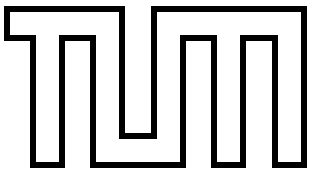


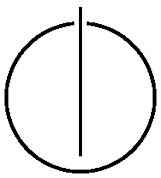
FAKULTÄT FÜR INFORMATIK

DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Interdisciplinary Project within the minor subject
Medicine
as part of the Bachelor studies in Informatics

RF2B - RF to B-Mode Ultrasound

Author:	Andre Hong Lam Dau
Advisors:	Prof. Dr. Nassir Navab
	Dipl.-Inf. Univ. Christian Wachinger
	MSc Athanasios Karamalis
Date:	February 15, 2010



Contents

Chapter 1

Introduction

1.1 Motivation

Diagnostic sonography is an important ultrasound-based medical imaging technique. It is easy to apply and the risk of exposure is relatively low. Therefore it is widely used and approved as a standard diagnostic imaging technique.
There is still research going on involving ultrasound imaging. When using an ultrasound system for different purposes and under different conditions it has to be adapted to the specific requirements. Especially research environments can vary considerably. The adjustment requires a certain amount of understanding regarding the ultrasound system and its image processing pipeline.
Furthermore images can only be interpreted properly by the researcher if he or she can distinguish image artifacts from real structures. Therefore it is important that the researcher knows the exact processing steps performed by the ultrasound system. However, most vendors do not reveal the exact processing and filtering methods used. They sell their systems as a set of black boxes which makes it difficult to determine the exact processing steps as well as the effects they have on the image.
The project focuses on the part of the system which generates brightness modulation (B-Mode) images out of digitalized radio frequency (RF) data obtained by the ultrasound probe. Brightness modulation images are used to make reflections of ultrasound waves within the body visible for humans. They show a two-dimensional map of the scanned body region based on ultrasound data. In clinical diagnostics physicians normally work with B-Mode images.

1.2 Objectives

The primary objective of the project is to develop a tool in MATLAB which should assist researchers in determining the necessary processing steps to transform digital radio frequency data to brightness modulation images. These transformation steps represent only one part of the whole processing pipeline of an ultrasound system.
The tool will provide the user with a framework for the processing pipeline. The framework contains the coarse processing steps. The user can customize the pipeline by choosing different filters and processing methods which should be applied in the different steps.
It visualizes the effect of the processing methods selected after each step in order to help the user finding the optimal filter combinations and parameters. To further assist the user the tool provides additional information on the RF data and the ultrasound probe.

Chapter 2

The Ultrasound System

2.1 Specifications

The examined system is the Sonix RP ultrasound system from Ultrasonix with probe model C5-2/60.
The pipeline implemented by the tool is based on the Sonix RP ultrasound system. The information on the probe needed for several calculations is taken from the specifications of the probe model C5-2/60.

Specifications of the Probe C5-2/60:

Type:	convex
Elements:	128
Center Frequency:	3.5 MHz
Fractional Bandwidth:	Min 60 % (at -6dB)
Elevation Height:	13 mm
Element Width:	0.424 mm
Element Pitch:	0.4792 mm
Kerf:	0.055 mm
Elevation Focus:	97 mm
Radius:	60 mm

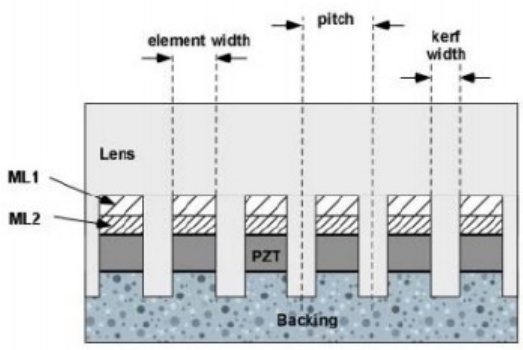
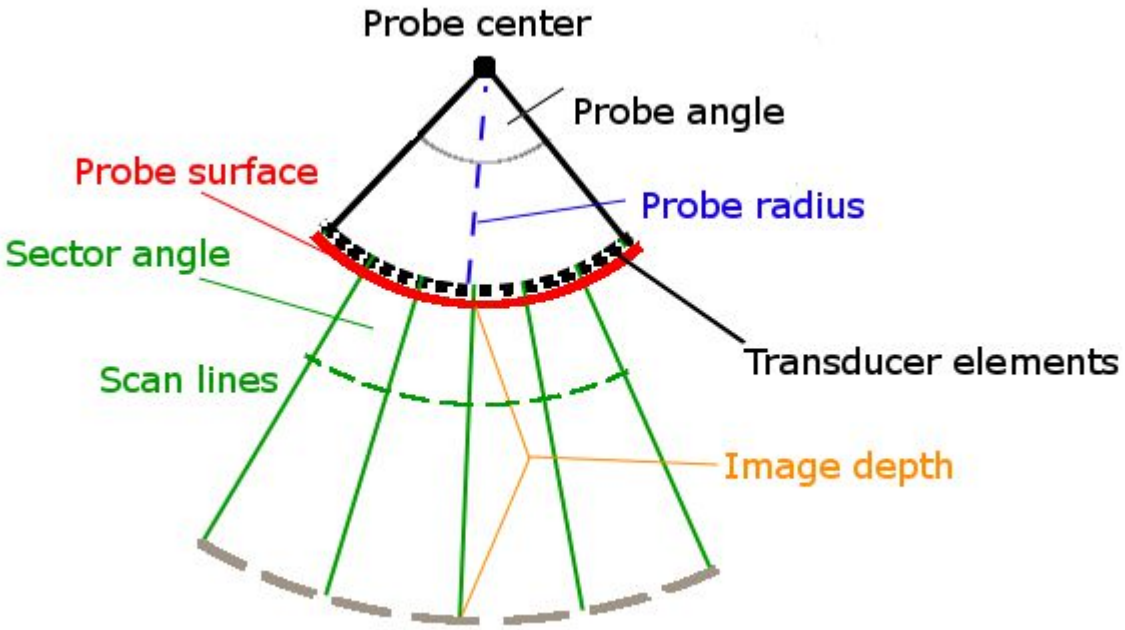


Fig.1 Structure of Elements

2.2 Terms and Definitions



2.2.1 Transducer Elements

The convex shaped probe model C5-2/60 of the Sonix RP ultrasound system consists of 128 transducer elements which transmit and receive ultrasound signals. An ultrasound beam is formed by activating multiple elements at once.

