NTNU – Trondheim
Norwegian University of
Science and Technology

TFE4930
MASTERS THESIS

GPU Implementation of Aliasing-Resistant Blood Flow Estimation Using Doppler Ultrasound

Emil Braserud

June 21, 2019

Abstract

In this thesis, a new method for aliasing correction of Doppler ultrasound measurements is described, and a pipeline from beamforming acquired data until blood flow estimation on a GPU/Tensorflow platform is presented. The method uses Doppler autocorrelation and cross correlation to produce velocity estimates for multiple unique transmit-receive pairs. The different estimates are then combined using a least squares method in order to produce angle-independent aliasing-resistant velocity estimates, implemented in the Tensorflow framework. The method is verified through comparison with a reference dealiasing method. The results indicate that the method produces qualitatively accurate and robust velocity estimates. Further analysis of the method is needed in order to quantitatively verify the method, but the results shown look promising for the future use of GPU-based processing for ultrasound Doppler applications.

Acknowledgements

I would first like to thank my supervisor Lasse Løvstakken and all the people at the NTNU ultrasound group. I was lucky I got to spend my last semester in such a great environment. A special thanks goes to Jørgen Avdal and Thomas Grønli for all the help you have given me during my thesis, it's hard to express how much I appreciate it.

I would also like to thank my family for all the support and encouragement you have given me during my years as a student. Finally, thank you Malin for all the the hours you listened to my rambling about my thesis, and for always giving me perspective when things seemed impossible.

Contents

Abstract i

Preface ii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Aim of Study . . . 2
- 1.3 Outline of report . . . 2

2 Background 3
- 2.1 Ischaemic strokes . . . 3
- 2.2 Diagnostic Ultrasound . . . 4
  - 2.2.1 Sound Propagation . . . 4
  - 2.2.2 Speckle . . . 5
  - 2.2.3 Transducer . . . 6
  - 2.2.4 Doppler Effect . . . 6
  - 2.2.5 Pulsed-wave Doppler . . . 7
- 2.3 Beamforming . . . 7
  - 2.3.1 Focusing . . . 8
  - 2.3.2 Delay and Sum Beamforming . . . 8
  - 2.3.3 Block Matching . . . 9
- 2.4 Vector Velocity Estimation . . . 10
- 2.5 Graphical Processing Units . . . 10
  - 2.5.1 Compute Unified Device Architecture . . . 12
  - 2.5.2 Tensorflow . . . 12

3 Theory 13
- 3.1 Least-squares Regression . . . 13
- 3.2 Doppler Autocorrelation estimator . . . 13

4 Methodology 15
- 4.1 Blood Flow Estimation Using Plane-Wave Ultrasound Imaging . . . 15
  - 4.1.1 Least Squares Regression . . . 15
  - 4.1.2 Block Matching . . . 16
- 4.2 GPU based 2D blood flow estimation . . . 17
  - 4.2.1 Beamforming . . . 17

4.2.2 Dealiasing using cross-correlation . . . 18
4.2.3 Tensorflow Least-squares Regression . . . 18
4.3 Experimental setup . . . 20

5 Results **21**
5.1 Beamforming results . . . 21
5.2 Aliasing correction results . . . 22
5.3 Velocity estimates results . . . 23
5.3.1 Data-set 1 result . . . 23
5.3.2 Data-set 2 result . . . 24
5.4 Computational time . . . 25

6 Discussion **26**
6.1 Beamforming . . . 26
6.2 Anti Aliasing . . . 26
6.2.1 Final Velocity estimates . . . 28
6.2.2 Computational Cost . . . 28
6.3 Weaknesses and future work . . . 29

7 Conclusion **30**

1 Introduction

1.1 Motivation

Cardiovascular disease is the world's leading cause of death, and this will likely not change in the near future due to a globally aging population and lifestyle changes in developing countries. The reason for the high death toll can be contributed to several factors, one being the difficulty in detecting and thus preventing cardiovascular diseases. One of the most important tools for detection of cardiovascular diseases is ultrasound technology. The discovery of using ultrasound technology for medical applications with a focus on cardiovascular diseases dates back to the late 1940s, when a few scientists recognized the potential of ultrasonic energy to produce information that can be used for medical diagnosis(1). When the early enthusiasts showed the potential for the field the amount of resources and effort put into research increased rapidly, and in the early 1950's, the first medical uses were implemented. The use of ultrasound imaging for cardiovascular diseases has expanded since, and today ultrasound is one of the most commonly used diagnostic tools.

One area where diagnostic ultrasound imaging is widely used, is assessment of carotid artery stenosis. Stenoses in the carotid artieries may cause a cerebral infarction due to shredding of emboli. These emboli will follow the blood stream to smaller arteries in the brain, where they may cause a stroke by blocking the blood flow. In order to detect stenosis, the velocity of the blood is measured, and the risk for brain infraction can be assessed based on the resulting blood flow estimates. The specific technique most commonly used for this area, is Doppler ultrasound.

Although greatly successful, Doppler ultrasound has one significant limitation, as it is one-dimensional. True blood flow is three-dimensional. However, with Doppler ultrasound, only the velocity component in the direction of the ultrasound beam can be measured. As a result of this, the measured velocity is dependent on the beam-to-flow angle. To obtain an estimate of the true velocity, it is often assumed that the blood flow is parallel to the vessel axis, and angle correction of the 1D velocity estimate is performed. As blood flow through stenoses may not be parallel to the vessel, assumptions on the flow angle may cause considerable spread in velocity measurements and large interobserver variability, and a definite conclusion about the state of the cardiovascular health is harder to achieve(2).

1.2 Aim of Study

One way to address the limitations of 1D velocity estimation, is to use multiple angle measurements to produce 2D velocity estimates. A 2D velocity estimate can be used to visualize complex blood flow, which will yield more robust and conclusive results. One of the main drawbacks of multiple angle measurements is the increased processing time, as the same computations need to be performed for all angles. The goals for this project will be to implement a pipeline from beamforming plane wave ultrasound data, until producing final 2D velocity estimates on a GPU based platform. The method's robustness and accuracy will then be analyzed through a comparison with another already verified method. Program run-time will also be presented, as processing data in close to real-time is desired for a final product.

1.3 Outline of report

Chapter 2 covers background information about ischaemic stroke, diagnostic ultrasound, and presents the Graphical Processing Unit. Chapter 3 covers the theory aspect of the underlying methods the implementation. Chapter 4 presents the method, and steps needed for replication of the presented results. Results are presented in Chapter 5 and discussed in Chapter 6. Chapter 7 concludes the report.

2 Background

This chapter gives the background information needed to understand the implemented method, the issues it faces, and the problems it solves. The first section covers ischaemic stroke, which is the clinical motivation for this thesis. Then, it presents areas of diagnostic ultrasound used in the implemented methods, and finally a description of the framework used to implement the method.

2.1 Ischaemic strokes

Cardiovascular disease (CVD) is the most frequent cause of premature death in modern countries, and was responsible for 17.8 million deaths world wide in 2017(3). Stroke is included as a sub-category within cardiovascular diseases, and is estimated to cause slightly over 6 million deaths yearly. 87% of the strokes that occur are Ischaemic strokes, which occurs due to a restriction on blood flow to the brain. The underlying cause for Ischaemic stroke is **Atherosclerosis**(4), which in effect blocks the blood vessels in the neck. The danger with this, however, is that the atherosclerosis plaque may rupture and cause blood clots which can occlude the vessel where the plaque is located, or occlude smaller vessels downstream.

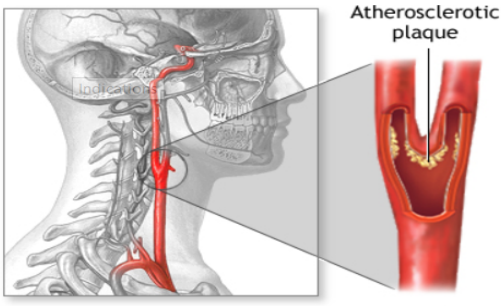

Figure 1: Atherosclerosis (5)

Figure 1 illustrates atherosclerosis in the carotid artery. Plaques builds up inside the artery, reducing blood flow. If the plaque ruptures, it can create a blood clot, which in turn can lead to a stroke.

