Increasing the Efficiency of Doppler Processing and Backend Processing in Medical Ultrasound Systems

by

Siyuan Wei

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2013 by the
Graduate Supervisory Committee:

Chaitali Chakrabarti, Chair
David Frakes
Antonia Papandreou-Suppappola

ARIZONA STATE UNIVERSITY

May 2013

ABSTRACT

Ultrasound imaging is one of the major medical imaging modalities. It is cheap, non-invasive and has low power consumption. Doppler processing is an important part of many ultrasound imaging systems. It is used to provide blood velocity information and is built on top of B-mode systems. We investigate the performance of two velocity estimation schemes used in Doppler processing systems, namely, directional velocity estimation (DVE) and conventional velocity estimation (CVE). We find that DVE provides better estimation performance and is the only functioning method when the beam to flow angle is large. Unfortunately, DVE is computationally expensive and also requires divisions and square root operations that are hard to implement. We propose two approximation techniques to replace these computations. The simulation results on cyst images show that the proposed approximations do not affect the estimation performance.

We also study backend processing which includes envelope detection, log compression and scan conversion. Three different envelope detection methods are compared. Among them, FIR based Hilbert Transform is considered the best choice when phase information is not needed, while quadrature demodulation is a better choice if phase information is necessary. Bilinear and Gaussian interpolation are considered for scan conversion. Through simulations of a cyst image, we show that bilinear interpolation provides comparable contrast-to-noise ratio (CNR) performance with Gaussian interpolation and has lower computational complexity. Thus, bilinear interpolation is chosen for our system.

To my parents for their love and support

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Chaitali Chakrabarti, for her continuous guidance and patience during the past year. I am lucky to have the opportunity of continuing to work with her in my future pursuit of PhD degree. I am grateful to Dr. Antonia Papandreou-Suppappola and Dr. David Frakes for their time and effort in reviewing my work.

I would like to thank my mentor Ming Yang for all his help. I could not have accomplished this without his patient explanation of signal processing and coding concepts. Also, I owe my gratitude to our collaborators in University of Michigan—Richard Sampson and Dr. Thomas Wenisch, for their insightful comments. I am also grateful to the other members of the Low Power System Laboratory, especially Lifeng Miao, Qian Xu, Chengen Yang and Yunus Emre, for their encouragement and their patience in answering my questions. Finally I would like to thank my friends and my family for their continuous support and all those who have helped me on my way toward this moment.

I gratefully acknowledge the financial support from NSF through CSR0910699.

TABLE OF CONTENTS

Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

Medical ultrasound imaging has been used in clinical diagnosis for many years. Compared to computed tomography (CT) and magnetic resonance imaging (MRI), ultrasound imaging has a longer history, and yet it is widely used because it is non-invasive, safe and cheap [1]. It also provides high-resolution images that are sufficient for many clinical applications, such as general abdominal imaging and color flow imaging [2].

Almost all the ultrasound imaging systems today are based on pulse-echo imaging. A sound wave is transmitted by a handheld transducer into the body. The wave is echoed by the tissue and blood, with part of the transmitted energy returning back to the transducer [2]. The echoes are detected and then processed to obtain information of the scatterers (tissue or blood). Given the sound speed in the tissue being investigated, the round trip delay of the sound wave can be calculated and the echo signals can be distinguished by their arrival time.

1.1 Background

The block diagram of a typical ultrasound imaging system is shown in Figure 1.1. A transducer array typically consists of hundreds of transducer elements. The piezoelectric transducer elements convert electrical energy into sound energy or vice versa. The T/R switch controls the flow between the transducer and analog front end blocks. The analog front end amplifies and filters the analog data obtained by the transducer. The ADC/DAC translates the analog signal into digital or vice versa. The

digital front end does the beamforming for both transmit and receive. There are several imaging modalities, such as A-mode, B-mode, M-mode and Color Doppler [3]. Our system is based on B-mode imaging which represents the brightness of the backscattered signals. Additionally we support Doppler processing for estimating the velocity of blood in vessels. At the end of the chain is backend processing which includes scan conversion and post processing modules to help display a clean ultrasound image on the screen.

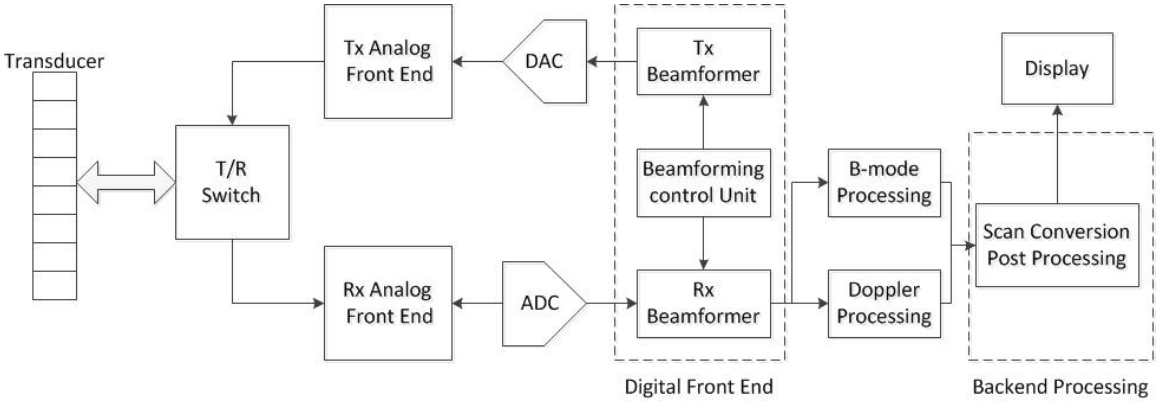


Figure 1.1 Ultrasound imaging system block diagram (adapted from [3])

There are generally two different kinds of ultrasound imaging systems: Conventional ultrasound imaging system based on linear array or phased array and synthetic aperture (SA) imaging system. Linear array system generates parallel image lines which are perpendicular to the transducer plane, while phased array system generates image lines starting from the center of the transducer, steered at different angles. Both systems utilize a linear array scanner; for a linear array system, only part of the transducer elements are active during transmit and receive, while for phased array system, all the transducer elements are active. Conventional systems acquire one image line at a time in sequence. Thus, the frame rate is limited by the speed of sound, c. If there are N image lines and the transmission depth is D, the maximum frames per second

(fps) is: $fps = c/2DN$. For instance, if we have 200 image lines and the transmission depth is 15cm, then for sound speed given by 1540m/s, the maximum fps is 51. The frame rate is significantly lower if 3D imaging is considered. Another disadvantage of conventional systems is that there is only one transmit focus, which means that the samples that are not close to the transmit focus are not properly focused.

Synthetic aperture (SA) imaging system creates low resolution images using less transducer elements for each transmission and then reconstructs the high resolution image by adding the low resolution images [1]. Typically one element transmits at a time and all elements receive. Unlike conventional imaging system, SA system creates a complete image in every transmission and thus decouples the number of scanlines and the frame rate [4]. Another advantage of SA system is that dynamic focusing is achieved in both transmission and receive compared to conventional system which has a single transmit focus. One of the weaknesses of SA system is the loss in SNR, because less elements are active during transmission, resulting in less signal power. This limits the penetration depth, since we cannot increase the transmit power too much, considering patients' safety. One solution is to combine several elements in one transmission. Certain apodization windows can help defocus the transmission to emulate a spherical wave [4]. In 3D imaging, subaperture processing method with sparse virtual sources has been proposed to achieve better trade-offs between image quality and hardware cost [5]. Another solution is to use coded excitation, such as orthogonal Golay code and orthogonal chirp [6]. Both can improve the SNR and penetration depth significantly. However, coded excitation method makes the system more sensitive to motion of patients

or examiners. In general, SA system overcomes the frame rate limitations and single transmit focus problem at the cost of higher hardware complexity.

1.2 Problem description

The overarching goal of our research is to build a 3D portable medical ultrasound imaging system that produces high quality images. The benefit of portable imaging devices is not only about convenience; the improvement of diagnosis outcomes has been proved in clinical applications [8]. 3D ultrasound imaging, compared to its 2D counterpart, provides images that are easier to interpret with multiple view angles so that technicians can locate relevant anatomy with less effort. 3D images of cysts and tumors have accurate and complete information which 2D images cannot provide. However, 3D imaging is very challenging in terms of hardware implementation. To create a 3D image, we need to use a 2D transducer array. This means the incoming raw data of a 3D system is about 100 larger than 2D. The number of focal points in a image is also increased significantly, resulting in much larger power needed for signal processing. The problem is particularly challenging since we have a rather low power budget (about 5W) for portability requirements.

In our earlier work with University of Michigan researchers, a 3D medical ultrasound imaging front end named "Sonic Millip3De" has been designed and implemented [9]. We were able to generate full 3-D images (50 X 50 scanlines, 4096 samples on each scanline) with 1 frame per second while staying below a system power of 20W in 45nm technology. The Sonic Millip3De architecture combines a massively pipelined hardware design with 3D die stacking techniques to achieve low power

consumption [9]. Sonic Millip3De is essentially an accelerator for the beamsum operation in the digital front end. In this thesis we studied two additional units to make the 3D imaging system more complete. These include a Doppler processing unit to estimate blood velocity and a backend processing unit to display the ultrasound image on the screen.

1.3 Contributions

In this work, we studied the performance and computational complexity of two schemes for Doppler processing, namely Conventional Velocity Estimation (CVE) [1] and Directional Velocity Estimation (DVE) [10]. CVE is based on computing beamsum along the scanline and estimating the velocity component based on the phase change of the signals. DVE is based on computing the beamsum along the flow line and estimating the velocity directly from the displacement of the scatterers. Our results show that DVE has better performance compared to CVE and can handle large beam to flow angles. However, it has higher computational complexity and includes square root and division operations which have a large hardware cost. To address this problem, we introduced approximation techniques, including Taylor expansion and Lagrange interpolation filter which replaced these operations with multiplications and additions. We showed that use of these approximations did not affect the velocity estimation performance.

Next we studied different blocks in backend processing, namely, envelope detection, log compression and scan conversion. We compared different envelope detection methods in terms of performance and computational complexity, and came to the conclusion that quadrature demodulation is desirable when phase information is

important and FIR based Hilbert transform method works better for a B-mode imaging system. Finally, we studied two interpolation methods for scan conversion, including bilinear interpolation and Gaussian interpolation. The results show that they are almost identical in terms of contrast-to-noise ratio performance, but bilinear interpolation has lower computational complexity.

1.4 Thesis report organization

The report is organized as follows: Doppler processing is described and the two competing schemes (CVE and DVE) are presented and their performance compared in Chapter 2. In Chapter 3, the different backend processing blocks are described. Chapter 4 concludes the report.

CHAPTER 2

DOPPLER PROCESSING

In clinical diagnosis, color flow imaging (CFI) based on Doppler Processing is quite popular. This imaging modality is effective in locating stenosis and occlusion and helps in the prevention of medical conditions such as stroke. In CFI, the velocity estimates in a region of interest are color coded and displayed. Sometimes, the CFI image is imposed on a B-mode image for display.

Current Doppler processing systems make use of pulsed waves. Multiple pulses are periodically transmitted in the same direction, and the blood velocity is estimated by measuring the phase shift between the received signals [1] or the movement of the scatterers over time [10].

A Doppler processing system can be built using both a linear array system [10] and a synthetic aperture system [11]. A linear array system is considered here since it has lower power requirement. In this system, the transducer elements are steered and focused during both transmit and receive. For each transmit and receive, delay-sum based beamforming is done as in B-mode imaging. The beamformed data is then used to estimate the velocity information.

Although similar to B-mode imaging in many ways, Doppler processing is more technically demanding. Doppler processing is typically used to measure blood flow, which has several orders smaller magnitude of backscattered signals compared to the surrounding tissues, resulting in lower SNR. Theoretically, the velocity can be estimated from only two transmissions. However, due to the stochastic nature of the echoes from

blood, more transmissions are used to generate good estimates in practice. As a result, CFI is usually done for only a part of the B-mode image.

Although our final goal is to measure more complex blood flow, in this work, several assumptions are made to simplify the velocity estimation problem. First, we assume the blood flow is laminar with a parabolic profile. Second, we assume that the blood flow is steady between transmissions. Last, we assume that the beam to vessel angle is known from the B-mode image.

2.1 Conventional Velocity Estimation (CVE)

2.1.1 Basic Method

Conventional Velocity Estimation is based on estimating the phase shifts between subsequent transmissions. Its firing scheme is the same as the linear array system in B-mode imaging. The block diagram for CVE [1] is shown in Figure 2.1.

