations even under adverse conditions.

IR barrier sensor were not further considered, as a large amount of sensors are required, making it difficult to install and possibly requiring structural changes to the supermarket. This is in violation of the programme of requirements as can be found in chapter 2.

LIDAR SYSTEMS

Lidar systems use emitted laser light to measure the distance to objects. The light emitted from the laser is reflected from an object and detected. The time-of-flight can be used to calculate a precise distance to an object. Due to the narrow laser-beam lidar offers ranging information at a very specific focus point. Using lidar in a scanning fashion, high resolution environment maps can be constructed. Because of this lidar is widely applied in fields of study geography and geology. More recently lidar has found applications in autonomous vehicles for object detection and avoidance.

Kim et al [10] proposed a lidar based railroad safety system. They introduce a laser rangefinder that is mounted on a mechanical system that can rotate and change the angle of the laser range finder. This makes it possible to scan an area for objects, creating a point cloud of the scanned data. The proposed system shows improved adaptability to weather changes and increased maintainability, as compared to existing systems.

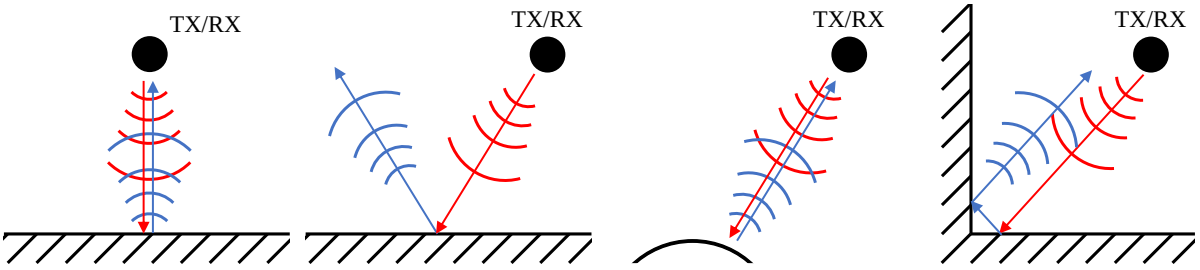
A lidar based system is deemed unsuitable for this project because the mechanical hardware involved in the scanning system would have a large impact on the battery life of the system. Also each measurement would generate a point cloud of scanned data, this would require large amounts of data which is not suitable for a distributed sensor system with simple hardware.

SONAR SYSTEMS

Active sonar uses a sound wave pulse to illuminate a certain area, the time of flight of the received echo is used to determine the distance between an object and sensor. Sonar is used in parking sensors, collision avoidance and environment mapping for autonomous robots, submarines navigation and medical applications like ultrasounds and echocardiography.

Wu, Abrahantes and Edgington [11] presented a multi-ultrasonic-sensor system for simultaneous localisation and mapping on the open source robot Kobuki Turtlebot. Their system includes 8 HY-SRF05 ultrasonic rangefinders. Their work demonstrated that ultrasonic sensors can be used as a cheap alternative to laser rangefinders for environment mapping.

Wu et al [12] implemented ultrasonic sensors in a real-time obstacle avoidance system for wheeled robots. Such that the robot can detect surroundings and avoid obstacles whilst moving towards a target. Their system included an ATMega162 embedded microcontroller as the computational core of the system.



(a) Flat object (b) Flat object under angle (c) Circular object (d) Corner object

Figure 1.2: Four different situations for acoustic reflections. The red wave represents a transmitted acoustic pulse, and the blue wave represents an echo.

Patkar and Tasgaonkar [13] designed a low cost object recognition system based on a horizontal array of ultrasonic sensors. They found that surface roughness is important in the recognition of objects with flat surfaces. Surface roughness improves the scattering of return pulses of the ultrasonic sensor. Their system was able to recognise a cylinder, cube and prism by measuring the object surface once. More complex objects like cones or pyramids required to be scanned at different heights.

Based on the information given above, sonar was selected as sensing technique for the detection of obstructions within a supermarket environment. Sonar is a technique that can relatively easily be implemented, requiring simple hardware. Sonar sensors require low power for operation, and when bought as simple piezo-electric transducers, they can be acquired cheaply. This allows for them to be implemented in sensor arrays to cover more surface, while not needing as many sensors as would be the case for IR barrier systems. The section below goes into a little more depth on the working principle of sonar.

1.3.2. SONAR WORKING PRINCIPLE

As was stated above, sonar uses sound wave pulses that are reflected from objects. For sound waves, reflections are generally called echos. Sound waves are produced when objects vibrate and transfer their vibrating motion to the surrounding medium. The result are longitudinal pressure waves that are travelling away from the source. The speed at which the sound waves travel depends on the medium. The speed of sound waves in air depends only on the composition and temperature, not on the pressure or density [14]. The speed of sound waves at room temperature is generally considered to be approximately 343 m/s [15]. The generation of sound waves for acoustic ranging is generally done using piezo-electric transducers. Capacitive transducer could prove to be more efficient, as their impedance is better matched to air, they are however not yet commercially available [16].

The propagation of sound in an elastic medium can be described by solving the wave equation. For sound waves with high frequencies (> 200 Hz), the propagation of the waves can be modelled using ray theory [17]. This allows for the use of the law of reflection, which states that for reflections, the angle of incidence equals the angle of reflection. Using this theory the reflections of sound waves can be estimated for different objects. in Figure 1.2, four situations are considered for the reflections of acoustic waves.

Figure 1.2a shows a sonar pulse hitting a flat object directly from above. The angle of incidence is 0 degrees, and thus so is the angle of reflection. The majority of the power in the acoustic wave is reflected back to the transceiver. Figure 1.2b again shows a sonar pulse hitting a flat object though this time under an angle. This could happen when the object is not directly in front of the transceiver or when the object has an angular surface relative to the normal of the transducer. (The normal being the direction in which the acoustic power is focused). In the case of the flat surface being hit under an angle, the majority of the acoustic power is reflected away from the transceiver. This makes it difficult to detect such an object. Figure 1.2c shows a circular (or spherical) object being hit by the sonar pulse. Circular objects have the property that there is always an angle that directly reflects the incoming pulse. Figure 1.2d shows the sonar pulse hitting an inside corner of an object. If an object has right inside corner, it acts as an corner reflector, having the property that it reflects waves back under the same angle, albeit translated.

1.4. THESIS OUTLINE

The subsequent chapters of this thesis describe the design process of the stand-alone obstruction detection system. In chapter 2 the programme of requirements is stated. Chapter 3 goes through the design of the system and is subdivided in the sensor selection 3.1, the sampling 3.2, the object detection 3.3 and the obstruction detection 3.4. In chapter 4 the implementation of the prototype is discussed. Chapter 5 discusses the testing of the prototype and presents the results. Lastly, in chapter 6 the final conclusions are drawn and some recommendations will be discussed.

2

PROGRAMME OF REQUIREMENTS

This chapter contains the functional requirements and the system requirements of the obstruction detection system. The aim of this project is the design of an Internet of Things sensor system that can detect obstructions of emergency exits and shopping aisles in a supermarket. A well functioning system should improve the safety within the store, and increase the customer satisfaction.

An obstruction here is defined as an object that remains static for an predetermined amount of time. These objects may be restocking trolleys or shopping carts that were abandoned in a shopping aisle or near an emergency exit.

When an obstruction is detected by the sensor, a notifications should be sent to a supermarket employee in order for the object to be removed. The notification and communication system falls however outside the scope of this design project.

2.1. FUNCTIONAL REQUIREMENTS

The following functional requirements were identified for the detection system:

1. The system should be able to detect (abandoned) objects near emergency exits and in shopping aisles within a supermarket environment.
2. The system should be easily installed without any major alterations of the supermarket.
3. The system should function autonomously, with as few human interaction as possible.

2.2. SYSTEM REQUIREMENTS

Based on the functional requirements, the system requirements were identified. The system requirements have been categorised as general requirements related to the performance of the system, and as environmental requirements, related to the environmental aspects in which the system should function.

General requirements:

1. The system should be able to to identify objects with dimension greater than 50x50x50 centimeters (LxWxH).
2. The system has to be able to discern a static object from a by-passer.
3. The system has to be able to measure the time an object remains static.
4. The emergency exit system should be able to detect an obstruction within 10 minutes.
5. The shopping aisle system should be able to detect an obstruction within 30 minutes.
6. The system should be able to function on battery power for a minimum of a year.
7. The system should have at most one false positive a day.

Environmental requirements:

1. The system should be ceiling mounted.
2. The system should be able to adjust to different surroundings and adapt to permanent changes to the environment.
3. The system should function with ceiling heights between 2.5 and 4 meters.
4. The emergency exit system should detect objects within a radius of 1 meters of the emergency exit.
5. The shopping aisle system should detect objects within aisle area of up to 4x2.2 meters (LxW).

3

DESIGN PROCESS

In this chapter the design process of the entire obstruction detection system will be discussed. The system will be gradually built up throughout this chapter. In section 3.1 different types of ultrasonic sensors are considered and the best sensor for this application is chosen. In section 3.2 sampling of the sensor data is discussed and an appropriate sampling rate is chosen. In section 3.3 the signal processing used prior to the object detection and the method for detecting objects is discussed. These sub-parts are integrated into an obstruction detection system for emergency exits and for shopping aisle. The algorithm used and the design of these systems is reported section 3.4.

Multiple sensing techniques were considered for the detection of static objects. These techniques were previously investigated in section 1.3.

3.1. SENSOR SELECTION

Static objects will be detected with the use of ultrasonic sensor data. This is done using a calibration measurement and comparing new measurements with the calibration. New objects will be marked and if left the system will communicate this as an obstruction. With this in mind different ultrasonic sensors were considered. Ultrasonic sensors come in various types and price ranges. They can differ in their directivity, range, the type of information they supply and their method of communication. The main criterion was the supplied information. This is considered the most important criterion because the supplied information is used to detect obstructions. When information supplied is taken as a category, ultrasonic sensors can roughly be divided in five sub-categories.

In their most basic form they are available as piezo-electric elements that can be used as a transmitter/receiver combination. Additional hardware is needed, like a pulse generator to actually send an ultrasonic pulse. They offer however, the most extensive information, as the raw measurements can be used.

On the other end of the spectrum are fully integrated ultrasonic ranging sensors. They provide a distance or time-of-flight to the closest measurable object within the beam-width. These sensors give limited information as only the distance to the closest object is communicated, they are however easy to acquire and cheap.