2.2 Diagnostic Ultrasound

Ultrasound is the term used to describe sound with frequencies above 20 000 Hertz(Hz), which is beyond the range of human hearing(6). Image formation using ultrasound is possible because of the difference in compressibility and density between materials in the body, causing reflection and scattering of ultrasound beams. The properties of a medium can be described by a given density ρ, and compressibility k(7). The equation illustrating pressure wave propagation can be derived by considering the conservation of mass and momentum. Assuming a homogeneous medium, and linear propagation where the displacement of scattering volumes is proportional to the change in pressure, the basic equation governing the propagation of a pressure wave $\rho(r,t)$ is given by:

$$\nabla^2\rho(r,t) - \frac{1}{c^2}\frac{\delta^2\rho(r,t)}{\delta t^2} - 0, \tag{1}$$

where r is a spatial position vector, t is time and c = $\frac{1}{\sqrt{\rho k}}$ is the speed of sound in the medium. The average speed of sound in human tissue is about 1540 m/s. The ultrasonic waves are attenuated as they travel through tissue, due to scattering losses, power absoption and geometric spreading of the ultrasound beam. Attenuation is frequency dependent, so high resolution results may be obtained using higher frequencies, but higher frequencies also lead to more attenuation. More attenuation leads to smaller penetration depth, and for this reason, different frequency ranges are used for examination of different parts of the body.

2.2.1 Sound Propagation

When sound propagates, it often interacts with different kinds of tissue. If the interaction is with objects that are large compared to the wavelength, there are two different outcomes, either reflection or transmission(8). A part of the incident energy (I_i) is reflected (I_r) while the rest is transmitted (I_t). The reflection coefficient α R is used to measure the reflection between two adjacent tissues, with different impedances, Z_1 and Z_2 (where $Z = \rho c$):

$$\alpha R = \frac{I_r}{I_t} = \frac{(\frac{Z_2}{cos\theta_t} - \frac{Z_1}{cos\theta_i})^2}{(\frac{Z_2}{cos\theta_t} + \frac{Z_1}{cos\theta_i})^2} \tag{2}$$

The sound wave that is not reflected, will be transmitted into the medium. The transmission coefficient αT is expressed as:

$$\alpha T = \frac{I_t}{I_i} = 1 - \alpha R \tag{3}$$

The equations governing the angles of incidence, θ_i, reflection θ_r and transmission θ_t are the following:

$$\theta_i = \theta_r, \frac{sin\theta_i}{sin\theta_t} = \frac{c_1}{c_2} \tag{4}$$

where c_1 and c_2 are the sound velocities in medium or tissue 1, respectively medium 2. In the case when $\theta_i = \theta_t = 0$, we can rewrite 2 depending on just the ration between Z_1 and Z_2:

$$\alpha R = \frac{I_r}{I_i} = \frac{(Z_2 - Z_1)^2}{(Z_2 + Z_1)^2} \tag{5}$$

In biological tissues, the medium is very rarely smooth, which will lead to the phenomenon of **scattering**. Scattering also occurs when the dimension of the target is negligible compared to the wavelength (e.g. blood cells). The scattering phenomenon can be seen in 2.

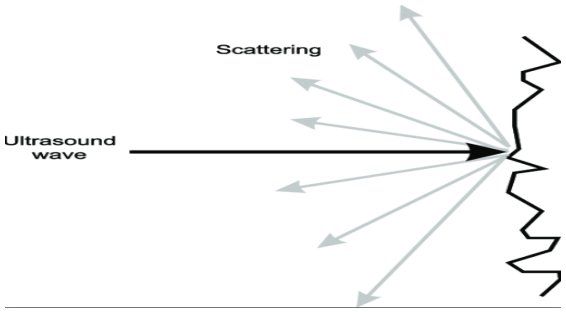

Figure 2: (9).

2.2.2 Speckle

A result of the scattering process (as described in 2.1.1) is the **speckle** pattern. Speckle patterns are the result of constructive and destructive interference of ultrasound backscattered from structures that are small compared to the wavelength. The speckle pattern that occurs in an image is random, but deterministic interference(10). The texture of the observed speckle pattern does not correspond to underlying structure, and it has a negative impact on ultrasound imaging quality. However, because of the deterministic nature of the speckle pattern, it follows the movement of underlying tissue in the image. Different speckle tracking methods are based on this attribute of speckle patterns, and they offer the ability to identify and track the same speckle occurring in multiple images. There are different algorithms used by different vendors for tracking speckle. Speckle tracking methods can be based on different techniques, e.g., block matching, differential based optical flow algorithms or conservation of gray value, i.e., it is assumed that gray values do not change over time(11).

2.2.3 Transducer

The transducer is responsible for the transmission and reception of ultrasonic pressure waves. A transducer typically consists of an array of piezoelectric elements(7). The piezoelectric elements vibrate in response to an external electric field, creating and transmitting ultrasonic waves. The backscattered pressure field is received by the same transducer elements. On receive, the piezoelectric elements vibrate in response to an external pressure, producing an electrical signal. Modern transducers come with controllable pulse emission timing and array element apodization, and allows for flexible beam shaping and electronic focusing and steering of the beam.

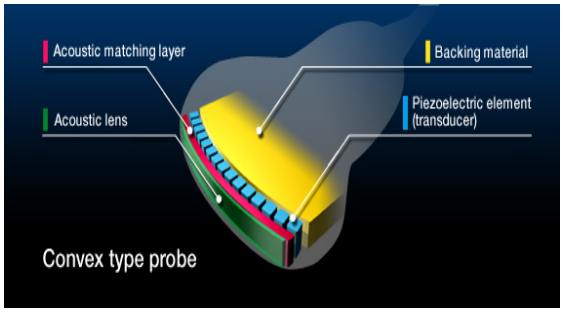

Figure 3: Illustration of a convex type transducer, highlighting the main elements(12).

2.2.4 Doppler Effect

When a transmitted ultrasound wave is scattered by moving objects in the body, the wave will experience a shift in frequency, and this phenomenon is called **the Doppler effect**(6). The Doppler effect was first presented by Christian Doppler in relation to the colors of double stars. The effect is used to explain changes in the frequency of waves emitted by moving objects as detected by a non-moving observer. The perceived frequency is lower if the object is moving away from the observer, and higher if the object is moving towards the observer. In ultrasound pulse-echo imaging, both of these cases occur. The scaling of the temporal axis can be given by:

$$\alpha = \frac{c + v\cos\theta}{c - v\cos\theta} \approx (1 + \frac{2v\cos\theta}{c}). \tag{6}$$

where θ is the angle between the ultrasound beam direction and the scatter velocity vector, and $v\cos\theta$ is the axial component of the scatter velocity, defined as positive towards the ultrasound transducer. The shift in frequency is then given by:

$$f_d = \alpha f_0 - f_0 = 2f_0\frac{vcos\theta}{c}. \tag{7}$$

where f_d is the doppler shift, and f_0 is the emitted frequency. The equation is valid as long as $v\cos\theta \gg c$. For blood the received signal from an insonified sample volume

is a sum of contributions from a large number of scatterers, each producing a Doppler shift according to their given velocity and direction(7). The received signal is therefore made up of a spectrum of different velocities. Further, as each scatterer is observed in a finite time interval, the estimated velocity spectra will broaden. This is termed the transit time effect.
Doppler imaging has many applications within diagnostic ultrasound, but is mainly used to image blood flow, and to measure movement of the cardiac muscle. The Doppler imaging techniques work by forming information regarding the location of each target in the body, corresponding to each pixel in the image, and analyzing the returning echoes in terms of Doppler shift or phase shift.