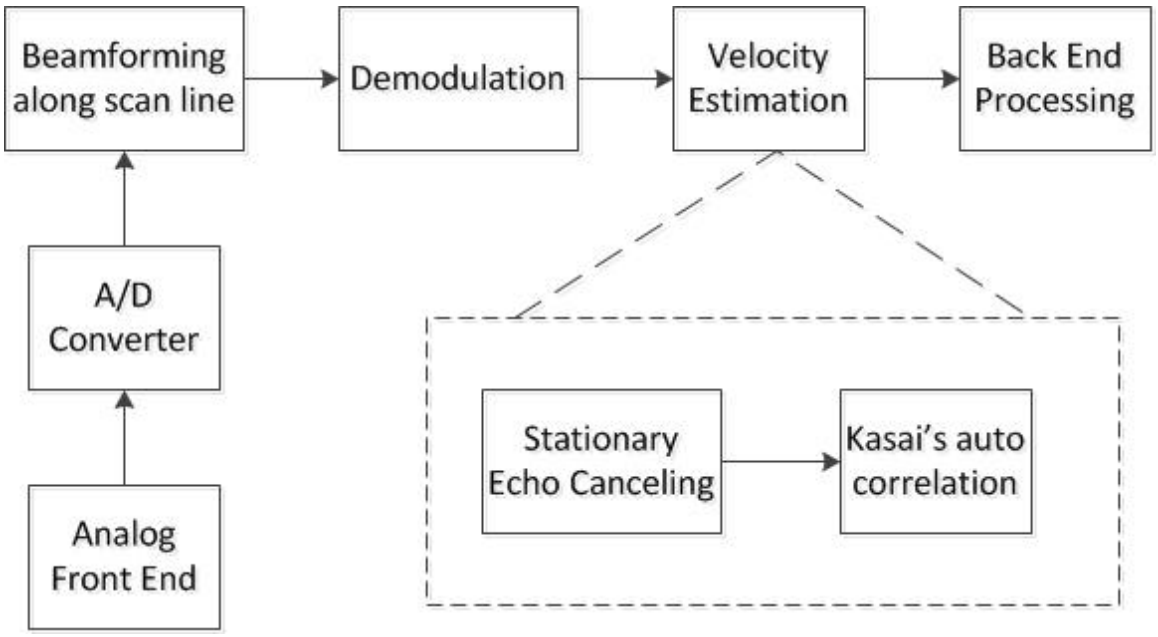


Figure 2.1 Block Diagram of CVE (adapted from [1])

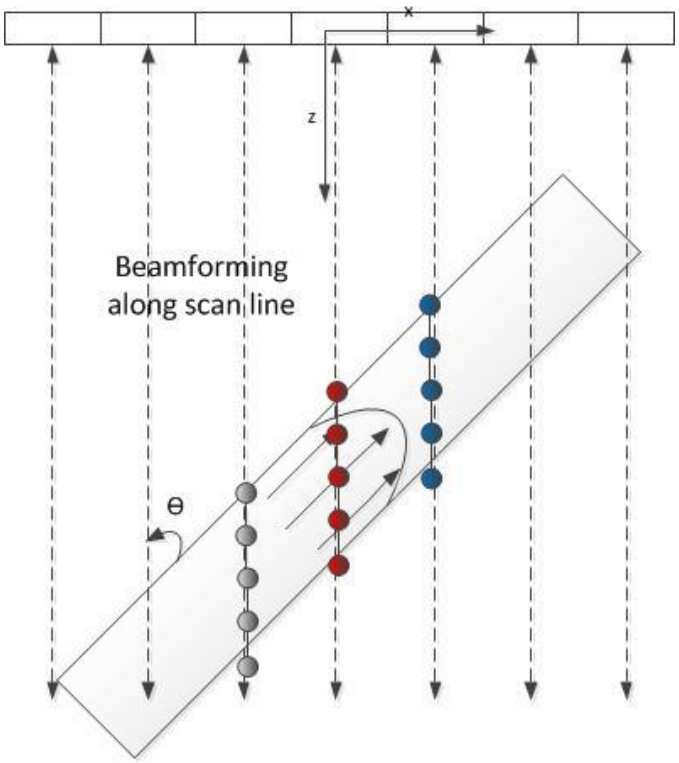


Figure 2.2 Beamforming along scan line in CVE

After A/D conversion, beamformation is done along the scan line as shown in Figure 2.2. This is the same as B-mode imaging and so no additional effort is required to do this stage. Next, demodulation is done to get the in-phase and quadrature phase signals. Quadrature demodulation is typically used for Doppler processing. The velocity estimation in this method is first done in the beam direction and then projected onto the flow direction. In the method in [10], first stationary echo canceling is done to isolate the stationary component from the blood flow component. This is done by subtracting the stationary signal, calculated using the mean of the signals from 16 firings, from the beamformed signal. In contrast, in [12], the stationary echo canceling is done by subtracting the beamformed results of successive firings. We tried both methods and found that echo canceling based on averaging the signals from all firings do not improve

the performance so the simulations presented here are based on echo canceling based on the results of consecutive firings. The modified beamformed signal $\tilde{g}_i(x')$ is then used to compute velocity using Kasai's auto correlation method.

For CVE, the velocity estimation is related to the phase change between consecutive received signals. If the blood flow velocity is denoted by v, the pulse repetition interval is denoted by t_{PRI}, and the angle between the beam and flow is θ, then the displacement in the beam direction between successive transmissions is $\Delta z = (v\cos\theta)t_{PRI}$. Thus, the delay between two successive received signals is given by $\Delta\tau = \frac{2\Delta z}{c_0} = \frac{2v\cos\theta}{c_0}t_{PRI}$ and $v = \frac{c_0\Delta\tau}{2t_{PRI}\cos\theta}$. If the velocity of the blood flow is assumed constant during the pulse repetition interval, the delay can also be expressed in terms of phase shift $\Delta\varphi$ as: $\Delta\tau = \Delta\varphi/2\pi f_c$ and the velocity estimate is given by

$$v = \left(\frac{c_0}{4\pi f_c t_{PRI}\cos\theta}\right)\Delta\varphi \tag{2.1}$$

The phase shift $\Delta\varphi$ can be estimated by computing the correlation between the signals of two successive transmissions. If more than two transmissions are used for a single scan line, the average of the correlation results is taken and then the phase change is obtained from the complex signal. If N_t is the number of transmissions per scan line, the estimated velocity is given by

$$v = \left(\frac{c_0}{4\pi f_c t_{PRI}\cos\theta}\right)\left(\tan^{-1}\frac{Im\left[\sum_{i=2}^{N_t}\tilde{g}_i(x')\tilde{g}_{i-1}^*(x')\right]}{Re\left[\sum_{i=2}^{N_t}\tilde{g}_i(x')\tilde{g}_{i-1}^*(x')\right]}\right) \tag{2.2}$$

While this method is fairly simple to implement, its performance is quite poor when the beam to flow angle is close to 90 degrees. This can also be seen from equation (2.2) which shows that $v \propto f(1/\cos\theta)$.

2.1.2 Computational complexity

Table 2.1 describes the parameters to characterize our system. These include N_t, the number of transmissions per estimation; N_{act}, the number of active elements; N_{est}, the number of points for estimation on one scan line; N_{foc}, the number of focal points on one scan line. Additional parameters R_{cc} and L_{cc} are related to computing the cross correlation function in DVE and will be explained in section 2.2.

Name	Description	CVE	DVE
N_t	Number of transmissions per estimation	16	8
N_{act}	Number of active elements	64	64
N_{est}	Number of points for estimation on one scan line	50	50
N_{foc}	Number of focal points on one scan line	1000	_
R_{cc}	Range of searching the maximum in cross correlation function	_	10
L_{cc}	Number of samples for cross correlation	_	200

Table 2.1 System parameters

Table 2.2 describes the computational complexity in terms of number of multiplications, additions and divisions. As in B-mode imaging, the beamforming part costs the most in terms of the number of computations. Velocity estimation requires divisions which are also costly in terms of hardware implementation.

	Multiplication	Addition	Division
Beamforming	$N_{foc} * N_t * N_{act}(delaysum)$ $= 2{,}048{,}000(delaysum)$		
Stationary echo canceling		$(N_t - 1) * N_{est}$ $= 750$	
Demodulation	$2 * N_{foc} * N_t$ $= 32000$	–	–
Velocity estimation	$4 * N_{est} * N_t$ $= 3200$	$3 * N_{est} * N_t$ $= 2400$	N_{est} $= 50$

Table 2.2 Computational complexity of CVE

2.2 Directional Velocity Estimation (DVE)

2.2.1 Basic method

DVE was proposed by Jensen in [10], to enable velocity estimation when the beam to flow angle is around 90 degrees. Unlike CVE, beamformation in DVE is done along the flow direction for a given depth within the vessel, as shown in Figure 2.4. The beamformed data is then used to estimate the velocity from the displacement of scatterers divided by the time between transmissions. Similar to CVE, the angle between beam and flow direction is assumed to be known from the B-mode image. The block diagram of the method in [10] is shown in Figure 2.3.

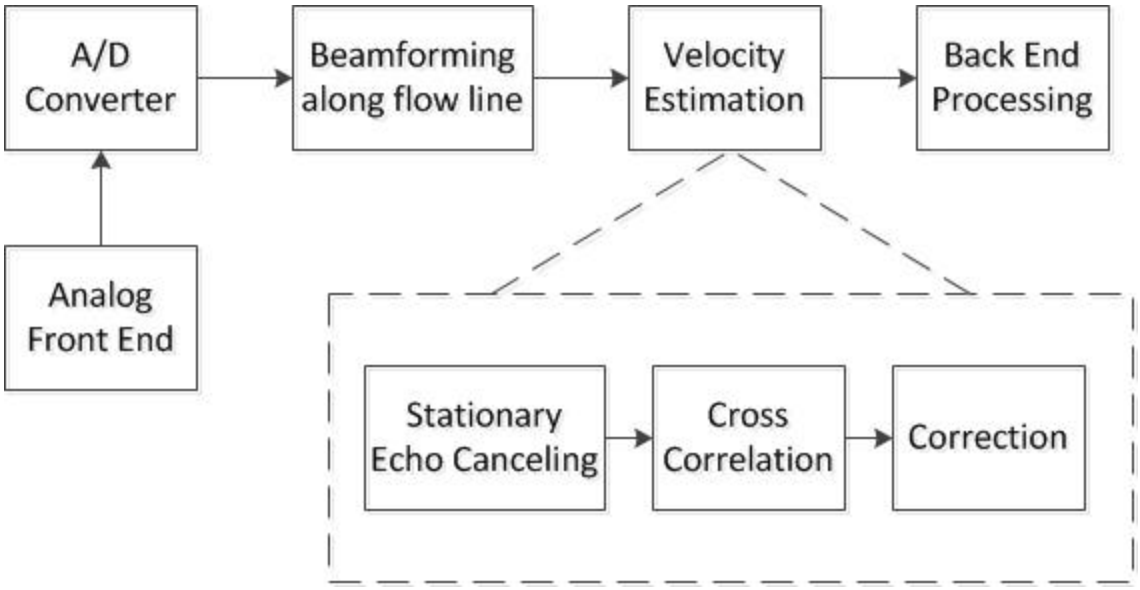


Figure 2.3 Block Diagram of DVE (adapted from [10])

To describe beamformation along a flow line, we introduce a coordinate system which is aligned along the flow direction as shown in Figure 2.5. The coordinate axes marked as x' and z' are placed at the center of the vessel; here the x' axis is parallel to the flow direction. The relation between the two coordinate systems is

$$x = x' \sin\theta\,;\; y = y';\; z = z' \cos\theta + z_o \tag{2.3}$$

where z_o is the offset between the origins of the two coordinate systems.

