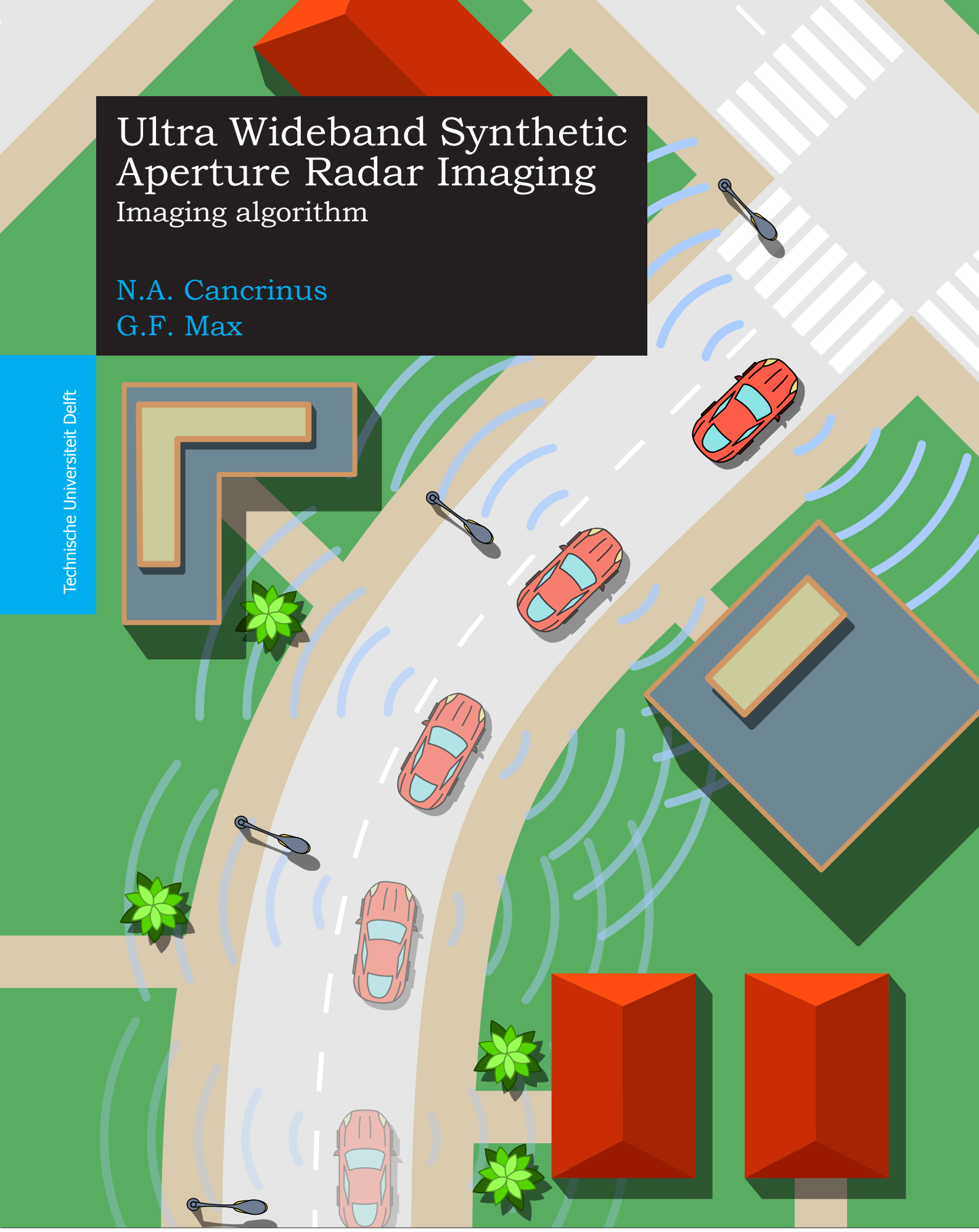
Ultra Wideband Synthetic Aperture Radar Imaging

Imaging algorithm

N.A. Cancrinus
G.F. Max

Technische Universiteit Delft



TU Delft Delft University of Technology

Challenge the future

Ultra Wideband Synthetic Aperture Radar Imaging

Imaging algorithm

by

N.A. Cancrinus
G.F. Max

in partial fulfillment of the requirements for the degree of

Bachelor of Science
in Electrical Engineering

at the Delft University of Technology,
to be defended publicly on Wednesday July 5, 2017 at 11:00 AM.

Student number:	N. Cancrinus	4377109
	G.F. Max	4321553
Project duration:	April 24, 2017 – July 7, 2017	
Supervisor:	Ir. P. Aubry	
Thesis committee:	Prof. dr. ir. A. van der Veen,	TU Delft
	Ir. P. Aubry,	TU Delft
	Dr. D. Cavallo,	TU Delft

TU Delft Delft University of Technology

Preface

The goal of this project was to implement the technique of Synthetic Aperture Radar to achieve higher localization accuracy of cars and to be able to obtain semi real-time information on the environment around roads. To achieve this, different imaging algorithms have been implemented, and a single method has been extended to directly accommodate the road-mapping application. Different measurements were done to test the algorithms and the results were promising. It is hoped that the use of this technology will become wide-spread and that it can help with creating a safer and more efficient world of traffic, where increased autonomy is facilitated.

N.A. Cancrinus
G.F. Max
Delft, June 2017

Contents

1

Introduction

In this chapter some of the most important concepts that are used in this project will be introduced. After this the goal of the entire project will be defined. The division of the project in subgroups will be explained and the goal of the subgroup of this thesis will be specified. Finally, the structure of the rest of this thesis will be described.

1.1. Basics of radar

In RADAR (RAdio Detection And Ranging) a certain electrical signal is offered to an antenna, which sends a corresponding electromagnetic signal. This electromagnetic signal interacts with the environment visible for the antenna, which will create some reflections. A part of these reflections are received by an antenna. This antenna can be the same one as the transmitting antenna, but can also be a different one. The result is a certain electric signal that can be analysed. This concept is shown as a diagram in figure 1.1.

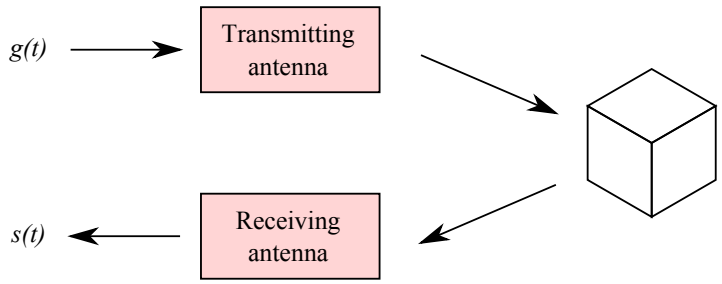


Figure 1.1: A diagram showing the basics of radar. $g(t)$ is the transmitted signal. $s(t)$ is the received signal.

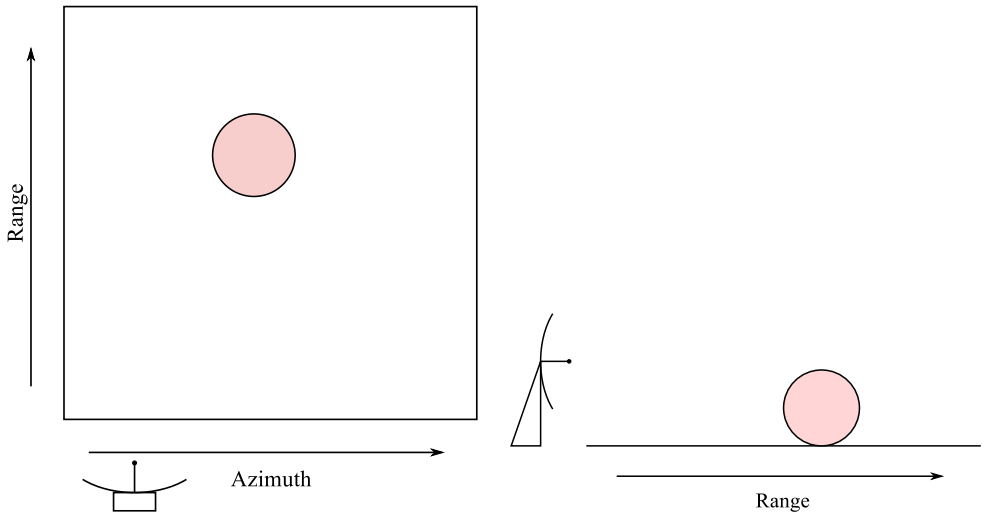
The time-points where the received signal contains reflections, give information about the distance between objects in the environment and the antenna. The directivity of the antenna can give some information about the direction where objects are located. The more directive an antenna is, the smaller the area in which a detected object can be located and vice versa.

1.2. Basics of Synthetic Aperture Radar

One of the disadvantages of using only a single radar is that there is little information about the direction. This can be improved by using a bigger radar aperture, i.e. with larger area where signals can be received. By doing so the resolution of the direction can be increased, however, building antennas larger than several meters is in most cases unpractical. It is possible to overcome this issue by using an array of smaller antennas. This way not only the construction costs can be reduced but also this structure has interesting properties. The received (or transmitted) signals can be directed to a specific direction which is called beam-forming. Thus, the data collected by each individual radar can be

processed to create a better image with much less SNR in comparison with using a single stationary radar.

The costs can be further reduced by using a single but moving radar with which the (virtual) array of smaller antennas can be reconstructed. The aperture created this way is synthetic and is much larger than that of a single radar, hence the name synthetic aperture radar or SAR. Figure 1.2 illustrates the situation. The fundamental idea behind SAR imaging is that the data acquisition happens at different places. Furthermore, broadside imaging is used meaning that the direction of movement (i.e. x direction) and the direction of the antenna (i.e. y direction) are perpendicular. When the antenna movement stays in one direction the technique is called stripmap SAR, when the antenna moves along a circle it is circular SAR. The x axis is often called in the literature as azimuth, cross-range or slow-time direction, while the y axis as (down-)range or fast-time direction.



(a) Top view. (b) Side view.

Figure 1.2: Basic setup for SAR. The antenna moves along the azimuth direction and takes measurements along the range direction.

The area being mapped by SAR is called the scene. If an object is not located in the plane of the scene while a reflection of that object is received, it will be still mapped to a location in the plane as a virtual target. In this case the distance between the virtual target and the measurement locations stay the same. Hence, if seen from above, the virtual target will be seen further away (see figure 1.3).

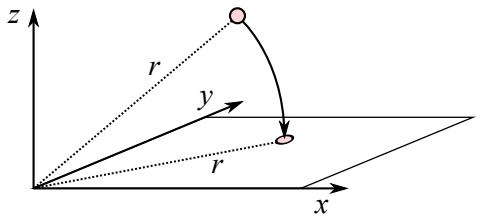
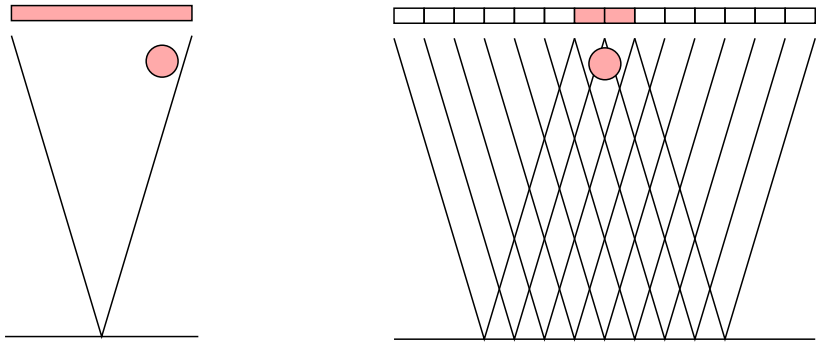


Figure 1.3: The effect of having a target out of the plane of view. The target maps to a virtual position that is further away than if the target was "dropped" onto the xy-plane.

The way in which the azimuth-resolution can be increased when multiple measurements are used can be described as follows. When a certain object gives a certain reflection at a certain measurement location, it can only be said that the object is within the area visible by the antenna at this measurement location. When the object is also within the area visible by the antenna at the next measurement location, it can be said that the object must be in a location visible from both measurement locations. If more measurement locations are used, the area in which the object can be becomes smaller and smaller. In essence the differences between the measurements are used to decrease the actual area in which the object can be. This decrease in area corresponds to an increase in azimuth-resolution. This

concept is also shown in figure 1.4. Figure 1.4a shows the case when only one measurement location is used. In this case area in which the object can be is rather large. Figure 1.4b shows the situation when multiple measurement locations are used. In this case the azimuth range at which the object can be located is a lot smaller.



(a) Resolution with one measurement. (b) Resolution with multiple measurements.

Figure 1.4: The effect of taking measurements from multiple distances as seen from the top. The target (red circle) lies within the range of the antenna. The possible locations for the target are shown above. This area is reduced and thus the azimuth-resolution is increased.

The literature about synthetic aperture radar (SAR) is quite developed, there are many well-written books and articles for novices [1–4]. The idea of SAR dates back in the 50's. With this technique, airborne – and later satellite – images of the surface could be made from large distances, which is also today the most widely used application of SAR. Nowadays thanks to the increased computational power, it is applied in different areas, such as geoscience and security applications.

To convert the data of different measurements into an image, a certain algorithm is needed. The algorithms that were used will be described later in chapter 3. Traditionally, the (digital) computational power was rather limited, therefore imaging algorithms were implemented by the means of analog systems [5, 6]. To process data, specially formed optical lenses were placed at well-defined places. As technology advanced, digital systems become increasingly important in SAR imaging [7]. At the moment the lack of computational power is significantly smaller and there is much more information on digital signal processing for SAR [8, 9]. There are many possibilities to construct an image using spectral estimation techniques [10], with a wide range of computational complexity and image quality.