2.2.2 Geometry

The transducer elements are aligned on an imaginary circle. The radius of the circle is called the probe radius. The angle between the outermost elements is called the probe angle and determines the maximum angle of the field of view. The part of the circle corresponding to the probe angle is called the probe surface. The pitch denotes the uniform distance between the centers of two adjacent elements.
The C5-2/60 has a radius of 6cm and a probe angle of approximately 58 degrees. The pitch between the elements is 0,4792 mm which means that the probe surface is approximately 6,09 cm long.

2.2.3 Field of View

The field of view is the part of the maximum probe angle which is actually analyzed during a single scan. The angle of the field of view is called the sector angle. The sector size is the size of the field of view relatively to the maximum field of view given by the probe angle. The depth of the image is the height of the field of view. So the actual distance of the farthest points of the field of view to the center of the circle is given by the image depth plus the probe radius.

2.2.4 Scan Lines

The sector angle is divided by scan lines which intersect in the center of the imaginary circle. A scan line corresponds to the propagation path of a beam transmitted by the probe. Reflected signals are measured along the scan lines. After the transducers receive ultrasound waves the analog signal is preprocessed and radio frequency (RF) data is generated for each scan line. The RF data contains the amplitude information of the received signal over time along each scan line. Therefore the RF data contains information about the reflectivity of interfaces along the single scan lines. By composing the information of all scan lines, a B-Mode image of the field of view can be calculated. The line density is the number of scan lines divided by the sector size.
The C5-2/60 has a default line density of 256.

2.2.5 Transmit Frequency and Bandwidth

The probe model C5-2/60 has its physical center frequency at 3.5 MHz which means it is designed to have a strong resonance at 3.5 MHz. Frequencies far away from the center frequency are attenuated during transmission and receiving. The bandwidth of the probe is centered around the center frequency. Frequencies lying within the bandwidth are less attenuated during transmission and receiving. The fractional bandwidth is the ratio between the bandwidth and the center frequency. A fractional bandwidth of 60% at -6dB therefore corresponds to a bandwidth of 0.6·3.5 MHz =2,1MHz around the center frequency. The frequencies left and right of the band are attenuated to -6dB or lower. Therefore the transmit frequency used for imaging must be near the center frequency of the probe.

2.2.6 Sampling Frequency

In order to digitize the received analog signal of each scan line the signals have to be sampled and discretized. In small intervals the analog signal is measured and the sampling points are stored digitally. The number of measurements per second is given by the sampling frequency. The higher

the sampling frequency the less is the information loss due to digitalization. According to the Nyquist-Shannon theorem the sampling frequency has to be at least double the highest frequency of the analog signal in order to discretize the signal properly.

2.2.7 Equations

Following equations hold:

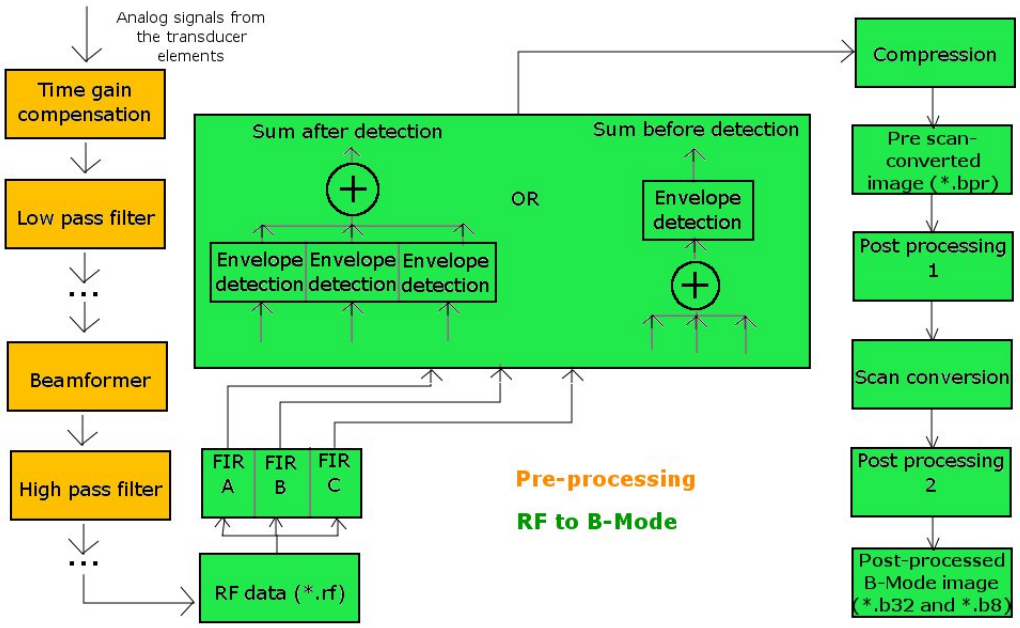
$$probe\_angle = \frac{probe\_surface}{probe\_radius}$$

$$probe\_surface \approx (prob\_elements - 1) \cdot pitch$$

$$depth \approx \frac{number\_samples\_per\_scanline \cdot speed\_of\_sound}{2 \cdot sampling\_frequency}$$

The factor 2 is necessary since the transmitted signal has to travel to the interface and back to the probe and therefore covers twice the distance to the interface.