While the beamformation is done along the flow line, the focusing scheme is the same as in CVE. Assuming x_i is the coordinate of the receive element, the delay for each focal point can be calculated by the round trip propagation distance of the wave divided by the speed of the sound, given as

$$\tau = (R_r + R_t)/c \tag{2.4}$$

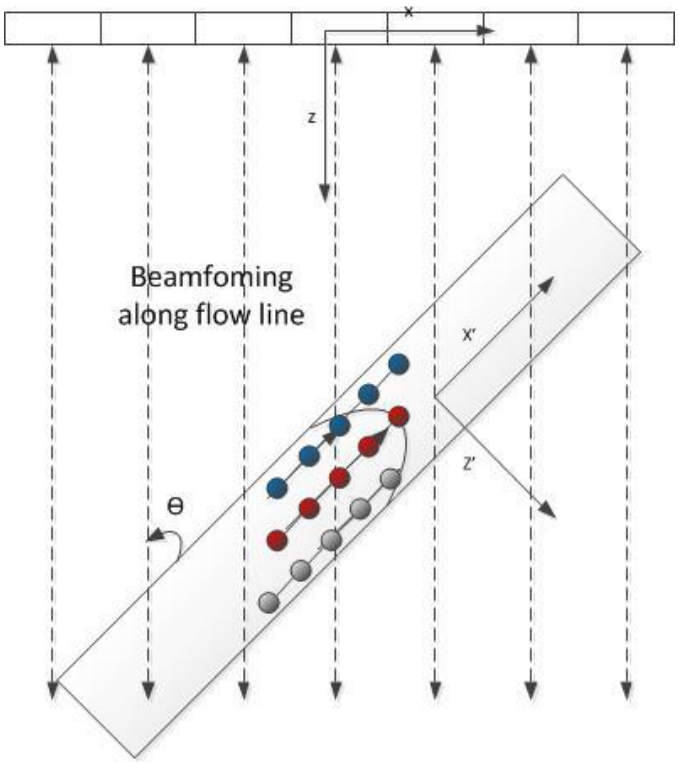


Figure 2.4 Beamforming along flow line in DVE

where R_r and R_t are the receive and transmit distance for each point to be beamformed. R_r and R_t can be expressed as

$$R_r = \sqrt{(z_o - x' \cos\theta)^2 + (x_i - x' \sin\theta)^2} \tag{2.5}$$

$$R_t = \sqrt{(z_o)^2 + x'^2 - 2z_o * x' \cos\theta} \tag{2.6}$$

Unlike CVE, demodulation is not needed for DVE, the RF data is directly used for the velocity estimation. As in CVE, we found that echo canceling based on averaging the signals from all firings do not improve the performance so the simulations presented here are based on echo canceling based on the results of the subtraction of two consecutive firings. Assuming that the first signal along the flow direction is $\tilde{g}_1(x')$, and

the second signal obtained after a pulse repetition interval $T_{PRI}$is $\tilde{g}_2(x')$, the relation between the two signals is given by:

$$\tilde{g}_2(x') = \tilde{g}_1(x' - v_{x'}T_{PRI}) \tag{2.7}$$

where $v_{x'}$ is the velocity in flow direction. Since the signals are discrete, the cross correlation signal is given by

$$R_{12}(k) = \sum_{x'=-L_{cc}}^{L_{cc}} \tilde{g}_1(x')\tilde{g}_2(x' + k) \tag{2.8}$$

where L_{cc} denotes the number of signal samples used for cross correlation. Since $\tilde{g}_2(x')$ and $\tilde{g}_1(x')$ are related (see equation 2.9), the cross correlation function in equation (2.10) becomes an autocorrelation function, expressed as $R_{11}(k - v_{x'}T_{PRI})$. The autocorrelation function has a global maximum at zero. Therefore, the position k_s, corresponding to the maximum value of the cross correlation function, is an estimate of the displacement. The velocity of the scatterers is then given by $v_{x'} = \frac{k_s}{T_{PRI}}$.

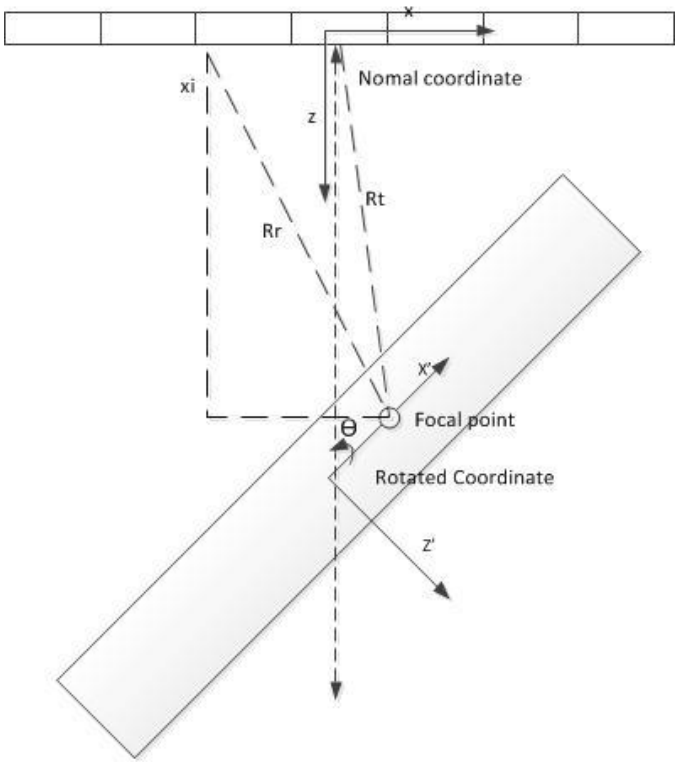


Figure 2.5 Illustration of delay calculations in DVE

Estimating the displacement based on cross correlation function is done for each pair of signals from two successive transmissions and the average of the results is reported. The searching range for the global maximum depends on the estimated maximum velocity. A smaller range not only reduces the computational complexity but also avoids false peaks [10].

The estimation of k_s is not accurate due to low spatial sampling rate. Figure 2.6 shows a scenario where the sampling is not dense enough and so the real peak is not identified. Again, increasing the spatial sampling rate leads to higher computational complexity for beamforming part and is not desirable.

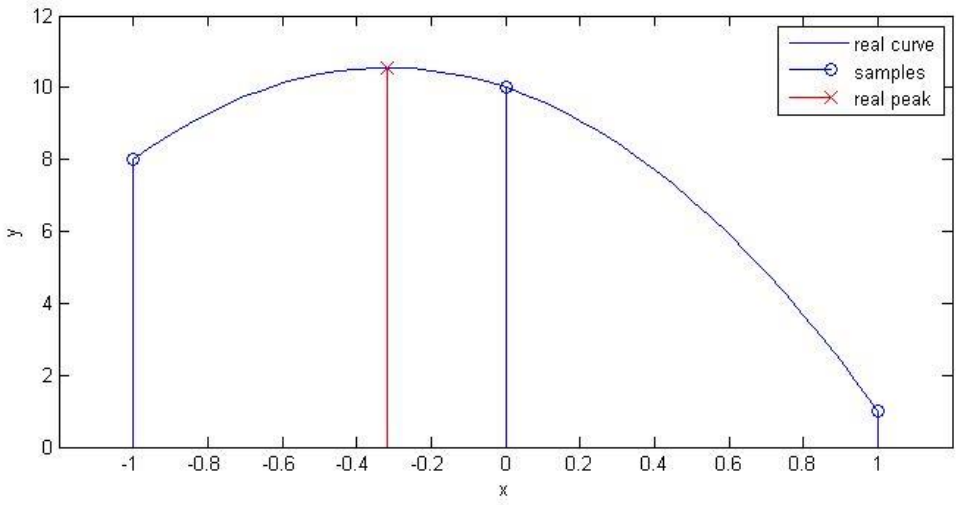


Figure 2.6 Need for correction in DVE

A correction method has been proposed in [10], based on quadrature interpolation method:

$$k_{int} = k_s - \frac{R_{12}(k_s+1) - R_{12}(k_s-1)}{2\left(R_{12}(k_s+1) - 2R_{12}(k_s) + R_{12}(k_s-1)\right)} \tag{2.9}$$

where R_{12} is the cross correlation function defined in equation (2.8). k_{int} is then used to find the displacement of scatterers which is then used to find the estimated velocity. If the displacement in terms of samples is denoted by S, and the spatial distance between two sampling points along the flow line is denoted as dx', the estimated velocity is then given by $v_{x'} = \frac{S*dx'}{t_{PRI}}$.

2.2.2 Computational complexity

The parameters to characterize our DVE system were given in Table 2.1. Table 2.3 describes the computational complexity of this method. Compared to the computational complexity of CVE system (Table 2.2), DVE system is more expensive in terms of hardware cost. For instance, for the most computationally complex block—beamforming,

DVE requires 5 times more delay-sum operations than CVE. DVE also requires more divisions than CVE. In the next section, we describe approximation techniques to remove the square root operation in delay calculations [3], using Taylor expansion. Techniques to remove divisions in the correction block will also be described.

	Multiplication	Addition	Division
Beamforming	$L_{cc} * N_{est} * N_t * N_{act}(delay - sum)$ $= 10{,}240{,}000(delay - sum)$		
Stationary echo canceling		$(N_t - 1) * N_{est}$ 350	
Cross correlation	$R_{cc} * L_{cc} * N_{est} * N_t$ $= 800{,}000$	$R_{cc} * L_{cc} * N_{est} * N_t$ $=800{,}000$	–
Correction	–	$4 * N_{est} * N_t$ $= 1600$	$N_{est} * N_t$ $= 400$

Table 2.3 Computational complexity of DVE

2.2.3 Techniques to reduce complexity

Delay calculations involve square root operations which are very expensive so we approximate the square root calculation by Taylor expansion around $x_i = x' \sin\theta$:

$$R_r = f(x' \sin\theta) + f'(x' \sin\theta)(x_i - x' \sin\theta) + \frac{f''(x' \sin\theta)(x_i - x' \sin\theta)^2}{2} + \cdots$$

If only the first three terms are taken for simplification, we have:

$$R_r = (z_o - x' \cos\theta) + \frac{1}{2(z_o - x' \cos\theta)}(x_i - x' \sin\theta)^2$$

Since $z_o \gg x' \cos\theta$, we can further simplify this as:

$$R_r = (z_o - x' \cos\theta) + \frac{(1 + x' \cos\theta / z_o)}{z_o}(x_i - x' \sin\theta)^2 \qquad (2.10)$$

Similarly, for the transmission delay calculation (equation 2.7), Taylor expansion is done around $x' = 0$:

$$R_t = z_o - x' \cos\theta + (x' \sin\theta)^2/(2z_o) \tag{2.11}$$

The performance penalty of both these approximations will be examined in the next section.

Division is another computationally expensive operation that occurs in the correction method of [10] as seen in equation (2.9). To remove division without affecting the performance too much, FIR interpolation filter is introduced. Specifically, we utilize Lagrange interpolation filter which is widely used in digital-to-analog converters and image processing. We design the second order Lagrange interpolation filter using MATLAB. Zeros are added between the samples based on the interpolation factor, and convolution is done between the zero padded signal and the Lagrange interpolation filter. Figure 2.7 shows the results using a second order Lagrange interpolation filter with interpolation factor of 5. This method helps identify a more accurate peak position. The interpolation factor can be selected to meet different requirements of accuracy. For instance, a larger interpolation factor is needed when the beam to flow angle is large.

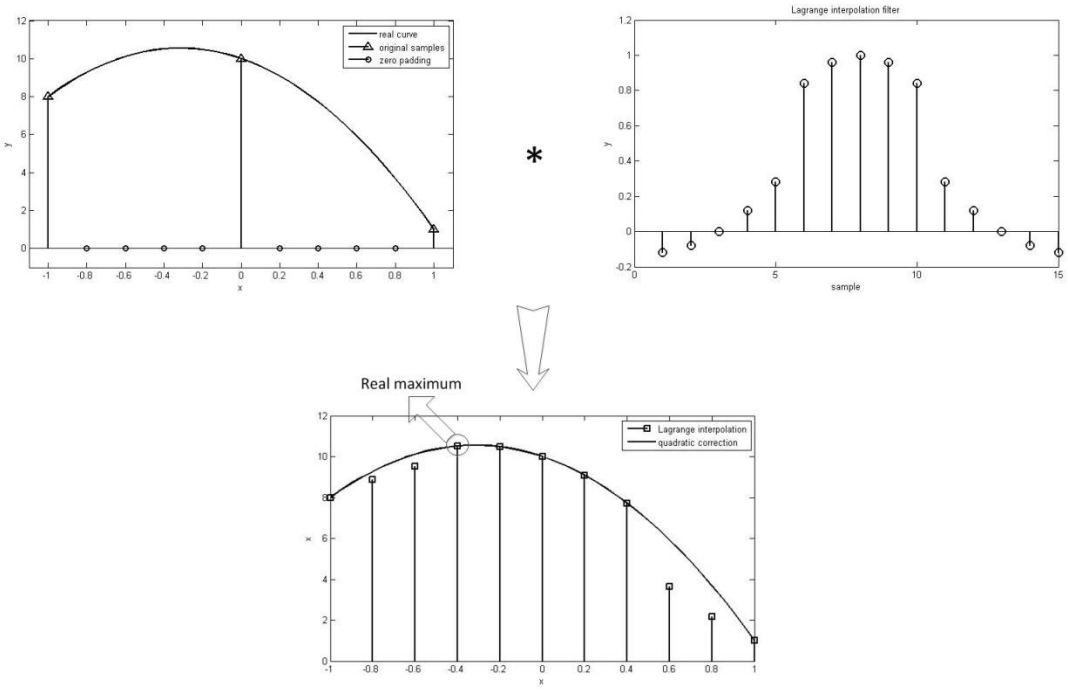


Figure 2.7 Lagrange interpolation

The proposed approximation methods remove the square root and division operations at the expense of additional multiplications and additions. Table 2.4 summarizes the computations needed with and without approximations. For delay calculations of each focal point, including transmit and receive, it costs 3 more multiplications and 1 more addition to avoid a square root operation. For the correction module, a division is replaced by 24 multiplications and 12 additions if we use a second order Lagrange interpolation filter with interpolation factor of 5.