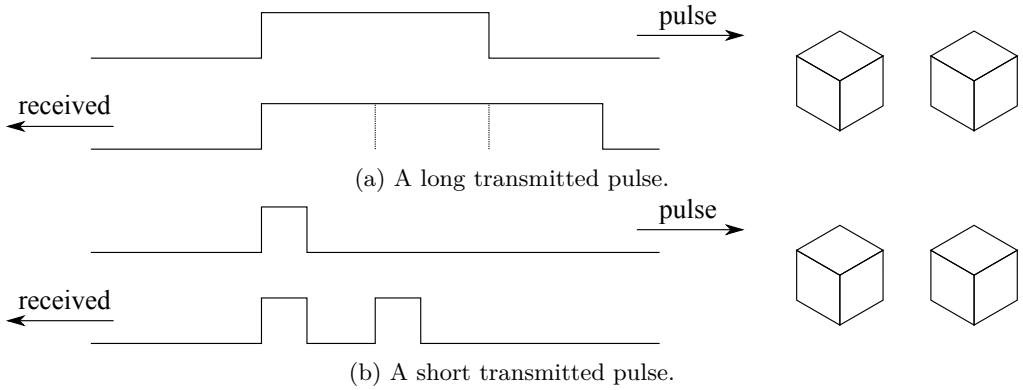
Another more frequently used algorithm is compressed sensing [11]. When handling large amount of data the computational time increases and may become unfeasible to create an image within realistic time. This algorithm helps significantly reducing the size of the data with minimal loss of details and consequently the image reconstruction becomes faster than ordinary methods [12, 13].

SAR can also be used to create 3D images [14–16]. The strength of such image is that the depth of the scene also becomes visible, which makes it practical for mapping inaccessible areas and (military) reconnaissance. Furthermore, the signal can penetrate through the surface, when applying applicable frequencies, thus make it a potential candidate for detecting mines [17] or for collecting scientific data.

1.3. Basics of Ultra Wideband radar

Whereas the azimuth resolution is determined by the synthetic aperture and the spacing between measurements, the range resolution is determined by the time-duration of the transmitted pulse. When a pulse is shorter, two received reflections can lie closer together without having overlap between them. In the ideal case, a delta-pulse is transmitted, because it gives infinitely high range resolution. The difference between short and long pulses is illustrated in figure 1.5. Figure 1.5a shows the situation when a relatively long pulse is used. In this case the two objects cannot be distinguished because the reflections overlap. Figure 1.5b shows the situation when a relatively short pulse is used. In this case the objects can be distinguished because the reflections do not overlap. So, in general, the resolution increases when a shorter transmitted pulse is used.

A short pulse in time-domain translates to a wide band in frequency domain. For this reason, a system that works with a relatively short pulse-duration is called an ultra wideband (UWB) system.



(a) A long transmitted pulse.

(b) A short transmitted pulse.

Figure 1.5: A figure that shows that the range-resolution increases when a shorter pulse is used. In (a) the two objects can't be distinguished because the reflections overlap. In (b) the objects can be distinguished because the reflections do not overlap.

When a transmitted pulse is very short, it is difficult to put a reasonable amount of power into it. Therefore in practice often a longer transmitted signal is used, that can be easily converted to a short pulse. In [18] the effect of two most commonly used signals (chirp and Gaussian) are evaluated in terms of the achieved resolution.

One of the advantages of using (microwave) UWB is its wall penetrating property. Near-gigahertz frequencies have a good compromise between the resolution of the image and the ability to propagate through solid (non-conducting) obstacles such as walls or the ground. By making use of these properties, the interior of a house can be mapped from outside [19], which has undeniable potential in military applications. Care should be taken when mapping closed spaces as multipath reflections can lead to virtual image(s) and/or inaccurate position of the target. These unwanted effects can be compensated for [20, 21].

Another potential application for UWB SAR is in the medical field. Since below-infrared (e.g. gigahertz to terahertz) frequencies are non-ionizing they can be harmlessly used for cancer tissue detection [22]. Furthermore, the resolution achieved by these microwave signals is sufficiently large to identify unhealthy tissues thus providing a safe substitute for X-ray.

1.4. The goal of the project

Current technologies (e.g. GPS) do not provide sufficient accuracy for localization of vehicles in urban areas or in closed spaces. In most cases, e.g. on highways and in rural areas, imperfect localization do not lead to problems. However, it may become inconvenient for the driver when the wrong path is taken in an unknown environment because of incorrect localization. Nuisance of turning the wrong way may grow to dangerous or even life-threatening situations in case the vehicle is engaged in autopilot, let alone, if it is fully autonomous. To tackle this problem a radar can be mounted on the vehicle which can take measurements on-the-go. The data that is obtained by SAR measurements can be used to map its surroundings [23] improving localization accuracy.

The goal of this project is to implement the technique of SAR to achieve higher localization accuracy of vehicles and to be able to obtain semi real-time information on the environment around roads. UWB is suitable for this since centimeter (millimeter) range resolution can be achieved with relatively low cost. For this application higher spatial resolution is unpractical. The SAR technology is particularly suitable since the vehicle moves. Besides that, majority of the new cars – both self-driving and traditional – come with some kind of built-in radar system, mostly for collision detection and/or avoidance. The fact that the key components are already present in vehicles, further facilitates the implementation of the system.

Features of the environment can be extracted from the created images and then used for accurate localization of the car [24] eventually together with other systems. When more cars are equipped with such system the map "discovered" by a single car can be shared between other cars. Later, based on these maps, a collective database can be built for localization purposes. This becomes especially important as self-driving cars with full autonomy must have vision about where they located at. Furthermore,

the map can be updated in case of unexpected obstacles on the road (such as trees falling on the road, construction works, or even accidents) and other vehicles as well as emergency services can be alerted real-time. It stays outside of the scope the thesis, however, [25] shows that (satellite) SAR images can be used for mapping purposes.

1.5. Problem definition

There are several must-haves that need to be dealt with in order to achieve the previously mentioned goal. It is necessary for this technology to be able to take measurements from different positions, and the coordinates, where these measurements were taken, are of great importance. The radar must be able to transmit and receive wide-band signals. For that an antenna and data acquisition system with applicable properties should be utilized. As last, the data has to be processed to create an image.

The project is divided into smaller groups where each group tackles different problems of the realization. Figure 1.6 shows the different groups. The movement of the radar system is handled by the hardware group. Data acquisition is performed by the antenna group. The task of the imaging group, our group, is to convert the obtained data to an image that depicts the scene.

Figure 1.6: The overview of the project.

The theses of the hardware group and data acquisition group can be found in [26] and [27] respectively. The inputs of the subsystem explained in this thesis are the measurements taken from different locations and the coordinates of these measurements. The output of the subsystem is an image that represents the scene. Therefore, the task of the imaging group can be summarized as follows:

Given SAR-measurements of a scene and the location from which they were taken, produce an image that corresponds to the scene.

1.6. Structure of thesis

The thesis is structured as follows. In chapter 2 the program of requirements for road mapping will be described. In chapters 3 and 4 the designed imaging algorithms are explained. Chapter 3 contains 3 general algorithms that can be used for creating an image as well as evaluation of these algorithms with simulation and measurements. Chapter 4 builds a connection between the chosen practical application and the algorithms, illustrates how they can be applied in the field, and finally, shows and discusses the results of the measurements. Chapter 5 gives a conclusion together with future work.

2

Program of requirements

2.1. Requirements for the entire system

The goal of the project is to use SAR-techniques to improve the localisation and navigation of cars. The idea behind this is that antennas mounted on the cars can be used to do multiple subsequent measurements. The data of these measurements can then be used to generate an image that represents the environment. This image can then be compared with a database of known images to determine up to higher resolution where a car is. The newly scanned images can also be used to update the database and thus obtain semi real-time information about the road system. These last two points fall outside the scope of this project. The focus of this project is the generation of the SAR-images.

The requirements for the entire system can be divided in several parts. The functional requirements are the specifications that describe what the system must do and what is fundamentally important when doing this. These are as follows:

[1.1] The system should be able to do multiple subsequent measurements.

[1.2] The data of these measurements should be converted into an image that represents the scanned environment.

[1.3] The system should be able to detect reflections within distances that are common in road environments. This means that the transmitted power and the measurement repetition frequency should be adequate.

[1.4] The system should be scalable to larger datasets.

[1.5] The resolution of the system should be high enough to be able to distinguish objects in a normal road environment. These objects could be buildings, fences, lamp posts or landmarks.

The implementation requirements are prerequisites that make the system easier to use in real applications. They are as follows:

[2.1] The needed hardware should be easy to implement in an actual vehicle.

[2.2] The image formation should be done semi real-time. This means that the time required to make the image should be smaller or in the same order as the time it takes to do the measurements.

The representation requirements are boundary conditions that are important to facilitate functional use of the system. They are as follows:

[3.1] The produced image should be easily understandable by humans and should be usable for comparison with a larger dataset.

The safety requirements indicate how safe the system should be. They are as follows:

[4.1] The radiated power should be of such a level that there are no health dangers for people in the proximity of the antennas.

2.2. Requirements for the imaging group

The goal of this subsystem of the project is to implement an algorithm that creates an image from the dataset obtained by the antenna. This algorithm must satisfy certain requirements. These requirements can be divided in several parts.

The functional requirements are the requirements that describe what the subsystem must do and what is fundamentally important when doing this. These are as follows:

[A.1] The subsystem must make an image where the amplitude of a location in the image represents the probability that there is a reflecting surface in the corresponding physical location. The inputs used to do this are the obtained measurement-data and the locations where measurements are taken.

[A.2] The resolution in the range-direction should be at least 2 cm.

[A.3] The resolution in the azimuth-direction should be at least 2 cm.

The implementation requirements are requirements that make the subsystem easier to use in real applications. They are as follows:

[B.1] The algorithm should be usable when the computational power is limited. This makes it possible to implement the algorithm on a small device or embedded system.

[B.2] The system should be easily scalable to larger planes, so the algorithm should be implemented in an efficient way. Pure brute force algorithms should be avoided.

[B.3] The algorithm will be implemented in MATLAB and should work without external applications.

The representation requirements are boundary conditions that are important to facilitate functional use of the subsystem. They are as follows:

[C.1] The result should be presented using a colour map.

[C.2] The axes should match the actual physical dimensions of the plane that is imaged, so that no extra operation is required.

3

General imaging algorithms

In this chapter three imaging algorithms will be implemented. First, the elementary variables used throughout this chapter are introduced. Then the sum-and-delay algorithm will be explained. After that the backprojection algorithm will be elaborated. Finally, the optical algorithm will be described. These algorithms are assessed by the means of simulation and experimental results will be shown.

3.1. Definition of variables

The setup for the radar-measurements is illustrated in figure 3.1. The x coordinates of the measurements taken by the moving antenna are stored in vector $\mathbf{m}$. At each measurement point m_i the received data is represented as $\mathbf{s}_i$, a vector with N samples. The elements of $\mathbf{s}_i$ are the time-sampled, quantized amplitude values of the received electromagnetic signal. The first and last samples of $\mathbf{s}_i$ correspond to real distances y_{begin} and y_{end}, respectively. It is worth noting that y_{begin} can have negative value if, for example, the received data is zero-padded at its beginning. The region of interest, which is a subset of the measurements, is located between y_{min} and y_{max}.

Having an antenna with certain directivity, these measurements correspond to the reflection of the objects in the scene under the "visible" beam angle, θ. In a real antenna, reflections outside of the beam angle are also received, however, these are significantly smaller than the reflections inside of the beam angle. Furthermore, reflections near the sides of the beam angle have less power in comparison with reflections in the middle of the beam angle. Throughout the thesis, it is assumed that the antenna pattern characteristics resembles a square wave. This means that every object can be detected within the beam angle with equal reflected power at a given distance, and no object can be found outside of the beam angle.

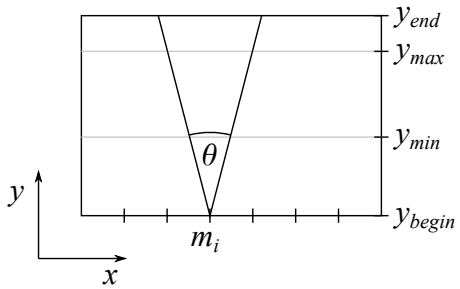


Figure 3.1: Overview of the measurement. x, y are Cartesian coordinates, m_i is the position from which the ith measurement is done, y_{begin} and y_{end} are the (range) distances corresponding to the first and last samples, y_{min} and y_{max} define the range of interest, θ is the beam angle of the antenna.

As mentioned earlier, the task of this group is to produce an image based on measurements taken at different locations. The collection of these measurements are made available in a matrix form:

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_M \end{bmatrix} \tag{3.1}$$

As $\mathbf{S}$ contains time-sampled information about the amplitude of the received signal with sample frequency F_s, it will become helpful to define a function that converts real-life distances to samples. The electromagnetic (EM) wave, that is reflected from an object at distance d, travels $2d$ from and to the antenna. For the given application it can be assumed that the EM wave travels in free-space, thus the corresponding time-sample can be then given as:

$$f(d) = \frac{2dF_s}{c} \tag{3.2}$$

It is worth noting that when the EM wave has to travel through material(s) with different electromagnetic properties (i.e. $\epsilon_r, \mu_r \neq 1$) this equation will give improper results regarding the sample. Furthermore, it is possible that $f(d)$ produces a non-integer number. Since it is not possible to select a non-integer element of $\mathbf{s}_i$, (linear) interpolation of the two closest samples will be taken as the result.

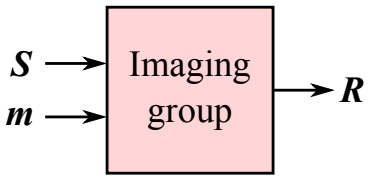


Figure 3.2: The inputs and output of this subgroup of the project.

The image to be created is a matrix $\mathbf{R}$ whose elements correspond to the reflected power in that point. Figure 3.2 summarizes the inputs and output of the imaging group.