2.3 Overview of the Sonix RP Processing Pipeline



2.3.1 Preprocessing

Firstly a time gain compensation function is applied to the analog signals. The amplitude of the sound wave decreases exponentially over time. Therefore distant interfaces with the same reflectivity as near ones appear weaker. To compensate for this effect analog signals from distant interfaces are amplified.
The next important component is the beamformer. Due to the convex shape of the probe surface and the circular propagation of sound waves echoes from interfaces will hit the probe elements at different times. In order to get a synchronous consistent signal for each scan line and each echo the received signals have to be delayed for some transducer elements. This is called beamforming.
During the preprocessing the analog signal undergoes several other filters, for example high-pass and low-pass filters.
The resulting signal before the application of the FIR filters is the raw RF data which is the starting point of this project.

2.3.2 RF to B-Mode

The RF to B-Mode pipeline can be divided into six general steps.

- **RF Filter**
  Filtering of the raw RF data to reduce noise and limit the signal to the working bandwidth.

- **Envelope Detection**
  Since only the amplitude of the signal is of interest and not the frequency information itself, the envelope of the signal is calculated. Basically the envelope is a non negative curve that connects the peaks (negative peaks are inverted) of the signal.

- **Compression**
  To visualize the envelope signal of a scan line the value of each sample point is mapped to a color. In B-Mode imaging usually an 8 bit grayscale color map is used. The result of the compression is a first B-Mode image, each column of pixels corresponding to a scan line.
  In order to improve the contrast of the image the brightness values of the B-Mode image have to be adjusted to the dynamic range of the human eye. Usually, non-linear mapping functions, which keep the order of the values but reduce the difference between high peaks in the signal and lower values and therefore increasing the overall brightness of the image, are used .

- **Pre-scan-converted Post-processing**
  The compressed image most likely contains various image artifacts and speckles. In order to improve the image quality some post-processing has to be performed. Some operations are applied before the scan conversion and some after.

- **Scan Conversion**
  The RF data contains the single scan lines as single columns. Since these scan lines are usually not parallel they have to be arranged geometrically correct in the final image depending on the geometry of the probe.

- **Post-scan-converted Post-processing**
  Image operations to enhance the quality of the scan converted image.

Chapter 3

RF to B-Mode in detail

This section explains the used filtering methods of the RF pipeline in more detail. It explains basic concepts as well as how the Sonix RP system and the RF Processing tool implement them.
The first four filtering steps up to the log compression are performed on each scan line individually. The post-processing steps and the scan conversion operate on multiple scan lines at once.

3.1 RF Filtering with FIR Filters

3.1.1 Background - FIR Filter

FIR filter stands for finite impulse response filter. An FIR filter is a (usually digital) frequency filter with a finite impulse response which means the impulse response returns to zero in a finite time. The impulse response is the output of the filter when supplying a short impulse as the input. Often used impulses are the Dirac delta function for continuous and the Kronecker delta function for discrete signals. Simply speaking, they produce infinitely short impulses with infinitely high peaks. Such an impulse comprises equal portions of all possible excitation frequencies. Therefore in this case the impulse response represents directly the attenuation behavior of the filter.
The impulse response of an FIR filter contains a phase shift. Therefore the response seems to be delayed. The phase shift can be different for each frequency contained in the response. If the phase shift is equal for all frequencies then the filter is called a linear phase filter. The impulse response is then in phase.
In case of a linear phase FIR filter the shift is simply $\frac{N}{2}$ sample points where N is the order of the filter.

Equation for FIR filters

$$y[n] = \Sigma_{i=0}^{N} b_i x[n-i]$$

$x[n]$ is the input signal at position n
$y[n]$ is the output signal at position n
b_i are the filter coefficients which define the filter behavior
N is the order of the filter
Since only the last $N+1$ sample points are considered when calculating the output the impulse response dies to zero $N+1$ samples after the last input signal.

Advantages

Because FIR filters have an finite impulse response they are always stable and not susceptible to uncontrolled oscillations regardless of the chosen parameters.
Furthermore they normally require no feedback, which means rounding errors are not carried and amplified through multiple iterations.
FIR filters can be implemented as linear phase filters relatively easy by making the coefficient sequence symmetric.

Disadvantages

The main disadvantage of FIR filters compared to IIR (infinite impulse response) filters is speed. FIR filters usually need more computational power to achieve the same level of filtering.

3.1.2 Sonix RP system

The Sonix RP system uses three FIR bandpass filters to reduce the noise in the signal. The bands are chosen according to the working ranges of the ultrasound system which in turn depend on the chosen transmit frequency and the bandwidth of the probe. Each filter is applied to the original signal resulting in three different filtered signals. These signals are merged by summation at a later point.

3.1.3 RF Processing-Tool

The tool uses the Parks-McClellan method to calculate the filter coefficients b_i for an FIR filter. MATLAB provides the function firpm for this purpose.
The function takes the desired filter order, a vector of bands and a vector of desired amplitudes corresponding to the defined bands. The algorithm then calculates optimal filter parameters b_i to satisfy the desired bands specifications using the specified filter order. The higher the order the better is the approximation of the filter response to the specifications but the slower is the computation. The optimal filter order depends on the needs of the user. The regions between the bands are transition regions and are not considered in the filter parameter calculation. The actual filter may

amplify or attenuate frequencies within the transition regions.
In order to make the filter design more convenient for the user, the tool uses the firpmord function provided by MATLAB to estimate the minimal filter order that would be needed to obtain a filter with a limited deviation from the optimal desired impulse response. The firpmord function takes as input a vector of bands, a vector of desired amplitudes corresponding to the bands, a vector of maximum deviations corresponding to the bands and the sampling frequency. It returns all parameters needed by the firpm function to calculate an optimal filter.
In order to make the number of phase shift sample points integer the filter order should be even. Furthermore an odd order of the filter introduces unwanted roots in the filter response function. To prevent this the order is increased by one if firpmord returns an odd filter order.
The user interface of the implemented filter allows the specification of exactly one passband (desired amplitude is 1 and hence maximal) and two stopbands (desired amplitude 0) left and right of the passband. The stopbands extend to 0 on the left and to the maximum frequency at the right. The passband must lie between the stopbands. The interface allows the specification of the maximum stopband amplitude compared to the passband, called attenuation. The passband ripple defines the maximum deviation within the passband.
To help the user find the optimal working range the tool offers frequency information on the RF data. The user can choose the scan line which should be analyzed. The tool calculates the frequency spectrum based on a fast Fourier transform and the RF signal amplitude along the scan line. Furthermore an average frequency spectrum of all scan lines is calculated. In order to make general peaks in the spectrum more visible the amplitude values of the single frequency spectra are exponentiated by four before calculating the mean value. By using the power function high values have more weight when calculating the sum. To rescale the values the fourth root of the mean amplitude values are calculated after summation. The tool also displays the impulse response of the FIR filter under the given parameters which allows the user to match the response visually to the frequency spectrum of the signal.
Practical advices
The transition regions between stopbands and passband should be of the same size. Otherwise the impulse response may show unwanted behavior in the transition regions like amplifying the transition regions unreasonably strong.
The smaller the transition regions the slower is the image processing.