In between these two extremes are sensors which are integrated but also allow for the raw data to be read out. MaxBotix is a manufacturer of sensors that allow this. They supply sensors with a so called 'analog envelope output'. The sensor can output the real time reflected signal data which is measured within the sensor. This raw data can then be sampled and examined to detect (multiple) objects. Figure 3.1 shows a typical output response from the AE (analog envelope) output. This figure is acquired from the Maxbotix XL-MaxSonar datasheet [18]. The figures show that the distance and multiple objects can be acquired from this AE output.

When detecting obstructions the closest object may not always be the obstruction, it could also be part of the static environment, shelves for instance. The sensors which only return distance do not allow easy data manipulation, it's not possible to choose which objects to neglect as only the first reflection time or distance is returned to the user. Designing an own sensing unit is the best option as it allows the best output with raw measurements. However given the time frame this is not a feasible option. The sensors with the analog envelope output are a good alternative, the measured signal can be sampled and used to depict objects from the measurements. A limit of this sensor, is that is not clear to what extend prepocessing is done on the AE-output. The manufacturer will not release information about this fact.

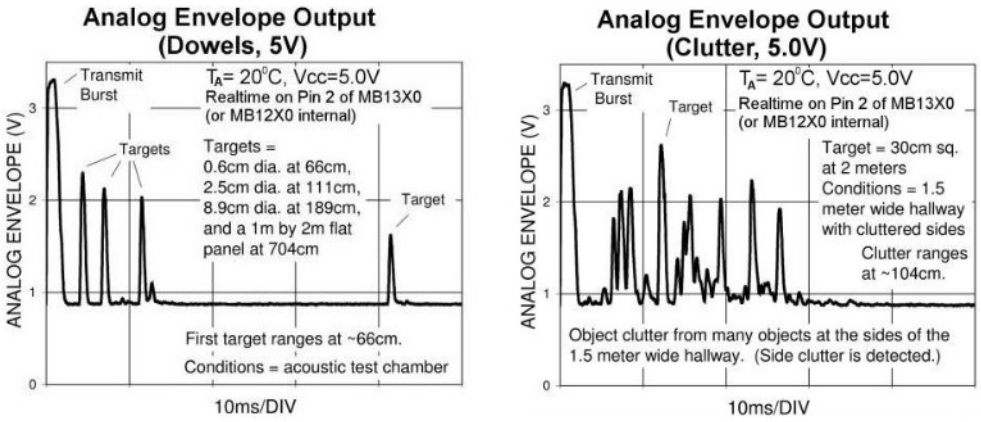


Figure 3.1: Typical AE-output waveforms [18]. Reflected echos cause peaks, the x-axis represents the time in which these reflections are received and is proportional to the distance.

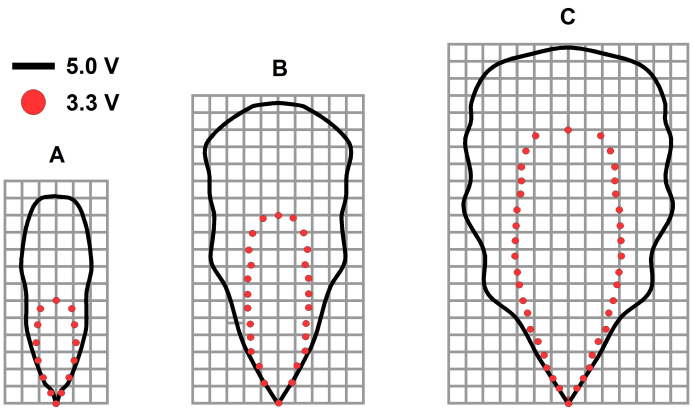


Figure 3.2: Beam patterns MB1300, 3-5 V supply voltage for different objects. Pattern A: 6.1 mm diameter dowel, Pattern B: 2.54 cm diameter dowel, Pattern C: 8.89 cm diameter dowel. One box represents 30x30 cm. [18]

Maxbotix has a variety of sensors that have the ability of an AE output. Within this selection we will look at range and beam width. The sensor will eventually be ceiling mounted, monitoring the supermarket floor. The PoR 2 states that the sensor must function with ceiling heights between the 2.5 and 4.5 meters. All AE sensors operate within this range. When looking at the beam width a larger width translates into more surface area that can be scanned with one sensor. Since signal processing is done shelves which fall into this larger beam width can be neglected. The MB1300 has the largest beam width and is the sensor that is chosen. Figure 3.2 shows the beam pattern of the MB1300. The sensor can operate between 3.3 and 5V. The figure shows that under the right circumstances a maximum floor diameter of approximately 4m can be reached.

3.2. SAMPLING

The ultrasonic sensor has an AE output that is sampled in order to perform digital signal processing and ultimately depict obstructions from the measured values. In this section the sampling of the analog signal is discussed. First the possible sampling rates are determined. After this the different operation modes of the ADC are depicted. This is then used in collaboration with test results to determine the appropriate sampling rate and operating mode for this implementation.

Table 3.1: ADC clock rates, prescaling factors and resulting sampling rates.

Prescaling factor	ADC clock rate (kHz)	Sampling rate (kHz)
2	8000	571.4
4	4000	285.7
8	2000	142.8
16	1000	71.43
32	500	35.71
64	250	17.85
128	125	8.929

SAMPLE RATE

According to the Nyquist-Shannon sampling theorem all information within an analog signal can be fully recovered if the signal is sampled using a sampling frequency of at least twice the highest frequency present in the signal [19]. For the sampling of the analog envelope signal full information recovery is not the aim. The sampling should be sufficient to be able to detect an object, and distinguish it from possible other objects. There's a trade-off in determining the sampling frequency. A higher sampling frequency will result in more samples per measurement. A larger amount of samples will take up more memory space, and increase computation time. Limited memory plays a large role when the system is to be implemented on microcontroller hardware. On the other hand, reducing the sampling frequency will reduce the spatial resolution, making it harder to distinguish between objects of the same height.

The sampling frequency that will eventually be used is constrained by the available hardware. For the prototype implementation an Arduino Mega 2560 is used. Further details on the prototype implementation can be found in chapter 4. The Arduino is based on the Atmel ATmega 2560 microcontroller [20]. The microcontroller has a built-in ADC (Analog to Digital Converter), which converts the analog voltage to a 10-bit digital value. The micro-controller has a 16MHz system clock. The ADC has a separate clock which runs on a lower frequency; the system clock is divided using a prescaling factor. The sampling frequency of the ADC is determined by the prescaling factor and the operating mode of the ADC.

The ADC has three different operation modes: Single conversion, free running and Autotriggerd. The function used on the Arduino to sample analog data is the analogRead() function. This function is defined in the wiring_analog.c library [21]. Using this function, the ADC operates in single conversion mode.

In single conversion mode, one ADC conversion takes 13 ADC clock cycles, except for the first conversion which takes 25 cycles due to the initialisation of the analog circuitry. After the ADC conversion the result can be retrieved from the ADC, and a new conversion can be started upon the next ADC clock cycle. When starting conversions one after the other, conversions can thus be done every 14 ADC clock cycles. A timing diagram for the ADC operating in single conversion mode can be found in appendix A.

Table 3.1 gives an overview of the available prescaling factors and the theoretical sampling rates, given that conversions are done every 14 cycles. The sampling rates were calculated using equation 3.1. In this equation f_s is the sample rate, and pf the prescaling factor.

$$f_s = \frac{16MHz}{pf \cdot 14} \tag{3.1}$$

Figure 3.3 shows four ranging measurements using the MB1300 sensor, where the analog envelope envelope signal is measured at different sampling rates. The sample rates used are 100kHz (Figure 3.3a), 35.71 kHz (Figure 3.3b), 17.85 kHz (Figure 3.3c) and 8929 Hz (Figure 3.3d). The 100 kHz sampling was performed using a Tektronix TDS2022C oscilloscope. The other measurements were taken directly using the Arduino. During the measurements, the sensor was placed in front of a wall at a distance of approximately 2.2 meters. The sensor was elevated from the floor by approximately 1 meters. The first peak that is visible in the graphs shows the transmit burst and and marks the start of the ranging measurement. The second largest peak in the figures, around sample 200 in Figure 3.3d, represents the echo returned from the wall.

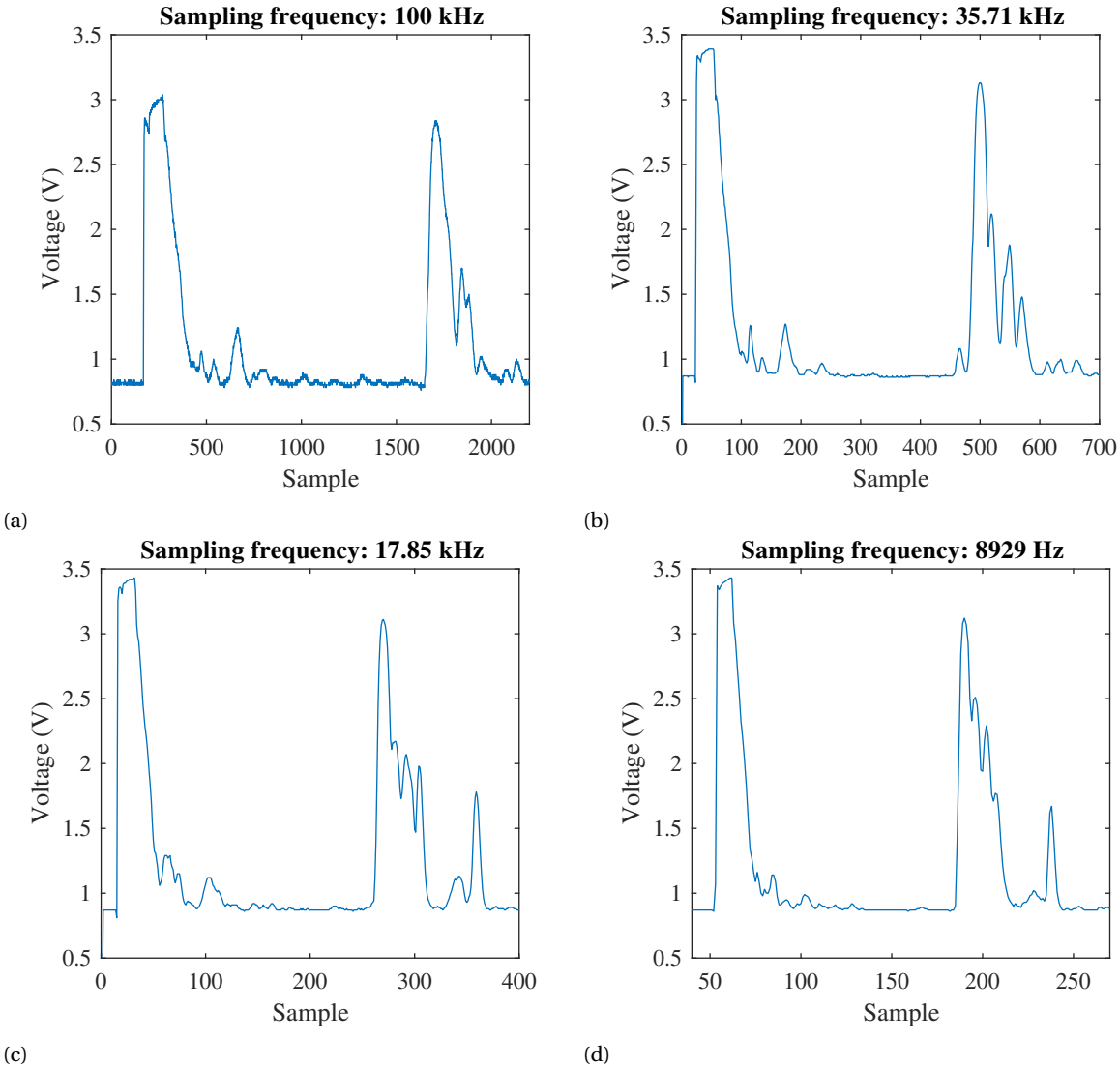


Figure 3.3: Four ranging measurements using the MB1300 sensor. The Analog Envelope output is measured using different sampling frequencies