3.2. Pulse Compression

As was explained before in section 1.3, when a delta-pulse is transmitted it is hard to give this pulse an adequate amount of power. For this reason not a delta-pulse is used, but a certain other signal shape, also see the thesis of the data acquisition group [27]. However, this signal shape could have a relatively low amplitude, even when the power is high. Therefore, the absolute signal amplitude might be lower than the noise, which could cause that the received reflection is poorly visible. To improve this, information about the shape of the transmitted signal can be used.

This is achieved by taking the correlation of the transmitted pulse and the received signal. At the time-points where a reflection was received there will be a match between the pulse and the received signal and the correlation will give a relatively high value. There will not be a good match between the noise and the transmitted pulse and therefore the relative noise value will be suppressed. The pulse compressed data $\mathbf{p}_i$ of measurement $\mathbf{s}_i$ can be written as:

$$\mathbf{p}_i[n] = (\mathbf{s}_i * \mathbf{g}')[n] \tag{3.3}$$

where $\mathbf{g}'[n] = \mathbf{g}[-n]$ is the mirrored signal of the transmitted, time-sampled pulse $\mathbf{g}[n]$. This equation holds relevant information only if the indices of the first (non-zero) element of $\mathbf{s}_i$ and $\mathbf{g}$ match. In this case, the elements of $\mathbf{p}_i$ preserve the corresponding real life distances, i.e. the distance corresponding to $\mathbf{s}_i[k]$ will be the same distance as for $\mathbf{p}_i[k]$. Since the convolution lengthens $\mathbf{p}_i$, it is truncated such that it has the same length as $\mathbf{s}_i$ while maintaining the consistency between indices and distances. Similarly to $\mathbf{S}$, the data after pulse compression is called $\mathbf{P}$ where each column is a vector of pulse-compressed data:

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \dots & \mathbf{p}_M \end{bmatrix} \tag{3.4}$$

As a result of the pulse compression $\mathbf{P}$ has the same dimensions as $\mathbf{S}$, as well as the distances corresponding to the samples in $\mathbf{S}$ are the same as in $\mathbf{P}$.

3.3. Sum-and-Delay algorithm

One algorithm to convert pulse compressed data into an image is the sum-and-delay algorithm. This algorithm works as follows. A certain pixel in the scene (x_j, y_k) has a certain distance to a measurement

point $\mathbf{m}_i = (x_{m_i}, 0)$. It is worth noting that the spacing and location of (x_j, y_k) can be arbitrary as long as they are within the visibility of the measurements.

To determine whether there is a reflecting surface at pixel (x_j, y_k), for every $\mathbf{p}_i$ the time-sample can be selected that corresponds to the distance between (x_j, y_k) and $\mathbf{m}_i$. If there is a reflecting surface at (x_j, y_k), for every measurement point $\mathbf{m}_i$ a sample will be taken from the corresponding measurement $\mathbf{p}_i$ that has a relatively high value. For this reason it can be said that if all the samples of different $\mathbf{p}_i$ corresponding to (x_j, y_k) are added and the result has a high value, the probability is high that there is a reflecting surface at (x_j, y_k). This procedure can be repeated for every j and k. When doing this, it is important to check whether a pixel is within the directivity of the antenna. Every pixel will give a result $\mathbf{R}[j, k]$ that represents the probability that there is a reflecting surface at (x_j, y_k). In this way the desired image is formed.

This procedure can be described mathematically as follows. The distance between a certain measurement point $\mathbf{m}_i = (x_{m_i}, 0)$ and a pixel in the scene (x_j, y_k) is given by:

$$\Delta d_i(x_j, y_k) = \sqrt{(x_j - x_{m_i})^2 + y_k^2} \tag{3.5}$$

The sample of a signal that corresponds to $\Delta d_i(x_j, y_k)$ can be found using function f in equation 3.2. The result at pixel (x_j, y_k), $\mathbf{R}[j, k]$, is found by adding the samples of different pulse-compressed measurements that correspond to the distance between given measurement and the pixel of interest (x_j, y_k). This is given by the following equation:

$$\mathbf{R}[j, k] = \sum_i \mathbf{p}_i[f(\Delta d_i(x_j, y_k))] \tag{3.6}$$

An advantage of the sum-and-delay algorithm is that it is accurate, a relative probability-value is determined for every pixel separately. Another advantage is that the implementation of the method is straightforward. A disadvantage is that the algorithm determines the relative probability-values for all pixels one by one. This is a brute-force method and is not very efficient. This property makes this approach not very scalable, the algorithm can become very slow for large datasets. Another disadvantage is that it is necessary to wait before all the measurements are done, before the method can be started, even though it is technically possible to do most of the work while measuring. For these reasons it is desirable to implement some changes or an entirely different technique.

3.4. Back-projection algorithm

To overcome the limitation of the sum-and-delay algorithm that it can only be started after all measurements are done, another approach should be developed. It is possible to implement the sum-and-delay algorithm in a slightly different way, which partially solves the problem. This other implementation of sum-and-delay is called the backprojection algorithm. Whereas in sum-and-delay the result is acquired per pixel, in backprojection the result is acquired per measurement. Changing the order causes that a large part of the required operations can be done during the measurements. The only step that is still required after all the measurements have been done is the addition of all the partial results to form the final result $\mathbf{R}$. The sum-and-delay algorithm in principle gives the same results as the backprojection algorithm.

The backprojection algorithm can be described mathematically as follows. The distance between measurement point $\mathbf{m}_i = (x_{m_i}, 0)$ and the coordinate (x_j, y_k) that corresponds to pixel $[j, k]$ is given by equation 3.5. For every measurement a matrix $\mathbf{Q}_i$ is made that represents the scene. The entry of $\mathbf{Q}_i$ at pixel $[j, k]$ that corresponds to the coordinate (x_j, y_k), is the sample of the measurement $\mathbf{p}_i$ that corresponds to the distance difference $\Delta d_i(x_j, y_k)$. Similarly to before, function f can be utilized to find this sample:

$$\mathbf{Q}_i[j, k] = \mathbf{p}_i[f(\Delta d_i(x_j, y_k)] \tag{3.7}$$

This should only be done when a pixel is within the antenna beam of the measurement of interest, just as was the case with the sum-and-delay algorithm. Another similarity is that the spacing of (x_j, y_k) can be arbitrary as long as it stays in the scene. As said before, when the distance does not directly match a sample of the signal, linear interpolation is used to find the adequate signal value. The final result $\mathbf{R}$ is then made by adding the different $\mathbf{Q}_i$:

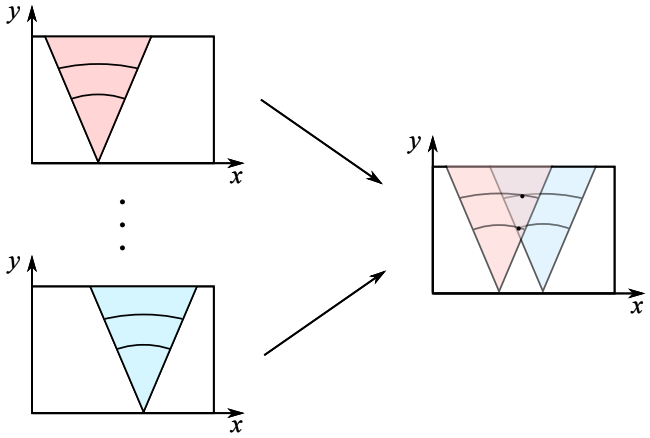


Figure 3.3: The function of the backprojection algorithm. For a certain measurement a part of the result is obtained. Doing this for all measurements and adding the parts will yield the final result.

$$\mathbf{R} = \sum_i \mathbf{Q}_i \tag{3.8}$$

This procedure is also shown in figure 3.3. A certain measurement gives a certain part of the result.

Besides the options for doing measurements and calculations simultaneously, another advantage of the backprojection algorithm is that it can be extended without much effort to other SAR-configurations such as for example spotlight-SAR, which will be explained later. Even though there are added parallellization possibilities, there is still the disadvantage of a relatively large brute-force element, just as was the case for the sum-and-delay algorithm. This means that the scalability might still be improved.

3.5. Optical algorithm

A different algorithm that can be scaled more easily, is the optical algorithm. The reason that this algorithm called the optical algorithm is that it is a digital implementation of the original, optical method that was used to form SAR-images. The original method was optical, because there was limited digital processing power at the time. For this algorithm the data first is represented in a certain way and then some operations are used to convert the representation to an image. In the following subsections a description in more detail will be given.

3.5.1. Data representation

To apply the optical algorithm, the data is first represented in a different way. An image is used where column i is equal to the pulse compressed measurement $\mathbf{p}_i$, so this image is a direct representation of $\mathbf{P}$. Consequently, this approach is fundamentally different from sum-and-delay and backprojection algorithms. This gives an image of the actual scene, but with some modification due to the effect of SAR-imaging. To recreate the actual scene the effect of the SAR-imaging should be removed.

The effect of SAR-imaging can be understood with the help of figure 3.4. When the antenna moves along the synthetic aperture there will be a certain measurement where a certain object will be visible for the antenna for the first time. At this measurement the distance between the object and the antenna is relatively large. Consequently, the time before a reflection from the object is received will also be relatively large. At later measurements the distance between the antenna and the object will be lower, and therefore, the time until the corresponding reflection will be lower. The distance will be minimal when the antenna is below the object and will increase once the antenna moves further away from the object.

When the object is located at (x_0, y_0), the distance d to the antenna at position $(x, 0)$ can be given mathematically as:

$$d = \sqrt{(x - x_0)^2 + y_0^2} \tag{3.9}$$

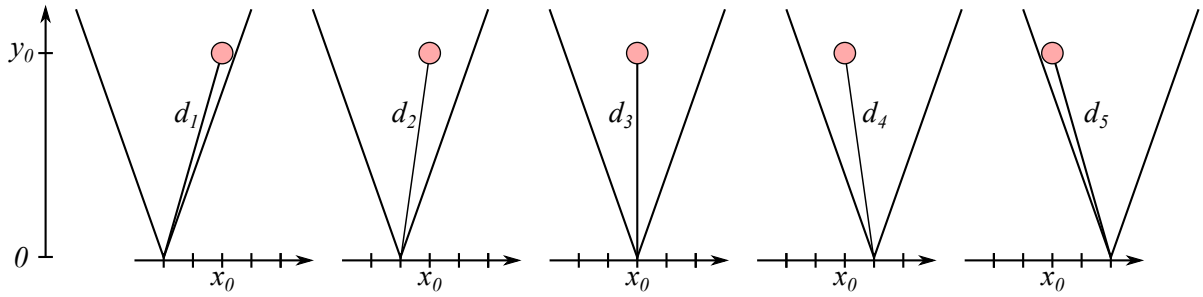
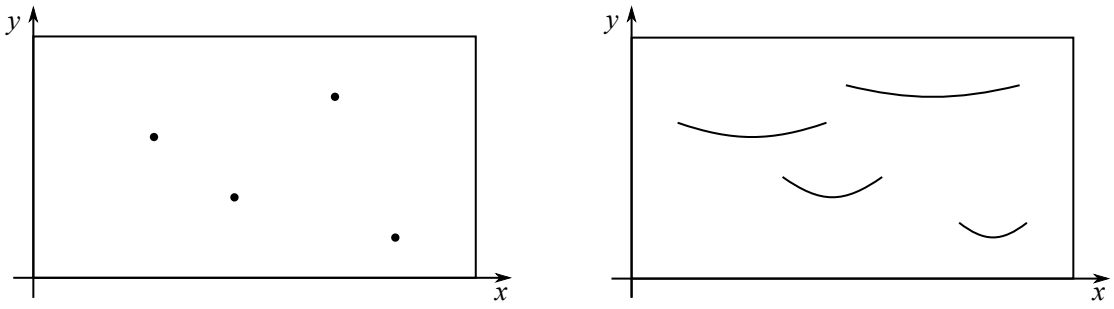


Figure 3.4: The difference in the measured distances in case of a single (stationary) target located at (x_0, y_0). At the first and last measurements the distances are the largest, when the antenna is positioned exactly "below" the target (x_0) the distance is the smallest.

This variation in the distance will lead to a curvature-effect in the data-representation. For example, if the scene would look like figure 3.5a, SAR-imaging would lead to the image with curves shown in figure 3.5b. The width of the curves is determined by the beam width of the antenna, which determines the moment at which an object is visible for the antenna for the first time.



(a) Simple example scene with multiple point targets.

(b) The pulse compressed data of the scene.

Figure 3.5: The goal of the optical algorithm is to create the image (a) using the measured and pulse compressed data (b).

The goal of the optical algorithm is to remove the effect of SAR-imaging by applying a couple of operations. These operations should convert the curves to points at the minima of these curves. This is the case because the minima correspond with the actual locations of objects in the scene. So the image in figure 3.5b should be converted back to the image of the scene in figure 3.5a.

In the optical algorithm this is done in two steps. In the first step the curves are converted to straight lines, to remove the effect of the distance-variation when the antenna moves along the synthetic aperture [28]. This step is called range migration. The closer objects are to the radar, the more important this is, because the distance-variation then is relatively large compared to the length of the curves. Therefore, in the application of road-mapping this can be crucial. In the next step the straight lines are converted to points that represent the locations of objects in the scene. This step is called azimuth compression. These two steps will be explained in more detail in the following two subsections.

3.5.2. Range migration

In range-migration, the different parts of the SAR-curves need to be shifted along the range-direction in a certain way. However, how much certain curves should be shifted in a certain place depends on the actual locations of the relevant objects, but in the first instance it is not known beforehand where the objects in the scene actually are. For example, the part at the end of a curve should be shifted more than a part around the part middle of the curve. Because the locations of the objects and thus the curves are unknown, a guess has to be made.