	Without approximations				With approximations			
	Mult	Add	Sqrt	Div	Mult	Add	Sqrt	Div
Delay calculations	7	5	2	0	10	6	0	0
Correction	0	4	0	1	24	16	0	0

Table 2.4 Extra computations to implement the approximation methods (per operation)

2.3 Simulation results

The performance of CVE and DVE based methods has been investigated using Field II program [22][23][24]. The simulations are done for a single scanline. Since linear array system is used for both methods, the estimation results on a single scanline is representative of the performance of the two methods.

Assuming the flow is laminar and parabolic, the velocity profile is expressed as:

$$v(r) = v_0\left(1 - \left(\frac{r}{R}\right)^2\right), \tag{2.8}$$

where r is the radial distance from the center line of the vessel, $v_0$is the peak velocity and R is the vessel radius. The scatterers are propagated during the pulse repetition interval based on the velocity profile. Most of the simulation results are for beam to flow angle of 45 degrees though we do consider larger beam to flow angles in the end.

Typical simulation parameters are shown in Table 2.5. For both CVE and DVE, the velocity estimates are made on the center scanline; the distance between two estimate points is 0.5 mm. The estimated velocity is then compared to the true velocity to measure the performance of estimation. We use root mean square error defined as: $RMSE = \sum_{i=1}^{N_{est}} v_{true}(i) - v_{est}(i)$, where $v_{true}(i)$ and $v_{est}(i)$ are the true velocity and the estimated velocity of the ith estimate points respectively, and N_{est} is the number of

estimate points along a scanline as described in Table 2.1. Smaller RMSE indicate better estimation. Maximum error is also reported in order to identify the outliers which could result in mis-diagnosis.

Name	Description	Value
f_c	Transducer center frequency	4 MHz
c	Speed of sound	1540 m/s
$\lambda = c/f_c$	Wavelength	0.385 mm
$p = \lambda$	Pitch of transducer element	0.385 mm
h_e	Height of transducer element	5 mm
$w = 0.95\lambda$	Width of transducer element	0.368 mm
N_{act}	Number of active elements	64
N_t	Number of transmissions per estimation	8
f_s	Sampling frequency	120 MHz
f_{prf}	Pulse repetition frequency	10 kHz
R	Radius of vessel	10 mm
v_0	Peak velocity of flow	1 m/s

Table 2.5 Common parameters for both CVE and DVE

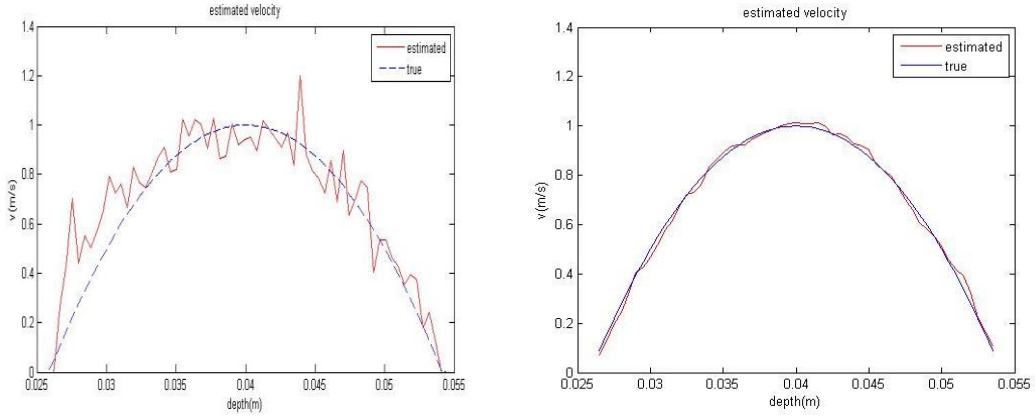
2.3.1 Comparison between CVE and DVE

Figure 2.8 shows the estimation results for both CVE and DVE, for a system with 64 active transducer elements and beam to flow angle of 45 degrees. No approximation techniques are used for DVE here. It is obvious that DVE gives more accurate estimation

than CVE. The RMSE and maximum error are shown in Table 2.6. We see that the RMSE of DVE is about 20% of that of CVE.

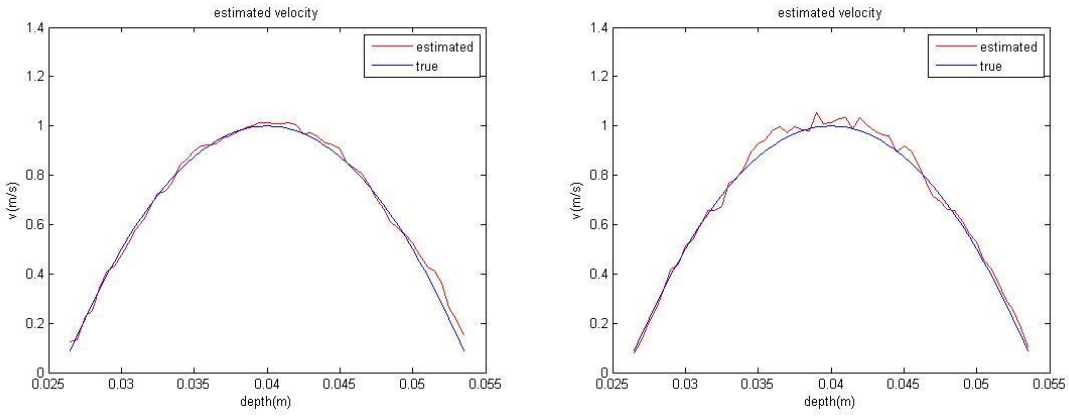
	CVE (64 elements)	DVE (64 elements)	DVE (128 elements)
Maximum error (m/s)	0.3502	0.0539	0.0556
RMSE (m/s)	0.1138	0.0257	0.0289

Table 2.6 Performance of CVE and DVE



(a) (b)

Figure 2.8 Estimation results and the true velocity for (a) CVE (b) DVE



(a) (b)

Figure 2.9 Estimation results and the true velocity for (a) DVE with 64 active elements (b) DVE with 128 active elements

2.3.2 Effect of number of active transducer elements in DVE

Typically, the number of active transducer elements in a linear array is either 64 or 128. Since a larger number of transducers increase the power consumption significantly, we compared the performance of a 64 element array with a 128 element array. Figure 2.9 shows that there is no obvious difference between the performance of the two systems. The RMSE and maximum error measurements tell the same conclusion. Since 64 element system consumes about half the transmission power compared to 128 element system, it is clearly a better choice.

2.3.3 Effect of correction in DVE

Here we test the performance of Lagrange interpolation filter as a substitute of the quadrature correction method proposed in [10]. Four cases are investigated:

Case 1: DVE with no correction

Case 2: DVE with quadratic correction

Case 3: DVE with Lagrange interpolation with interpolation factor of 5

Case 4: DVE with Lagrange interpolation with interpolation factor of 9

From Figure 2.10 and Table 2.7, we see that the estimation results are close enough to the true velocity except for the one without correction. Lagrange interpolation filter proves to be a good substitute for the correction method proposed in [10]. The performance of interpolation factor of 5 is almost as good as that of interpolation factor of 9, with smaller computational cost.

	No correction	Quadratic correction	Lagrange with factor of 5	Lagrange with factor of 9
Maximum error (m/s)	0.2800	0.0539	0.0968	0.0502
RMSE (m/s)	0.1612	0.0257	0.0330	0.0302

Table 2.7 Performance of correction methods

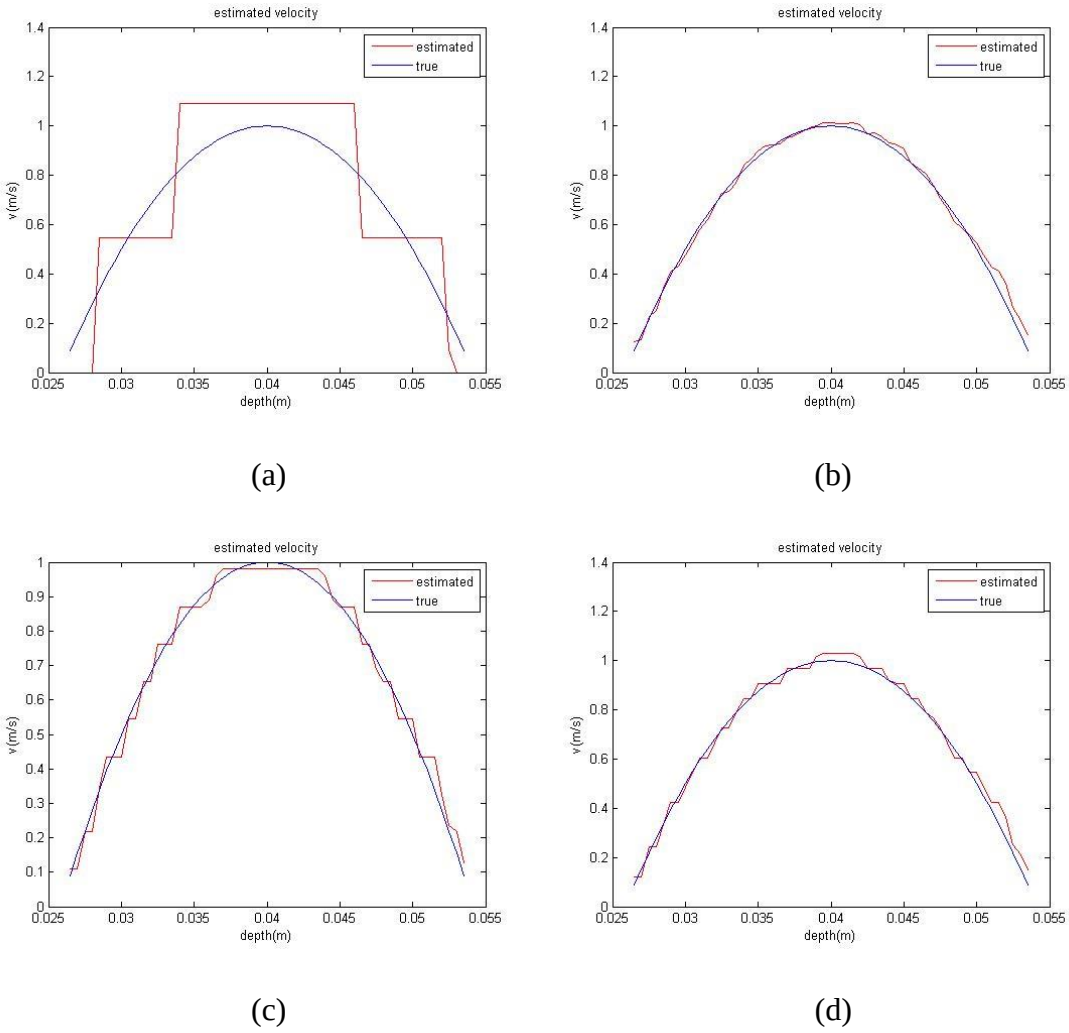


Figure 2.10 Simulation results for different correction methods (a) no correction; (b) quadratic correction; (c) Lagrange interpolation with factor of 5; (d) Lagrange interpolation with factor of 9.

2.3.4 Effect of approximations in delay calculations for DVE

Figure 2.11 shows the simulation results using the Taylor expansion approximations for delay calculation. Note that interpolation filter with a factor of 5 is used for correction here. The RMSE is 0.0385 and the maximum error is 0.0683, which are almost identical to the results without approximation.