3.2 Envelope detection

The envelope detection filters out the high frequency information of the RF signal by calculating an envelope of the signal. The envelope is basically a non-negative curve that connects the peaks of the original signal. Therefore the envelope describes the developing of the amplitude. The values of the envelope signal can then be mapped to a grayscale color map.
The Sonix RF system uses a Hilbert transformation to calculate the analytical signal of the RF signal. The envelope is the absolute value of the analytical signal.
The tool offers a filter which calculates the envelope using the Hilbert transform. The envelope detection can be applied to the three signals of the RF Filtering step individually before merging them or it can be applied to the merged signal.

3.3 Compression

3.3.1 Background - Compression

After the envelope detection the signal values can be rescaled to fit the grayscale color map in order to display the signals as a B-Mode image. But usually the image will be very dark and structures will be hard to see. This is due to the fact that there are usually some relatively high peaks in the envelope signal corresponding to materials with a very high reflectivity. When rescaling the whole envelope this peaks cause a down-shift of the other values. As a result the image looks dark. To compensate for such peaks the signal values are not rescaled linearly. There are various possibilities to rescale the image non-linearly. All of them have in common that higher values are more attenuated than lower ones in order to increase the overall brightness while preserving the order of the signal values. Two popular methods are calculating the square root or the logarithm of the envelope signal before rescaling the values. The derivatives of both functions decrease with high values and therefore high values are more attenuated.
Usually the value range is compressed to 8 bits since the ability of the human eye to distinguish grayscale colors is limited anyway.

3.3.2 Sonix RP system

The Sonix RP system uses a logarithmic lookup-table to map the envelope signal to the corresponding grayscale value. The function given by the table is not purely logarithmic and deviates from the logarithmic curve in order to provide a better contrast. The table was probably generated as a result of optimizing several test images.

3.3.3 RF Processing tool

The tool offers a simple logarithmic function of the form

$$y = log(x + a)$$

a can be specified by the user. The higher a is, the darker is the image. Afterwards the values are rescaled to fit 8 bits which corresponds to a value range from 0 to 255.
In addition the tool offers a second truncation filter which cuts off very high and very low values. The user can specify the cutoff value relatively to the actual value range. Afterwards the values are rescaled to use the whole 8 bit value range. This way the dynamic range of the image can be windowed which allows to view a range of interest with better contrast.

3.4 Post-processing

3.4.1 Background

Ultrasound images contain artifacts. Some are useful and wanted, such as shadows behind solid tissues and brightening behind liquids, because they provide information on the material and structure of the tissue. Others are unwanted such as frequency noise due to multi-reflections or discretization errors. The objective of the post-processing step is to enhance the image quality by reducing unwanted artifacts while preserving the information contained in useful artifacts. The post-processed image should help the physician in finding the right diagnosis while lowering the risk of false diagnosis.
Basic post-processing includes smoothing and edge enhancing. Smoothing is performed to make the structure of tissues look smoother and to reduce noise artifacts. This should help physicians in distinguishing real abnormalities in the anatomy from errors generated during the image acquisition. The edge enhancement should help the physician in determining borders of objects and organs.
However, if too much post-processing is performed the information of the image could be lost or, even worse, altered. For example smoothing should not alter the characteristic reflection pattern of a certain material too much. Edge enhancing could sharpen borders around little image artifacts and mislead the physician in interpreting them as real objects.

3.4.2 Sonix RP system

In the documents provided by Ultrasonix it is only stated that the post-processing includes smoothing and edge enhancing. It is not revealed which methods are actually used. A primary application of the RF Processing tool

is the decryption of the post-processing step. Ultrasonix provides the post-processing function as a black box library called Mucro. The library offers the two processing modes ”generic” and ”predefined”. In the generic mode the user can specify a post-processing level from 0 to 100. 0 is the slowest level and 100 the fastest. The predefined mode offers submodes ranging from 0-20.
It seems that the Mucro lib is set up to operate only on 8 bit images with good results. Therefore the compression step is important before applying the Mucro post-processing.

3.4.3 RF Processing tool

The tool offers just three filters so far. Other filters can be easily implemented because of the modular design of the tool. The three filters offered are a wrapper for the Mucro library, anisotropic diffusion and a Gaussian low-pass filter.
Anisotropic diffusion
Anisotropic diffusion is a method to smooth an image while preserving edges.
The first filter uses a script by Daniel Lopes under the BSD license. The script is based on papers by Perona and Malik. The user can specify the number of iterations, the integration constant, the gradient modulus threshold and the filter mode.
The higher the number of iterations, the slower is the filter and the stronger is the smoothing.
The integration must be between 0 and $\frac{1}{7}$. According to the documentation of the script the value should usually be set to its maximum value to ensure numerical stability.
The gradient modulus threshold controls the conduction. In an example of the script the value is set to 30 however other users report that setting this value to small values like 0.2 gave better results.
The filter implements two conduction coefficient functions which are proposed in Perona&Milk's paper. Setting the option to one uses a function which privileges high-contrast edges over low-contrast ones. The second option privileges wide regions over smaller ones.
Gaussian low-pass filter
The implemented Gaussian low-pass filter is a 2D image filter. The filter is defined by a matrix which contains weights. The sum of all weights is 1. The new filtered value of a single pixel at a position (x, y) of the image is calculated by summing up the pixel values of the neighborhood of (x, y) using the corresponding weights of the filter matrix. The weight corresponding to the pixel at (x, y) is stored in the center of the matrix. The size of the matrix determines how many pixels of the neighborhood are considered in the sum. The Gaussian filter uses a two dimensional version of the

normal distribution to calculate the weights. The maximum of the function is located at the center of the matrix. The standard deviation of the normal distribution can be specified by the user as well as the size of the matrix. A larger matrix takes more pixels of the neighborhood into consideration and therefore extends the smoothing effect to a wider area. A high standard deviation decreases the weight of the center pixel and increases the blur effect.