This is done by selecting a certain subpart of the pulse-compressed data $\mathbf{P}$, a certain window $\mathbf{W}_i$. The window is selected in such a way that there is a possibility that a SAR-curve lies in this window, see figure 3.6. A filter is applied to this window, that applies a certain distance-shift. If there was

a SAR-curve in the middle of the window, the filter will transform this curve into a straight line. In case there is no curve in the window or if the curve is not centered in the middle of the window, the distance-shift that is accomplished by the filter will not transform the curve into a straight line, but into something else. This incorrect shift will cause some distortion in the final image. The same steps are repeated for all possible windows within the pulse compressed data **P**. In this way, for every window is determined whether there was a curve inside the window. These filtered-windows are then passed on to the azimuth-compression step, where further processing is done.

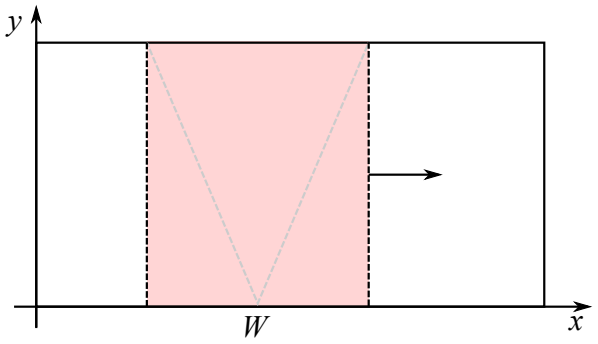


Figure 3.6: One of the possible range migration windows, a window has width W.

The width of the windows is determined by looking at the length of the SAR-curves. This length is determined by the directivity of the antenna, as is shown in figure 3.6. The width of the windows should be chosen in a way that an entire curve fits into it, even at the largest distance of interest y_{max} (also see figure 3.1. This is important, because when curves are cut off, information about the objects is thrown away. It is also important to not choose the window-width way too large, because then more noise will be in the window. The beam angle also causes that the curves close to the antenna are smaller than the curves far away from the antenna. In theory this leads to less strong image formation close to the antenna. This effect is however compensated by the fact that the electromagnetic waves are attenuated less at this distance. Therefore, it was chosen to leave this as is. The width of the window in meters is W and the width in pixels is W_p.

Every window is centered around a certain x-coordinate. A window will provide the part of the result **R** with this x-coordinate, as will be explained later in more detail. To obtain information in the result for every x-coordinate for which a measurement was done, it is necessary that a window is centered around all the measurement positions. If a window is centered around an x-coordinate that belongs to one of the first or the last columns of **P**, there will not be enough other columns around this column to fill an entire window. This is a problem, because every column of a window needs to contain information, none can remain empty.
This problem is solved by zero padding the left and right of the pulse-compressed data **P**. This is also shown in figure 3.7. The amount of zeros necessary corresponds to a width of around $W/2$ meter on each side. In this way as many windows can be made as there are measurement points m_i. So

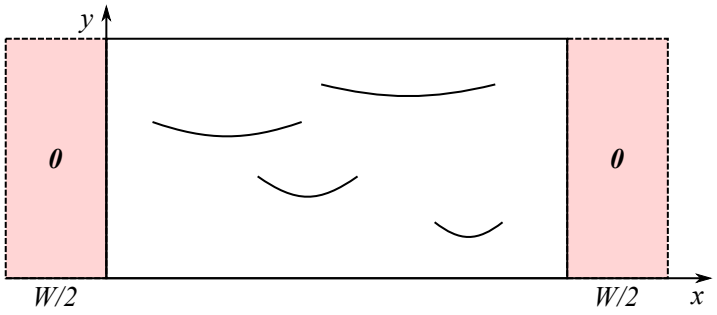
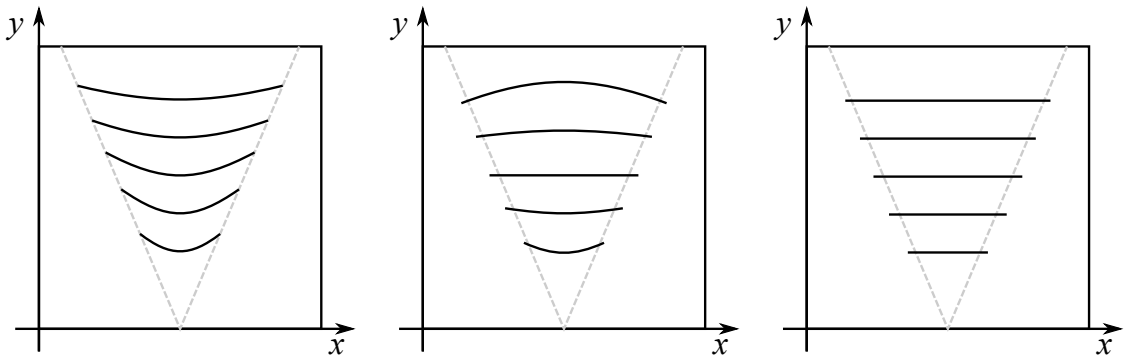


Figure 3.7: Zeros are added to both sides of the original pulse-compressed data **P** to be able to select enough windows.



(a) No focus point. (b) One focus point. (c) Multiple focus points.

Figure 3.8: In (a) a set of the original data-curves is shown. In (b) the range-migrated data-curves are shown, when only one y_{focus} is used. It can be seen that only one of the curves turns out straight, the other ones are out of focus. this will cause some distortion in the final image. In (c) the range-migrated curves are shown when multiple different y_{focus} are used. In this case all curves have become straight.

mathematically the window selection can be described as follows. Zeros are added to the sides of the matrix $\mathbf{P}$, creating a zero-padded version $\mathbf{P}^{zp}$:

$$\mathbf{P}^{zp} = \begin{bmatrix} \mathbf{0} & \mathbf{P} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^{zp} & \mathbf{p}_2^{zp} & \cdots & \mathbf{p}_{M+W_p-1}^{zp} \end{bmatrix} \tag{3.10}$$

A window $\mathbf{W}_i$ is formed by taking a subset of the columns of $\mathbf{P}^{zp}$:

$$\mathbf{W}_i = \begin{bmatrix} \mathbf{p}_i^{zp} & \mathbf{p}_{i+1}^{zp} & \cdots & \mathbf{p}_{i+W_p-1}^{zp} \end{bmatrix} \tag{3.11}$$

When a set of windows is created, a filter has to be designed that performs the right operation on these windows. This filter is applied in the frequency-domain but there are possibilities to do this in the spatial-domain as well. The filter should perform the operation that converts the image in figure 3.8a to the image in figure 3.8c. The distance-shift that is required at a certain position in the image can be determined with the help of equation 3.9. This equation gave the distance between an object at position (x_0, y_0) and the antenna at position $(x, 0)$. This distance should be changed to the distance between the object and the antenna when the antenna is below the object. This distance is y_0. This means that the following distance-difference should be removed:

$$\Delta d = \sqrt{(x - x_0)^2 + y_0^2} - y_0 \tag{3.12}$$

This equation shows that the distance-shift that is necessary depends on the height of the object y_0. This means that at different y-values in the window, a different distance-shift is required. This cannot be achieved with a single filter in the frequency-domain, because this would mean that the same spatial frequencies (similar curves) are treated differently. This is not possible, because the filter can only change the amplitude and phase per frequency. Later a certain trick will be applied to solve this problem, but for now it will be assumed that it is required to set y_0 to a constant value y_{focus}, which will determine how all curves in the window will be shifted. So all curves are shifted in the same way as the curve that would be located at $y = y_{focus}$. This will lead to some distortion because the curves that are not at this y will be shifted with a certain error that grows larger when the object is further away from $y = y_{focus}$. Figure 3.8b illustrates the effect of being out of focus. The distance-shift that corresponds with y_{focus} is given as follows:

$$\Delta d = \sqrt{(x - x_0)^2 + y_{focus}^2} - y_{focus} \tag{3.13}$$

Because the distance-shift in the spatial domain is only in the y-direction, it is enough to do the Fourier-transform on the window only in the y-direction. After this, the transform can be multiplied with the filter, that also is in the frequency-domain for only the vertical direction. After this the range-migrated window can be obtained by doing the inverse Fourier-transform, also only in the vertical direction. The Fourier-transform in the y-direction of the window is as follows:

$$\mathcal{W}_i = \mathcal{F}_y\{\mathbf{W}_i\} = \begin{bmatrix} \boldsymbol{w}_i & \boldsymbol{w}_{i+1} & \dots & \boldsymbol{w}_{i+W_p-1} \end{bmatrix} \tag{3.14}$$

Where the curly letters indicate that the Fourier-transform has been done and where it holds that:

$$w_i[k] = \sum_{n=0}^{N-1} p_i^{zp}[n] e^{-j2\pi kn/N} \tag{3.15}$$

The filter can be obtained by using the fact that a distance-shift in the spatial domain corresponds to a certain frequency-dependent phase-shift in the (spatial) frequency-domain. This property is shown in the following equation:

$$\text{If } g(t) \overset{F}{\longleftrightarrow} G(f) \tag{3.16a}$$

$$\text{Then } g(t-\tau) \overset{F}{\longleftrightarrow} G(f) e^{-j2\pi\tau f} \tag{3.16b}$$

In other words: a shift in the time (space) domain corresponds to a frequency-dependent phase shift in the (spatial) frequency domain. Since the required distance-shift was given by equation 3.13, this means that the filter H is given as follows, where x and y are the variables in the spatial domain and u and v are the corresponding variables in the spatial frequency domain:

$$H(x,v) = \exp(j2\pi\Delta dv) = \exp\left(j2\pi\left(\sqrt{(x-x_0)^2 + y_{focus}^2} - y_{focus}\right)v\right) \tag{3.17}$$

This is a space-continuous function. Therefore, before the filter can be applied to the windows, sampling is required. The space between samples can be determined using the function f (equation 3.2). The indices in the x- and y-direction will be called m and n respectively. So the range-migrated window in the (x,v)-domain is as follows:

$$\mathcal{W}_i^{RM}[m,n] = \mathcal{W}_i[m,n] H[m,n] \tag{3.18}$$

Where RM stands for range-migrated. To obtain the range-migrated window in the (x,y)-domain, the inverse Fourier transform in the vertical direction is taken:

$$\mathbf{W}_i^{RM} = \mathcal{F}_y^{-1}\{\mathcal{W}_i^{RM}\} \tag{3.19}$$

So if it holds that:

$$\mathcal{W}_i^{RM} = \begin{bmatrix} \boldsymbol{w}_{i,0}^{RM} & \boldsymbol{w}_{i,1}^{RM} & \dots & \boldsymbol{w}_{i,W_p-1}^{RM} \end{bmatrix} \tag{3.20}$$

And:

$$\mathbf{W}_i^{RM} = \begin{bmatrix} \mathbf{w}_{i,0}^{RM} & \mathbf{w}_{i,1}^{RM} & \dots & \mathbf{w}_{i,W_p-1}^{RM} \end{bmatrix} \tag{3.21}$$

Than, using the definition of the inverse DFT, it holds that:

$$\mathrm{w}_{i,j}^{RM}[n] = \frac{1}{N} \sum_{k=0}^{N-1} w_{i,j}^{RM}[k] e^{j2\pi nk/N} \tag{3.22}$$

Because $\mathcal{W}_i^{RM}$ and $\mathbf{W}_i^{RM}$ do not have matching columns, two indices are required to distinguish them. This is caused by the applied filter.

3.5.3. Azimuth compression

The range-migration step gives a set of range-migrated windows to the azimuth-compression step. The windows that correspond to the places where objects are located will contain straight lines. These straight lines will be centered around the middle of the window. This can also be understood by looking at figure 3.8. This set of windows has to be converted to an image that represents the actual scene.

A simple way to do this conversion is to sum for each window the elements of every row. In this way every window is converted to a column of data. This is also represented in figure 3.9. The column that is created for a certain window belongs to the x-value that corresponds to the middle of the window. So if the result $\mathbf{R}$ is written as:

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \dots & \mathbf{r}_M \end{bmatrix} \tag{3.23}$$

Then window $\mathbf{W}_i$ centered around the x-value of measurement position m_i will give column $\mathbf{r}_i$ of the final result:

$$\mathbf{r}_i = \sum_{n=1}^{W_p} \mathbf{w}_{i,n}^{RM} \tag{3.24}$$

All range-migrated windows together can be used to produce the entire final result $\mathbf{R}$.

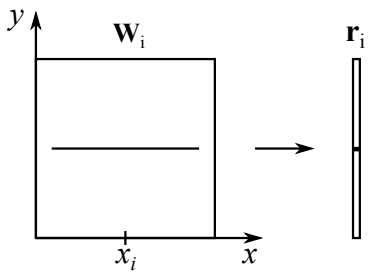


Figure 3.9: The idea behind azimuth compression. Every window is used to create one column of the final result. This is done by summing all the elements along the rows of the window.

This method works because for a certain window, the rows with straight lines correspond to objects. In these rows many elements with large values will be added together, leading to a high resulting value. In the rows where there is no straight line, there was no object and the elements will add up to a relatively low resulting value. In this way indeed the probability that there is a reflecting surface at a position is represented.

3.5.4. Variable focus

In practice it is not desirable that only one focus-distance can be chosen because it causes that objects in the scene that are further away from y_{focus} are not imaged sharply. A solution to this problem is running the algorithm for a range of different y_{focus}, which will create a set of different results that all show another part of the scene in focus. So, for example, one of the results displays objects around $y_{focus,1}$ best, another result displays the objects around $y_{focus,2}$ best, and so on. Next, for each of these results the part around the corresponding y_{focus} is selected and all these parts are combined into one final result. This will ensure that all objects in the scene are shown in focus, even when they are at different y-coordinates. So the result of the algorithm with $y_{focus,1}$ gives the part of the final result around $y_{focus,1}$, the result of the algorithm with $y_{focus,2}$ gives the part of the final result around $y_{focus,2}$ and so on. This procedure is also shown in figure 3.10.

The number of focus-points used is an adjustable parameter. When more focus points are chosen, the quality of the final result will be higher, but the speed of the algorithm will be lower. Therefore, it is important that a good trade-off between quality and speed is made.