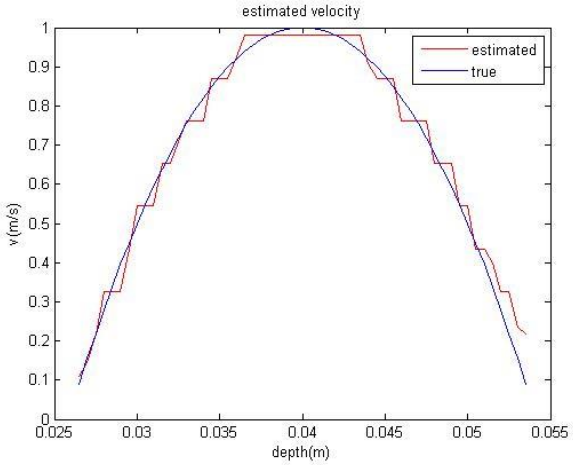
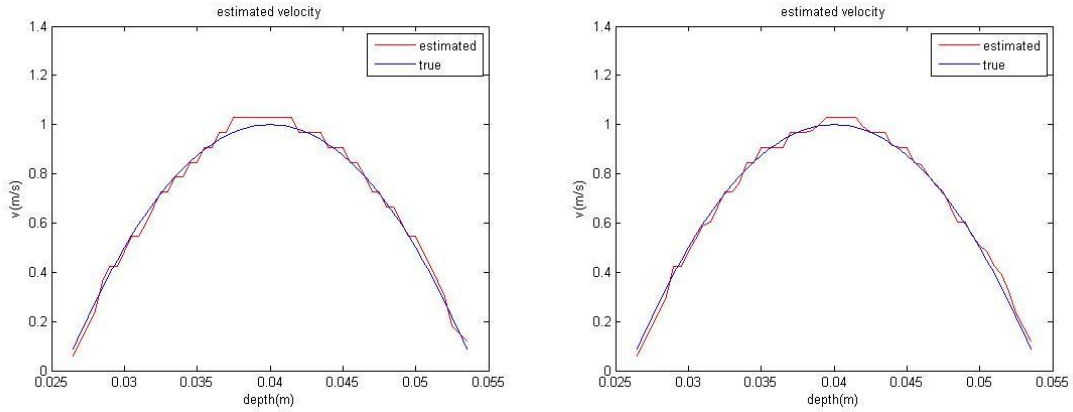


Figure 2.11 Velocity estimation performance with approximations in delay calculations

2.3.5 Effect of number of firings

We observe that the difference between the velocity estimation of successive firings of a DVE system is small. Therefore, it is possible to maintain the performance with fewer firings in one direction. We investigate two scenarios: 3 firings and 8 firings, with beam-flow angle of 45 degrees. The interpolation factor for both scenarios is 9. Figure 2.12 and Table 2.8 shows the estimation results, which suggest that the performance loss is very small with less number of firings. Less firings results in fewer computations and improves frame rate. However, a system based on fewer firings could be susceptible to noise. Therefore, next we investigate the effect of noise on the performance of our system.



(a) (b)

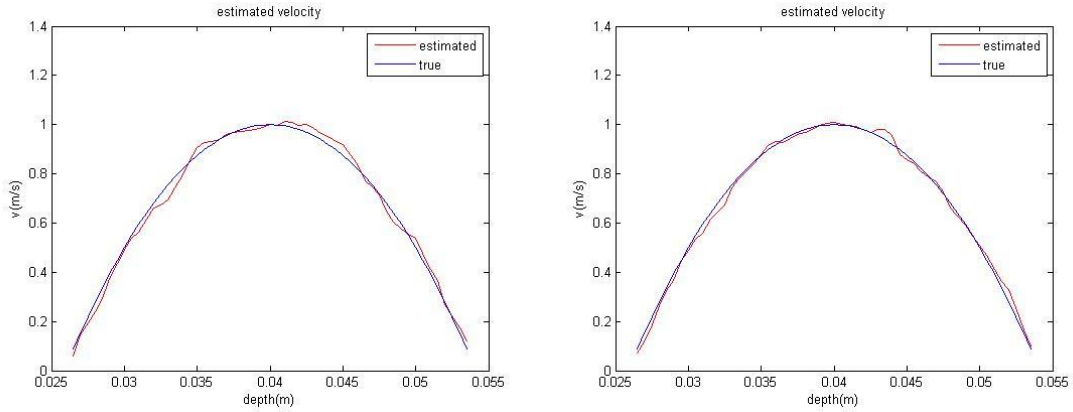
Figure 2.12 (a) 3 firings in one direction (b) 8 firings in one direction

	3 firings	8 firings
Maximum error (m/s)	0.0902	0.0683
RMSE (m/s)	0.0393	0.0385

Table 2.8 Effect of number of firings

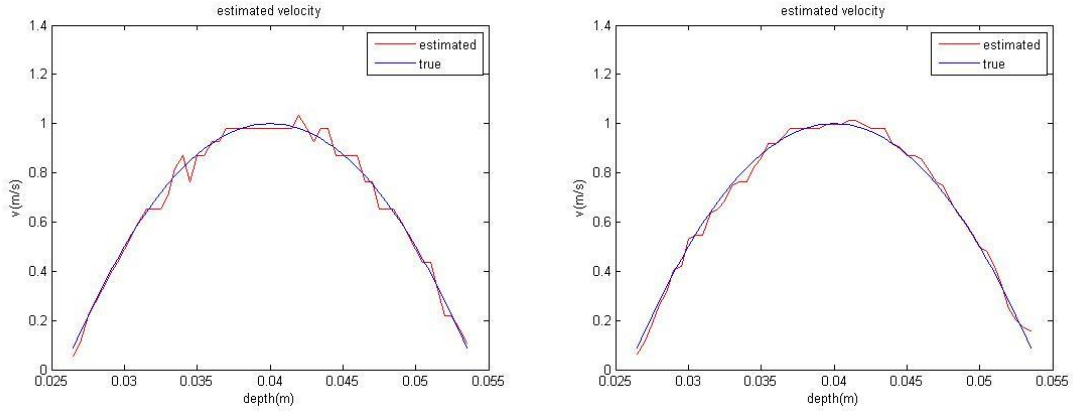
2.3.6 Effect of noise

We investigate two scenarios, one with 3 firings and one with 8 firings; the other system settings are the same. White noise is inserted before the beamformation. Three SNR configurations are considered: 23 dB, 13 dB and 3 dB. The results are shown in Figure 2.13, Figure 2.15 and Table 2.9. In all cases, we see that estimations with 3 firings are slightly worse than those with 8 firings. But considering the reduction in computational complexity, a system with 3 firings is worth pursuing. Noise did not affect the performance very much unless it has almost the same magnitude with the signal (in the case when SNR = 3 dB).



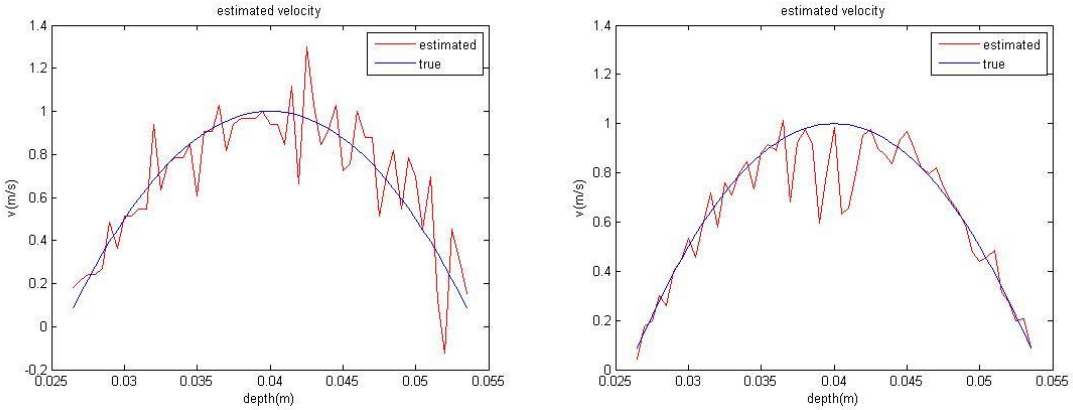
(a) (b)

Figure 2.13 Velocity estimation with SNR = 23 dB (a) 3 firings (b) 8 firings



(a) (b)

Figure 2.14 Velocity estimation with SNR = 13 dB (a) 3 firings (b) 8 firings



(a) (b)

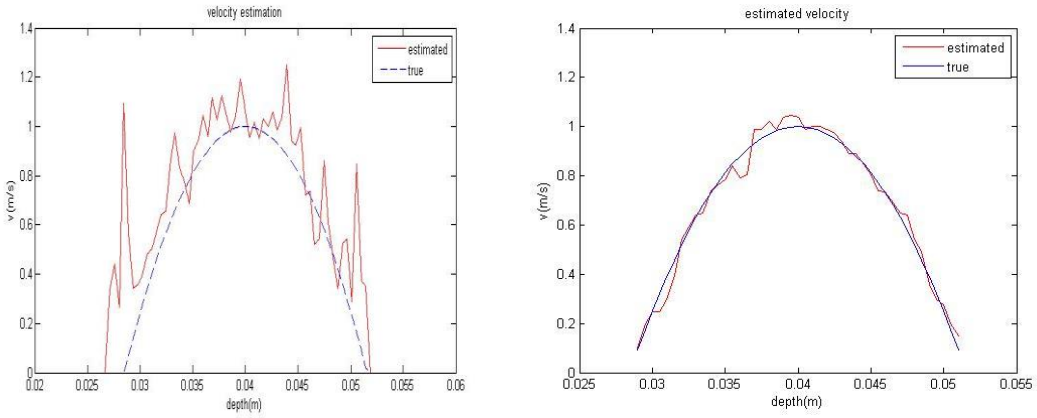
Figure 2.15 Velocity estimation with SNR = 3 dB (a) 3 firings (b) 8 firings

	SNR = 23 dB		SNR = 13dB		SNR = 3 dB	
	3 firings	8 firings	3 firings	8 firings	3 firings	8 firings
Maximum error	0.0683	0.0514	0.0865	0.0668	0.3889	0.3987
RMSE	0.0342	0.0321	0.0396	0.0365	0.1443	0.1114

Table 2.9 Effect of noise on velocity estimation performance

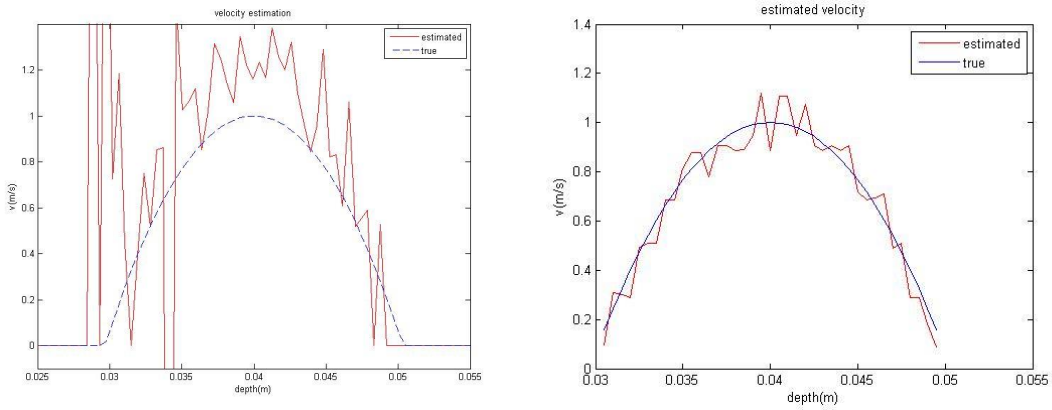
2.3.7 Results with different beam to flow angles

In this section, beam to flow angle of 60 degrees and 75 degrees are investigated. From Figure 2.16 and Figure 2.17, it is obvious that with the increase of the beam to flow angle, the estimation performance degrades severely for CVE, while DVE still provides accurate estimations. In CVE, this is because when the beam to flow angle increases, the velocity component on beam axis becomes smaller and thus harder to detect. For large beam to flow angles, a larger interpolation factor is necessary. These simulation results are based on interpolation factor of 9. In DVE, for beam to flow angle of 60 degrees, RMSE is 0.0393 and standard deviation is 0.0669. For beam to flow angle of 75 degrees, RMSE is 0.0700 and standard deviation is 0.1222.



(a) (b)

Figure 2.16 Results for beam to flow angle 60 degree for (a) CVE (b) DVE



(a) (b)

Figure 2.17 Results for beam to flow angle 75 degree for (a) CVE (b) DVE

2.4 Conclusion

We investigated the performance of CVE and DVE for estimating velocity. CVE provides acceptable estimation results when the beam-flow angle is 45 degrees, while DVE generally gives accurate estimation results when the beam-flow angle is 45 degrees and acceptable results when the beam-flow angle is 60 or 75 degrees, at the cost of higher computational complexity. A direct implementation of DVE includes division and square root operations, which are very expensive in terms of hardware cost. Therefore, techniques to remove those operations are described and implemented with small extra cost. We find that the estimation accuracy is hardly affected by the approximation techniques. We can even use fewer firings than the typical 8 firings in DVE. The performance of 3 firings is comparable to that of 8 firings, with or without inserted noise. With larger beam to flow angle, DVE can still provide valid estimation results. However, the interpolation factor needs to be larger to ensure reasonable performance.

CHAPTER 3

BACKEND PROCESSING

Backend processing is done after the acquisition of the beamformed radio frequency data (RF-data). For B-mode imaging, backend processing generally consists of envelope detection, log compression and scan conversion, as shown in Figure 3.1. In this chapter, all three blocks are described and their computational complexity and performance trade-offs are analyzed.