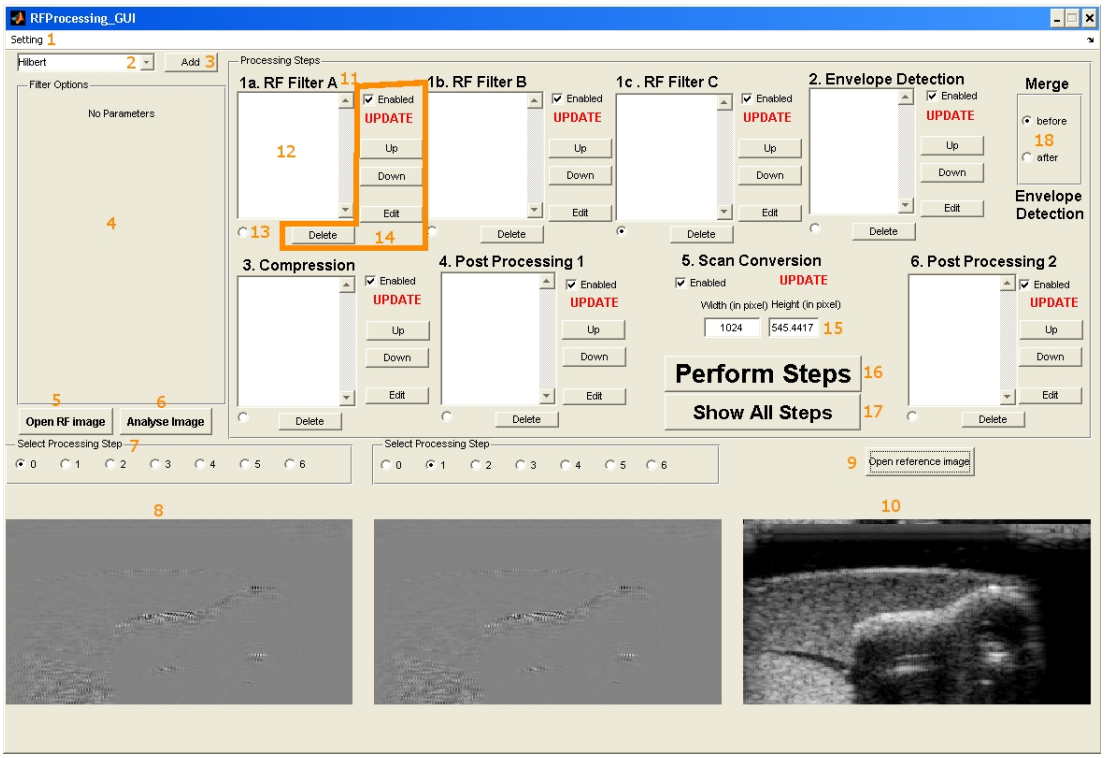
3.5 Scan conversion

Since the probe used has a convex shaped curved linear transducer array the scan lines are not parallel. Up to now the scan lines are displayed as parallel columns of the image. Because of this the image is distorted. To correct the distortion the actual geometrical alignment of the scan lines has to be calculated.
The scan lines intersect at the probe center and diverge with growing distance to the center.This means with increasing depth the space between two scan lines increases. Therefore some pixel values of the new scan-converted image have to be calculated through interpolation between the scan lines.
Ulrasonix provides the Pando library to perform the scan conversion and the interpolation. It uses various information of the probe and the RF image. The exact algorithm is unknown.
The Pando library needs 8 bit images as input to work properly. Therefore the compression step is important before applying the Pando scan conversion.
The RF Processing tool uses a wrapper for the Pando library. The user has to specify only the size of the new image in pixels. The tool automatically keeps the correct ratio between width and height under the assumption that the height of a pixel equals its width. All other information needed by the Pando library are automatically calculated or taken from the probe specifications and the image header.
The wrapper was compiled under Win32. To use it on other systems the wrapper has to be compiled on this systems. The source files as well as the instructions can be found on:
http://www.research.ultrasonix.com/viewtopic.php?f=11&t=443
The important part of this post is also cited in the appendix at the end of this document.