2.2.5 Pulsed-wave Doppler

In pulsed-wave Doppler (PW-Doppler), a series of pulses are emitted into the tissue at a constant pulse repetition frequency (PRF), phase-coherent with respect to the transmission carrier frequency f_0, and range-gated on receive to achieve range resolution(7). The received signal is sampled after a predetermined delay τ, given by the round-trip time for the echo from scatterers in distance z; $\tau = \frac{2}{c_0}z$, where c_0 is the velocity of ultrasound in blood(13). The echo from each blood scatterer is an attenuated copy of the transmitted pulse, and will contribute to the sampled receive signal when the round-trip time is in the range $T \pm \frac{1}{2}T_p$. The axial measurement size region is given by the transmitted pulse length according to the equation:

$$\Delta z = \frac{c_0}{2}T_p \tag{8}$$

In order to avoid interfering echoes from previous pulses, the time interval between pulse transmissions must be larger than the round-trip time for the echos from the depth range z of interest. This means that the PRF must be limited to:

$$PRF < \frac{c_0}{2z} \tag{9}$$

The maximum measurable velocity is called the Nyquist velocity, and is inversely proporsional with depth range z and ultrasound frequency f_0. The maximum measurable velocity for PW-doppler is called the Nyquist velocity. The Nyquist velocity can be found with the equation:

$$vNyquist = \frac{cPRF}{4f_0} \tag{10}$$

If the velocity exceeds the Nyquist limit, aliasing in the spectrum will occur.

2.3 Beamforming

The beamformer is the primary driver for image formation in an ultrasound system. Beamforming enables the selectivity of acoustic signals reflected from some known po-

sitions, while attenuating the signals from other positions. This is often done by delaying (focusing) and applying specific weights to the reflected signals.

2.3.1 Focusing

During transmission, delays are applied to ensure that the contributions coming from all transducer elements are focused in a given point, called focal point in emission. During reception, the echoes received by the elements are delayed such that the sum of contributions are coming from the same given point in the medium(8).

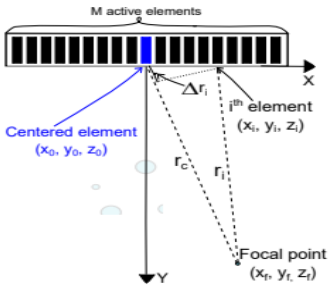

Figure 4: Beamforming Focusing example(8).

An example that shows how transmit focusing works is shown in 4. The probe contains M active elements with a central element highlighted in blue. The distance from the i-th element to the focal point is r_i. The distance from the central element to the focal point is r_c. The delay that is applied to the signal emitted by the i-th elements is Δr_i.
The received signals are also delayed during image formation, as described in section 2.3.2.

2.3.2 Delay and Sum Beamforming

Depending on the calculation of the weights applied to the output array of the reflected signals, beamformers can either be data-independent (fixed) or data-dependent (adaptive). The Delay-And-Sum is an data-independent beamforming method(8).

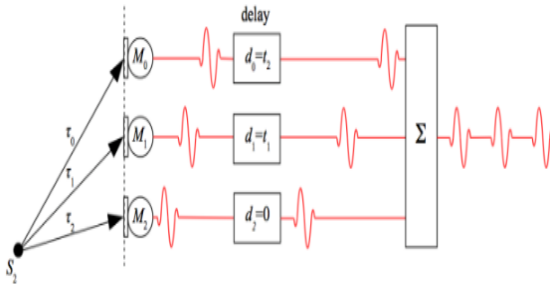

Figure 5: Delay and Sum illustration(14).

The Delay-And-Sum method is illustrated in 5. The echoes received by the elements of the ultrasound probe, also called channel data, are delayed in order to compensate for the time-of-flight differences. The signals are also weighted (using a weighting function, also called apodization function) and further summed to form one beamformed signal.

2.3.3 Block Matching

Motion estimation using block matching takes advantage of the fact that successive image frames are often highly correlated(15). In the block matching motion estimation process, each frame is divided into overlapping blocks of size n*n. Then, for the largest motion displacement of 'p' pixels per frame, the current block is matched with a corresponding block in the previous frame with many sets of displaced coordinates. The highest correlation value yields the estimated displacement.

Figure 6: (16).

When the value of n increases, the number of total blocks that needs to be processed in each frame will decrease, therefore, the computational complexity decreases as well. However, finding a valid match for the given block is difficult with a large n, and the

probability of not finding a match in a corresponding frame increases with n.
The most straightforward block matching approach is the *full search* methods, which exhaustively searches for the best matching block within the search window. The disadvantage of this approach is the high associated computational load, which often makes it the bottleneck for real-time applications. Another approach is the *fast search* methods. The fast search methods exploit certain properties of visual communication applications, and eliminates areas in the search window based on different assumptions. An example of such a property can be that there is little motion between the adjacent frames in a real-time application, and therefore, a large percentage of zero-motion blocks are encountered. The number of static blocks per frame could be easily as high as 70%, which can eliminate much of the computational cost.

2.4 Vector Velocity Estimation

2D vector velocity imaging has the potential to visualize complex blood flow patterns for medical applications. The method should also provide accurate, angle-independent velocity estimates and cover a large velocity span. Maniatis et al. (17) shows in his study that there can be benefits for having multiple angle measurements where the true flow velocity is unknown, and that the dominant factor determining the velocity estimation accuracy is the angle between the observations.

Vector doppler imaging is based on the autocorrelation technique, which is limited by the PRF, which sets the limit for the velocity span that can be measured. If the velocity exceeds this limit, aliasing can occur. The maximum PRF is ultimately determined by the imaging depth, and a compromise between the measurable velocity span and the number of transmit beams that can be used to generate the velocity vectors must be made. Multiple angle transmit measurements can increase the accuracy of the velocity estimate, but on the cost of a smaller measurable velocity span due to the reduced PRF.

2.5 Graphical Processing Units

During the past several decades, there has been a major shift from sequential computing to parallel computing. One of the most important methods for improving the performance of consumer computing devices has been to increase the speed at which the processor clock operated. But due to the phenomenon known as Dennard Scaling, this is no longer a feasible method of improving Performance. In the early years of parallel computing, much was limited to exotic supercomputers or mainframes(18). The interest for parallel computing started back in the 1950s, which lead to the advancement of supercomputers throughout the 1960s and 1970s. The early machines were **shared memory** multiprocessors, with multiple processors working side-by-side on shared data. In the late 1980s, clusters came to compete and replaced the current

supercomputers within many domains. A cluster is a type of parallel computer built from large numbers of off-the-shelf computers connected by an off-the-shelf network. Today, parallel computing is becoming mainstream as even the low-end netbook machines, cell phones and a number of microcontrollers use the multicore paradigm.

When the central processors evolved both in clock speeds and core count, there were also made large improvements to the state of of graphics processing. In the late 1980s and early 1990s, the growth in popularity of graphically driven operating systems such as Microsoft Windows helped create a marked for a new type of processor. In the early 1990s, users began purchasing 2D display accelerators for their personal computers. These display accelerators offered hardware-assisted bitmap operations to assist in the display and usability of graphical operating systems. Around the same time, in the world of professional computing, a company by the name of Silicon Graphics spent the 1980s popularizing the use of three-dimensional graphics in a variety of markets, including government and defence applications and scientific and technical visualization, as well as providing the tools to create magnificent cinematic effects. In 1992, Silicon Graphics opened the programming interface to its hardware by releasing the OpenGL library.

By the mid 1990s, the demand for consumer applications employing 3D graphics had escalated rapidly, setting the state for two fairly significant developments. First, the release of first-person games which ignited the quest to create progressivley more realistic 3D environments for PC gaming. Att the same time, companies such as NVIDIA, ATI Technologies and 3dfx Interactive began releasing graphics accelerators that were affordable enough to attract widespread attention. When NVIDIA released their GEForce 256, it further pushed the capabilities of consumer graphics hardware. For the first time, transform and lighting fast computations could be performed directly on the graphics processor, thereby enchanting the potential for even more visually interesting applications. From a parallel computing standpoint, the release of NVIDIA's GeForce 3 series in 2001 might have been the biggest breakthrough in GPU technology. The GeForce 3 series were the first GPUs to implement Microsoft's then-new DirectX 8.0 standard. The standard required that compliant hardware contain both programmable vertex and programmable pixel shading stages. For the first time, developers were able to control the exact computations that would be performed on their GPUs.