3.5.5. Schematic representation

A block diagram that describes the optical algorithm is shown in figure 3.11. The functioning of the algorithm can be repeated shortly as follows: First the zero-padded, pulse-compressed data is converted

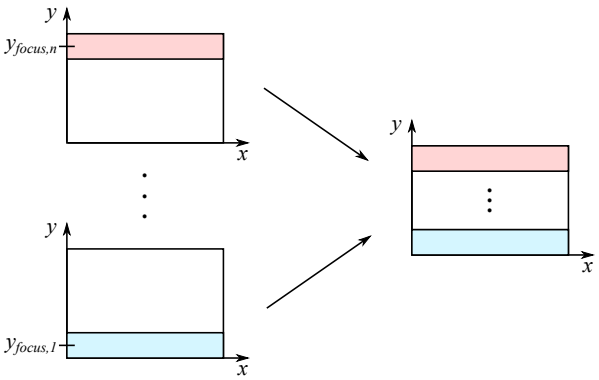


Figure 3.10: The idea behind the variable focus. Many different results with a different y_{focus} are created and the best parts from each are combined to form the final result.

into a set of windows $\mathbf{W}_i$. Next the range migration is done on every window, using a filter in the frequency-domain. After this every range-migrated window is converted to a single column by applying azimuth compression. All these columns together form the final result $\mathbf{R}$.

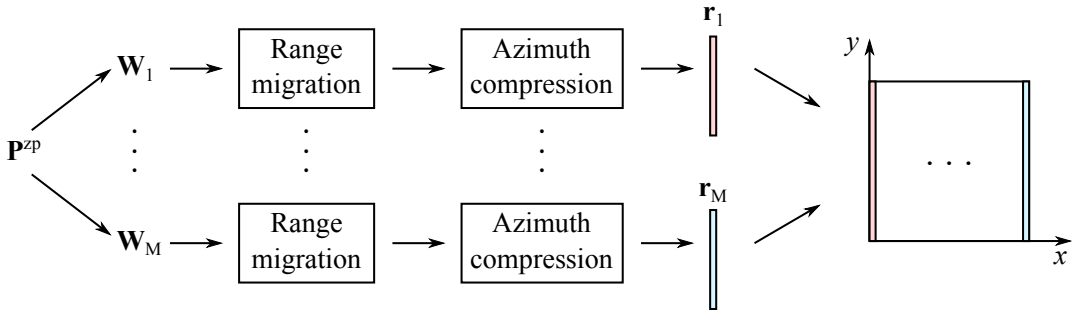


Figure 3.11: Summary of the optical method. The zero-padded pulse compressed data $\mathbf{P}^{zp}$ is first divided into smaller overlapping windows $\mathbf{W}_i$. Then range migration is applied to these windows individually. Azimuth compression ensures that the horizontal lines in the window are mapped to a single point. These results $\mathbf{r}_i$ are placed after each other creating an image $\mathbf{R}$.

3.6. Results

3.6.1. Simulator

To test the algorithms described in the previous chapter a simulator was created. In an ideal case a single delta pulse is transmitted and the reflections of that pulse are recorded. With this simulator the received pulse(s) from (multiple) point-like targets could be generated and thus the functionality of the algorithms could be tested. This way these algorithms could be evaluated in a noise and distortion-free environment. The working of the simulator follows these steps:

1. Generate point-like targets in the scene.
2. Generate synthetic aperture ($\mathbf{m}$).
3. Generate received data ($\mathbf{s}_i$) for each measurement point m_i based on the beam angle (θ) of the antenna and the targets defined in the first step.
4. Apply the algorithm to be tested.

In the first step the (x, y) coordinates of N_{target} number of targets are generated. Here simple geometric objects should be thought of, such as line (segments) and circle (segments). After that, the measurement locations are determined. For simplicity, the y coordinates of the the measurements are

Table 3.1: Coordinates of simulated targets

	Point target #1	Point target #2	Point target #3
x [m]	1.1	2.2	2.5
y [m]	3.3	2.1	2.1

taken as zero. In the following step the Cartesian distances are calculated between m_i and each target that lays inside the beam angle defined at m_i. The attenuation of the signal over this distance is accounted for. These distances are then converted to samples and the attenuated delta pulse is shifted accordingly. The simulator does not compensate for the fact that the closer the target is to the borders of the beam, the less power is received, i.e. the directivity of the simulated antenna resembles a square wave.

In the following sections simulated results will be shown. There were 3 point-targets simulated whose coordinates are given in table 3.1. This measurement setup will be later used in real-life measurements (see 3.6.2). In figure 3.12 the received data $\mathbf{S}$ is depicted, which in this case equals to $\mathbf{P}$ as the transmitted (and thus the received) pulse is already compressed to a delta spike. Here the curves can be seen as a result of differences in the distances explained in 3.5.1. This ideal data is then fed to the sum-and-delay algorithm, of which the results are shown in figure 3.13a, and the backprojection algorithm, of which the results are shown in figure 3.13b. Both of these algorithms give practically the same result, as was expected, since both methods perform the same operations, but in a different order. The spots in the result where the values are highest, i.e. the spots where the probability on a reflecting surface is highest, correspond well with the positions where the point-targets are actually located. It can also be seen that there are some sidelobes around these spots. These are inherent to SAR-imaging and are caused by the fact that for a single radar-measurement it is not exactly known where an object is within the antenna beam. When this is not known, a certain probability has to be allocated to every position within the antenna beam that is at the same distance from the antenna as the object.

The ideal simulated data was also given to the optical algorithm. When doing this, in the first instance the variable focus is turned off, to show the effect of this functionality. This means that only one y_{focus} is chosen. All objects that are approximately at this y will be displayed relatively sharp. Other objects will be displayed increasingly less sharp, dependent on how far they are from y_{focus}. This was first done for a focus-distance of 2.1 m, at the y-coordinate of the closest two point-targets. The result is shown in figure 3.15a. It can be seen that the two closest point-targets are imaged sharply and that the point target in the back is out of focus.

Next the algorithm was applied with a focus-distance of 3.3 m, which is the y-coordinate of the single point target furthest away. the result is shown in figure 3.15c. It can be seen that the single point-target in the back is now imaged sharply and that the other two point-targets are out of focus.

Finally the algorithm was applied for a variable focus. The result is shown in figure 3.15e. It can be seen that now all three targets are well visible. The result is similar to the results of the sum-and-delay and backprojection algorithms. Again the point-targets are mapped to the right location and again some side-lobes are visible.

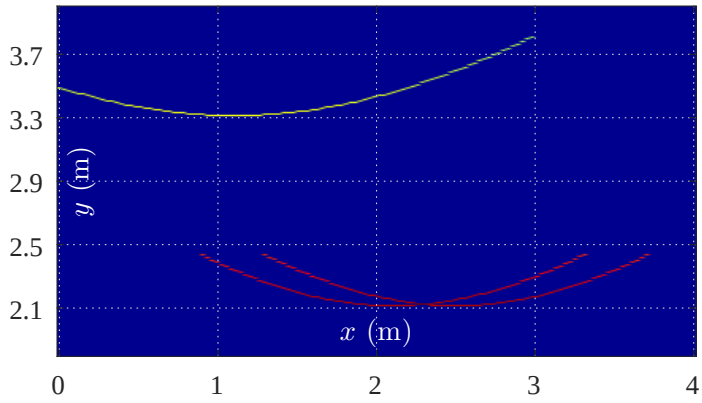


Figure 3.12: Simulation of 3 point-like targets.

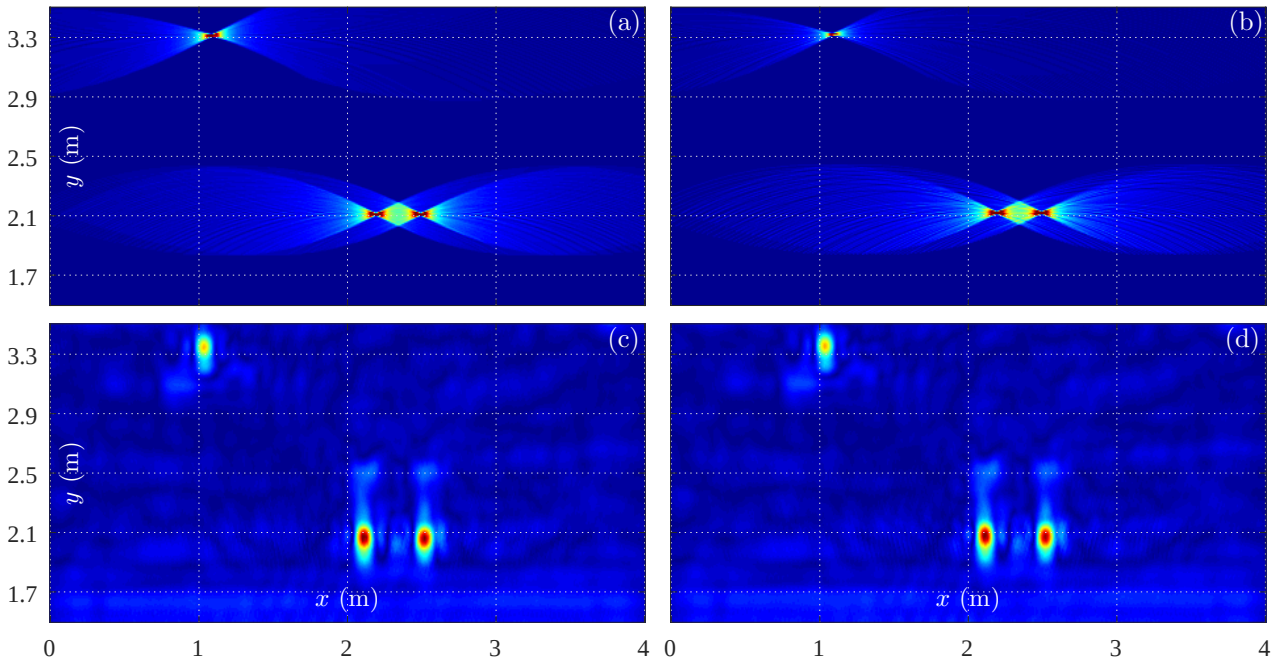


Figure 3.13: The results of simulated and measured data of 3 point-like targets with sum-and-delay and backprojection algorithm. All images given here are on the same scale and have the same axes. Top row shows simulated data, bottom row shows the corresponding image with real measurements. In the left column sum-and-delay algorithm was used while in the right column backprojection algorithm was used.

3.6.2. Results of real data

To test the imaging capabilities of the algorithms the same scene with 3 point-like targets was used during measurements as in the simulations (see table 3.1). This real-life scene is shown in figure 3.14. The set-up that was used to move the antennas is described in [26] and the antennas that were used are described in [27]. The obtained data was first pulse compressed before the algorithm was applied. The result when the sum-and-delay algorithm was used is shown in figure 3.13c. The result when the backprojection algorithm was used is shown in figure 3.13d. Just as in the simulation, these two results are again practically the same. The cans are imaged clearly and can be distinguished well from the noise. The values corresponding to the cans in front are higher to those corresponding to the cans in the back. The reason for this is that the transmitted electromagnetic pulse has been attenuated less when it reaches the front two stacks. The received reflections that correspond to these stacks are stronger.

When testing the optical algorithm on this same simple measurement, again in the first instance the variable focus is turned off, to show the effect of this functionality. Figure 3.15b shows the result when $y_{focus} = 2.1$ m, the y-coordinate of the two stacks of cans in front. As expected, these stacks are displayed sharply and the cans in the back of the scene are out of focus.
Figure 3.15d shows the result when $y_{focus} = 3.3$ m, the y-coordinate of the stack of cans in the back. As expected, the can in the back is displayed sharply and the two cans in front are out of focus.
The optical algorithm with variable focus gives the result that is shown in figure 3.15e. In this result both the cans in font and the cans in the back are shown in focus. The result is similar to the results of the sum-and-delay and backprojection algorithms.

Apart from a measurement with three stacks of cans relatively far apart, also a more complex measurement was done, where the letters TU were spelled out with cans that are closer together. The measurement set-up is shown in figure 3.16. In this case the optical algorithm was used to obtain the result, that is shown in figure 3.17. All the different stacks of cans can be distinguished well from the noise, even though some of them are behind other stacks. The reason that this is not a problem in this case, is that measurements are taken from many positions, and that all stacks are visible from at least some of these measurement positions.

Figure 3.14: The simple set-up used to initially test the algorithms.