The measurements taken using the Arduino show slightly larger peaks than the measurement using the oscilloscope. This is probably due to a scaling difference. The ADC of the Arduino returns a 10 bit value, an integer between 0-1023. These bits represents a value from 0V to Vcc, where Vcc is 5 V. In order to convert the ADC output into a voltage, the value is multiplied by $5/1023 \approx 4,9mV$. As long as this scaling difference is constant, it has no impact on the object detection. Furthermore, for the object detection, the 10 bit ADC output is not converted into a voltage. This is because the 10 bit value can be represented by an integer, requiring only two bytes, whilst a voltage would be represented by a float, requiring twice the amount of memory. Converting to a voltage does not increase the detection accuracy, as no information is added.

The measurement taken using a sampling frequency of 8929 Hz (Figure 3.3d) shows no significant reduction in spatial resolution as compared to the measurement using a sampling frequency of 35.7 kHz (Figure 3.3b). The measurements show slightly different peaks, but this was because the sensor was displaced slightly during the measurements. According to equation 3.2 a sampling rate of 8929 Hz results in a spatial resolution of approximately 2 cm. In equation 3.2, f_s is the sampling rate and v_s the speed of sound. The speed of sound in air is approximately 343 m/s. The factor $\frac{1}{2}$ is included because an echo needs to travel twice the distance to an object.

$$resolution = \frac{1}{2} \cdot \frac{1}{f_s} \cdot v_s \tag{3.2}$$

Using equation 3.3 the amount of samples necessary to cover a ranging distance of distance d, can be calculated. In this equation f_s again is the sampling rate and v_s the speed of sound, 343 m/s. A sampling rate of 8929 Hz would require 208 samples to cover a ranging distance of 4 meters. When saved as an array of integers this measurement would take up 416 bytes of memory.

$$N_{samples} = 2 \cdot d \cdot \frac{f_s}{v_s} \tag{3.3}$$

Given the information above, a prescaling factor of 128 was deemed sufficient for the purpose of detecting objects. Theoretically, this would result in a sampling rate of approximately 8929 Hz. Per measurement 250 samples are taken, this would cover a range of 4.8 meters. These samples are then saved in an array of integers. The 4.8 meters is slightly more than the required 4 meters. It should be noted however that the sampling does not start at exactly the same time as the transmit burst. The transmit burst generally starts at sample 5. This can be seen in Figure 3.5. The 250 samples also take into account the width of the floor pulse. The next section describes a measurement that was done in order to verify the sampling rate.

MEASURING THE SAMPLE RATE

A script written to measure whether the actual sampling rate equals the theoretical sampling rate. During the measurement, 300 samples are taken. The time before and after the sampling are saved and then subtracted from each other. This gives the total time for taking the 300 samples. This process was repeated 300 times.

At an ADC clock rate of 125 kHz one ADC clock cycle takes 8 μs. 300 samples would theoretically result in a total of $8 \cdot 14 \cdot 300 = 33600$ μs. The results showed that the time it takes for taking 300 samples was consistently more than 33600 μs. Because the ADC clock is not reset between conversions, the additional time between samples must be an integer multiple of 8 μs.

Figure 3.4 shows a histogram of the additional ADC clock cycles that were measured over 300 samples. The additional ADC cock cycles were calculated by subtracting 33600 μs from the measured time, dividing it by 8 μs and rounding the result down to the nearest integer. On average 300 samples takes an additional 4 ADC cycles. There's one outlier that took an extra 14 cycles. This can be explained by the fact that the first AD conversion takes 25 clock cycles instead of 13. The additional clock cycles show a flaw in the sampling method, but did not pose any problems for the detection of objects as is described in the next section. It is however recommended to further look into the sampling, in order to improve upon this problem. This is also discussed in section 6.2.

3.3. OBJECT DETECTION

This section describes how objects are detected from the sampled measurements.

Figure 3.5 shows two typical measurements taken with the MB1300 sensor suspended from the ceiling at a height of approximately 3.10 meters in a simulated supermarket environment. The testing environment is further discussed in chapter 5 and appendix B. The first peak in the graphs, around sample 20, represents the transmit burst. The peaks around sample 175 represent the echos returned from the floor. In fact, the first peak returned from the floor starts 160 samples after the start of the transmit burst. Using equation 3.3 this translates to a distance of 3.08 meters. The aim of the object detection process is to identify both the easy and the difficult to detect objects.

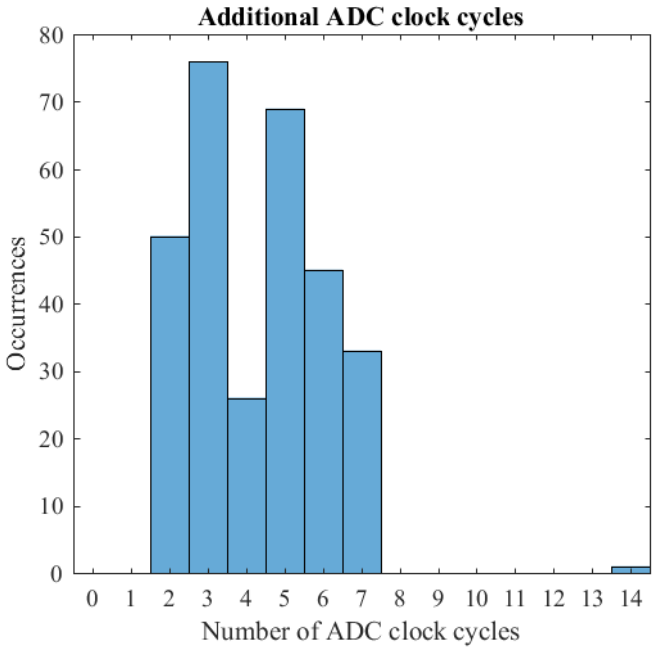
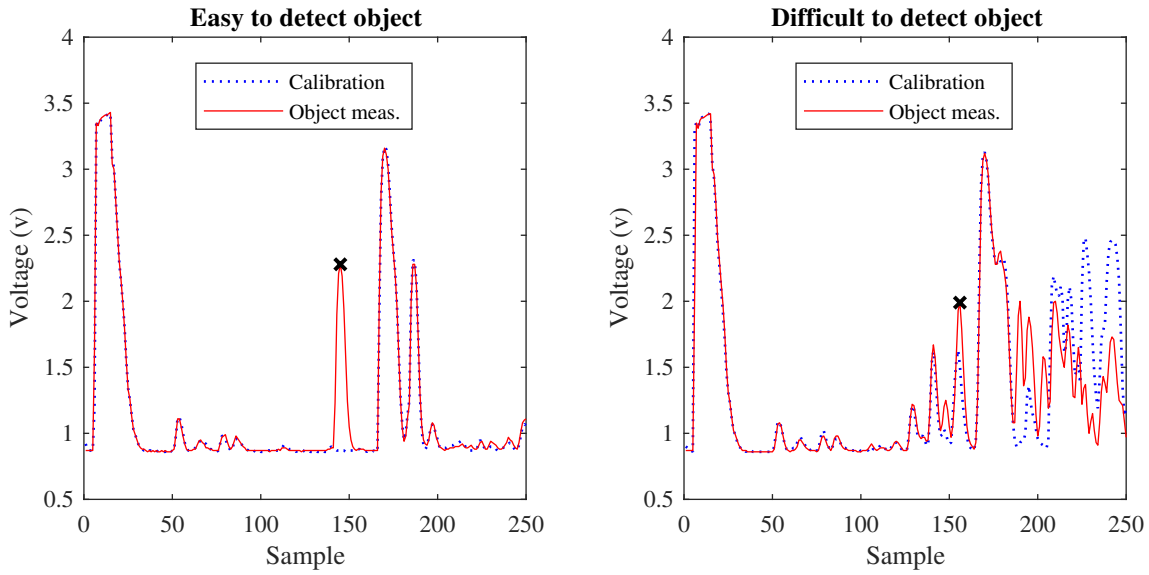


Figure 3.4: Histogram of the additional ADC clock cycles that where measured when taking 300 samples



(a) (b)

Figure 3.5: Typical measurements taken with the MB1300. An easy to detect object is an object directly underneath the sensor. A difficult to detect object is an object that is placed under an angle underneath the sensor. Both the calibration and object measurements are visible. The objects are marked with a black cross.

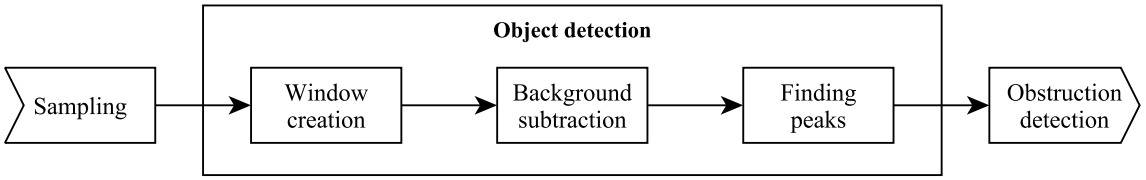


Figure 3.6: Graphical overview of the object detection precess.