Today, the GPU is a massively parallel device dedicated to generating and drawing graphical objects on a screen. It is used by the CPU as a coprocessor, by offloading graphics related processing tasks to the GPU. GPUs are necassary in tasks where the CPU cannot possibly be expected to satisfy all the computational demands related to graphical processing. Such examples are easy to find when it comes to graphical processing, as tasks like real-time video games and live video encoding / decoding needs a lot of parallel computations.

2.5.1 Compute Unified Device Architecture

Compute Unified Device Architecture (CUDA) is a general-purpose parallel compuing platform and programming model which is used to program NVIDIA GPU's in order to solve complex computational problems in a more efficient manner(19). The CUDA platform is accessible through CUDA-accelerated libraries, compiler directives, application programming interfaces and extensions to industry-standard programming languages like C, C++ and Python (as illustrated by 7.

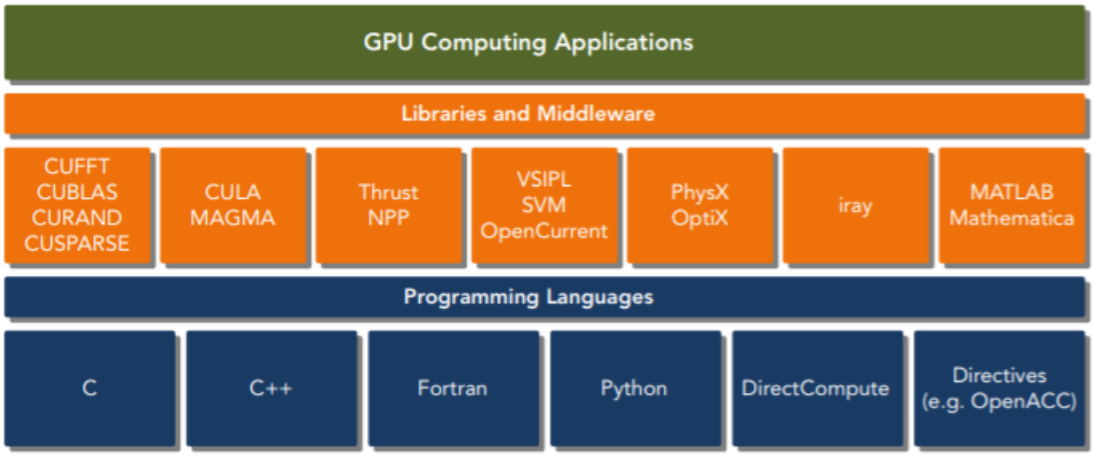

Figure 7: The CUDA platform(19)

2.5.2 Tensorflow

TensorFlow in an open source distributed numerical computation framework released by Google in 2015(20). The main intention is to reduce the amount of effort needed in order to implement a neural network manually, tensorflow instead gives building blocks to put together. The back-end of TensorFlow is implemented using CUDA, which provides low run-time computation of data. Even though TensorFlow originally was intended for neural networks, the framework is suited for other applications as well, such as ultrasound data processing.

3 Theory

3.1 Least-squares Regression

The linear regression model is a statistical method that estimates the linear relationship between two or more variables. This relationship gives the amount of change in one variable, that is associated with change in another variable. The model can also be tested for statistical significance, to predict if the linear relationship could have occurred by chance or not. **Least squares estimation** is a method used to combine all the information to give one solution which is "best" by some criterion. The method uses the criterion that the solution must give the smallest possible sum of squared deviations of the observed b from the estimates provided by the solution. The problem to be solved has the form:

$$\mathbf{Ax} = \mathbf{b} \tag{11}$$

where b is a vector containing observations, x is a vector containing the parameters to be estimated and the matrix A describes the linear relationship between x and b. The least squares estimate of x is then given by:

$$\hat{\mathbf{x}} = \mathbf{A_w b} \tag{12}$$

where:

$$\mathbf{A_w} = (\mathbf{A^T W A})^{-1}\mathbf{A^T W} \tag{13}$$

The least squares residual can then be calculated with:

$$\mathbf{r} = ||W^{\frac{1}{2}}(\mathbf{A}\hat{\mathbf{x}} - \mathbf{b})||_2 \tag{14}$$

$$= ||W^{\frac{1}{2}}(\mathbf{A A_w} - \mathbf{I})\mathbf{b}||_2 \tag{15}$$

Where r is a vector containing the residuals, which are a measure of the ability of the model to explain the observation. A lower residual value means a better fit.

3.2 Doppler Autocorrelation estimator

The Doppler autocorrelation estimator was presented by Namekawa and Kasai in 1980 (21), and is still a standard algorithm used in commercial ultrasound systems. The

autocorrelation approach estimates the three spectral parameters from the slow-time correlation function $R_x(m)$ at lag 0 and 1 as follows:

$$\hat{P} = \hat{R}_x(0),\ \hat{w}_d = \angle\hat{R}_x(1), \hat{B} = \sqrt{1 - \frac{|\hat{R}_x(1)|}{\hat{R}_x(0)}} \tag{16}$$

where $\hat{P}$ is the blood flow signal power indicating the presence of blood flow, $\hat{w}_d$ is the mean frequency of the Doppler spectrum, $\hat{B}$ is the frequency bandwidth of the Doppler spectrum and $\hat{R}_x(m)$ is given by:

$$\hat{R}_x(m) = \frac{1}{N_p - m} \sum_{k=0}^{N_p - m - 1} x(k)^* x(k+m) \tag{17}$$

where * denotes the conjugate operation. The mean axial velocity of blood is further obtained by a scaling factor:

$$\hat{v}_z = \frac{c_0\ \mathrm{PRF}}{4\pi f_0} \angle\hat{R}_x(1) \tag{18}$$

Where c_0 is the speed of sound in blood(1560 m/s), and f_0 is the received pulse center frequency. The main advantage of the autocorrelation estimator compared to other methods, is that it provides valid estimates of the mean velocity over the whole Nyquist range(13).

4 Methodology

In this work we will implement and compare two methods, and validate them using in vivo data. This chapter describes the implementation of the two methods, and presents the experimental setup. Section 4.1 presents an extended least squares vector Doppler method, which is used as a reference. A GPU implementation of the proposed method is presented in section 4.2. Chapter 4.3 describes the validation of the method. The two methods have been compared using two different data-sets, data-set 1 and data-set 2. The comparison will give an indication of the accuracy and robustness of the implemented method.

4.1 Blood Flow Estimation Using Plane-Wave Ultrasound Imaging

The method presented by Ekroll et al(22) uses an extended least squares method for robust, angle-independent 2D vector velocity estimation. A combination of least squares regression of Doppler auto-correlation estimates and block matching method yields aliasing-resistant vector velocity estimates.

The method consists of five main parts, which are plane wave transmissions for acquiring the signals, processing the acquired data with conventional beamforming and Doppler processing, using Least Squares Regression to find the alias patterns and velocity vector candidates, and Block Matching for resolving aliasing ambiguities and determining the true 2-D velocity vector.

4.1.1 Least Squares Regression

The area of interest is insonated from a small number of transmit angles (M = 1 - 5), fired successively in a coherent compounding setup. The Doppler shift is estimated for N different transmit-recieve angle combinations for each resolution cell in the ultrasound image, and each angle corresponds to a unique two-way Doppler angle.