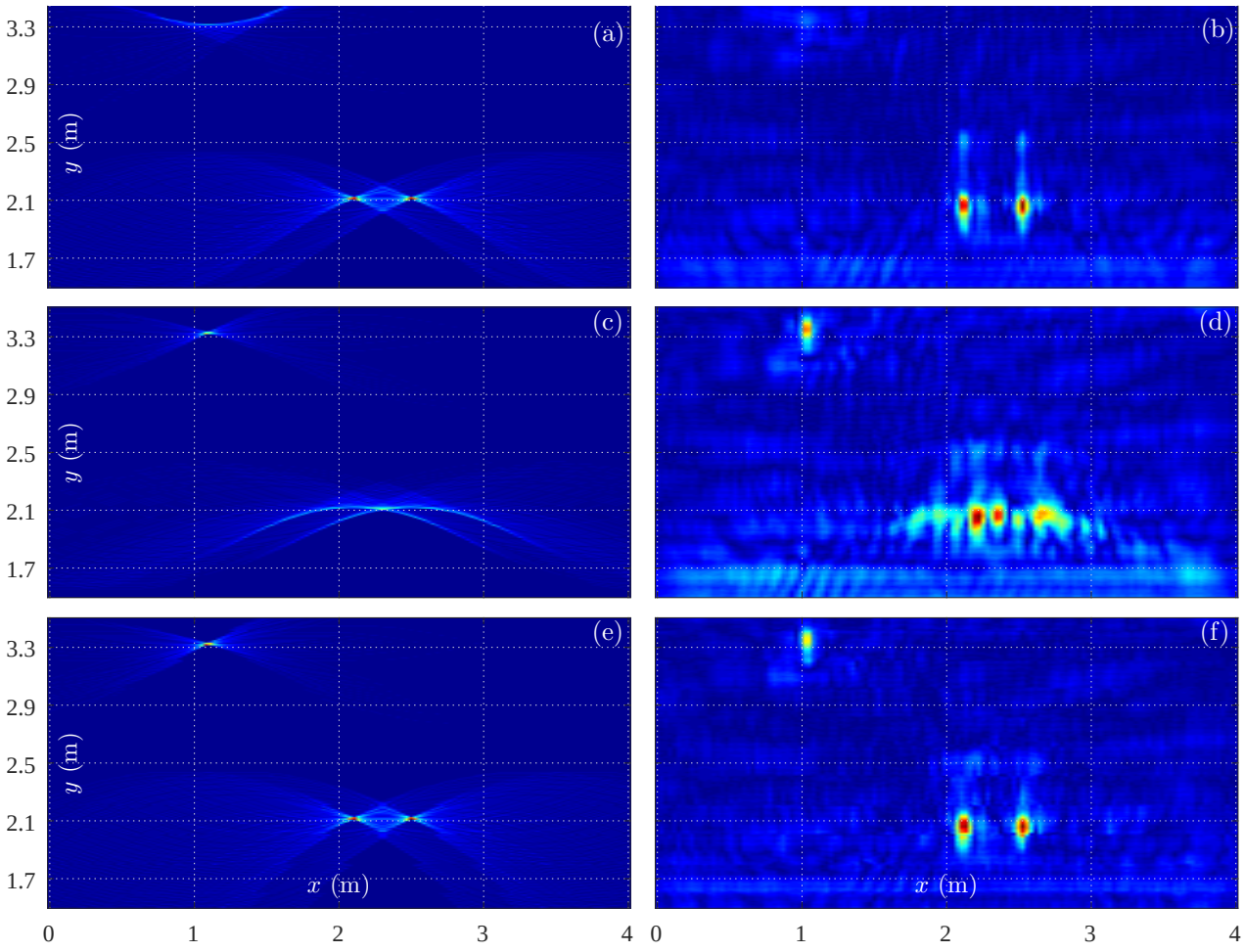


Figure 3.15: The optical method with simulation and real data. The images are shown in scale and with the exact same axis. Left column corresponds with simulation and right column with measured data. The focus distance in the top row is 2.1 m, in the middle row is 3.3 m, and in the bottom row multiple focus points were used.

Figure 3.16: More complex set-up with multiple stacks of cans in the form of the letters TU.

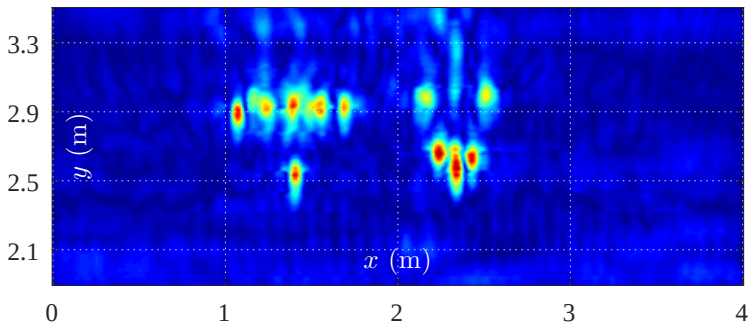


Figure 3.17: More complex image with multiple point-like targets in TU-shape. The optical algorithm was used for creating the image.

4

Application specific imaging algorithms

4.1. Introduction

The general goal of the bachelor graduation project was to implement the technique of SAR to achieve higher localization accuracy of cars and to be able to obtain semi real-time information on the environment around roads, as was given in section 1.4. Implementing the SAR-technique can be done by mounting the measurement antennas on the car. The movement of the antennas is then created automatically by the movement of the car. In this way a SAR-image of the area around the car can be made.

Doing this can give two major improvements with respect to the current system. The first improvement is that the location of cars can be determined up to a much higher accuracy. In urban areas where there are many buildings that reflect signals, GPS alone is much less accurate. Especially when different roads are close together, this could lead to some errors. This problem can be decreased by comparing the SAR-image made by the car with an earlier collected database of similar SAR-images. When there is a match between the measured SAR-image and the database, the location of the car can be determined with much higher accuracy.

The second improvement is that changes in the road-system can be detected semi real-time. When something happens, for example a car accident that causes a car to be stalled on the road, this can be seen on the SAR-image created by cars passing by. The database of SAR-images can be updated with this information and this can be used to inform drivers planning to drive in this direction that they should take another route.

These improvements are especially important in the case of self-driving cars where the insight of humans is not available. If a self-driving car doesnt know accurately where it is located navigation errors might occur and some faults in the localization might even lead to dangerous situations. It is also important that the system of self-driving cars knows on time that something has happened on the road, so that this can be taken into account.

The algorithms that were discussed up until now work only with stripmap-SAR. Cars in general do not drive only in straight lines, so the SAR-algorithms should be made more general to take this into account. Firstly, the direction in which the antenna looks should be made more general, instead of only perpendicular to the direction of motion. Second, the positions $\mathbf{m}_i$ from which the measurements are done should be generalized as well, since it is no longer given that the antennas move in a straight line. So instead of $\mathbf{m}_i = (x_{m_i}, 0)$, it holds that $\mathbf{m}_i = (x_{m_i}, y_{m_i})$. The algorithm where both the angle and location are entirely variable will be called the free-path situation.

In practice the information about the location and measurement angle that correspond to different measurements will be determined using saved information about the speed and the steering angle. Because these variables are only known up to a certain accuracy and because the errors are of a cumulative nature, the maximum size of an accurate SAR-image made by a car will be limited. In principle it is the case that when it is possible to make a larger accurate SAR-image, there is more material to compare and the localization will be more accurate.

4.2. Variable-angle SAR

The first step in making the SAR-algorithm more general is to make the direction in which the antenna looks variable. With this adjustment it is possible to compensate for the fact that the antenna might turn during the measurements, which could for example be the case when a car turns while driving. It was chosen to make the adjustment on the backprojection algorithm, since this was the most easy to do. The main difference with the original version of the algorithm is that the pixels visible for a certain measurement no longer depend only on the x-coordinate, but on the x-coordinate and the antenna direction. The functionality of this change was tested by doing spotlight-SAR measurements that turn the antenna in such a way that one point in the scene is in focus. The results that were obtained with these measurements are given in section 4.4.

4.3. Free-path SAR

The last extension that is required to make the SAR-algorithm generally applicable is that the measurement positions can have two coordinates instead of one, so that it is not necessary that all measurement locations lie along a straight line. Road-mapping applications need this addition as well, since the paths of car can have a relatively arbitrary form. The adjustment is made in the backprojection algorithm where the angle already was made variable, meaning that the resulting algorithm can process the data from arbitrary measurement locations with arbitrary antenna directions in a 2D-scene. Again it was chosen to use the backprojection algorithm, since it is the most easy to adapt. The change in the algorithm is similar to the change required to make the antenna direction variable. The pixels that lie within the visibility of a measurement no longer only depend on the x-coordinate and the antenna direction, but on the x-coordinate, y-coordinate and the antenna direction. This means that equation 3.5 can be rewritten as:

$$\Delta d_i(x_j, y_k) = \sqrt{(x_j - x_{m_i})^2 + (y_k - y_{m_i})^2} \tag{4.1}$$

Tests on the new algorithm were done by moving the antennas along a curved path. The results of this test are shown in section 4.4.

4.4. Results

To test whether the generalized algorithm works correctly, several tests were done. To test the variable antenna direction, a spotlight-SAR measurement was done. In such a measurement the antenna still moves along a straight line, but the antenna direction is no longer necessarily orthogonal to the direction of motion. The antenna is rotated in a way that it always looks at the same point in the scene, this is shown in figure 4.1. For this reason more information is acquired about the area around this point and less information about the other areas in the scene, which causes the resulting image to have an area of focus.

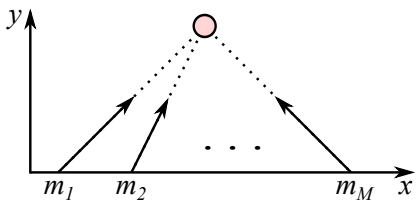


Figure 4.1: Visual representation of measurements with variable angle. Here the antenna faces the target in all measurements while the antenna moves along the x-axis.

To perform a spotlight-SAR test, a platform is required that can rotate the antennas. Elaboration on this platform is given in the thesis of the hardware-group [26]. A measurement where two stacks of cans were placed around the point of focus was done to see whether they could be imaged correctly. The result of this test is given in figure 4.2.

It can be seen that the two stacks of cans are brought into focus and can be distinguished. It can also be seen that everything behind the cans is more smeared out. This result indicates that the algorithm with variable angle works correctly.

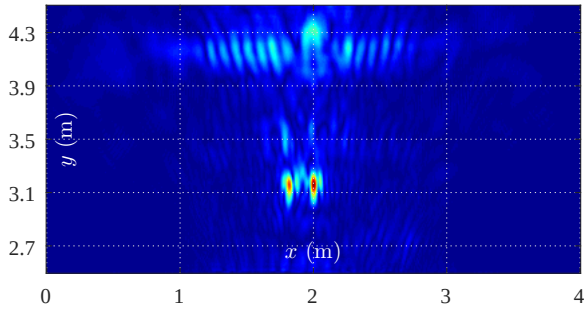
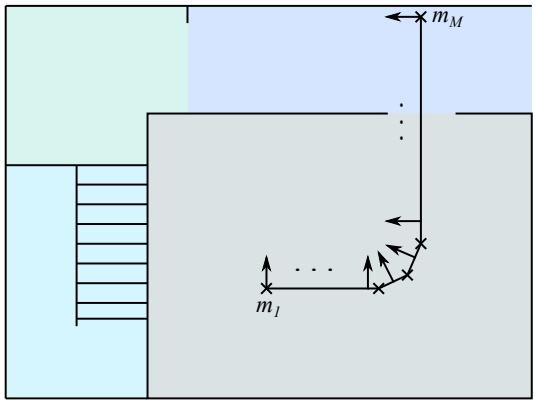


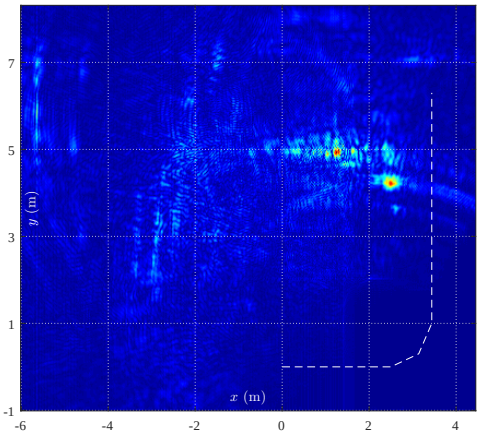
Figure 4.2: The result of measured data with variable angle. During the measurements two stacks of cans (similar to previous measurements) were placed at $(1.8, 3.1)$ m and $(2.0, 3.1)$ m.

To test the entire free-path algorithm, a test was done that resembles actual road-mapping applications, albeit on a slightly smaller scale. The antennas were placed on a manoeuvrable cart, which was then moved along a predetermined route through a room and corridor. The lay-out of the building and the route that was driven are shown in figure 4.3a. In this case measurements were only done to the left side of the cart. In the real road-mapping application it would be desirable to make radar-scans of both sides of the car, but since measurements on only one side can be extended directly to two sides, scans of one side suffice for testing purposes.

The result obtained when the data of this test was given to the free-path algorithm, is shown in figure 4.3b. The walls and doorways are visible in the result, therefore a map of the building is created. The accuracy is however limited. It is suspected that this is caused by doing the test on a smaller scale, making constant errors relatively large. Another factor that could play a role is that the measurement positions and corresponding antenna angles are only known up to a limited accuracy. When more accurate information on the steering angle and speed of vehicles is used and the test is done on a more realistic scale, the images could improve and they can be used for localization more easily.



(a) The schematic of the measurement with the taken path indicated.



(b) The resulting image.

Figure 4.3: The result of free-path measurement. (a) shows the route of the measurement with schematic of the environment. The route consists of 4 linear sections marked between the crosses. At each separate section the antenna was set perpendicular to the movement direction. (b) shows the result of the free-path algorithm applied on the collected data. The route is shown as white dashed line. The origin of the coordinate system was set to the first measurement, m_1.

5

Conclusion and future work

5.1. Conclusion

The goal of the BAP-project was to implement UWB SAR techniques to improve localization accuracy in road-navigation and to provide possibilities for semi real-time road status updates. To this end, several SAR-algorithms have been implemented. The first one was the sum-and-delay algorithm, which is relatively easy to implement, but is not optimal in terms of efficiency. A slight improvement of this technique is the backprojection algorithm, which is better in the sense that a large part of the required operations can be done during the measurements. The final implemented method is the optical algorithm, which is more efficient than the first two techniques, but of a higher complexity. The three techniques were tested using a set-up where the antennas move along a rail while doing measurements. It was possible to image objects such as cans, so the resolution seemed adequate for mapping applications. The three algorithms were however not yet directly usable for road-mapping applications, since they only work when the measurements are done along a straight line and the antenna direction is perpendicular to the direction of motion, which is not necessarily the case for a driving car.

To improve this the measurement position and the antenna direction possible as input to the algorithm have been generalised. This has been done only for the backprojection algorithm for simplicity reasons. To test the variable antenna angle, a spotlight-SAR measurement was done. It was again possible to image an object like a can, proving that the adjustment was made correctly. Finally a free-path measurement was done that was similar to an actual road-map measurement, but on a smaller scale. It was possible to map the walls of the room and corridor up to a certain accuracy, showing that the adjustment where the position was made variable worked as well. This result also proved the concept in general, because when it is possible to apply this technique on such a small scale, it also seems possible to do it on the scale of a real road. The integration of SAR-techniques with cars can give many improvements in localization of cars and semi real-time mapping of surroundings. These can be especially advantageous in the case of self-driving cars, where knowledge of location and environment is of vital importance.