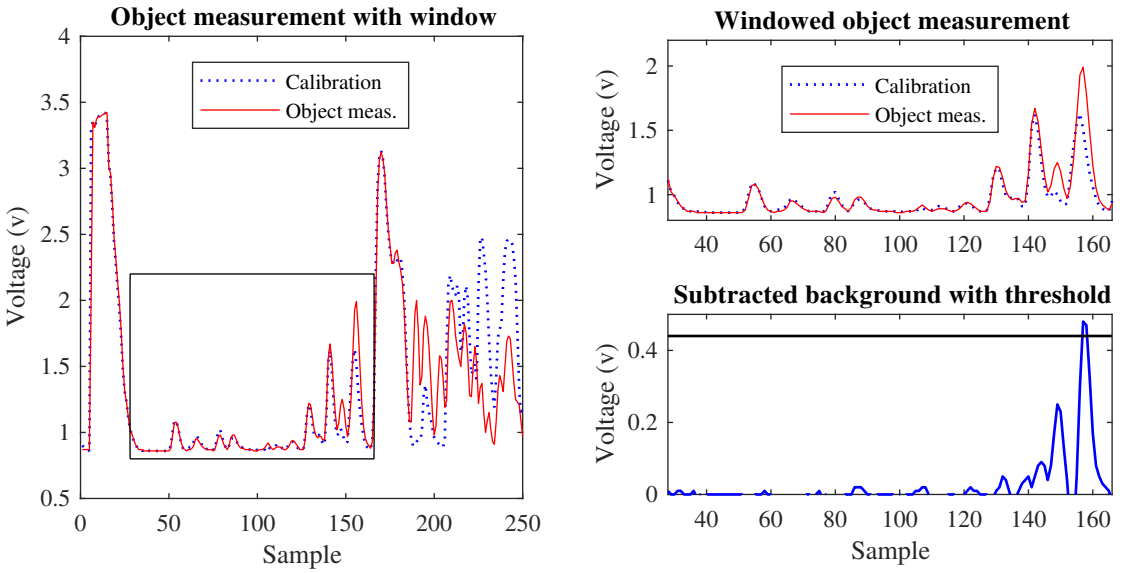


Figure 3.7: Measurement of a difficult to detect object including window (left graph). Measurement zoomed in on the window (upper right graph). And the background subtracted signal including a threshold of 0.44 volts.

Figure 3.6 shows a graphical overview of the object detection process. During the sampling an array is filled with the samples representing the analog envelope of a ranging measurement. First a calibration measurement is made and stored. This calibration measurement will be used to avoid the detection of objects that are part of the environment. Next, new measurements are made to detect objects. In the object detection process, first a window is created in which objects will be detected. Next a background subtraction is performed, and lastly the peaks representing objects are detected. Below, this process is described in more detail. Also, the code implementing the object detection can be found in appendix D.

WINDOW CREATION

Figure 3.7a shows the same measurement as Figure 3.5b. The figure shows that the sensor receives echos even beyond the echo returned from the floor (the peak around sample 175). This can be explained by the fact that the sensor also receives indirect echos that have travelled further than the direct echo from the floor.

In order to make sure the detection algorithm only looks for objects between the ceiling and the floor, a window is created. Outside this window, peaks will be disregarded.

For the creation of the window, the calibration measurement is used. First the transmit burst is detected by looking for the largest value in the array. By definition, the largest value corresponds to the transmit burst. Next, the floor echo is detected, by looking for the largest value between sample 100 en 250. This corresponds to a distance between approximately 1,8 and 4,7 meter, when taking in consideration that the transmit burst starts around sample 5. The window is then created using the position of the transmit burst and the floor.

In Figure 3.7a the window is marked by the rectangular box. The upper graph of Figure 3.7b shows the measurement zoomed in at the window.

BACKGROUND SUBTRACTION

Figure 3.5b shows that besides the object, there are other peaks prominently present. These can be echo returns from parts of the scenery. In the case of a shopping aisle, these could be echos from the product shelves. The calibration measurement is used as a background map of the environment in which the sensor operates. The background (calibration) is subtracted from the new measurements to avoid the detection of parts of the environment. The lower graph in Figure 3.7b shows the windowed measurement in which the background is subtracted.

For the background subtraction it is essential that the calibration measurement and the new measurement are exactly aligned. Otherwise peaks might pop up that are actually part of the background. Test results showed that sometimes there is a mismatch with the calibration measurements, where a new measurement leads by one sample. This did not yet lead to problems with the detection of objects. However, this could start to play a role when more precision is required from the object detection process. It is recommended to further look into this problem, see also section 6.2.

FINDING PEAKS

The last part of the object detection process is the detection of the peaks in the background subtracted signal. The criterion for a peak is: The value of the sample should be larger than the values of both the previous and next sample. Apart from that, the value of the sample should exceeds a certain threshold. The value for the threshold is a parameter that determines the sensitivity of the object detection. With a larger threshold certain objects with poor reflections might go unnoticed. However with a smaller threshold, peaks might be falsely marked as objects. The threshold value used in the final prototype implementation is 90. This number is on a scale from 0-1023, and corresponds to $90 \cdot 5/1023 \approx 0.44V$. The lower graph in Figure 3.7b shows this threshold as a black horizontal line. Using this threshold the object as marked in Figure 3.5b can successfully be identified.

3.4. OBSTRUCTION DETECTION

Once the objects have been successfully identified by the sensor, a check has to be performed to determine if the object is static. Next, if the object remains static for a certain amount of time, it has to be labelled as an obstruction. This section describes the algorithm that was designed for the detection of obstructions.

DATA STRUCTURE 'OBJECT'

In order to store the detected objects for later reference, a new data structure was introduced that represents detected objects. The data structure 'Object' holds the following information:

- Location [Integer]
- Lifetime (Lt) [Integer]
- Last Seen (LS) [Integer]
- Check Flag [Boolean]

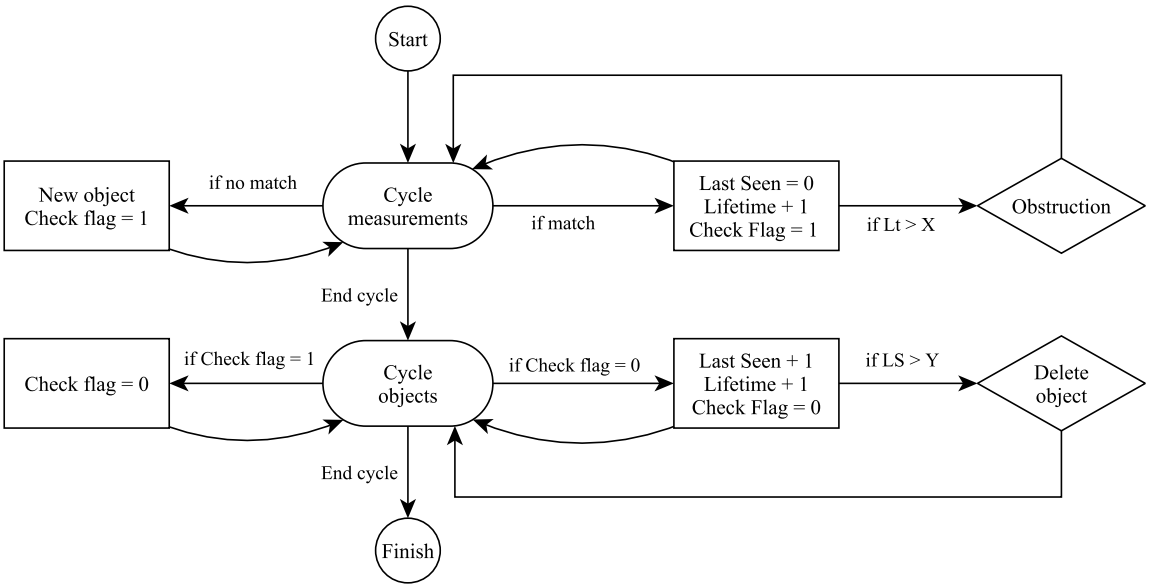


Figure 3.8: Flowchart of the obstruction detection algorithm.

The **Location** holds the sample number of the peak in the measured signal. This is proportional to the absolute distance between the sensor and the object.

The **Lifetime** counts how long an object has remained static.

The **Last Seen** counts the time since an object was last detected.

The **Check Flag** is used in the obstruction detection algorithm as is explained in below.

ALGORITHM

Figure 3.8 shows a flowchart representation of the obstruction detection algorithm. First a cycle is made through the newly detected objects, or measurement. Each detected object is compared with existing objects; objects detected earlier. If a match is found based on the object location, the objects information is edited. The Last Seen time is reset, the Lifetime increased and the Check Flag is set 'true'. Additionally, if the Lifetime exceeds a certain threshold X, the object will be marked as an obstruction. If a newly detected object does not find a match, a new object should be created and saved.

Next, a cycle is made through the existing objects in order to determine whether an object has been removed from the environment. Here, only object are considered that weren't newly created or edited in the first cycle, thus only objects with a Check Flag that has remained 'false'. With the object detection, false negatives can occur as a result of larger objects casting an acoustic shadow over the smaller objects. An object can then go undetected while it's actually still present. In order to not instantly delete such an object, first only the Last Seen and the Lifetime counters are increased. If the object goes undetected long enough, the Last Seen counter exceeds a certain threshold Y, then the object will be removed from the memory. During the object cycle, also all Check Flags are reset for the next measurements cycle.

EMERGENCY EXIT AND SHOPPING AISLE

The obstruction detection is designed to work for both the emergency exit and shopping aisles. The dynamic window and background subtraction features allow it to be mounted on any ceiling within the height requirements. stated in the PoR (chapter 2). The amount of by passers and the importance of an obstruction are different for the emergency exit and shopping aisles. A shopping aisle is much more prone to by passers, obstructions are less important. With emergency exits the obstructions are important as they tend to cause safety issues, there will be less people walking in range of the sensor. With this in mind and the PoR the following thresholds and timing intervals were set.

Emergency exit:

- lifetime threshold: 3
- last seen threshold: 3
- measure interval: every 2 minutes

The system performs an iteration or measurement every 2 minutes, when an object is detected, lifetime = 0 on the first detection. An obstruction will be marked after five iterations. This results in marking a static object as an obstruction after 10 minutes. This corresponds to the PoR. Shopping aisle:

- lifetime threshold: 4
- last seen threshold: 3
- measure interval: every 5 minutes

The same can be said for the shopping aisle thresholds. An object that is static for 30 minutes is marked as an obstruction.

These thresholds and intervals are starting points. Excessive testing of the sensor in real life situations and power consumption tests need to be done in order to verify these thresholds.