A least-squares regression method is then applied on order to calculate the velocity vector $\mathbf{v} = [vx, vz]$ that corresponds to the best measured Doppler shift. However, aliasing can still be an issue for one or more of the N Doppler frequencies. To account for

aliasing, separate least squares problems are solved for all possible aliasing patterns. As seen in (23), each least squares problem can be written as:

$$kAv_i = \hat{f} + g_i \tag{19}$$

where i iterates though all the aliasing patterns and their corresponding solutions, k is a constant factor converting velocity to normalized frequency. $\hat{f} = [f_0, ..., f_{n-1}] + \epsilon$ is a vector containing normalized Doppler frequency estimates from N different transmit-receive combinations, ϵ is the measurement noise.
The matrix $A = A_t x + A_r x$ has dimensions N X 2 and is the sum of the projection matrices onto the transmit and receive Doppler directions, respectively. The rows of A are given by:

$$a_n = [-\sin\alpha_n - \sin\beta_n, \cos\alpha_n + \cos\beta_n] \tag{20}$$

where α_n and β_n are the steering angles of the angle pair n, on the transmit and receive, respectively. The frequencies estimates in are corrected by the aliasing vectors g_i for all aliasing pattern candidates, and all the elements in these vectors represent a frequency bias of an integer number of PRFs. g_i runs through all the combinations of integers with an absolute value smaller than **L**. L is a parameter dependent on the applications PRF and the maximum that represents the maximum aliasing order to be investigated. The general least squares solutions are as following:

$$v_i = k^{-1} A_w (f + g_i) \tag{21}$$

where:

$$A_w = (A^T W A)^{-1} A^T W \tag{22}$$

The weighting matrix W is typically used to account for differences in the variances of the autocorrelation estimates. For each solution v_i of (1), (3) can be used to calculate the least squares residual:

$$r_i = ||W^{\frac{1}{2}}(kAv_i - (\hat{f} + g_i))||_2 \tag{23}$$

$$= ||W^{\frac{1}{2}}(AAw - I)(\hat{f} + g_i)||_2 \tag{24}$$

The solution v_m with the smallest residual r_m is selected as the dealiased solution.

4.1.2 Block Matching

One significant drawback with the least squares method is that it cannot reliably distinguish between candidate vector solutions v_i and v_j if the difference between their respective residuals r_i and r_j is small compared with the norm of the measurement noise ϵ. Also, the triangle inequality states that for any triangle, the sum of the lengths of any two sides must be greater than or equal to the length of the remaining side. Thus, the difference between r and r_j has an upper bound given by:

$$|r_j - r_i| \leq B_{ij} \tag{25}$$

where:

$$B_{ij} = ||W^{\frac{1}{2}}(AAw - I)(g_j - g_i)||_2 \tag{26}$$

It follows from (6) that a small value of Bij increases the probability of selecting g_j in pixels where the correct aliasing pattern is g_i, and vice versa. Thus, the solutions v_i and v_j will be indistinguishable if B_{ij} is small. In practice, when $cos\alpha_n \approx 1$ for all steering angles, B_{ij} is very small(<0.005) if:

$$g_j = g_i + [l, l, ..., l] \tag{27}$$

where l is an integer and $|l| \leq L$.. When $B_{ij} \approx 0$, the method will select g_i and g_j with equal probability. As a consequence, any solution v_i actually belongs to a set S_i up to 2L + 1 vectors that are in practice equivalent using the least squares approach, where S_i is the set of candidate vector solutions $v_{i,l}$ to (5) with aliasing patterns $g_{i,l}$ given by:

$$g_{i,l} = g_i + [l, l, ..., l], max|g_{i,l}| \leq L. \tag{28}$$

The ambiguities were solved using block matching with 2L + 1 candidate displacement vectors. The candidate with the smallest residual was selected as the dealiased estimate. Block matching is performed as described in 2.3.4.

4.2 GPU based 2D blood flow estimation

The second implemented method is based on the work presented in (22), and it uses the same mathematical concepts in order to produce a 2D velocity estimate. One major difference between the methods is the platform in which it is implemented on. The new method is implemented on a GPU-based platform, and all major calculations are performed on a GPU. The programming language used in order to interface with the GPU is Python. In general, the method consists of 3 main parts, which is beamforming of acquired data, cross correlation in order to eliminate aliasing, and a machine learning based least-squares regression using tensorflow.

4.2.1 Beamforming

The acquired data are beamformed using a DAS beamformer, as described in section 2.3.2. The beamforming is performed using a skewed grid as shown in figure 8. This was done in order to use a feature of the speckle tracking library that performed axial cross-correlation.

The beamformed data is then filtered in order to suppress the clutter signals from stationary and slowly moving tissue.

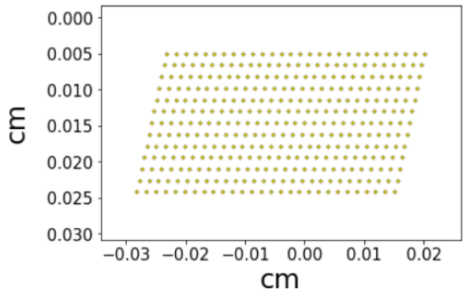

Figure 8: Skewed grid illustration of data-set 2

4.2.2 Dealiasing using cross-correlation

The block-matching step first calculates the blood velocity using the autocorrelation method as described in section 3.2. Then, it calculates the cross correlation between beams in consecutive frames in the image. For each transmit-receive pair, only axial displacements matching the Doppler shift or its aliased copies are evaluated. The displacement yielding the highest correlation is selected as the dealiased candidate.

4.2.3 Tensorflow Least-squares Regression

The least-squares step uses a weighted root mean squared error method on the dealiased velocity component estimates in order to produce one final velocity estimate. First, a new grid C is created, and filled with random numbers ≈ 0. The following error function is calculated for each image point:

$$error = w(\mathbf{I}(x) \cdot \mathbf{b} - V_d)^2 \tag{29}$$

where I(x) is the velocity estimate interpolated to position x, Vd is the velocity estimate, b is a unit vector in the beam direction and w is the weight for the given point. The weight w is proportional to the signal power in each pixel after clutter filtering, as shown in section 3.2. The error is then minimized by adapting C to a value that best fits the data in the surrounding area.

Because of the skewed grid, the velocity estimates for the different angles are on different grids, and a pixel-to-pixel based least squares approach was not suited to find the best velocity estimates. The velocity estimate of each pixel p in C was produced by spline interpolation of a small spatial region surrounding p in all the grids.

For display of velocity estimates, a manually segmented mask was created from the mean of the B-mode image from all angles, as seen in figure 9 and 10. The mask was

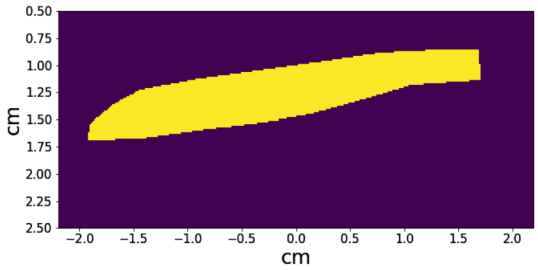

Figure 9: Binary mask data-set 1

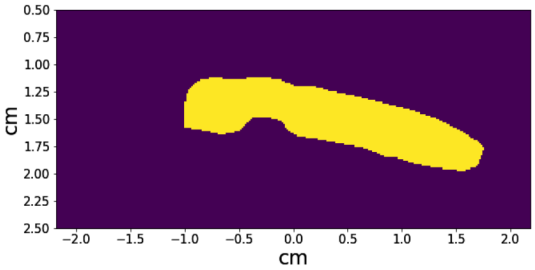

Figure 10: binary mask for data-set 2

then interpolated and applied to the grid, excluding all velocity estimates outside the mask.

A threshold was then selected for the minimum energy level indicating blood flow, resulting in a mask as shown in figure 11. This mask was applied to the velocity estimates V_d before executing the least squares regression step.

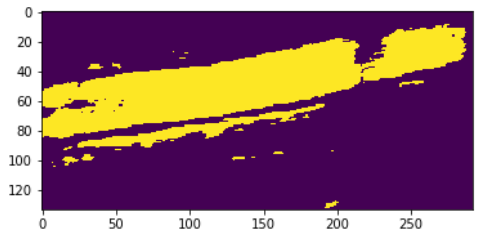

Figure 11: Data-set 1 masked based on signal energy level. Yellow indicates 1, black indicates 0.

4.3 Experimental setup

Parameter	Data-set 1	Data-set 2
PRF[Hz]	6448	6000
c[m/s]	1540	1540
$f_0[Hz]$	5111350.5	4808000
Probe type	Linear	
Apodization window	Tukey75	
Apodization f number	1.1 for Tx and Rx = 0, 1.4 for rest	
Transmit angles	-15, 15	
Receive angles	-15, -3.5, 6, 15, 0, -6, 3.5, 15, 0	

Table 1: Experimental setup

Two different data-sets were used when conducting the experiments. Both were in vivo recordings of the carotid artery. Data-set 1 is of a healthy person with no atherosclerotic plaque, with normal blood velocities. Data-set 2 is a recording where atherosclerosis plaque is present, which may lead to higher blood velocity and more complex blood flow pattern.