Chapter 4

RF Processing Tool - Interface

4.1 Main Window



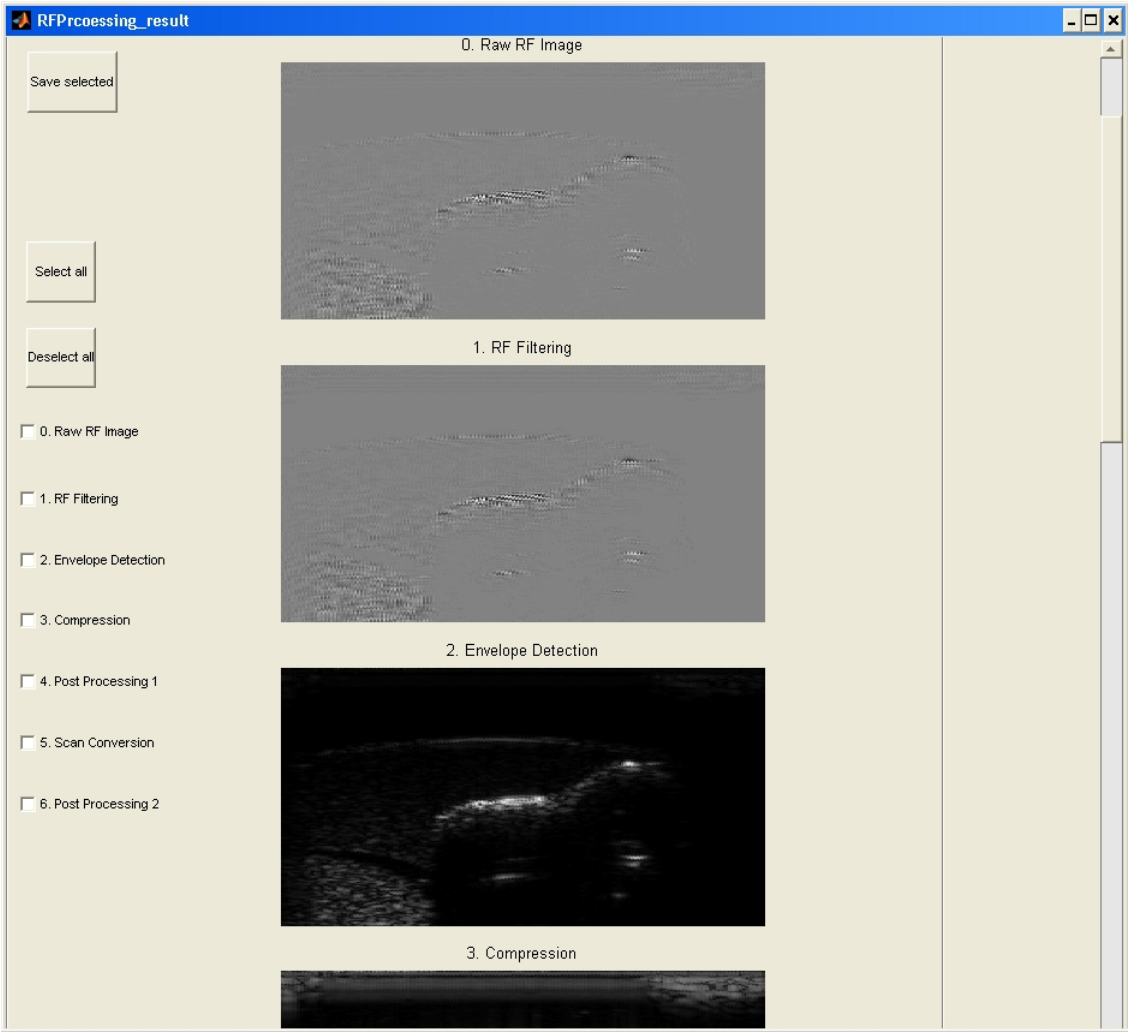
1. **Setting**: Save and open filter settings
2. **Filter selection**: Select a filter from the drop-down menu
3. **Add filter**: Add filter to selected listbox
4. **Filter options panel**: Define filter parameters and options
5. **Open RF image**: Open raw RF image for image processing

6. **Analyze RF data**: Show image header and additional information of the RF image and the probe
7. **Select processing step**: Choose processing step after which a preview image should be shown
8. **Preview image**: Resulting image after the selected processing step
9. **Open reference image**: Opens an reference image to which the processed image can be compared
10. **Reference image**: Reference image
11. **Name of processing step**: The processing steps are enumerated in the order of execution
12. **Filter listbox**: The listbox contains all filters which are performed in this processing step. The filters are applied from top to bottom.
13. **Listbox selection**: Makes the current listbox the active listbox. New filters will be added to the active listbox.
14. **Listbox controls**:
    Enabled: If not checked, the processing step will be left out during image processing
    Update status: If something changed which makes it necessary to reapply the filters from the processing step the status is set to UPDATE (e.g. filter added/deleted/edited, filter order changed, previous processing steps changed). Otherwise the processing step is not performed again and the status is set to DONE.
    Up/Down: Change the filter order by moving the selected filter of the listbox up or down
    Edit: Edit selected filter.The corresponding filter UI will be shown in the filter options panel and the current filter parameters are copied to the UI components.
    Delete: Delete the selected filter from the listbox.
15. **Scan converted image size**: Specify the size of the image after scan conversion. The tool automatically keeps the correct ratio between width and height if one of them is changed.
16. **Perform steps**: Applies all filters from the processing steps which need to be updated. This can take some time. During this time the GUI will not respond to user inputs. The processing is finished when all checked processing steps change their status to DONE.
17. **Show all steps**: Performs all processing steps, if necessary, and shows the images from the single processing steps in a new window. The window offers the option to save the images as png files.

18. **Merge**: Merge the filtered RF signals before or after the envelope detection. Merging the signals before envelope detection causes the tool to calculate the envelope detection after summation of the single RF signals. Otherwise the envelope of all three signals are calculated individually and the envelopes are summed up.