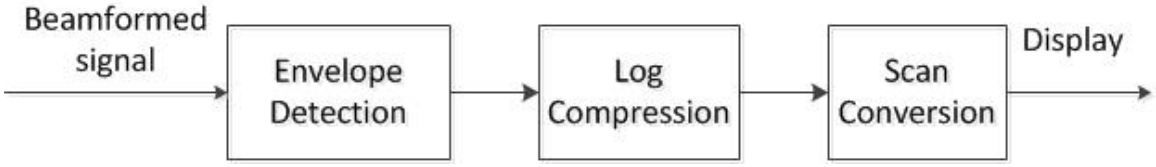


Figure 3.1 Block diagram of backend processing (adapted from [3])

3.1 Envelope Detection

The RF-data acquired after beamforming are oscillating signals with a limited bandwidth and no DC component. The envelope of the RF-data represents the low-frequency variation due to the different scattering properties of the tissue structures. The envelope can be ideally detected using the Hilbert Transform (HT) followed by computation of the magnitude of the complex signals [13]. For a discrete beamformed signal $r(n)$, the HT acts as an ideal 90-degree phase shifter, which provides the imaginary term $r_H(n)$in the analytic representation of the signal, given by $r_a(n) = r(n) + jr_H(n)$. The envelope $E(n)$ is then obtained by computing the magnitude of $r_a(n)$:

$$E(n) = \sqrt{r^2(n) + r_H^2(n)} \tag{3.1}$$

There are several techniques for envelope detection. These include FFT based Hilbert Transform [14], FIR based Hilbert Transform and Quadrature demodulation [15].

3.1.1 Computing the analytic signal using FFT

The Fourier Transform of a real-valued signal is complex symmetric. By removing the negative frequency of the spectrum, we can obtain the analytic representation of the signal [14]. For the RF data acquired, we can replace the FFT coefficients corresponding to negative frequencies with zeros and then implement inverse FFT, as described in Figure 3.2. The envelope of the signal is then found by computing the magnitude of the analytic signal.

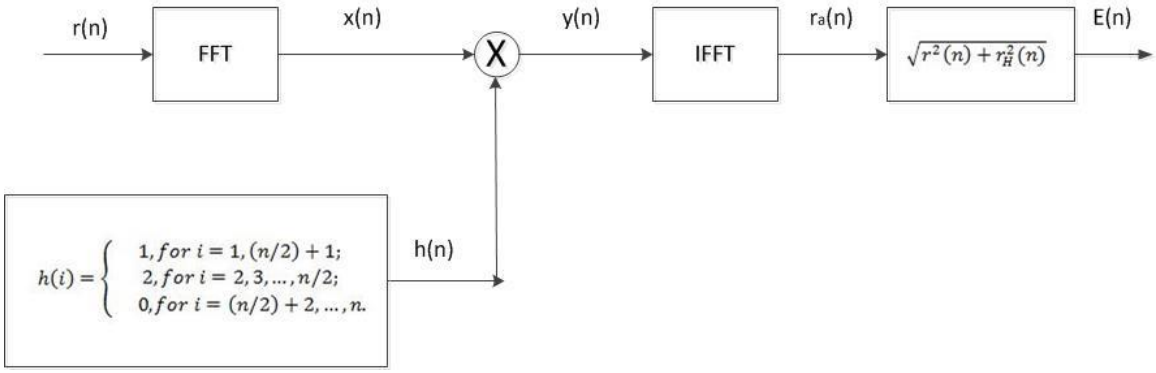


Figure 3.2 Computing the discrete-time analytic signal using FFT [14]

3.1.2 FIR based Hilbert Transform

Hilbert Transform can also be implemented by using a FIR filter or IIR filter [17]. A design method of linear phase FIR Hilbert transform using eigen filter was proposed in [16]. Least square and minimax fitting based methods were used for FIR and IIR filter design in [17]. We designed the FIR based Hilbert filter using MATLAB. The imaginary term $r_H(n)$ is obtained after filtering. The envelope can be computed using equation (3.1). The block diagram of this method is shown in Figure 3.3.

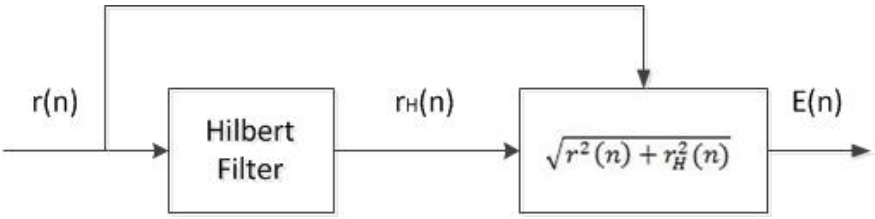


Figure 3.3 Block diagram for FIR based Hilbert Transform

3.1.3 Quadrature Demodulation

Quadrature demodulation generates the in-phase and quadrature components by multiplying with sine and cosine carriers followed by low pass filtering. Let I(n) and Q(n) be the in-phase and quadrature components. Assume that the carrier frequency ω_0 is known and does not vary with time. Then the envelope is obtained by:

$$E(n) = \sqrt{I^2(n) + Q^2(n)} \tag{3-2}$$

The block diagram of this method is shown in Figure 3.4, where T_s is the sampling period.

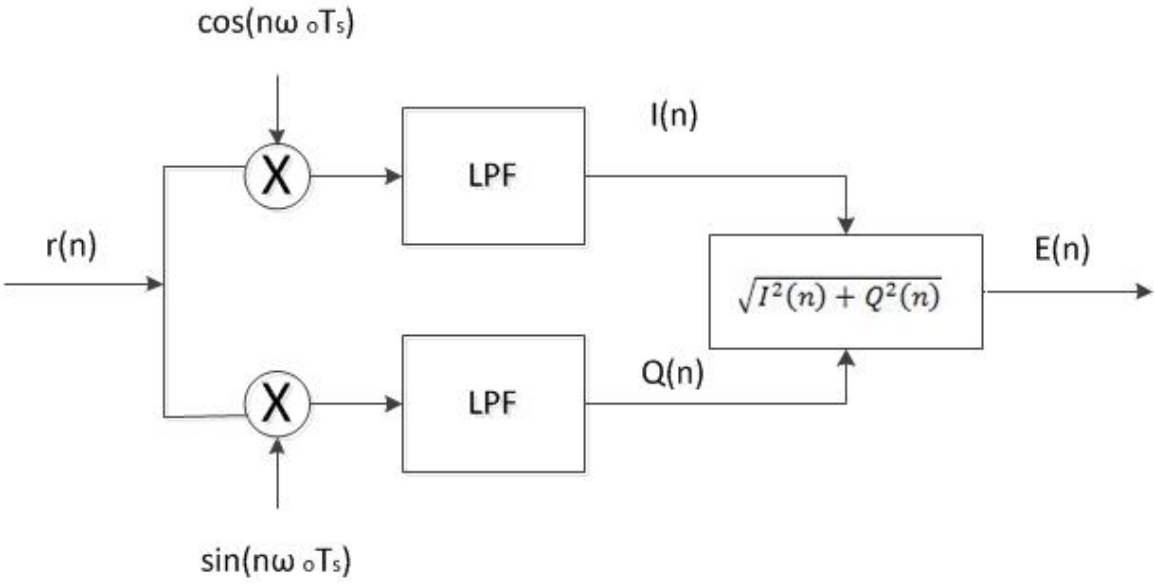


Figure 3.4 Block Diagram for Quadrature Demodulation (adapted from [15])

3.1.4 Simulation results

Simulations are done for a randomly selected scanline of a synthetic aperture system for all three methods: FFT based Hilbert Transform, FIR based Hilbert Transform and quadrature demodulation. The parameters of the synthetic aperture system are described in Table 3.1. The simulation results are shown in Figure 3.5. The dashed line is the original signal and the others are the envelopes detected by the three methods. In this figure, 100 successive samples of the original signal and the detected envelopes are shown to have a clear display.

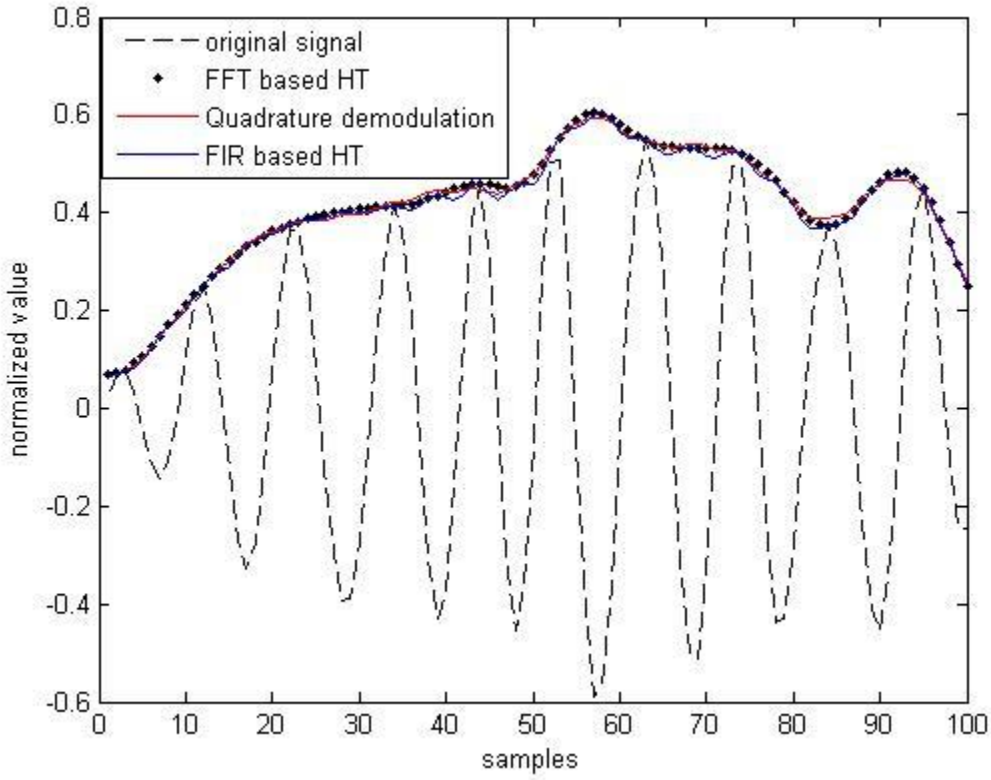


Figure 3.5 Envelope detection performance comparisons

The results of these three methods are almost identical, as shown in Figure 3.5. If the FFT based Hilbert transform method is considered ideal, we can calculate the RMSE of the

other two methods with respect to it. Accordingly, the FIR based Hilbert transform method has an RMSE of 0.0097 and the quadrature demodulation based method has an RMSE of 0.0076.

Name	Description	Value
N	Number of receiving elements	128
M	Number of transmission elements and number of transmission	32
B	6dB bandwidth of transducer	4 MHz
fs	A/D sampling frequency	40 MHz
fs'	Sampling frequency after interpolation	120 MHz
c	Speed of sound in body tissue	1540 m/s
R	Number of focal points in one scanline	3117
Q	Number of scanline in one image	200
θ_{range}	Angle range of the image	45 degrees

Table 3.1 Parameter definitions and values for synthetic aperture based imaging system

3.1.5 Computational complexity

Assume R is the number of samples of the RF data. For the FFT based Hilbert Transform method, an FFT and an inverse FFT are needed. The FFT is computed for N samples, where N is the next power of 2 of R. Thus, the complexity of this part will be in the order of 2*log(N)*N. Since log compression is done after taking the absolute value, the square root part can be avoided. Therefore, we only need 2*R multiplications, R additions, to obtain the magnitude of the signals.

For the FIR based Hilbert Transform method, if the order of the FIR filter is K_h, $K_h * R$ multiplications and additions are needed for filtering. Another $2 * R$ multiplications and R additions are needed to calculate the magnitude of the signals.

For the quadrature demodulation based method, $2 * R$ multiplications are needed to acquire the in-phase and quadrature components. Since the signals are shifted on base band, the Nyquist sampling rate is lower and thus we can downsample the signal by a factor of 5 to save computations. Assume the lowpass filter has an order of K_l, $2 * K_l * R/5$ multiplications and additions are needed for the filtering. Another $2 * R/5$ multiplications and $R/5$ additions are needed to calculate the magnitude of the signals.

In this work, simulations are done with R of 3117, K_h of 14, and K_l of 26. K_h and K_l are selected according to the performance requirement. For the proposed simulation setup, FFT based method requires about 105k multiplications, FIR based method requires about 50k multiplications, and quadrature demodulation requires about 40k multiplications. We see that FFT based Hilbert transform method is more computationally expensive and thus not preferred in our system. FIR based Hilbert transform method is close enough to FFT based Hilbert transform method and has number of computations comparable with quadrature demodulation. However, the phase information is lost if FIR based Hilbert transform method is used for envelope detection. Quadrature demodulation also provides accurate results and keeps the phase information in the in-phase and quadrature components of the signal. Therefore we choose to use quadrature demodulation as the envelope detection method in our system when phase information is necessary and we choose to use FIR based Hilbert transform method when phase information is not needed.