The GPU used in this experiment was a NVIDIA Titan V Graphical Processing Unit. The Processor used was a Intel Xeon E5-1600/E5-2600 v2. A run-time comparison was conducted comparing the GPU pipeline with the same version ran on a CPU. The cross correlation step, however, requires a GPU to run. The data set used for measurements were a 274x126 grid for all 9 angles, and 270 frames were used. Parameters used for acquisition and processing of data are given in table 1.

5 Results

This chapter presents the results obtained for each step in the implemented pipeline, the final velocity estimates for the two data-sets, and the computational time for the different steps either using a GPU or a CPU.

5.1 Beamforming results

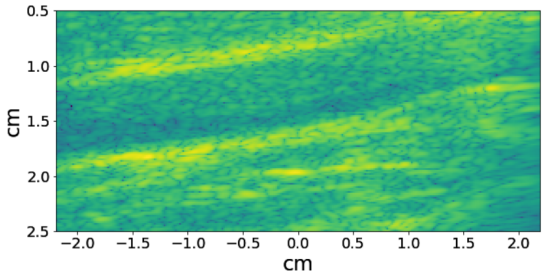

Figure 12: B-mode image from data-set 1

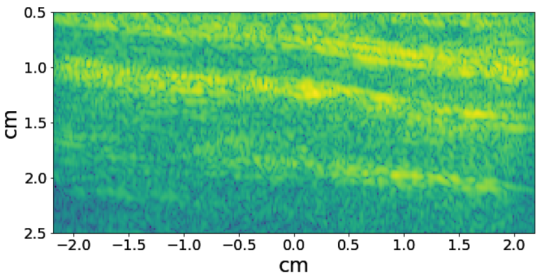

Figure 13: B-mode image from data-set 2

Figure 12 and 13 shows the beamforming results for data-set 1 and 2 respectively. With data-set 1, you can see a clear indication of where the carotid artery is from the B-mode image, as the inside of the vessel has lower intensity. For data-set 2, the upper wall of the artery is clearly delineated, whereas the lower wall is unclear in parts of the image. Some narrowing of the vessel can be seen in the middle of the artery.

5.2 Aliasing correction results

Figure 14 shows the velocity estimates from using the autocorrelation method on data-set 2.

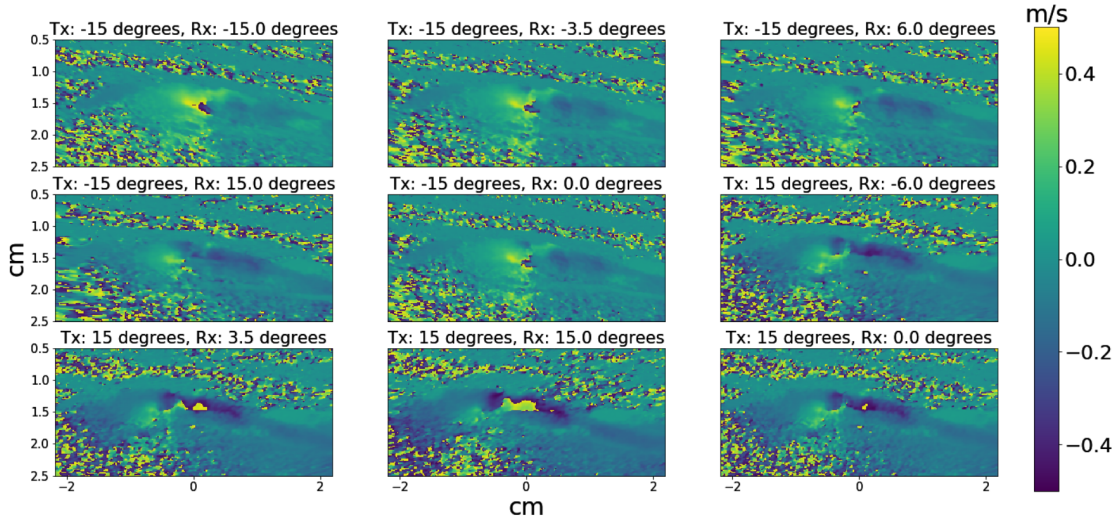

Figure 14: Velocity estimate for 9 different angles using the autocorrelation method.

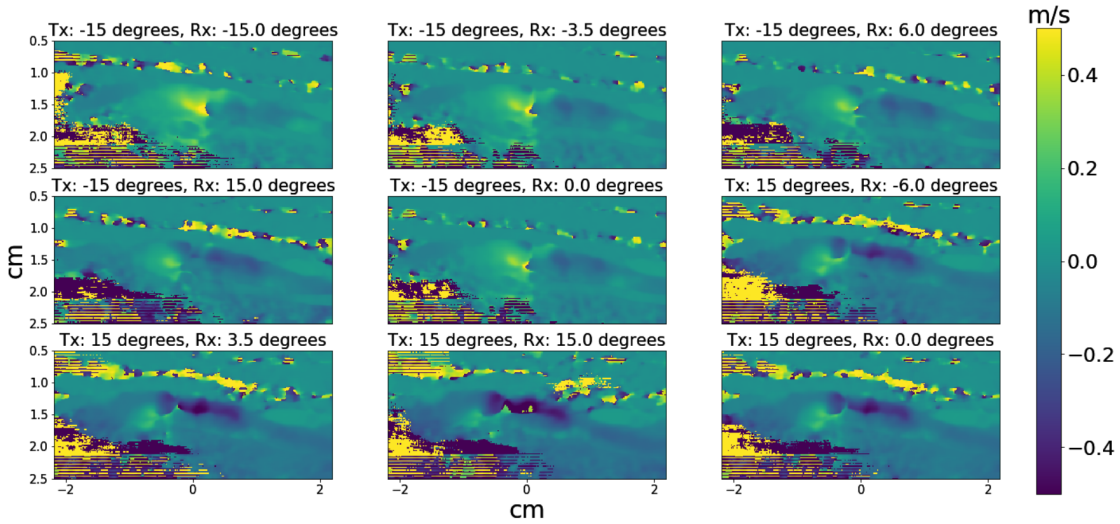

Figure 15: Velocity estimate for 9 different angles using autocorrelation and cross correlation.

Figure 15 shows the corresponding velocity estimates after performing the cross correlation step for aliasing correction, as described in Section 4.2.2. The velocities are similar to the ones in Figure 14, except in regions with apparent aliasing in the central part of

the image. Visual inspection shows that aliasing is corrected in these regions, with only a few exceptions.

5.3 Velocity estimates results

Figure 16, 17, 18 and 19 shows gray scaled B-mode images, overlayed by a colorplot indicating blood flow velocity. The arrows indicate blood flow direction and velocity.

5.3.1 Data-set 1 result

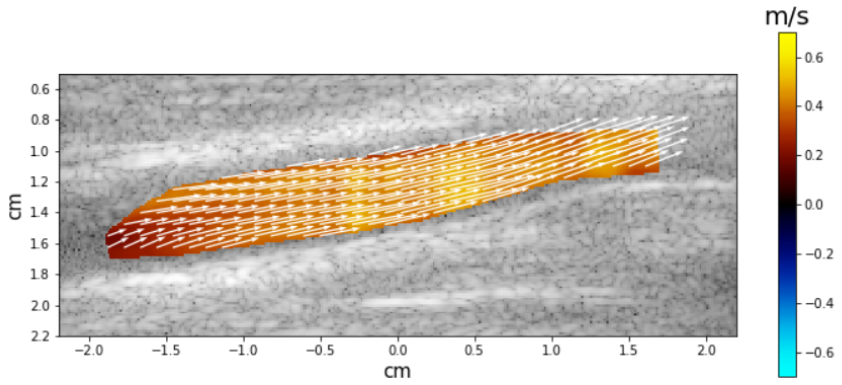

Figure 16: Data-set 1 velocity estimate from method described in section 4.1.