5.2. Future work

The result of the free-path SAR measurement seemed promising. The mapping accuracy is still limited, but it is suspected this will improve when accurate information about the speed and steering angle of the vehicle is used to create more accurate knowledge on the measurement positions. At the moment the technique already seems applicable up to a certain extent, however, to reach the ultimate goal of a fully integrated system several tasks still can be done:

- At higher speeds it might be possible to take the Doppler-shift into account to improve the images. It might be possible to do this by making the pulse compression more general.
- In the optical algorithm, at the moment a rectangular window is used, even thought the antenna beam is more triangular. This causes that the window takes pixels into account that cannot contain any relevant curves, which will cause some distortion in the result. The result might be improved when a window is used that matches the shape of the antenna better.

- The range-migration in the optical algorithm is done using a filter in the spatial frequency domain. It is also possible to do this with a simple shift in the spatial domain. A comparison can be done between these two methods to determine which is faster and better scalable.
- At the moment only linear interpolation is used. Results might improve when another, more complex, interpolation is done.
- At the moment pixels are assumed to be either visible for the antenna or not visible by the antenna. In reality some pixels will give a stronger reflection than others, the directivity is a smooth function. Taking this into account can improve the images.
- An algorithm should be made that converts the information about steering angle and driving speed into an estimation of the path that was driven by the car.
- A system should be designed that can upload the data received by different cars to servers at a sufficient speed. This system could for example look like a collection of transceivers next to the road that forward data to the cloud in a controlled way.
- An algorithm should be designed to determine which data can be used for updating the database and which data should be ignored. This algorithm should also combine the suitable images in a way that noise is decreased and errors occur less.
- A collaboration with car designers should be started to integrate adequate antennas in the cars or to make the already integrated antennas more suitable.
- Instead of the backprojection algorithm, a more scalable SAR algorithm should be implemented to create free-path images.
- The algorithm should be implemented on a circuit that can be easily integrated into a car. It might be desirable when this chip can compute rapidly in parallel, for example like a GPU, since large parts of the algorithms can be done simultaneously. When the processing of the data is faster, the spacing between measurements can be made smaller and the cross-range resolution can be increased.
- The range-resolution might be increased by optimizing the used frequency band, by optimizing the used antennas and by optimizing the data acquisition.

If these changes are implemented, the system can cause that the increasingly complex world of traffic can keep growing in autonomy, which will improve overall safety and efficiency.

5.3. Acknowledgements

We would like to thank Mr. Pascal Aubry and Mr. Faruk Uysal for their support with this project.

A

Appendix - MATLAB-codes

The MATLAB-codes created throughout the project:

A.1. MATLAB code - Pulse compression

```
function [  data_correlation ] = pulse_compression( data, pulse, r_min, r_max, Fs)
% Find the correlation between the received signal and the pulse, to find
% out at which point there was a high probability that a reflection was
% received.
%
% Inputs:
%   data:   The matrix with the measured data, it is assumed that the data
%           starts at t = 0, so when this is not the case, this has to be
%           corrected before calling this function.
%   pulse:  The transmitted pulse
%   r_min and r_max: The correlation will be cut off in a way that the
%           samples correspond to distances between r_min and r_max.
%           Therefore r_min and r_max should be selected in a way that the
%           region of interest lies between them.
%   Fs:     Rate at which the received signal was sampled.
%
% Output:
%   data_correlation:   The useful part of the correlation between the
%                       received signal and the pulse
%
% Date: 11-06-2017

% Initialize
c = 3*10^8; % Speed of light
[num_measurements, data_length] = size(data);   % Size data-matrix
pulse_length = length(pulse);   % Length of the transmitted pulse

% Mirror the pulse
pulse_flip = fliplr(pulse); % p[t] = p[-t]

% Convolve signal with mirrored pulse
for i=1:num_measurements
    convolved(i,:) = conv(data(i,:),pulse_flip);
end

% Pick the interesting values from data_correlation
data_correlation = convolved;

index_min = floor(r_min*2/c*Fs);
index_min_conv = index_min + length(pulse) + 1;
index_max = floor(r_max*2/c*Fs);
index_max_conv = index_max + length(pulse) + 1;
data_correlation = data_correlation(:,index_min_conv:index_max_conv);

end
```

A.2. MATLAB code - Sum-and-delay algorithm

```
function [ result ] = sumanddelay_algorithm( received, x_rec, r_min, r_max, Fs,
    beam_angle, crossrange_pixels, range_pixels)
%% Sum-and-delay (SaD) algorithm
% Function:
%    This function generates image from SAR measurements using SaD. The
%    contribution of the antenna to a single point (pixel) is calculated by
%    delaying the received signal corresponding to the distance. These
%    individual contributions are then added together.
%
% Inputs:
%   received:   Time-sampled data of each measurement (MxN matrix)
%   x_rec:      x coordinates of the measurements (1xM vector)
%   r_min:      Distance corresponding to y = 0 in the image / the
%               distance between the antenna and the FIRST measured data
%               point (in m)
%   r_max:      Distance corresponding to y = y_res in the image / the
%               distance between the antenna and the LAST measured data
%               point (in m)
%   Fs:         Sample frequency of the data
%   beam_angle: HALF beam angle of the antenna in radians
%   crossrange_
%       pixels: Number of pixels of the result in the x direction
%   range_
%       pixels: Number of pixels of the result in the y direction
%
% Outputs:
%   result:     A (crossrange_pixels x range_pixels) matrix where each
%               element contains the result of delayed and summed signals.
%
% Note(s):
%   It is assumed that the measurements were taken along the x-axis.
%
% Date: 14-06-2017

c = 3e8;   % speed of light

% Compensate for the distance of the first element in 'received'
shift_constant = max([floor(-2*r_min/c*Fs), 0]);

% Size of the data
max_pixel = size(received,2);

% Allocate memory for 'result'
result  = zeros(crossrange_pixels, range_pixels);

% Calculate real coordinates of the pixels
x_pixel = linspace(x_rec(1),x_rec(end),crossrange_pixels);
y_pixel = linspace(r_min,r_max,range_pixels);

%% Generate image with delay and sum
for i=1:crossrange_pixels
    for j=1:range_pixels
        % Calculate distance and angle between current pixel and all of the
        % measurements
        r_pixel = sign(y_pixel(j))*sqrt((x_rec-x_pixel(i)).^2+y_pixel(j).^2);
        angle_pixel = atan2(y_pixel(j),x_rec-x_pixel(i))-pi/2;
        is_in_range = abs(angle_pixel)<beam_angle;

        % Help variable for linear interpolation
        discretization = mod(2*r_pixel/c*Fs,1);

        for k=1:length(x_rec)
            % If pixel is in range of measurement k, the contribution of
            % that measurement is added to the previous value
            if (is_in_range(k))
                % Select the correct index
                index = min([floor(2*r_pixel(k)/c*Fs)+shift_constant,max_pixel]);

                % Linear interpolation
```

```
                pixel_contribution = received(k,index)*(1-discretization(k))+ ...
                   received(k,min([index+1, max_pixel]))*discretization(k);
                result(i,j) = result(i,j) + pixel_contribution;
            end
        end
    end
end
```

A.3. MATLAB code - Backprojection algorithm

```
function [ result ] = backprojection_algorithm( received, x_meas, theta_meas, Fs, x_min
    , x_max, y_min, y_max, beam_angle, r_begin, resolution)
% This function applies the backprojection algorithm to stripmap SAR
% measurement data. Only one iteration is done, because this makes it easy
% to apply the algorithm while doing the measurements. The inputs and
% outputs are as follows:
%
% Inputs:
%   result: The result of the previous iteration. The new information will
%           be added to this old result to create the new result.
%   received: The data of the measurement.
%   x_meas: The x-coordinates of the measurement positions.
%   y_meas: The y-coordinates of the measurement positions.
%   theta_meas: The direction of the antenna for the different
%           measurements.
%   Fs: The rate at which the received signal was sampled.
%   x_min: The x-value at which the scene of interest starts.
%   x_max: The x-value at which the scene of interest ends.
%   y_min: The y-value at which the scene of interest starts.
%   y_max: The y-value at which the scene of interest ends.
%   beam_angle: Half of the entire beam angle.
%   r_begin: The range that corresponds to the first sample of the data.
%   resolution: Spatial resolution of the image in meters
%
% Date: 15-06-2017


% Initialization
c = 3e8; % Speed of light

% Information on the distances corresponding to samples
sample_spacing = c/Fs/2; % The distance between two samples.

% Choose right amount of pixels
x_size = floor((x_max-x_min)/resolution);   % Number of pixels in the x-direction
y_size = floor((y_max-y_min)/resolution);   % Number of pixels in the y-direction

% Initialize matrices
result = zeros(x_size,y_size); % Result
range_signal = result; % Matrix that describes which pixels are visible for the antenna

% Vectors with the x- and y- coordinates that correspond to the samples
x_pixel = linspace(x_min,x_max,x_size);
y_pixel = linspace(y_min,y_max,y_size);

% Min & max angles bepalen
theta_min = theta_meas - beam_angle; % Angle at which the beam starts with respect to
    horizontal
theta_max = wrapToPi(theta_meas + beam_angle); % Angle at which the beam ends with
    respect to horizontal

for i=1:length(x_meas) % Loop over the measurements

    % Reset temporary matrix
    range_signal = ones(x_size,y_size);

    % Determine for a certain height, at which x-value the beam starts and
    % determine which pixels are then not visible by the antenna before the
    % beam.
    if (theta_min(i) > 0)
```

```
        for j=1:y_size
            x_theta_min = y_pixel(j)/tan(theta_min(i))+x_meas(i);
            x_theta_min = min([x_theta_min, x_max]);

            range_signal(:,j) = range_signal(:,j).*(x_pixel <= x_theta_min)';
        end
    end

    % Determine for a certain height, at which x-value the beam ends and
    % determine which pixels are then not visible by the antenna after the
    % beam.
    if (theta_max(i) > 0)
        for j=1:y_size
            x_theta_max = y_pixel(j)/tan(theta_max(i))+x_meas(i);
            x_theta_max = max([x_theta_max, x_min]);

            range_signal(:,j) = range_signal(:,j).*(x_pixel >= x_theta_max)';
        end
    end

    % For all the pixels that are visible for the antenna
    parfor xx=1:x_size
        for yy=1:y_size
            if range_signal(xx,yy) % ~= 0

                % Calculate the distance between measurement position and
                % pixel
                distance_from_measurement = sqrt((x_pixel(xx)-x_meas(i))^2+y_pixel(yy)
    ^2);

                % Find the corresponding signal value with the help of
                % interpolation and add this to the result
                sample_actual = ((distance_from_measurement - r_begin)/sample_spacing);
                sample_actual_floor = max([floor(sample_actual), 1]);
                sample_actual_mod = mod(sample_actual,1);
                result(xx,yy) = result(xx,yy) +...
                    (1-sample_actual_mod)*received(i,sample_actual_floor)+...
                    sample_actual_mod*received(i,sample_actual_floor+1);

            end
        end
    end
end
```

A.4. MATLAB code - Range migration

```
function [ H, window_pixel, h ] = range_migration( x_res, L, y_res, r_min, r_max,
    beam_angle, r_focus )
% Range migration
% Function:
%   This function generates a matched filter that compensates for the
%   changes due to range difference. The antenna receives signals within a
%   wide angle (beam_angle) and objects that are not placed directly
%   perpendicular to the viewing angle are mapped to a signal "further
%   away". To compensate for this a shift is needed, that can be done in
%   either in the time-domain as convolution or as multiplication in the
%   frequency domain. The inputs and outputs of this function are:
%
% Inputs:
%   x_res:      Resolution in the x direction (same as the "length" of the
%               measured data), in pixels (or measurements)
%   L:          The length of the synthetic aperture in the x direction (in
%               meter)
%   y_res:      Resolution in the y direction (same as the "width" of the
%               measured data) in pixels
%   r_min:      Distance corresponding to y = 0 in the image / the
%               distance between the antenna and the FIRST measured data
%               point (in m)
%   r_max:      Distance corresponding to y = y_res in the image / the
%               distance between the antenna and the LAST measured data
```

```
%                point (in m)
%   beam_angle: HALF beam angle of the antenna in radians
%   r_focus:    Distance from the antenna where image should be the
%               sharpest (in m)
%
% Outputs:
%   H:          Frequency domain representation of the matched filter
%   window_pix: The number of pixels of the window
%   h:          Time domain representation of the matched filter
%
% Date: 15-06-2017

% Properties of a window
window_width = min([2*r_max*sin(beam_angle), L]);
window_pixel = floor(x_res*window_width/L);

% Make window_pixel odd (otherwise problematic image filtering)
if mod(window_pixel,2) == 0
    window_pixel = window_pixel+1;
end

% Vectors representing the x- and y-coordinates of the window
x = linspace(-0.5*window_width,0.5*window_width,window_pixel);
y = linspace(r_min,r_max,y_res);

%%
% Vector with distance in meters to be shifted for different x
f_help = sqrt(r_focus^2 + x.^2) - r_focus;
% The same vector, but in pixels instead of meters
f_help = f_help/(r_max-r_min)*y_res;

% Create the filter h in the space-domain
for i=1:window_pixel
    h(i,:) = circshift([1; zeros(y_res-1,1)],-floor(f_help(i)))';
end

% Do a fft to obtain the filter in the frequency domain
H = fft(h,y_res,2);
```

A.5. MATLAB code - Optical algorithm

```
function [ result ] = optical_algorithm( received, x_rec, Fs, y_min, y_max, beam_angle)
% This function applies the optical algorithm to convert a set of
% measurements into an image.
%
% Inputs:
%   received:   The matrix with measured data.
%   x_rec:      The vector with the x-coordinates of the measurements
%   Fs:         The sample rate
%   y_min:      The start of the scene in the y-direction
%   r_max:      The end of the scene in the y-direction
%   beam_angle: The angle that represents the beamwidth of the antenna
%
% Output:
%   result:     The generated image
%
% Date: 15-06-2017

% Initialization
c = 3e8; % speed of light
delta_r = c/Fs; % Step size

% Size of the matrix with measurements
[x_res, y_res] = size(received);

% Create vector with focus distances
n_focus = 1;
focus_space = (y_max-y_min)/n_focus;
r_focus = linspace(y_min+focus_space/2, y_max-focus_space/2, n_focus);