3.2 Log compression

Considering 256 gray levels, the dynamic range of human eyes is in the order of 30dB [3]. To display the B-mode image, a log compression is necessary in order to provide more detailed information for lower magnitude data. Log compression is generally done by first dividing the magnitude obtained from envelope detection by a fixed number (predetermined based on the expected maximum magnitude possible), and then adjusting the values with a threshold operator based on the actual dynamic range. Typically, the dynamic range used in B-mode ultrasound image is 40dB or 60dB, depending on the accuracy requirement [3]. This block is typically implemented by a look-up table.

3.3 Scan conversion

The scan lines from a phased array or synthetic aperture system are often in the polar-coordinate system. In order to display the image on a regular screen, which is under the Cartesian-coordinate system, we need scan conversion. To translate the input data under polar-coordinate system into the output data under Cartesian-coordinate system, interpolation is needed. In this section, two different methods for interpolation are described and the trade-offs analyzed.

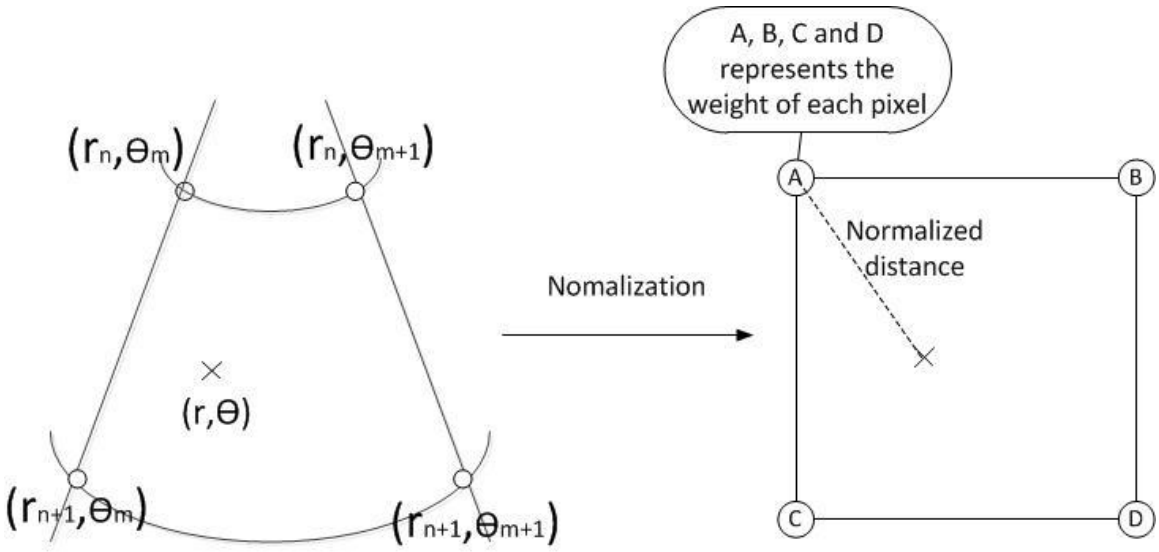


Figure 3.6 Interpolation in scan conversion

Let (r, θ) be the coordinates of the pixel whose intensity will be determined by the intensities of its four neighboring pixels whose coordinates are $(r_n, \theta_m), (r_n, \theta_{m+1}),$ $(r_{n+1}, \theta_m), (r_{n+1}, \theta_{m+1})$. Figure 3.6 describes the above configuration, which corresponds to an interpolation window of size 2×2. The intensity of pixel at (r, θ) is given by:

$$I(r, \theta) = A * I(r_n, \theta_m) + B * I(r_n, \theta_{m+1}) + C * I(r_{n+1}, \theta_m) + D * I(r_{n+1}, \theta_{m+1}) \quad (3\text{-}3)$$

where A, B, C and D are the weighting coefficients of the four neighboring pixels. The calculation of the weighting coefficients is related to the normalized geometric distance between input and output data. For instance, the normalized distances from point (r_n, θ_m) to point (r, θ) can be computed as:

$$D_r = \frac{|r - r_n|}{\Delta r}$$

$$D_\theta = \frac{|\theta - \theta_m|}{\Delta \theta} \quad (3\text{-}4)$$

where $\Delta r = |r_{n+1} - r_n|$, and $\Delta \theta = |\theta_{m+1} - \theta_m|$.

3.3.1 Bilinear Interpolation

Bilinear interpolation is commonly used as the method for polar-Cartesian coordinate conversion because of its relatively low computational complexity. The weighting coefficients are calculated based on the product of the normalized distances along $r\ and\ \theta$ [19]. For example, weight A is given by $\frac{|r-r_n|}{\Delta r} * \frac{|\theta-\theta_m|}{\Delta\theta}$, weight B is given by $\frac{|r-r_n|}{\Delta r} * \frac{|\theta-\theta_{m+1}|}{\Delta\theta}$ and so on.

3.3.2 Gaussian Interpolation

The weighting coefficients in Gaussian interpolation are chosen according to the Gaussian function. For instance, the weighting coefficient A can be expressed as:

$$A = exp\left[-\left(\frac{D_r^2}{\sigma_r^2} + \frac{D_\theta^2}{\sigma_\theta^2}\right)\right] \tag{3-6}$$

where $D_r\ and\ D_\theta$ are defined by equation (3.4) and $\sigma_r$and $\sigma_\theta$are the standard deviations of corresponding dimensions [20].

3.3.3 CNR calculation

In order to compare the performance of the two interpolation methods, we use Contrast-to-noise ratio (CNR) as the performance metric. The CNR is calculated after envelope detection and log compression. We analyze the performance of the different schemes using a cyst image. The CNR is given by

$$CNR = \frac{|\mu_{cyst}-\mu_{background}|}{\sqrt{\sigma_{cyst}^2+\sigma_{background}^2}} \tag{3-7}$$

where cyst is the black area without scatterers in the center and background is the area surrounding it, as shown in Figure 3.7. The mean μ and standard deviation σ are calculated for the cyst and the background [21]. In this work, the background area is

considered as the ring area with three times larger radius than the cyst. However, in order to avoid the fuzzy area near the cyst-background boundary, we consider 10%, 20% or 30% smaller radius of the cyst. The corresponding CNRs are named CNR1, CNR2, CNR3, as shown in Figure 3.7.

3.3.4 Simulation results

The simulations are done with MATLAB R2012a, and the Field II simulation platform [22][23][24]. The cyst is at a depth of 60 mm, and has a radius of 5 mm. The phantoms around the cyst consist of 20,000 random scatterers. Synthetic aperture with 32 transmissions and 128 receive elements is used. The dynamic range is 60 dB. Other system parameters were described in Table 3.1 [7]. After the acquisition of the beamformed data, multiple scan conversion settings are tested on the same data for fairness of comparison. An example of the displayed image is shown in Figure 3.8.

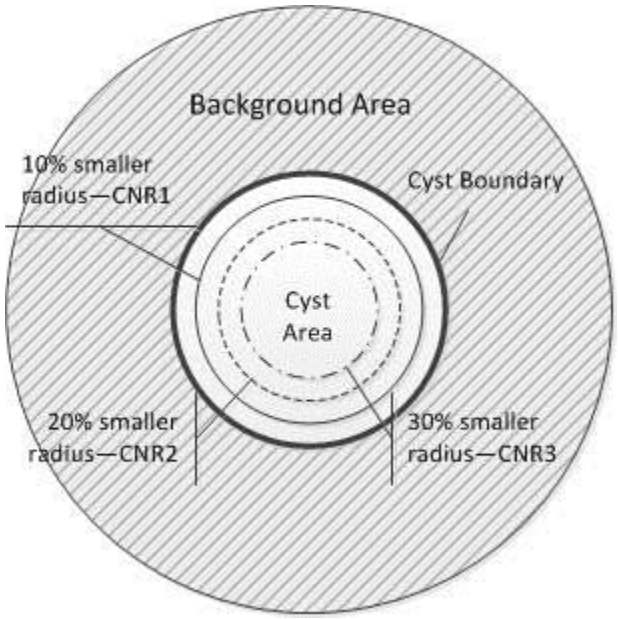


Figure 3.7 CNR calculation

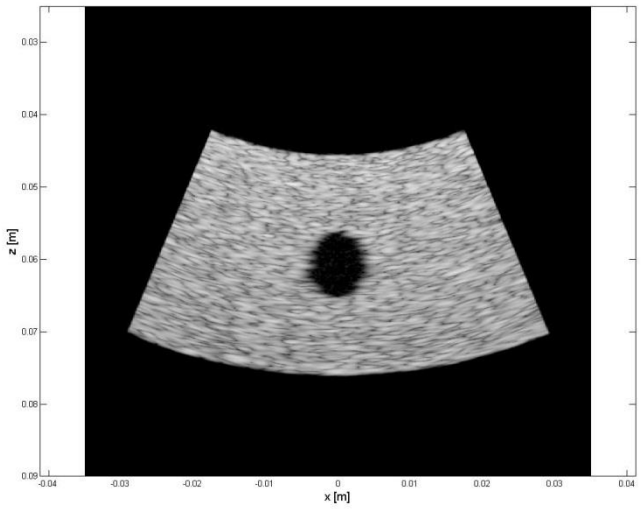


Figure 3.8 Resultant image after scan conversion

3.3.4.1 Analysis on σ

For Gaussian interpolation, σ needs to be carefully selected in order to produce reasonable results. Since the distances from the input data points to the point to be interpolated at the center are normalized, σ for both dimensions should be the same. Therefore, in this section, only one σ will be discussed. Figure 3.9 shows a parametric analysis of σ based on CNR performance.

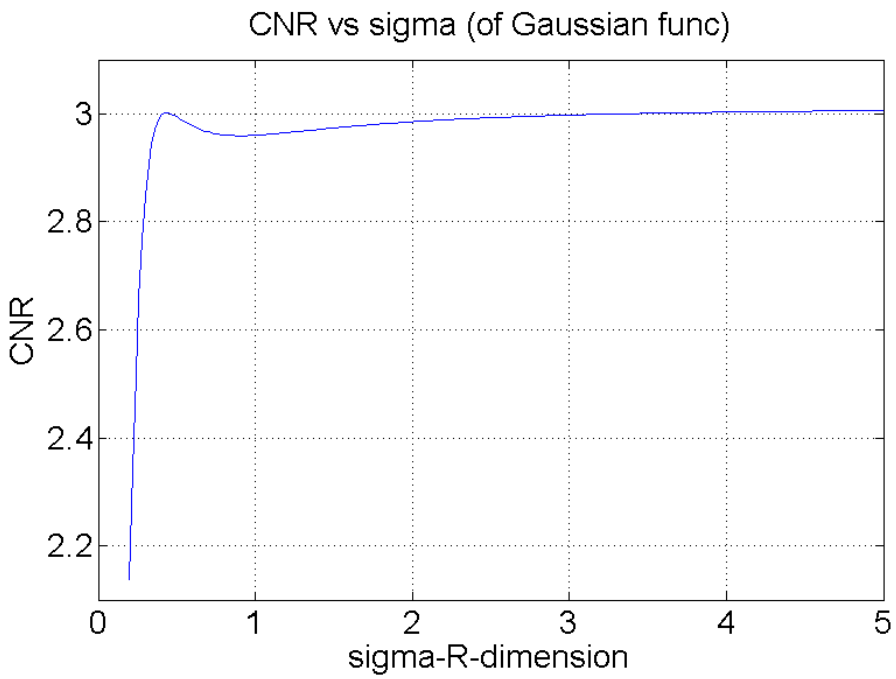


Figure 3.9 Analysis of CNR as a function of standard deviation

We can see there is a local minimum of σ at 0.44. Even though we can achieve the same CNR when σ is larger than 5, we do not consider it. This is because a large σ makes the Gaussian interpolation approximately equivalent to taking the average of the neighboring pixels and is not effective when the pixels are far apart.

3.3.4.2 Analysis on window size

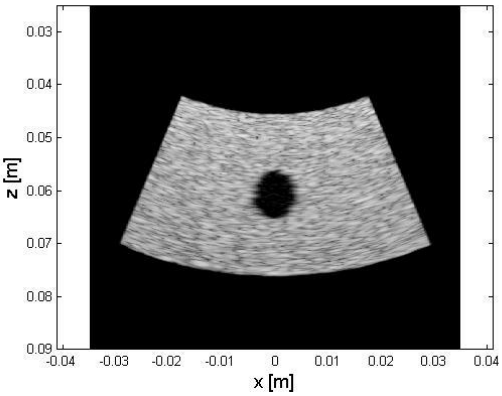
Earlier, the interpolation schemes for window size 2×2 have been described. Larger window sizes have the potential of providing better results since more information is taken into consideration. First we investigate the scenario with 200 scanlines and 3117 samples on each scanline. The other system parameters are described in Table 3.1. Bilinear interpolation and Gaussian interpolation with different window sizes are used for scan conversion in both scenarios. The resultant figures after scan conversion are shown in Figure 3.10. The CNR performance results are shown in Table 3.2.