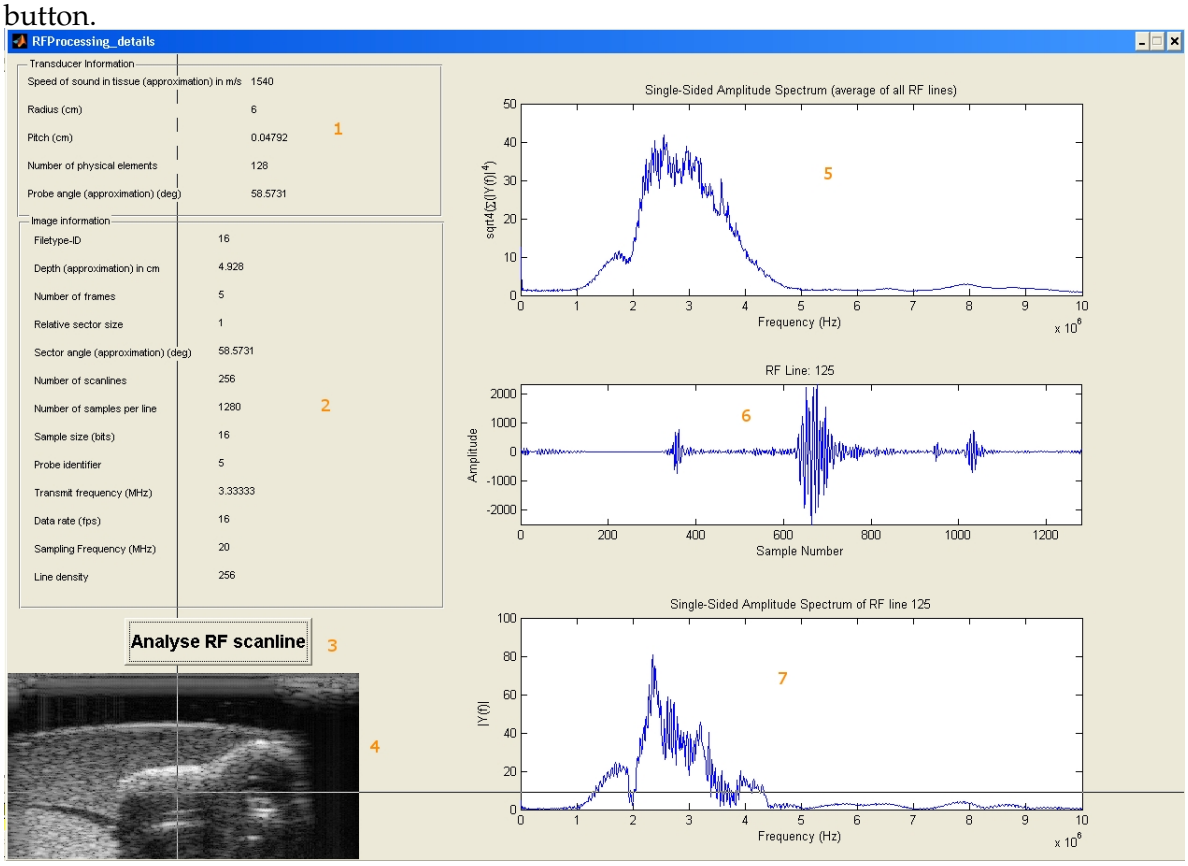
4.2 Processed Images Window

This window is shown when the user clicks the 'Show All Steps' button.



4.3 RF Information Window

This window is shown when the user clicks the 'Additional information' button.



1. **Transducer information**: General information on the probe
   Speed of sound in tissue: (not really a transducer property) This value is used as an approximation for the speed of sound in further calculations
   Radius: The radius of probe is the distance between the center of the probe and the transducer elements. The center of the probe is the point, where all scan lines intersect.
   Pitch: The pitch is the distance between the centers of two adjacent probe elements on the transducer array.
   Number of elements: The number of elements on the transducer array used for ultrasound transmission and receiving
   Probe angle: The angle between the two outermost transducer elements. It is also the maximum angle of the field of view.

2. **Image information**: Header information and additional information on the raw RF image

Filetype-ID: An ID which defines the file type within the Sonix system.
Depth: The image depth is the distance between the farthest sample point and the transducer array. It is calculated using the sampling frequency and an approximation for the speed of sound in tissue.
Number of frames: The number of frames contained in the RF file. Only the first frame is used for image processing.
Relative sector size: The relative size of the field of view angle compared to the probe angle.
Sector angle: The angle of the field of view.
Number of scan lines: Number of scan lines used to scan the the field of view.
Number of samples per line: The number of discretization samples taken from each scan line.
Sample size: The size of a single sample value taken from one scan line.
Probe identifier: Probe ID used to identify the probe type within the Sonix system.
Transmit frequency: The frequency used to stimulate the transducers during transmission.
Data rate: The number of frames taken per second.
Sampling frequency: The number of sample values taken from one scan line per second.
Line density: The number of scan lines divided by the sector size.

3. **Analyze RF scan line**: After clicking the button the user can select a vertical scan line within the RF image. The RF information of the scan line is shown on the right hand side as well as frequency information.

4. **RF image**: The RF image on which the user can click in order to select a scan line to be analyzed.

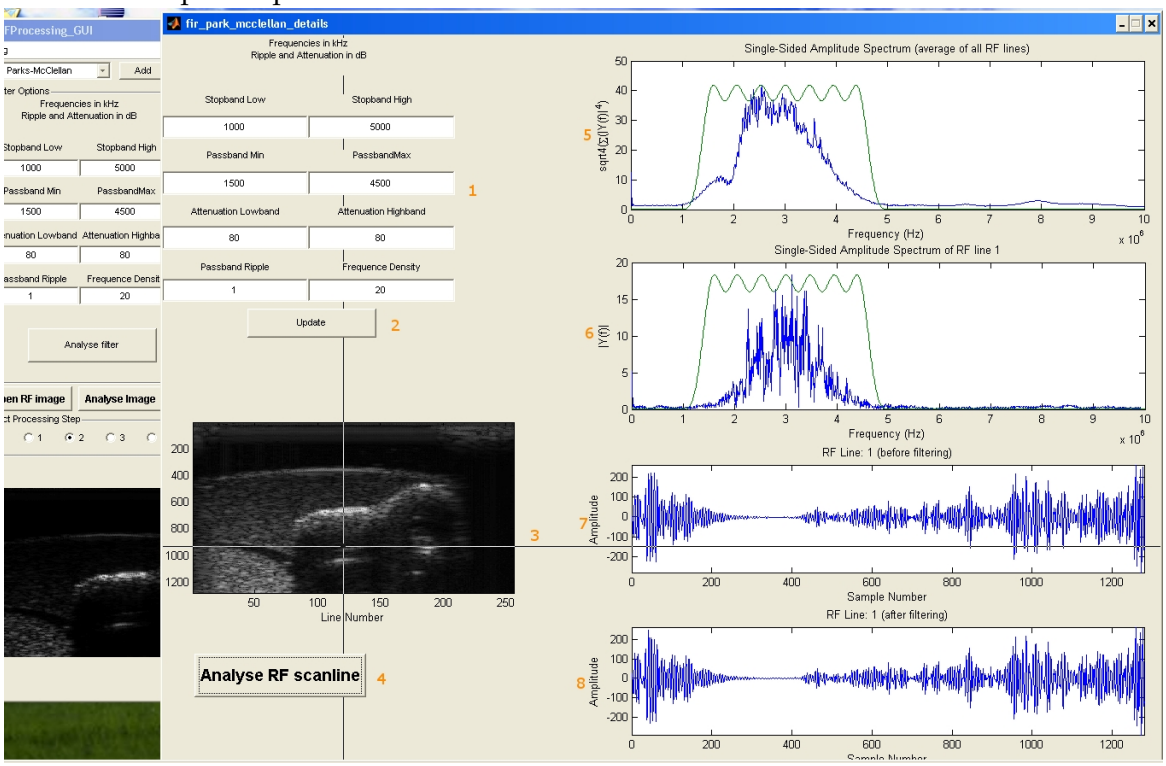
5. **Average frequency spectrum**: The mean value of the frequency spectra of all scan lines in the RF image. In order to make peaks within the single scan lines more visible in the average spectrum, the single spectra are exponentiated by four before summing them up. To rescale the values the fourth root is taken after calculating the mean value of the spectra.