4

PROTOTYPE IMPLEMENTATION

In this section the design from the previous chapter is integrated into a prototype. This chapter holds the top level description of this prototype. Section 4.1, elaborates with which hardware the prototype is built and why. The purpose of the prototype and it's key features are discussed in section 4.2. A picture of the prototype can be found in the appendix (Figure B.1a).

4.1. HARDWARE

For the prototype the Arduino Mega 2560 is used [22]. The Arduino development platform was chosen because it allows easy connection of sensors. In addition to this, Arduino supports C/C++ which is a familiar language and can be used for object oriented programming. For the type of Arduino, the Mega 2560 was chosen, it has more memory that other Arduino boards and was already in our personal possession. The extra memory is needed for storing the sampled signal, objects and preforming signal processing. The sensor used is the MaxBotix MB1300, as stated earlier it is chosen for it's beam-width, range and because of it's Analog Envelope output.

Figure 4.1 shows an overview of how the prototype is wired. The sensor is fed 5 volt DC from the Arduino board. The sensor could also be fed with 3 volt however this is not chosen because it will decrease the range and beam-width of the sensor as can be seen in Figure 3.2. A ranging measurement is initiated by pulling pin 3 of the MaxBotix sensor high for a minimum of 20 ms, this is done by the digital output of the Arduino. Ultrasonic sensor pin 2 outputs the analog envelope, this output is read by the analog input of the Arduino. The Arduino is connected to the computer via USB, this is mainly done to retrieve sensor data for debugging and testing. This is also the prototypes source of power. In the final product the system will send it's data wirelessly and operate on battery power.

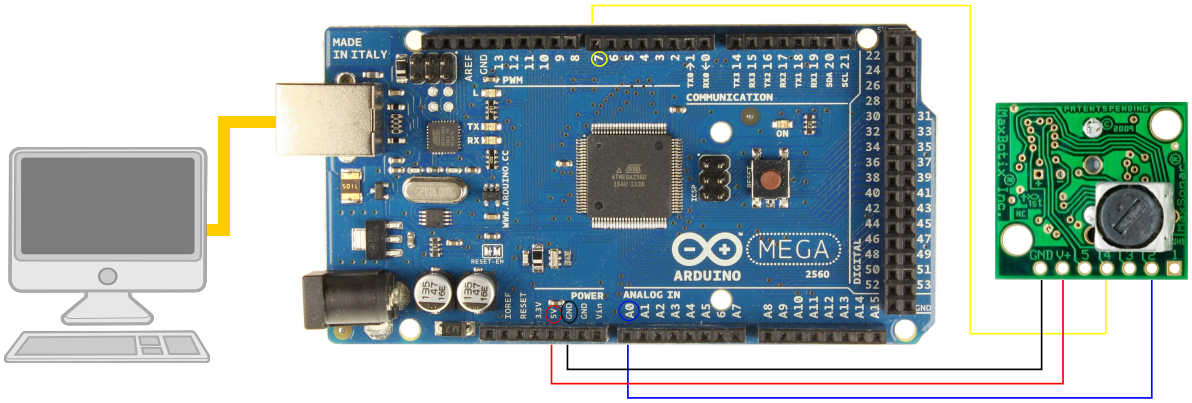


Figure 4.1: Overview of prototype components including the wiring, pictures (from left to right) attained from:
http://clipartall.com/img/clipart-224900.html,
http://www.electroschematics.com/7963/arduino-mega-2560-pinout/,
https://www.maxbotix.com/Ultrasonic_Sensors/MB1300.htm

4.2. PROTOTYPE FEATURES

The prototype is more than a proof of concept it is developed in order for the system to be tested and further developed. The code used for this prototype can be found in chapter D of the appendix. We will not go into this code too much as it's simply an implementation of the different design features discussed earlier.

Important to note are the different output modes. When testing, the prototype can output data via serial communication to the computer. This is used for debugging and observing the outputs throughout the design of this system. Matlab is used to read out the COM port and graphically display the desired outputs. Two possible output modes will be discussed which are both used for the testing in the next chapter:

- Object detection mode: The system conducts measurements, all signal processing is done, the found peaks are communicated to the computer. This mode is used for testing the range of the sensors and the effectiveness of the signal processing.
- Obstruction detection mode: The system does the same as with Object detection mode, only now the obstruction algorithm is also used. The found peaks are communicated to the obstruction detection algorithm. The system outputs the objects that are created. This is used in order to evaluate the obstruction detection algorithm. The final system will only communicate if there is an obstruction.

5

EVALUATION

In order to evaluate the performance of the obstruction detection system a series of tests have been conducted. In this chapter these test are elaborated upon, results are evaluated and compared to the test criteria which were formed using the programme of requirements stated in chapter 2. These test criteria are fundamental when analysing the system performance and are coupled to every test in this chapter. Throughout this chapter, the following format is used: the test method and setup are discussed and the test criteria applicable are coupled, the results are stated followed by a discussion on how well the criteria are met and if there is room for future improvement.

The chapter starts with the testing of sensor coverage area, this is done in section 5.2. In section 5.3 the tests performed on the emergency exit are stated. This section also holds an extensive analysis of the object detection algorithm. In section 5.4 the shopping aisle obstruction detection performance is tested, this section is more focused on the false positives and negatives.

5.1. TEST CRITERIA

The system performance will be evaluated using the following criteria. The system must be able to:

1. General:
    1.1. Detect typical supermarket objects, the smallest detectable object should have the dimensions 4x25x30 cm (LxWxH).
    1.2. Detect new objects whilst neglecting the background (calibration) measurement.
    1.3. Be able to adapt to different environments and permanent changes to the tested environment.
2. Emergency exit obstruction:
    2.1. Detect a typical supermarket object within a radius of 1 meter in front of the emergency exit.
    2.2. Detect obstructions: An object that remains static for 2 minutes.
    2.3. Neglect dynamic objects.
    2.4. Have at most one false positive a day.
3. Shopping aisle obstruction:
    3.1. Detect a typical supermarket object within a shopping isle length of 4x2.20 meter (LxW).
    3.2. Detect obstructions: An Object that remains static for 2 minutes
    3.3. Neglect dynamic objects.
    3.4. Have a maximum of one false positive a day.

For the sake of testing in these test criteria the time in which a static object must be marked as an obstruction is lowered. The time between iterations (a total system cycle) is also lowered with respect to the PoR. An iteration is done every 20 seconds.

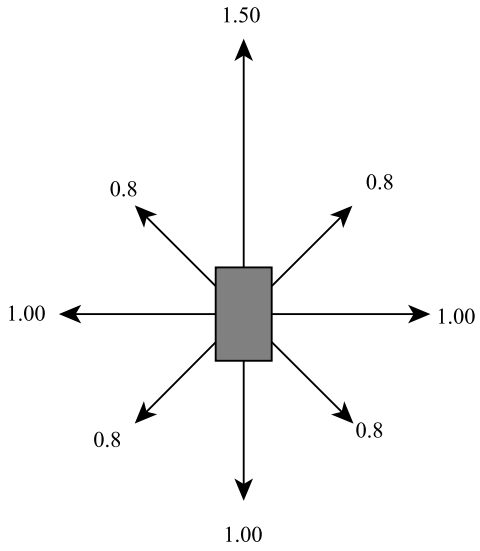


Figure 5.1: Top view mapping of range measurements in which a 40x25x30 cm cardboard box could be detected. Values in figure are in meters,

5.2. AREA COVERAGE

In order to evaluate the performance obstruction detection the limits of the object detection first need to be tested. In this test the area in which the prototype can detect an object will be determined. Design requirement 1.1 was tested. Also predictions on criteria 2.1 and 3.1 can be made after this test.

SETUP

For this test the sensor was mounted on the ceiling at a typical height of 3.1 meters. The used object is a 40x25x30 cm (LxWxH) cardboard box, which is used throughout all measurements in this chapter. The box represents a typical supermarket object and meets the size requirements. This box was moved over a floor grid. The box was kept in the same orientation throughout the test.The system was put in object detection mode in order to evaluate if the object was detected. This mode was discussed in the previous section.

RESULTS

Figure 5.1 shows a top view scale map in which the box can be detected. The values at the top of the arrows give the maximum range in which the object was able to be detected.

DISCUSSION

The results do not need further clarification as the test is rather simple. However there are a few important notes that can be made from these results. The area in which the object can be detected is not symmetrical. This could be due to the sensor accidentally being mounted with a slight angle, or due to the wooden block on which the sensor is mounted (Figure B.1b) blocks part of the ultrasonic pulse. Further testing is needed in order to pinpoint what causes this problem.

When comparing the maximum detectable floor diameter of 2.50 meter with the beam-width patterns from the Maxbotix data-sheet (Figure 3.2). It can be concluded that this object falls into the category B. It's important to know why this large object does not fall into category A. There are two possible explanations for this. The preprocessing and detection don't not work properly: the threshold is set too high and objects that could

have been seen in the analog envelope are not presented. The other explanation could be that the object orientation/material/size is not ideal for detection. After closely examining the measurements it became clear that when an object could be observed in the analog envelope or "raw" measurements, it was also detected by the system. The poor range is caused by the object. In chapter 1 it was already mentioned that the material and shape of an object is fundamental for it's detection. The material cardboard and the box shape are not ideal for the detection using ultrasonic sensors. However the cardboard box is a typical supermarket object. It was chosen because of it's poor shape and material in order to test the limits of the sensor system. When the box is set at the edges of the testing grid ultrasonic waves bounce off it's flat surface, this was explained with Figure 1.2b. Other supermarket objects for instance a shopping kart or a shelve stocking trolley have more corners and will most likely be detected at a larger distance from the sensor. In section 5.4 this is put to the test when a shopping kart is used as an object.

What does this mean for the requirement stated earlier. Detecting a typical supermarket object within an aisle area of 4x2.20 (LxW) meter can not be guaranteed. This means that not all requirements for the shopping aisle detection will be met. For the emergency exits there does not seem to be a problem, the detection range of 1 meter around a emergency exit door can still be achieved.

5.3. OBSTRUCTION DETECTION: EMERGENCY EXIT

In this section the prototype performance of the system mounted on the ceiling above an emergency exit is evaluated. design criteria 1.2, and all design criteria of the emergency exit are tested. This is done in multiple tests. First the setup of these tests are discussed followed by the results. The section is concluded with a discussion of the results.