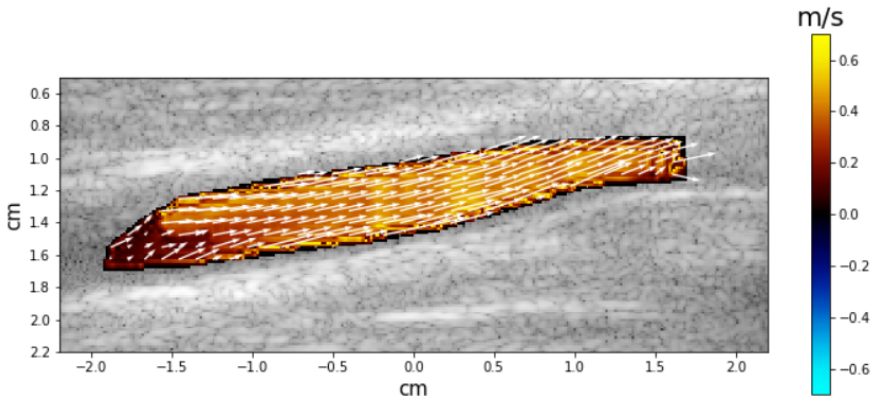

Figure 17: Data-set 1 velocity estimate from method described in section 4.2.

Figures 16 and 17 show the velocity estimates in data set 1 after aliasing correction, for the reference method and the implemented method, respectively. Velocity estimates look qualitatively similar, except along the edges of the flow region.

5.3.2 Data-set 2 result

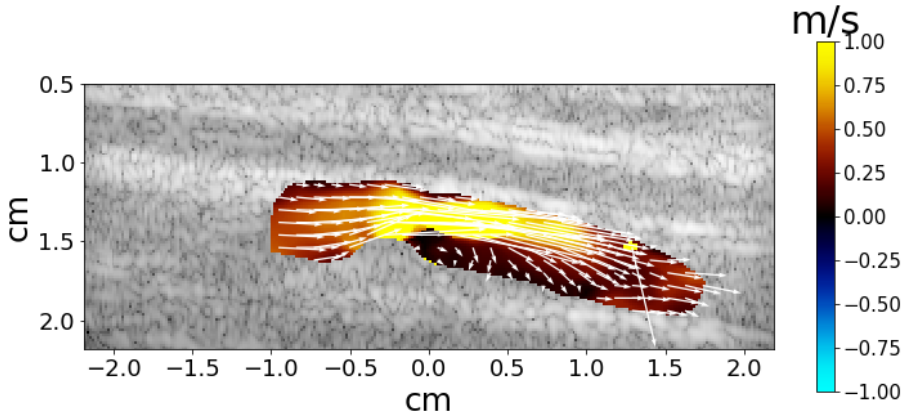

Figure 18: Data-set 2 velocity estimate from method described in section 4.1.

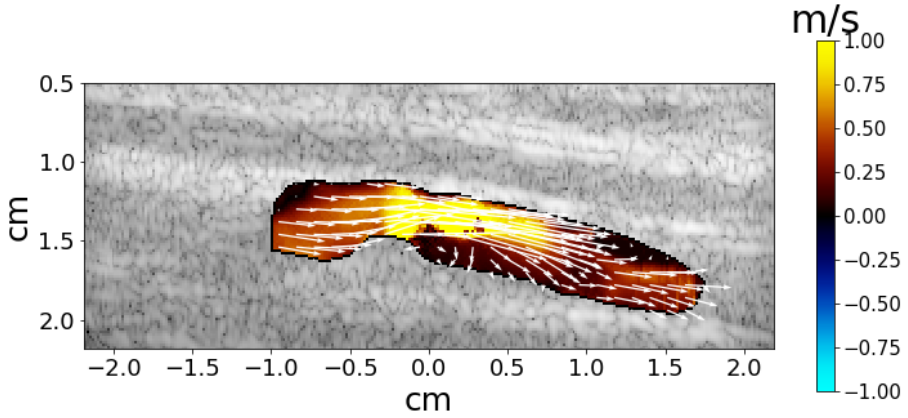

Figure 19: Data-set 2 velocity estimate from method described in section 4.2.

Figures 18 and 19 show the corresponding dealiased velocity estimates for data set 2. Again, the results look qualitatively similar, and for both methods, it is possible to see the inflow region, high velocity stenotic jet flow, and vortex formation in the outflow region of the jet.

5.4 Computational time

Processing Unit	Beamforming	Cross correlation	Least squares	Total
CPU run-time[s]	262	-	114	-
GPU run-time[s]	30	18	14	118

Table 2: Computational time

Table 2 shows the computational time used for the indicated steps using either a CPU or a GPU. No run-time results could be obtained for the cross correlation step, as the library used only is compatible with GPU's.

6 Discussion

Two methods for 2D vector velocity estimation using plane wave ultrasound imaging has been presented in this work. The pipeline described in section 4.3 utilizes GPU technology for all major calculations. The results indicate that aliasing correction is performed correctly, as the vector velocity estimates are similar to the reference method.

6.1 Beamforming

The beamforming process produced the correct results with different apodization used in the 9 different transmit-receive angles. The difference between the angles is clearly displayed in Figures 14 and 15. There is a clear difference between the velocity estimates for the different angles, especially when comparing the velocity estimate with transmit and receive angle of -15 degrees with the estimate with transmit and receive angle of 15 degrees.

6.2 Anti Aliasing

Figure 14 shows velocity estimates from 9 different transmit-receive pairs using only the autocorrelation method. Figure 20 shows a subset the same velocity estimate with receive angle and transmit angle both at -15 degrees.

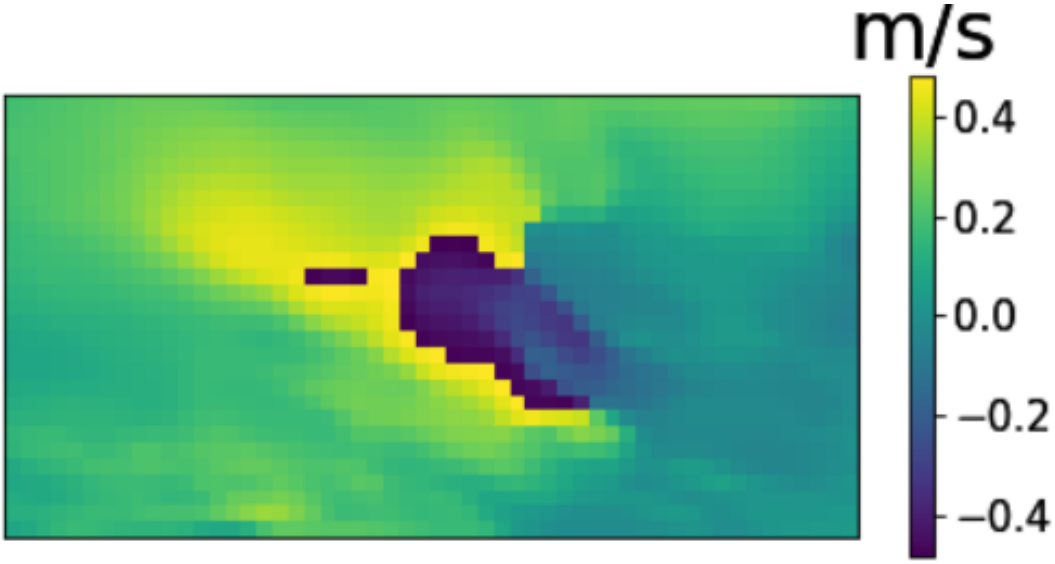

Figure 20: Autocorrelation velocity estimates with aliasing.

The velocity changes from vNyquist to -vNyquist within two neighboring pixels, which indicates that aliasing has occurred in a substantial area in the image. Figure 21 shows the same area as Figure 20 after performing cross correlation. It shows that the cross correlation step removes the aliasing, and all values near -vNyquist have been given values close to vNyquist in the correct direction. This shows that the cross correlation step using speckle tracking works as intended.