```

```
% Create one matched filter to determine the required window size:
[~, window_pixel, ~] = range_migration(x_res, max(x_rec)-min(x_rec),...
        y_res, y_min, y_max, beam_angle, r_focus(1));

% Take the fft of the received data and add zeros to both sides:
rec_fft = fft(received,y_res,2);
rec_zero_padded = [zeros(floor(window_pixel/2), y_res); rec_fft; zeros(floor(
    window_pixel/2), y_res)];

% Initialize the final result and different window steps
result = zeros(x_res, y_res);
window_ffty = zeros(window_pixel,y_res);
window_range_migrated = window_ffty;
window_fftx = window_ffty;
window_range_migrated_focus = zeros(window_pixel, y_res);

% Calculate all the matched filters for all the different r_focus
for j = 1:n_focus
    [H(j,:,:), window_pixel, h] = range_migration(x_res, max(x_rec)-min(x_rec),...
        y_res, y_min, y_max, beam_angle, r_focus(j));%-1.7);
end

% For a certain xi, take the desired window, then create the range
% migrated version for a certain r_focus. Finally, add the right parts of
% the different range migrated windows to create the final windows. Use
% these windows to create the final result.
for i = 1:x_res
    for j = 1:n_focus
        % Window of the data:
        window_ffty = rec_zero_padded(i:(i+window_pixel-1),:);
        % Range migrated version of this window for r_focus(j):
        window_range_migrated_focus = ifft(window_ffty.*reshape(H(j,:,:),window_pixel,
    y_res),[],2);

        % Add the right parts of different windows that are differently
        % focussed.
        selection_vector = (floor((j-1)*focus_space*y_res/(y_max-y_min))+1):floor(j*
    focus_space*y_res/(y_max-y_min));

        % Update the final result:
        result(i,selection_vector) = sum(window_range_migrated_focus(:,selection_vector
    ));
    end
end

end
```

A.6. MATLAB code - Freepath backprojection

```
function [ result ] = backprojection_freepath( result, received, x_meas, y_meas,
    theta_meas, Fs, x_min, x_max, y_min, y_max, beam_angle, r_begin, resolution)
% This function applies the backprojection algorithm to a single
% measurement to obtain a part of the final result. Only one iteration
% is done, because this makes it easy to apply the algorithm while doing
% the measurements. The inputs and outputs are as follows:
%
% Inputs:
%   result: The result of the previous iteration. The new information will
%           be added to this old result to create the new result.
%   received: The data of the measurement.
%   x_meas: The x-coordinates of the measurement positions.
%   y_meas: The y-coordinates of the measurement positions.
%   theta_meas: The direction of the antenna for the different
%           measurements.
%   Fs: The rate at which the received signal was sampled.
%   x_min: The x-value at which the scene of interest starts.
%   x_max: The x-value at which the scene of interest ends.
%   y_min: The y-value at which the scene of interest starts.
%   y_max: The y-value at which the scene of interest ends.
%   beam_angle: Half of the entire beam angle.
```

```
%   r_begin: The range that corresponds to the first sample of the data.
%   resolution: Spatial resolution of the image in meters
%
% Date: 15-06-2017

% Initialization
c = 3e8; % Speed of light

% Information on the distances corresponding to samples
sample_spacing = c/Fs/2; % The distance between two samples.
r_end = r_begin + sample_spacing*size(received,2); % The distance that corresponds to
    the last sample

% Choose right amount of pixels
x_size = floor((x_max-x_min)/resolution);   % Number of pixels in the x-direction
y_size = floor((y_max-y_min)/resolution);   % Number of pixels in the y-direction

% Vectors with the x- and y- coordinates that correspond to the samples
x_pixel = linspace(x_min,x_max,x_size);
y_pixel = linspace(y_min,y_max,y_size);

% Angle at which the beam of the antenna starts with respect to the
% horizontal
theta_min = wrapToPi(theta_meas - beam_angle);

parfor xx=1:x_size % Iterate along all the pixels
    for yy=1:y_size
        % Check whether a pixel is within the beam of the antenna
        theta_pixel = atan2(y_pixel(yy)-y_meas,x_pixel(xx)-x_meas); % Angle towards
    pixel
        if(wrapTo2Pi(theta_pixel - theta_min) < 2*beam_angle) % If within beam

            % Check whether a pixel is within the range of the antenna
            distance_from_measurement = sqrt((x_pixel(xx)-x_meas)^2+(y_pixel(yy)-y_meas
    )^2); % Distance towards pixel
            if (distance_from_measurement < r_end) % If within range

                % Use interpolation to obtain the correct signal value and
                % add this to the result.
                sample_actual = ((distance_from_measurement - r_begin)/sample_spacing);
                sample_actual_floor = floor(sample_actual);
                sample_actual_mod = mod(sample_actual,1);
                result(xx,yy) = result(xx,yy) +...
                    (1-sample_actual_mod)*received(sample_actual_floor)+...
                    sample_actual_mod*received(sample_actual_floor+1);
            end
        end
    end
end
```

B

Appendix - Fulfillment of the requirements

The program of requirements was given in chapter 2. In this appendix will be returned to this program to check whether the requirements have been fulfilled. First the global requirements for the entire project will be checked and then the requirements for the image formation subgroup.

B.1. Global requirements for entire project

These requirements were achieved by the group as whole. Consequently, some parts are described more in detail in the corresponding thesis.

Functional requirements:

[1.1] This requirement has been fulfilled by the Data Acquisition group (see [27]).

[1.2] This requirement has been fulfilled. The used method as well as such images are shown in this thesis.

[1.3] This requirement can be achieved theoretically, but has not been tested thoroughly. The longest successful (indoor) detection was ≈10 m.

[1.4] The imaging algorithms are able to handle larger datasets. However, it takes longer to produce an image if larger datasets are provided.

[1.5] Smaller objects have been successfully detected with sufficient accuracy. The cans that were used for the majority of the measurements are adequate replacements for lampposts and suchlike. Objects smaller than that are uncommon in road environments.

Implementation requirements:

[2.1] This global requirement can be achieved theoretically, but has not been implemented yet.

[2.2] The freepath algorithm can be used to process the data while the vehicle is moving, however, further optimization of hardware and/or software is necessary to eliminate any delay at high speeds.

Representation requirements:

[3.1] This requirement has been partially fulfilled. The generated images clearly show where objects are located whereas these images have not been tested in a form of comparison with larger datasets.

Safety requirements:

[4.1] This requirement has been fulfilled by the Data Acquisition group (see [27]).

B.2. Requirements for image formation subgroup

The requirements for the imaging group were given in section 2.2. The degree to which they have been fulfilled is described in the following list:

Functional requirements:

[A.1] This requirement is fulfilled. For stripmap-SAR three different algorithms have been implemented that do this: The sum-and-delay algorithm, the backprojection algorithm and the optical algorithm. The backprojection algorithm has also been extended to variable-angle SAR and free-path SAR.

[A.2] The algorithm is able to achieve this resolution. This constraint now lies with the bandwidth of the transmitted electromagnetic pulse and not with the algorithm.

[A.3] The algorithm is able to achieve this resolution. To improve this further the spacing between consecutive measurements should be decreased. The constraint no longer lies with the algorithm.

Implementation requirements:

[B.1] The sum-and-delay algorithm can only be started after all the measurements have been done. The backprojection algorithm and optical algorithm can be implemented in a way that a part of the required operations can be done during the measurements, which makes the time period available to apply the algorithm larger. For this reason, these methods impose smaller computational constraints. Also see the closely related next point.

[B.2] The sum-and-delay algorithm has a large brute-force element and is not very scalable. The backprojection algorithm has the same problem, even though it has some options for parallelization. The optical algorithm seems to be more suitable for scaling to larger planes.

[B.3] This requirement has been fulfilled.

Representation requirements:

[C.1] This requirement has been fulfilled.

[C.2] This requirement has been fulfilled.

Bibliography

[1] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, *A tutorial on synthetic aperture radar,* IEEE Geoscience and remote sensing magazine **1**, 6 (2013).

[2] J. P. Fitch, *Synthetic Aperture Radar* (Springer Science & Business Media, 2012).

[3] C. Oliver and S. Quegan, *Understanding Synthetic Aperture Radar Images* (SciTech Publishing, 2004).

[4] C. V. Jakowatz, D. E. Wahl, P. H. Eichel, D. C. Ghiglia, and P. A. Thompson, *Spotlight-Mode Synthetic Aperture Radar: A Signal Processing Approach* (Springer Science & Business Media, 2012).

[5] L. Cutrona, E. N. Leith, L. Porcello, and W. Vivian, *On the Application of Coherent Optical Processing Techniques to Synthetic-Aperture Radar,* Proceedings of the IEEE **54**, 1026 (1966).

[6] K. Tomiyasu, *Tutorial review of synthetic-aperture radar (SAR) with applications to imaging of the ocean surface,* Proceedings of the IEEE **66**, 563 (1978).

[7] D. A. Ausherman, *Digital versus optical techniques in Synthetic Aperture Radar (SAR) data processing,* Optical Engineering **19**, 157 (1980).

[8] J. C. Kirk, *A Discussion of Digital Processing in Synthetic Aperture Radar,* IEEE Transactions on aerospace and electronic systems , 326 (1975).

[9] I. G. Cumming and F. H. Wong, *Digital processing of synthetic aperture radar data,* Artech house **1**, 3 (2005).

[10] S. Ramakrishnan, V. Demarcus, J. Le Ny, N. Patwari, and J. Gussy, *Synthetic aperture radar imaging using spectral estimation techniques,* (2002).

[11] D. L. Donoho, *Compressed sensing,* IEEE Transactions on information theory **52**, 1289 (2006).

[12] X. Dong and Y. Zhang, *A Novel Compressive Sensing Algorithm for SAR Imaging,* IEEE Journal of selected topics in applied earth observations and remote sensing **7**, 708 (2014).

[13] J. Fang, Z. Xu, B. Zhang, W. Hong, and Y. Wu, *Fast Compressed Sensing SAR Imaging Based on Approximated Observation,* IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **7**, 352 (2014).

[14] J. M. Lopez-Sanchez and J. Fortuny-Guasch, *3-D Radar Imaging Using Range Migration Techniques,* IEEE Transactions on antennas and propagation **48**, 728 (2000).

[15] O. Ponce, P. Prats-Iraola, M. Pinheiro, M. Rodriguez-Cassola, R. Scheiber, A. Reigber, and A. Moreira, *Fully Polarimetric High-Resolution 3-D Imaging With Circular SAR at L-Band,* IEEE Transactions on Geoscience and Remote Sensing **52**, 3074 (2014).

[16] D. Reale, G. Fornaro, A. Pauciullo, X. Zhu, and R. Bamler, *Tomographic Imaging and Monitoring of Buildings With Very High Resolution SAR Data,* IEEE Geoscience and remote sensing letters **8**, 661 (2011).

[17] S. Vitebskiy, L. Carin, M. A. Ressler, and F. H. Le, *Ultra-Wideband, Short-Pulse Ground-Penetrating Radar: Simulation and Measurement,* IEEE Transactions on Geoscience and Remote Sensing **35**, 762 (1997).

[18] D. Oloumi, J.-W. Ting, and K. Rambabu, *Design of Pulse Characteristics for Near-Field UWB-SAR Imaging,* IEEE Transactions on Microwave Theory and Techniques **64**, 2684 (2016).

[19] C. Le, T. Dogaru, L. Nguyen, and M. A. Ressler, *Ultrawideband (UWB) Radar Imaging of Building Interior: Measurements and Predictions,* IEEE Transactions on Geoscience and Remote Sensing **47**, 1409 (2009).

[20] P. Setlur, M. Amin, and F. Ahmad, *Multipath Model and Exploitation in Through-the-wall and Urban Radar Sensing,* IEEE Transactions on Geoscience and Remote Sensing **49**, 4021 (2011).

[21] T. Jin, B. Chen, and Z. Zhou, *Image-Domain Estimation of Wall Parameters for Autofocusing of Through-the-Wall SAR Imagery,* IEEE Transactions on Geoscience and Remote Sensing **51**, 1836 (2013).

[22] E. C. Fear, S. C. Hagness, P. M. Meaney, M. Okoniewski, and M. A. Stuchly, *Enhancing Breast tumor Detection with Near-Field Imaging,* IEEE Microwave magazine **3**, 48 (2002).

[23] C. Lundquist, L. Hammarstrand, and F. Gustafsson, *Road Intensity Based Mapping Using Radar Measurements With a Probability Hypothesis Density Filter,* IEEE Transactions on Signal Processing **59**, 1397 (2011).

[24] M. Lundgren, E. Stenborg, L. Svensson, and L. Hammarstrand, *Vehicle Self-localization Using Off-the-shelf Sensors and a Detailed Map,* in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE* (IEEE, 2014) pp. 522–528.

[25] S. Liu, Z. Cao, H. Wu, Y. Pi, and H. Yang, *Target detection in complex scene of sar image based on existence probability,* EURASIP Journal on Advances in Signal Processing **2016**, 114 (2016).

[26] L. de Gelder and R. Koch, *Ultra Wideband Synthetic Aperture Radar Imaging: Radar Movement*, Bachelor thesis, Delft University of Technology (2017).

[27] R. Arriëns and T. Wieffering, *Ultra Wideband Synthetic Aperture Radar Imaging: Data Aquisition and Antenna Analyses*, Bachelor thesis, Delft University of Technology (2017).

[28] E. Leith, *Complex spatial filters for image deconvolution,* Proceedings of the IEEE **65**, 18 (1977).