SETUP

The sensor was mounted above a typical emergency exit at a height of 2.50 meter. Figure B.1d in the appendix shows a picture of the emergency exit the sensor is mounted above. The following tests were conducted:

1. The range of the sensor was tested, criteria 2.1 is applicable. This is done by moving the cardboard box in a radius around the emergency exit. The system was put in object detection mode.
2. Secondly the background subtraction was tested. this is done for two environments: The "empty" emergency exit and the emergency exit with objects around it. The system is re-calibrated as if the objects were permanent changes of the environment. criteria 1.2 and 1.3 are tested with this test. The system is again put in object detection mode to check if false positives occur (criteria 2.4). A false positive occurs when an obstruction is detected while there is no obstruction present.
3. With this test the obstruction detection was evaluated (criteria 2.2). The system is put in obstruction detection mode. In this mode detected objects are stored, static objects should be observed and dynamic objects should be deleted. The system iterates every 20 seconds, this is for testing purposes. In the real supermarket situation the system should iterate less often to improve power consumption. With this test an office chair is first wheeled in front of the emergency exit and stays there for 2 iterations. This object is interesting because of it's shape, multiple peaks will be detected. The object detection and deletion is tested. After this the cardboard box is set under the sensor, the box is left there as an obstruction for 8 iterations. Finally people will walk randomly in range of the sensor in order to test if the system can neglect by passers. At the end all objects are removed. The output of the sensor is observed and put into a table.

RESULTS

In this section the test results of the obstruction detection for emergency exits are stated.

Area coverage The box is moved in-front of the emergency exit the maximum distance where the object can be detected is equal to 1.20 meters right in front of the exit and at 1 meter maximum to the sides. This coincides with the statement made in the previous section, that the cardboard box falls into the category B of the beam pattern figure (Figure 3.2.

With respect to adjusting to different environments the system performs well. The permanent changes to the background were not detected as objects when the system was re-calibrated. The system ran for 20 iterations without a false positive. In addition to this the difference in ceiling-height with respect to the previous test does not seem to be a problem. Test criteria 1.3 is met.

To give a clear view of what the obstruction detection algorithm does, all objects that were detected during this test were placed in Table C.1 found in the appendix. This table gives an insight into what factually happens during the obstruction detection process. A full description of the test results can be found in the appendix.

DISCUSSION

With the tests conducted the obstruction detection system for emergency exits performs well. The object detection range is within the required range. When it comes to the obstruction algorithm static objects are detected and dynamic objects are neglected. There were no false positives in this test.

There are some improvements that can be made. The Lifetime and Lastseen thresholds need further testing, also the margin in which a peak can fall in order to be considered a match should be better defined. In the test there was a situation where a new object was made where this was not necessary.

More important is the problem of the false positives. When an object is detected the Lifetime counter keeps increasing even when an object is not present. This was done to prevent a static object from being removed that has fallen into the shadow of a larger dynamic object. However this has a high possibility of causing a false positive, something that should be avoided. The better design choice is to only increment the Lifetime counter when the object is actually detected. Objects will take longer to be marked as obstruction and have the chance of being removed when still present. This is however preferred over the detection of false positives. When this is implemented more tests should be done in real life environments, testing for false positives and negatives.

5.4. OBSTRUCTION DETECTION: SHOPPING AISLE

In this section the performance of the obstruction detection in the shopping aisles is evaluated. Again multiple tests were performed.

SETUP

A test setup was built which represents a shopping aisle, a schematic view of this setup is shown in Figure 5.2. The shelves are wooden shelves, they are set at the maximum required width of 2.20 meters. The maximum aisle length with the materials we were given was 2 meters. More elaboration and pictures of the testing setups can be found in the appendix B. The sensor is again mounted at a height of 3.10 meters. In the ideal situation the tests would be in a real supermarket, this was not realisable.

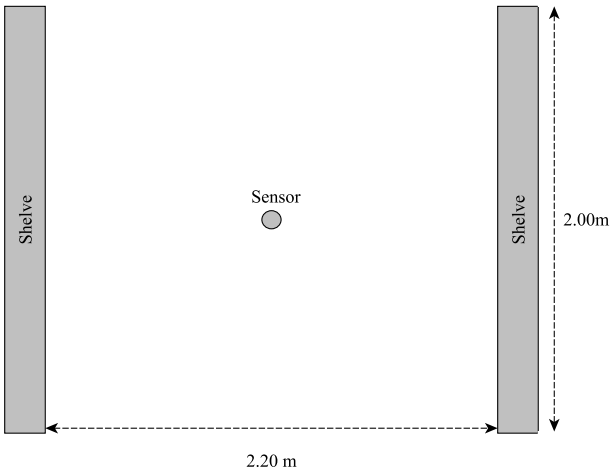


Figure 5.2: Top view of test setup used for obstruction detection in a shopping aisle.

The following tests were done:

1. Range tests are done with two objects. This is in order to further test criteria 3.1. The objects are: the cardboard box which is also used in previous measurements and a shopping kart with groceries. The kart and box are placed in different parts of the testing area in order to measure where the objects can be detected.

2. This test evaluates the performance of the obstruction detection system, as the performance of this system was already elaborated thoroughly with the obstruction detection of the emergency exits, this test will focus on the false positives and negatives and tries to determine by what these are caused. For testing the obstruction detection system four tests were done. The tests simulate a supermarket situation and correspond to the design criteria.

    2.1. No objects present for 40 iterations

    2.2. The cardboard box was placed directly under the sensor for 20 iterations

    2.3. No objects were present. However for 40 iterations a person with shopping cart randomly walks through the shopping aisle. This is to simulate a busy supermarket without while checking for false positives.

    2.4. The cardboard box was placed at the edge of the sensors detection reach. In addition a person with shopping cart walks through the shopping isle whilst measuring for 30 iterations

RESULTS

The results of the range measurements were as expected. The cardboard box could only be detected in the same range as shown in Figure 5.1. The cart however was detected throughout the whole testing region shown in Figure 5.2

The test results of the obstruction detection can be found in Table 5.1.

Table 5.1: False positives and negatives for four different measurements

Measurement	No object	Object, ideal	No object, people present	Object, people present
Number of iterations	40	20	40	30
False positives obstructions	0	0	1	1
False negatives obstructions	0	0	0	0

For the first two measurements no anomalies were observed; there were no false positives or negatives. For the third measurement, in which a busy supermarket is simulated one false positive occurred. In this test 39 peaks (objects) were detected throughout the 40 iterations. The false positive occurred in iteration 22, at this moment the Lifetime timer of that object was at 8. this means the object was not measured continuously but there were breaks in which it disappeared. The false obstruction was removed (deleted) in iteration 24.

In the last test, which is a repetition of test three only now with an object present, was done for 30 iterations. The object was detected and marked as an obstruction. A false positive occurred at iteration 14. In this iterations two peaks were detected, one at sample 133 and another at sample 135, the static object was at 133 and already marked as an obstruction, sample 135 falls within the margin and is thus noted as an extra obstruction. This peak is however not stored as an extra object.

DISCUSSION

The range of the sensor does not meet its requirement for all objects, the cardboard box could not be detected at the edges of the testing field. The obstruction algorithm again performs quit well. The first false positive would be solved if the Lifetime timer only increments when the object is detected, as was discussed in the discussion of the emergency exits. Extra testing is needed when this is implemented. The situation which caused the false positive can cause the lifetime counter to increment faster when multiple peaks are within the margin, this is a bug that needs to be fixed. When multiple peaks fall within the margin of an object they should be treated as one object.

5.5. GENERAL DISCUSSION OF THE RESULTS

A variety of tests have been done in this chapter and important observations were made. In this last section a summary of these observations will be given. Also tests that need to be performed in the future are discussed.

The system performs well on most of the design criteria stated above. There are however problems with the area in which objects can be detected, also there have been false positives caused by dynamic objects which should have been neglected. This poses a problem for the obstruction detection of shopping aisles. These false positives need to be avoided. As indicated in the discussion of the emergency exit section, the Lifetime counter should only be incremented when an object is detected. This will decrease the amount of false positives.

The threshold set for the minimum peak height works well, all object-peaks that were observed in the analog envelope measurement could also be observed in obstruction detection algorithm as objects.

There are a few important requirements that have not yet been tested. first of all power consumption. It was not possible to conduct a test for this in the given time frame. This will be the first thing to do as the system will need to function as a IoT device which operates at low power. Another important test or design implementation is the time intervals in which the sensors measures. The measurement intervals follow from the power consumption.

6

CONCLUSIONS AND RECOMMENDATIONS

The design of the thesis is concluded in this chapter. The overall results of this thesis will be discussed and compared to the goals and objectives stated in chapter 1.

6.1. CONCLUSIONS

The following project goal was set:

Develop a proof of concept for a stand alone obstruction detection system for supermarket environments

And the following 5 objectives were formulated:

1. Depict applicable sensing techniques and choose the best for the given problem.
2. Design a sensing system that can detect objects.
3. Further develop a sensing system that can detect obstructions near emergency exits.
4. Further develop a sensing system that can detect obstructions in shopping aisles.
5. Develop a functioning prototype.

In the time frame of eight weeks a stand alone obstruction detection was designed in addition a prototype was developed that was tested in chapter 5. In this chapter it was discussed that the obstruction detection of the emergency exits performed as stated in the design requirements. The shopping aisles obstruction detection had issues with false positives and objects not always being detected at the edges of the test setup. We can conclude that the project goal was partially met. In addition the "stand-alone" part of the design goal was not yet met. This part of the design goal was however considered with every design choice made throughout this project. The system has to be able to function as an IoT application: Low power, wireless and stand alone are features that have been taken into account when designing this system, however some steps need to be made to finalise this product. These steps are stated in the next section.

6.2. RECOMMENDATIONS

Based on the results obtained during this project, the following recommendations are given.

- As stated above, further testing needs to be done on the power consumption of the sensing system. This determines whether the sensor would be able to run on battery power, and what kind of battery would be needed. This should however be done in combination with the communication system as the communication would be partially responsible for the power consumed. The power consumption of the prototype would however still be an estimate of the power consumption of a finalised product. The Arduino board used for the prototype was designed for easy prototyping, not specifically low power consumption. Further development of the system would involve the selection of a more specialised microcontroller, as well as the design of a custom circuit board.

- In section 3.3 the problem of a mismatch between the calibration measurement and a new measurement was highlighted. Testing results showed that it sometimes occurs that a new measurement leads the calibration measurement by one sample. This could lead to problems when the precision of the object detection needs to be improved. It is recommended to further look into the origin of this problem. The problem might be a result of the fact that the first AD conversion after the ADC is turned on takes up more time than subsequent conversions. Or the additional ADC clock cycles between samples. This was touched upon in section 3.2. It is recommended to improve upon the sampling function. Other possible solutions might be to synchronise two measurements before the background subtraction, or to discard the first few measurements after the ADC is turned on.
- More research could be done into peak heights and widths of the returned echos. These parameters might improve the object identification algorithm, making the obstruction detection system more reliable and reducing the occurrence of false positives.
- A multisensor system might be used to cover more floor area for the detection of obstructions in shopping aisles. Research should be done into the avoidance of cross talk that could occur when multiple sensors are installed closely together.
- Ultrasonic sensors proved to be useful for the detection of obstructions near emergency exits. For shopping aisles the usefulness of ultrasonic sensors is more doubtful. Many sensor would be required in order to cover the full area of an shopping aisle. An option would be to place the sensor in places that are often obstructed. Another option would be to reconsider the use of CCTV images of existing existing security camera's. The detection of abandoned objects in CCTV images has extensively been researched for public safety reasons. This might not be an IoT solution but could prove to be more accurate and even simpler to implement.