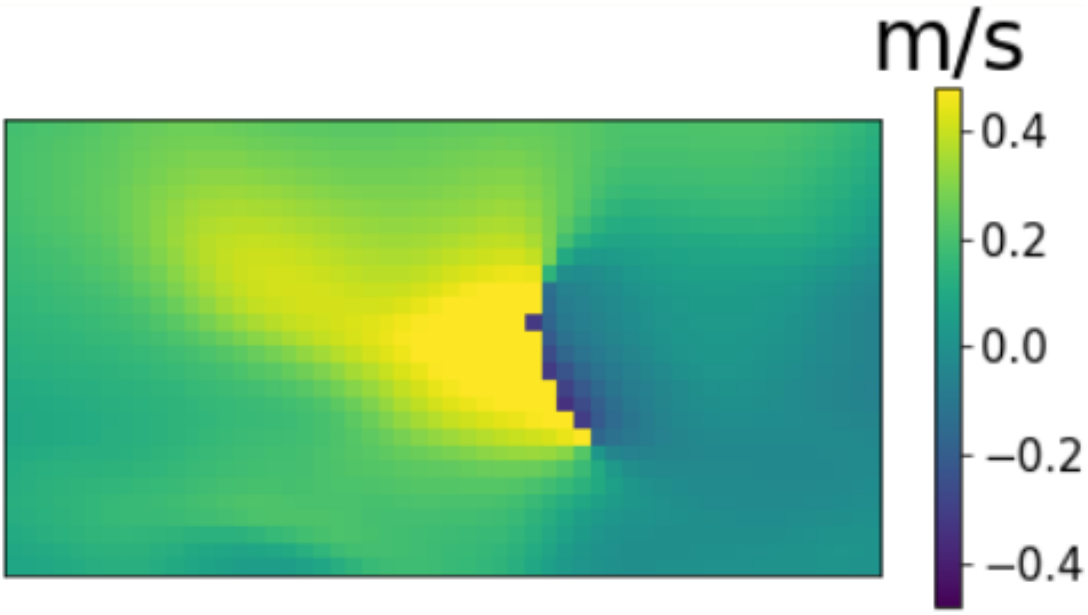

Figure 21: Aliasing corrected velocity estimate.

6.2.1 Final Velocity estimates

In the following the method presented in 4.1 will be called method 1, and the method presented in 4.2 will be called method 2. Figure 16 and 17 shows the estimated velocities for data-set 1. The blood flow direction is qualitatively similar in both methods, as indicated by the white arrows in the image. The velocity ranges are also similar in the two methods. Exceptions can be found in the edges of the estimation area, where method 1 shows a higher velocity estimate. This is mostly due to a more restrictive mask used in method 2 in the weighted root mean squared error step, as values near the edges got a weight of 0.
The velocity estimates for data-set 2 can be seen in figure 18 and 19. It shows that the blood flow direction is consistent in the two methods. The presence of plaque causes a narrowing in the artery, which leads to an increase in velocity in the narrow area. Right after the narrowing, parts of a vortex flow patter can be observed, which is as expected near the outflow region of a jet. A slight difference in velocity estimates can be seen, however, as method 1 displays about 0.4-0.6 m/s higher peak velocities. One potential explanation for this is that the use of different grids in method 2 leads to more spatial smoothing than in method 1.

6.2.2 Computational Cost

The GPU based method shows a clear speed up where comparison was possible. It shows a approximately 8 times speedup on both the beamforming and the least squares

regression. It also shows that the method uses 56 seconds on the steps before, in between and after beamforming, cross correlation and least squares, which is slightly below half of the processing time. This shows that the bottleneck for this method no longer is the main parts (beamforming, cross correlation and least squares), but rather the small and many computations in between needed. Many optimizations are still needed in order to achieve real-time computations.

6.3 Weaknesses and future work

As the artery walls will not be static during measurements over a longer time period, the masks shown in 9 and 10 will not be valid for each frame in the recording. An approach for improving how to exclude the noise from the data should be investigated for the method, as it might be the biggest source of erroneous results. It is especially critical when using skewed grids, as velocity estimates for one pixel is affected by the surrounding area. Areas near the artery wall (which often consist of noise) will therefore impact the velocity estimates near the edges of the artery, and produce wrong estimates.
A weakness of the study is that the two methods only have been compared qualitatively, so a conclusion of the robustness and accuracy of the method can not be made. Also, only 270 frames of two different data sets have been tested, so even though dataset 2 tested the aliasing-resistance of the method, it is not sufficient to completely verify the functionality of the method. Further work should include a quantitative analysis of the method using a larger bulk of data. A study in an environment where the correct results are known would also be beneficial for tuning and improving the accuracy of the method.
The computational costs are also all approximations, as the code has not yet been correctly profiled. The functions should be ran extensively on different data sets to provide a correct run time estimate.

7 Conclusion

A method for aliasing correction in Doppler ultrasound was implemented on a GPU-based platform using autocorrelation, cross correlation and least squares regression. A quantitative study is still needed for an thorough verification of the method, however, the qualitative results indicate that the presented method produces correct velocity estimates, as both flow direction and velocity is qualitatively equal to the reference method. The vast difference in computational time also shows that a GPU/Tensorflow framework provides a large speedup compared to the same version of the code using a CPU.

Bibliography

[1] A. F. B.B Goldberg, R. Gramiak, "Early history of diagnostic ultrasound: The role of american radiologists," 1982.

[2] I. Ekroll, "Ultrasound imaging of blood flow based on high frame rate acquisition and adaptive signal processing," 2013.

[3] "Causes of death." [Online]. Available: https://ourworldindata.org/causes-of-death

[4] M. W. E. B. R. Mota, J.W Homeister, "Atherosclerosis: Pathogenesis, genetics and experimental models," 2017.

[5] "Carotid artery surgery - series - indications." [Online]. Available: https://medlineplus.gov/ency/presentations/100124_2.htm

[6] W. H. Organization, "Manual of diagnostic ultrasound," 2011.

[7] L. Løvstakken, "Signal processing in diagnostic ultrasound: Algorithms for real-time estimation and visualization of blood flow velocity," 2007.

[8] T. Szasz, "Advanced beamforming techniques in ultrasound imaging and the associated inverse problems," 2017.

[9] B. J. S.J. Shin, "Principle and comprehension of ultrasound imaging," 2013.

[10] T. Szabo, "Diagnostic ultrasound imaging: Inside out," 2014.

[11] G. T. M.E. Anderson, "A seminar on k-space applied to medical ultrasound," 2000.

[12] "Basic principle of medical ultrasonic probes (transducer)." [Online]. Available: https://www.ndk.com/en/sensor/ultrasonic/basic02.html

[13] L. L. H. Torp, "Doppler ultrasound for ttk 4165," 2013.

[14] R. S. D. Kroll, A. Lorenc, "Detecting laterality and nasality in speech with the use of a multi-channel recorder," 2015.

[15] A. H. N. S. R. Yaakob, A. Aryanfar, "A comparison of different block matching algorithms for motion estimation," 2013.

[16] "Block matching." [Online]. Available: http://www.ques10.com/p/18886/write-a-short-note-on-exhaustive-block-matching-al/

[17] K. J. T.A. Maniatis, R.S.C. Cobbold, "Two-dimensional velocity reconstruction strategies for color flow doppler ultrasound images," 1994.

[18] E. K. J. Sanders, *CUDA by Example*, 2010.

[19] T. M. J. Cheng, M. Grossman, *Professional CUDA C Programming*. John Wiley Sons, Inc, 2014.

[20] "Tensorflow." [Online]. Available: https://www.tensorflow.org

[21] A. K. R. O. C. Kasai, K. Namekawa, "Real-time two-dimensional blood flow imaging using an autocorrelation technique," 1983.

[22] A. S. H. T. L. L. I.K Ekroll, J. Avdal, "An extended last squares method for aliasing-resistant vector velocity estimation," 2016.

[23] L. P. K. L. P. J. Flynn, R. Daigle, "Estimation and display for vector doppler imaging using planewave transmissions," 2011.