		Bilinear interpolation	Gaussian interpolation
Window size 2×2	CNR1	2.06	2.06
	CNR2	3.08	3.09
	CNR3	4.68	4.68
Window size 4×4	CNR1	--	2.06
	CNR2		3.08
	CNR3		4.67

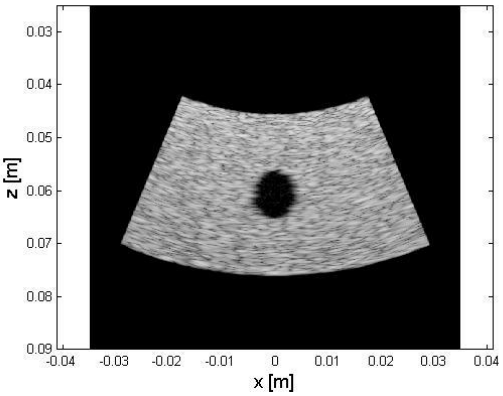
Table 3.2 CNR results for different interpolation methods and different window sizes, when there are 200 scanlines and 3117 samples on each scanline.

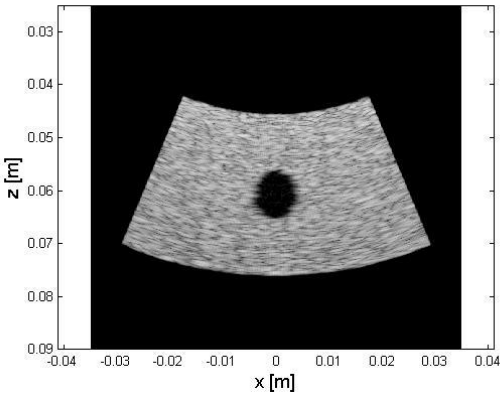
		Bilinear interpolation	Gaussian interpolation
Window size 2×2	CNR1	2.08	2.08
	CNR2	3.14	3.15
	CNR3	4.89	4.90
Window size 4×4	CNR1	--	2.09
	CNR2		3.15
	CNR3		4.92

Table 3.3 CNR results for different interpolation methods and different window sizes, when there are 100 scanlines and 623 samples on each scanline.



(a)





(b) (c)

Figure 3.10 Comparison of the interpolation performance with (a) window size 2×2; bilinear interpolation (b) window size 2×2; Gaussian interpolation (c) window size 4×4; Gaussian interpolation, when there are 200 scanlines and 3117 samples on each scanline.

We see that both interpolation methods are sufficient in terms of performance. Window size 4×4 of Gaussian interpolation provides slightly better CNR values but requires about 4 times more computations than window size 2×2.

Next we repeat this experiment for a scenario with 100 scanlines and 623 samples on each scanline. Table 3.3 presents the CNR results for this scenario. We see that in this case a 4×4 window results in a minor improvement in the CNR value. Our overall

conclusion is that for a dense sampling grid, such as the one used in our system, a 2×2 window is sufficient to attain high CNR values.

3.3.4.3 Computational complexity

From Table 3.2, we can see that the performance of bilinear interpolation and Gaussian interpolation are almost identical. However, Gaussian interpolation needs 3 more multiplications, 1 more addition, and an extra look-up table for the exponential function compared to bilinear interpolation. Furthermore, Gaussian interpolation could result in interpolated pixel value being larger than the dynamic range since the weighting coefficients do not always add up to 1. This requires additional processing to adjust the dynamic range after interpolation. Bilinear interpolation does not have the problem as the coefficients always add up to 1. Therefore, bilinear interpolation is a better choice considering both performance and computational complexity and we choose to use it in our system.

3.4 Summary

Backend processing, which includes envelope detection, log compression and scan conversion, has been discussed in this chapter. For envelope detection, both FIR based Hilbert transform and quadrature demodulation provide identical results with the FFT based Hilbert transform, which is more computationally expensive. FIR based Hilbert transform is a better choice when phase information is not needed after envelope detection, while quadrature demodulation is chosen if phase information is necessary. Log compression is implemented with a look up table, and the square root operation in envelope detection is avoided because of log compression. In scan conversion, bilinear

and Gaussian interpolation provide comparable results. We choose to use bilinear interpolation because it is less computationally expensive. Also, for our system a 2×2 window interpolation provides sufficiently good performance.

CHAPTER 4

CONCLUSION

This thesis report described our work on increasing the efficiency of implementing Doppler processing and backend processing in ultrasound imaging systems. Two different Doppler processing methods, one with beamformation along the scan line (CVE) and one with beamformation along the flow direction (DVE), were compared. Simulations were done with a simple parabolic velocity profile in a linear array system. While DVE provides better estimation performance and is effective for large beam to flow angles, it has higher computational cost. Not only are the number of multiplications and additions larger, a direct implementation also involves operations such as square root and division, which have a large hardware cost. We proposed two approximation techniques to replace square root and division operations with multiplications and additions. Specifically we used the Lagrange interpolation filter to replace division and the Taylor series expansion to avoid square root. For beam to flow angle of 45 degrees, a second order Lagrange interpolation filter with interpolation factor of 5 is sufficient, however if the beam to flow angle is larger (60 or 75 degrees), a larger interpolation factor is necessary to achieve high estimation performance. Furthermore, we studied the effect of number of firings and found that 3 firings are sufficient as opposed to 8 firings used in [10].

The purpose of backend processing is to properly display the ultrasound image on the screen. There are three major blocks for backend processing: envelope detection, log compression and scan conversion. In this work, three different envelope detection methods were compared with respect to performance and hardware cost. Among them,

FIR based Hilbert Transform is considered to be the best choice when phase information is not needed, while quadrature demodulation works better if phase information is necessary. Bilinear and Gaussian interpolation were considered for scan conversion. In the simulations with a cyst image, bilinear interpolation was shown to have comparable contrast to noise ratio (CNR) performance with Gaussian interpolation. Since bilinear interpolation has lower computational complexity, we choose to use bilinear interpolation in our system.

There are several related problems that will be considered in the near future. First, we plan to consider velocity estimation in the presence of stenosis. The velocity profile is no longer parabolic through the constricted region. However we can exploit the fact that the maximum velocity at the center of the vessel will be larger in presence of stenosis. We plan to do several experiments on automatically generated flow profiles as well as real data to develop a robust algorithm to identify stenosis [27].

Next we plan to derive a method to remove the dependence on prior knowledge of beam to flow angle θ. In both the methods considered here, θ was considered known from the B-mode image. Transverse oscillation approach has been proposed to generate vector flow images [25]. Alternatively, if we can measure the velocity components in two orthogonal directions, we may be able to find the true velocity vector combining them.

Another direction of future research is to develop a volumetric analysis based on the velocity information. The flow volume through a certain transection gives more direct information [25]. It helps in determination of bifurcation and abnormal blood flow.

Finally, we plan to extend this work to handle 3D velocity estimation. We will build an accelerator that will work with our 3D B-mode imaging architecture [9]. The

computational flow will have to be further optimized to reduce the power consumption of the hardware system.

REFERENCES

[1] R. Cobbold, Foundations of Biomedical Ultrasound. Oxford University Press, USA, 2007.

[2] J. U. Quistgaard, "Signal Acquisition and processing in medical diagnostic ultrasound," IEEE Signal Processing Magazine, vol. 14, no. 1, pp. 67-74, Jan. 1997.

[3] Murzata Ali, "Signal Processing Overview for Ultrasound Systems for Medical Imaging," TI White Paper, Nov. 2008.

[4] J. A. Jensen, S. I. Nikolov, K. L. Gammelmark, M. H. Pedersen, "Synthetic aperture ultrasound imaging," Ultrasonics, Volume 44, Supplement, 22 December 2006, pp. e5-e15.

[5] M. Karaman, I. O. Wygant, O. Oralkan, and B. T. Khuri-Yakub, "Minimally redundant 2-D array designs for 3-D medical ultrasound imaging," IEEE Transactions on Medical Imaging, vol. 28, no. 7, pp. 1051-1061, July 2009.

[6] T, Misaridis and J. A. Jensen, "Use of modulated excitation signals in medical ultrasound. Part I: basic concepts and expected benefits," IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control, vol. 52, no. 2, pp. 177-191, Feb. 2005.

[7] M. Yang, S. Wei and C. Chakrabarti, "Reducing the complexity of orthogonal code based synthetic aperture ultrasound system," IEEE Workshop on Signal Processing Systems (SiPS), pp.270-275, Oct. 2012.

[8] D. Weinreb and J. Stahl, "The introduction of a portable head/neck CT scanner may be associated with an 86% increase in the predicted percent of acute stroke patients treatable with thrombolytic therapy," Radiological Soc. of N. America, pp. 1-2, Dec. 2008

[9] R. Sampson, M. Yang, S. Wei, C. Chakrabarti, and T. F. Wenisch, "Sonic Millip3De: Massively parallel 3D stacked accelerator for 3D ultrasound," 19th IEEE International Symposium on High Performance Computer Architecture, Feb. 2013.

[10] J. A. Jensen, "Directional velocity estimation using focusing along the flow direction I: Theory and simulation," IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control, vol. 50, no. 7, pp. 857-872, 2003.

[11] J. A. Jensen and S. I. Nikolov, “Directional Synthetic Aperture Flow Imaging.” IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control, vol. 51, no. 9, pp. 1107-1118, 2004.

[12] J. A. Jensen, “Stationary Echo Canceling in Velocity Estimation by Time-domain Cross Correlation,” IEEE Transactions on Medical Imaging, vol.12, no.3, pp.471-477, Sep. 1993.

[13] M. Schlaikjer, J. P. Bagge, O. M. Sorensen, and J. A. Jensen, “Trade off study on different envelope detectors for B-mode imaging,” in Proc. IEEE Ultrason. Symp., 2003, pp. 1938-1941.

[14] S. L. Marple, “Computing the Discrete-Time “Analytic” Signal via FFT, IEEE Transactions on Signal Processing, vol.47, no.9, pp.2600-2603, September 1999.

[15] C. Jin, J.T. Yen, K.K. Shung, "A Novel Envelope Detector for High-Frame Rate, High-Frequency Ultrasound Imaging," IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control, vol.54, no.9, pp.1792-1801, September 2007.

[16] S. C. Pei and J. J. Shyu, “Design of FIR Hilbert transformers and differentiators using eigenfilter,” IEEE Transactions on Circuits and Systems, vol.35, no.11, pp.1457-1461, Nov. 1988.

[17] M. Z. Komodromos, S. F. Russell and P. T. P. Tang, “Design of FIR Hilbert Transformers and Differentiators in the Complex Domain,” IEEE Transactions on Circuits and Systems—I: Fundamental Theory and Applications, vol.45, no.1, Jan. 1998.

[18] I. Kollar, R. Pintelon and J. Schoukens, “Optimal FIR and IIR Hilbert transformer Design via LS and Minimax Fitting,” IEEE Transactions on Instrumentation and Measurement, vol.39, no.6, pp. 847-852, Dec. 1990.

[19] X. Li, “Ultrasound Scan Conversion On TI’s C64x+ DSPs,” TI application Report, Mar. 2009.

[20] S. Kestur, S, Park, K.M. Irick, V. Narayanan, "Accelerating the Nonuniform Fast Fourier Transform Using FPGAs," 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), pp.19-26, May 2010.

[21] K. F. Üstüner and G. L. Holley, “Ultrasound Imaging System Performance Assessment,” Siemens Medical Solutions USA, Inc. Ultrasound Division, 2003.

[22] J. A. Jensen, “Field: A program for simulating ultrasound systems,” in 10th Nordicbaltic Conference on Biomedical Imaging, Vol. 4, Supplement 1, Part 1: 351-353, 1996, pp. 351-353.

[23] J. Jensen and N. Svendsen, “Calculation of pressure fields from arbitrarily shaped, apodized, and excited ultrasound transducers,” IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control, vol. 39, no. 2, pp. 262-267, Mar 1992.

[24] J. A. Jensen, P. D. Fox and L. Taylor, “simulation of non-linear ultrasound fields,” in Proceedings of IEEE Ultrasonics Symposium, 2002, vol. 2, 2002, pp. 1733-1736.

[25] J. A. Jensen, S. I. Nikolov, J. Udesen, P. Munk, K. L. Hansen, M. M. Pedersen, P. M. Hansen, M. B. Nielsen, N. Oddershede, J. Kortbek, M. J. Pihl and Y. Li, “Recent Advances in Blood Flow Vector Velocity Imaging,” 2011 IEEE Ultrasonics Symposium Proceedings, pp.262-271.

[26] O. D. Kripfgans, J. M. Rubin, A. L. Hall, M. B. Gordon and J. B. Fowlkes, “Mesurement of Volumetric Flow,” J Ultrasound Med, 2006, vol.25, pp. 1305-1311.

[27] D. H. Evans, J. A. Jensen and M. B. Nielsen, “Ultrasonic colour Doppler imaging,” *Interface Focus (2011)*, pp. 490-502, May 2011