BIBLIOGRAPHY

[1] *Proceedings of Conference on Intelligent Transportation Systems* (1997).

[2] Y. Tian, R. S. Feris, H. Liu, A. Hampapur, and M. T. Sun, *Robust detection of abandoned and removed objects in complex surveillance videos,* IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **41**, 565 (2011).

[3] *2016 International Computer Symposium (ICS)* (2016).

[4] *2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)* (2016).

[5] R. S. Kshetrimayum, *An introduction to uwb communication systems,* IEEE Potentials **28**, 9 (2009).

[6] *Proceedings of IEEE Sensors*, Vol. 2 (2002).

[7] A. Mroué, M. Heddebaut, F. Elbahhar, A. Rivenq, and J. M. Rouvaen, *Uwb radar and leaky waveguide for fall on track object identification,* in *2010 IEEE Radar Conference* (2010) pp. 573–577.

[8] J. Kämäräinen, A. Tenhunen, M. Pellinen, and T. Lehikoinen, *Experimental results on obstacle indication on impulse uwb radar imaging,* in *2014 IEEE Radar Conference* (2014) pp. 1075–1079.

[9] J. J. Garcia, C. Losada, F. Espinosa, J. Urena, A. Hernandez, M. Mazo, C. de Marziani, A. Jimenez, E. Bueno, and F. Alvarez, *Dedicated smart ir barrier for obstacle detection in railways,* in *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.* (2005) pp. 6 pp.–.

[10] *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery* (2012).

[11] X. Wu, M. Abrahantes, and M. Edgington, *Musse: A designed multi-ultrasonic-sensor system for echolocation on multiple robots,* in *2016 Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)* (2016) pp. 79–83.

[12] T. F. Wu, P. S. Tsai, N. T. Hu, and J. Y. Chen, *Use of ultrasonic sensors to enable wheeled mobile robots to avoid obstacles,* in *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing* (2014) pp. 958–961.

[13] A. R. Patkar and P. P. Tasgaonkar, *Object recognition using horizontal array of ultrasonic sensors,* in *2016 International Conference on Communication and Signal Processing (ICCSP)* (2016) pp. 0983–0986.

[14] R. P. Feynman, R. B. Leighton, and M. Sands, *The Feynman lectures on physics vol. I* (Addison-Wesley, 1963).

[15] W. M. Haynes, *CRC handbook of chemistry and physics*, 97th ed. (CRC press, 2017).

[16] C. Wykes, F. Nagi, and P. Webb, *Ultrasound imaging in air,* in *International Conference on Acoustic Sensing and Imaging, 1993.* (1993) pp. 77–81.

[17] A. Waite, *Sonar for Practising Engineers* (Wiley, 2002).

[18] Maxbotics, *XL-MaxSonar-AE Series,* https://www.maxbotix.com/documents/XL-MaxSonar-EZ_Datasheet.pdf (2015).

[19] J. G. Proakis and D. K. Manolakis, *Digital signal processing*, 4th ed. (Pearson, 2014).

[20] Atmel-2560/v, *8-bit Atmel Microcontroller,* http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf (2014).

[21] D. A. Mellis, *wiring_analog.c,* https://github.com/arduino/Arduino/blob/master/hardware/arduino/avr/cores/arduino/wiring_analog.c#L17 (2010).

[22] Arduino, *Arduino mega specifications,* https://www.arduino.cc/en/Main/ArduinoBoardMega2560.

A

ADC TIMING

Figure A.1 shows a timing diagram for an ADC conversion in single conversion mode. The conversion process starts at the first rising edge of the ADC clock after ADSC has been set. The sample and hold takes place during the first 1.5 ADC cycles. Next the quantization takes 11.5 ADC cycles. After the thirteenth ADC cycle the conversion is complete, an interrupt flag is set and the conversion result is written to the register ADCH and ADCL (MSB and LSB respectively). Next the ADSC can be set again to start a new conversion upon the next rising edge of the ADC clock. New conversions can thus start every 14 ADC cycles.

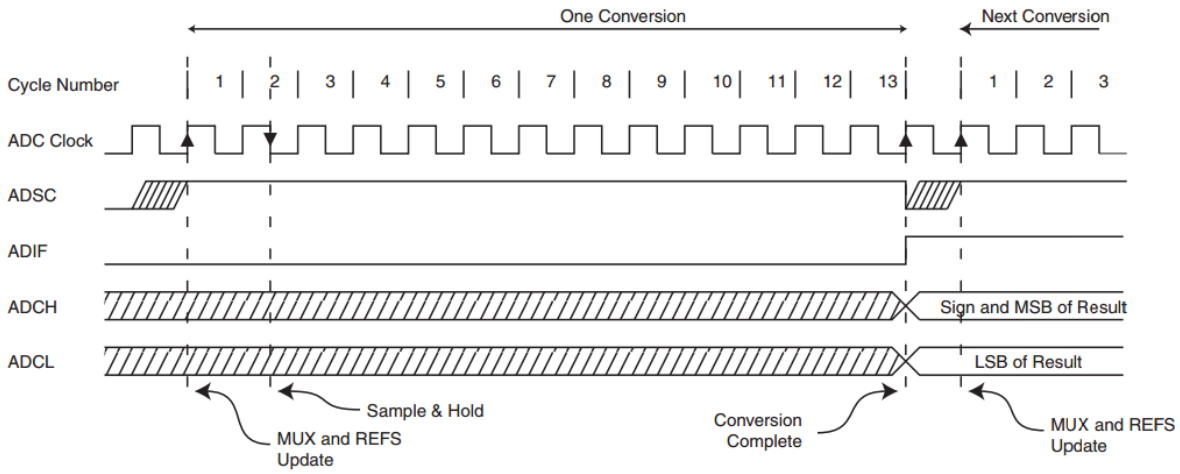


Figure A.1: Timing diagram of the ADC in single conversion mode [20].

B

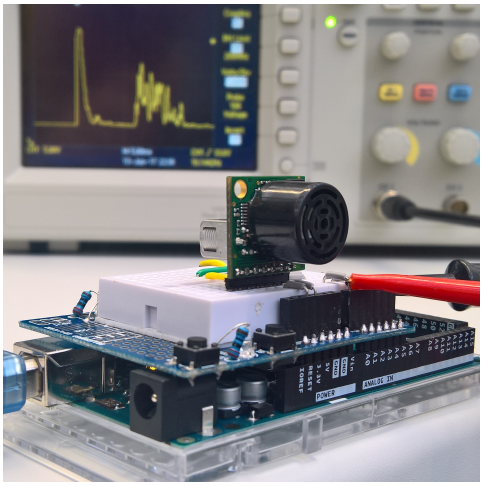
TESTING ENVIRONMENT

Figure B.1 shows some photos of the testing environment. Figure B.1a shows the prototype that was used during the design process and the testing. In the background the Analog Envelope can be seen on a oscilloscope.

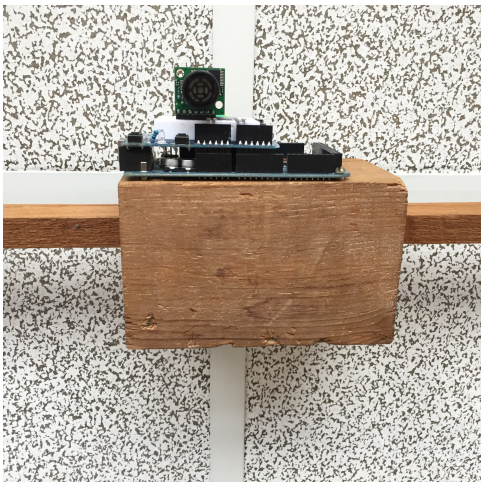
Figure B.1b shows the prototype being mounted on the ceiling for tests involving the simulated shopping aisle. Figure B.1c shows the simulated shopping aisle. The shelves are placed 2.2 meters apart, and have a height of 2 meters. In the upper part of the photo, the sensor can be spotted.

Figure B.1d shows the prototype mounted above an emergency exit for the test involving the emergency exit. The sensor is mounted at a height of 2.5 meters.

Figure B.1e shows the box that was used during the tests. It measures 40x25x30 cm (LxWxH). Figure B.1f shows the simulated shopping cart that was used during the tests.



(a) Sensor prototype



(b) Sensor mounted on ceiling

(c) Simulated shopping aisle

(d) Emergency exit

(e) Box

(f) Simulated shopping cart

Figure B.1: Photos of the testing environment

C

TEST RESULTS

In this appendix chapter the results of the emergency exit obstruction test are discussed. Table C.1 holds the results to this test. First the table layout and thresholds used are clarified. This is followed by the result elaboration. The setup and discussion of this test can be found in chapter 5.

The table is constructed as follows: the first and second row show the iteration number, this is followed by a row in which the sample location of the measured peaks are found. These are the peak locations inside the measured window. In the following rows the objects are placed. When a peak is detected it's location is stored inside a structure called object this object is placed inside a vector. The object numbers are simply locations in the vector. This means that if an object is removed the next object takes it place. In the table the objects are displayed with a status: D - Detected, M - Matched, R - Removed, O - Obstruction. Also the Lifetime and Lastseen is added, this is in order to give insight of when an object is removed or is marked as an obstruction. The following thresholds where used:

- Lifetime: 5 (Lifetime > 5 & detected = Obstruction)
- Lastseen: 3 (Lastseen > 3 = Remove)
- Margin:+/-2 samples (within 5 cm margin of Peak location = Matched)

The following observations are made. The office chair causes four peaks which are stored as four different objects. These are all removed when Lastseen exceeds it's threshold.

The peak created by the cardboard box is located at sample 96. In iteration 12 a person walks in range of the sensor, multiple peaks are detected which include a peak at sample location 99. This is probably the cardboard box. However it does not fall within the set margin of +/-2 samples, it is stored as new object. In the next two cycles a peak at 97 is detected, object 0 and object 4 now both have the status matched.

At iteration 16 a person walks into the range of the sensor to remove the cardboard box, due to this multiple peaks are detected. In iteration 17 the cardboard box is removed from the range of the sensor. The status now changes from ObstructionMatch to Obstruction and the object is removed from the memory three iterations later.

After iteration 17 The objects with peak 26 and 99 are moved up in the table because two objects are removed. In the real situation this happens one iteration earlier, directly when the other objects are removed.

Table C.1: Emergency exit obstruction test results

Iteration																				
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Detected peaks	-	65,82, 95,103	66,82, 95,103	-	-	-	-	-	96	96	96	15,26, 79,99	97	97	96	96,21, 27	-	-	-	-
Object 0		65							96											
Status		D	M				R		D	M	M		M	M&O	M&O	M&O	O	O	O	R
Lifetime		0	1	2	3	4	5		1	2	3	4	5	6	7	8	9	10	11	12
Last seen		0	0	1	2	3	4		0	0	0	1	0	0	0	0	1	2	3	4
Object 1		82										15					26			
Status		D	M				R					D				R				R
Lifetime		0	1	2	3	4	5					0	1	2	3	4	5	6	7	8
Last seen		0	0	1	2	3	4					0	1	2	3	0	1	2	3	4
Object 2		95										26					99			
Status		D	M				R					D				M		R		
Lifetime		0	1	2	3	4	5					0	1	2	3	4	5	6		
Last seen		0	0	1	2	3	4					0	1	2	3	0	3	4		
Object 3		103										79								
Status		D	M				R					D				R				
Lifetime		0	1	2	3	4	5					0	1	2	3	4				
Last seen		0	0	1	2	3	4					0	1	2	3	4				
Object 4												99								
Status												D	M	M						
Lifetime												0	1	2	3	4				
Last seen												0	0	0	1	2				

D

OBSTRUCTION DETECTION CODE

```
//This is code for an obstruction detection device, the code is written for an adruino mega 2560.
//the used sensor is a MaxBotix MB1300 ultrasonic range finder.
//Authors: Bart de Vos, Tjerk den Boer - 2017

#include "Functions.h" //includes vector
#include "Ranging.h"
#include "Detection.h" //includes Object.h

//250 samples ~4.5m range
int calibrate[250]; //Array containing calibrated measurements
int measured[250]; //Array which will contain the measured values
boolean calibration_bit = false; //bit used to recalibrate the system. In the near future this bit is declared one when recalibration is needed.
vector <Object> Object_vector; //dynamic array of peak objects
vector<int> window;            //dynamic window created between the starting pulse and the floor

void setup() {
  Serial.begin(57600); // set baudrate to 57600

  pinMode(RX, OUTPUT);
  digitalWrite(RX, HIGH); //initiates a ranging to start up the ultrasonic sensor
  delayMicroseconds(20);
  digitalWrite(RX, LOW);
  delay(150);
}

void loop() {
  Range_and_Sample(calibrate);

  int iteration = 1;

  Window(calibrate, window); //calls the window function, creates a window from the calibration measurement

  while (calibration_bit == false) {
    Range_and_Sample(measured);

    //re-sizes the measurements to the window size
    for (int i = 0; i < window.size(); i++) {
      measured[i] = measured[window[i]];
    }
    //subtraction
    for (int i = 0; i < window.size(); i++) {
      measured[i] = measured[i] - calibrate[i];
    }
    //removes negative values
    for (int i = 0; i < window.size(); i++) {
      if (measured[i] < 0) {
        measured[i] = 0;
      }
    }
    //remove all values larger than the window size
    for (int i = window.size(); i < 250; i++) {
      measured[i] = 0;
    }

```

```
  vector <int> Peaks_l; //dynamic array of peaks locations inside the window
  Find_peaks(measured, 90, Peaks_l);
  if (Peaks_l.size() > 0) {
    for (int i = 0; i < Peaks_l.size(); i++) {
      Serial.print("peaks ");
      Serial.println((float)(Peaks_l[i])); //for serial feedback, prints peak location within the window
    }
  }
  //starts the obstruction detection
  Obstruction_detection (iteration, Object_vector, Peaks_l);
}
}
```

```
#include <StandardCplusplus.h>
#include <vector>

//functions.h holds the functions window, findpeaks and maximum, which are all used to determine objects from the measurements

using namespace std;

//This function calculates the maximum of a given array.
//Returns the location (int) of the maximum.
int Maximum(int x[], int sizeX) {
  int max_location;
  int max_value = 0;
  for (int i = 0; i < sizeX; i++) {
    if (max_value < x[i]) {
      max_value = x[i];
      max_location = i;
    }
  }
  return max_location;
}
//This function finds all peaks in a given array.
//The locations of these peaks are stored in a vector
void Find_peaks(int x[], int threshhold_v, vector<int> & peaks) {
  int j = 0;
  for (int i = 1; i < 249; i++) {
    if (x[i] > x[i - 1] & x[i] > x[i + 1] & x[i] > threshhold_v) {
      peaks.push_back(i);
      j++;
    }
  }
}
//this function creates the operating window, a collection of samplelocations between the send pulse and the floor
//in this region peaks will be found which can be objects
//returns nothing but fills the vector 'window' with the correct values
int Window(int calibration[], vector<int> &window) {

  int floor_window[150];
  int window_min = 0;
  int window_max = 0;
  int scaler;

  //find the outgoing pulse, sets as maximum, adds until the min value is reached: the end of the pulse
  window_min = Maximum(calibration, 250); //location of maximum in measured data = outgoing pulse
  scaler = (calibration[window_min] - calibration[0]) * 0.1 + calibration[0];
  while (calibration[window_min] > scaler) { //10% of the maximum
    window_min++;
  }
  //create a window in which to look for the floor. between ~2 and ~5m <----- dit klopt niet helemaal 50 samples ervanaf trekken
  for (int i = 100; i < 250; i++) {
    floor_window[i - 100] = calibration[i];
  }
  //Floor is the maximum piek floor window.
  window_max = Maximum(floor_window, sizeof(floor_window) / sizeof(int));
  while (floor_window[window_max] > scaler) {
    window_max--;
```

```
  }
  window_max = window_max + 100; //correct for offset caused by the window

  //creates the window: A vector with in it's content the sampled values that are inside the window.
  for (int i = window_min; i < window_max; i++) {
    window.push_back(i);
  }
}
```

```
//Set pins:
const int AEin = 0; //Analog envelope: input: pin 0
const int RX = 7;   //Digital pin used to trigger a ranging pulse if high for a minimum of 20us , if left low no ranging takes place,

//Ranging function that activates the MB1300, conducts a ranging converts the sampled data to volts and stores this in an array.


void Range_and_Sample(int measured[])
{

  pinMode(RX, OUTPUT);
  digitalWrite(RX, HIGH); //initiates a ranging
  delayMicroseconds(20);
  digitalWrite(RX, LOW);
  delay(20);

  //sampling of the analog envelop data; done at 8.93 kHz
  for (int i = 0; i < 250; i++) {
    measured[i] = analogRead(0);
  }
}
```

```
#include "Object.h"
//detection.h contains the obstruction detection algorithm.


void Obstruction_detection (int &iteration, vector <Object> &Object_vector, vector <int> &Peaks) {
  Serial.print("Iteration ");     // for serial output feedback
  Serial.println(iteration);
  iteration++;

  boolean match = false;   // extra variable to keep track of whether a peak has found a match. If not, a new object must be created.

  int mode = 0; // 0 -> emergency exit, 1 -> Shopping aisle
  int X;
  int Y;


  if(mode==0){ // thresholds and measurement intervals for emergency exits can be set here:
    X = 5;
    Y = 3;
    delay(30000);
  }
  if(mode==1){ //thresholds and measurement intervals for shopping aisle can be set here:
    X = 10;
    Y = 3;
    delay(30000);
  }


  // Start of the algorithm
  for (int i = 0; i < Peaks.size(); i++) {  // cycle measured peaks
    match = false;                          // reset match
    for (int j = 0; j < Object_vector.size(); j++) {   // checking all objects for a match
      if (Peaks[i] < (Object_vector[j].location + 3) & Peaks[i] > (Object_vector[j].location - 3)) { // checks if Peak location matches object location with a margin of 2 samples
        Serial.println("match");          // for serial output feedback
        match = true;                     // set match variable
        Object_vector[j].last_seen = 0;   // if match: reset last_seen
```

```
        Object_vector[j].lifetime++;      // if match: increment lifetime
        Object_vector[j].check_flag = true; // if match: object marked as checked
        if (Object_vector[j].lifetime > X) {  // Check lifetime, if(Lt > X) -> Obstruction detected, action must be taken.
          Object_vector[j].Obstruction = true; //set obstruction as true
          Serial.println("OBSTRUCTION");  // for serial output feedback
          //Send Obstruction, to user.
        }
      }
    }
    if (match == false) {     // If match variable remained false, a new object should be created
      Object A;
      A.location = Peaks[i];
      A.check_flag = 1;
      Object_vector.push_back(A);
      }
  }
  for (int i = 0; i < Object_vector.size(); i++) { // cycle objects
    if (Object_vector[i].check_flag == false) {
      Object_vector[i].last_seen++;             // If object not detected, still increment last seen
      Object_vector[i].lifetime++;              // increment lifetime
      if (Object_vector[i].last_seen > Y) {     // Check last_seen, if (LS > Y) -> Delete object.
        Serial.print("DELETE OBJECT "); // for serial output feedback
        Serial.println(Object_vector[i].location);
        Serial.println(i);
        Object_vector.erase (Object_vector.begin() + i);
        i--; //correct i for when object is erased
      }
    }
    if (Object_vector[i].check_flag == true) {
      Object_vector[i].check_flag = false;      // reset check flags
    }
  }

  for (int i = 0; i < Object_vector.size(); i++) {     // for serial output feedback
    Serial.print("Object ");
    Serial.println(i);
    Serial.print("Location: ");
    Serial.println(Object_vector[i].location);
    Serial.print("Lifetime: ");
    Serial.println(Object_vector[i].lifetime);
    Serial.print("Last seen: ");
    Serial.println(Object_vector[i].last_seen);
    Serial.print("Check flag: ");
    Serial.println(Object_vector[i].check_flag);
    Serial.println();
  }
}
```

```
//This header contains the struct Object. This struct is used for storing objects when detecting obstructions

struct Object {
  public:
    int location = 0;  //location of the sample at which the peak is found
    int lifetime = 0;  //lifetime of the object. Is used to determine if an object lives long enough to be an obstruction.
    int last_seen = 0; // last seen, is incremented when object is not detected. Is used to determine if an object should be deleted
    boolean check_flag = false;
    boolean Obstruction = false;
};
```