6. **RF Line**: The amplitudes of the reflected ultrasound waves along the selected scan line.

7. **Frequency Spectrum**: The frequency amplitude spectrum of the selected scan line.

4.4 FIR Parks-McClellan Window

This window is shown when the user clicks the 'Analyze filter' button in the filter options panel when the FIR Parks-McClellan filter is active.



1. **Filter parameters**: The user can change the filter parameters within this window.

2. **Update**: This button updates the graphs on the right with the new filter parameters.

3. **Analyze RF scan line**: After clicking the button the user can select a vertical scan line within the RF image. The RF information of the scan line is shown on the right hand side as well as frequency information.

4. **RF image**: The RF image on which the user can click in order to select a scan line to be analyzed.

5. **Average frequency spectrum**: The mean value of the frequency spectra of all scan lines in the RF image. In order to make peaks within the single scan lines more visible in the average spectrum the single spectra are exponentiated by four before summing them up. To rescale the values the fourth root is taken after calculating the mean value.
   Furthermore the frequency spectrum of the impulse response of the

filter is shown. Therefore the user can see immediately which frequencies are attenuated.

6. **Frequency Spectrum**: The frequency amplitude spectrum of the selected scan line.
   Furthermore the frequency spectrum of the impulse response of the filter is shown. Therefore the user can see immediately which frequencies are attenuated.

7. **RF Line**: The amplitude of the reflected ultrasound wave along the selected scan line.

8. **Filtered RF Line**: The amplitude of the reflected ultrasound wave along the selected scan line after applying the FIR filter.

Chapter 5

RF Processing Tool - Program Files and the Filter Module System

5.1 Table of Files

Table 5.1: Root Folder

File name	Brief description
RFProcessing_GUI.m	Main GUI file - Contains the callbacks of the main GUI and the general program behavior
RFProcessing_GUI.fig	Corresponding MATLAB figure file to RFProcessing_GUI.m
RFProcessing_details.m	Image analyzing window - Figure with additional information on the RF image and the probe
RFProcessing_details.fig	Corresponding MATLAB figure file to RFProcessing_details.m
RFProcessing_result.m	Processed images window - Figure which shows the processed images of all steps; allows saving of the images
RFProcessing_result.fig	Corresponding MATLAB figure file to RFProcessing_result.m
External files	
RPread_noTag.m	Reads an Ultrasonix RF data or ultrasound images into a matrix with correct sizes; returns also header information

Table 5.2: root / filters

File name	Brief description
initFilters.m	New filters must be registered within this file

The filter subdirectories all follow a similar structure. A typical set of filter files is contained in the template folder. The files are explained using the template folder as an example. Some filters contain special files which will be listed in separate tables. An exception is the scan conversion folder which does not contain any of the typical files.

Table 5.3: root/filters/template

File name	Brief description
template_createGui.fig	Figure file which defines the user interface components available in the filter options panel of the main GUI
template_createGui.m	Registers the UI components created in the corresponding figure file for access by the main GUI script and displays the UI elements in the filter options field
template_editFilter.m	Initializes the user input fields in the filter options panel when the user wants to edit an already added filter
template_formatArgs.m	Reads the user input from the UI elements and returns a string representation of the arguments, usually a comma separated list
template_procFilter.m	Applies the filter to an image when given the argument string; the script is responsible for parsing the argument string

In the following tables only special files are listed.

Table 5.4: root/filters/fir_parks_mclellan

File name	Brief description
fir_parks_mcclellan_details.fig	Image analyzing window - Figure with additional frequency information on the RF data
fir_parks_mcclellan_details.m	Corresponding callback m-file

Table 5.5: root/filters/mucro

File name	Brief description
External files	
cvie.dll	Part of the Mucro library
cvltk.dll	Part of the Mucro library
cvmipconv.dll	Part of the Mucro library
cvrestore.dll	Part of the Mucro library
libmmd.dll	Part of the Mucro library
mucro.dll	Part of the Mucro library
utx_utils.dll	Part of the Mucro library
vc60.idb	Part of the Mucro library
cv.par	Part of the Mucro library
MucroWrap.dll	Part of the Mucro-Wrapper for MATLAB
MucroWrap.h	Part of the Mucro-Wrapper for MATLAB
MucroWrap.exp	Part of the Mucro-Wrapper for MATLAB
MucroWrap.lib	Part of the Mucro-Wrapper for MATLAB
MucroWrap_mfile.m	Part of the Mucro-Wrapper for MATLAB

Table 5.6: root/filters/scanconversion

File name	Brief description
External files	
pando.dll	Pando library
scan_conversion.mexw32	Mex-File for integration of the library in MATLAB
scan_conversion.m	Documentation on how to use the wrapper function

5.2 Filter Module Design

The design of the tool allows an easy integration of new filter modules. In the previous section a typical set of files defining a filter module was shown. This section provides an detailed guide on how to implement a new filter module.

5.2.1 Minimal Filter Module

Each filter module must implement at least four functions:

- UI creation: A script which is responsible for creating the user interface components displayed in the filter options panel of the main GUI. The function gets the filter options panel as a parameter, places the UI components within the panel and returns a handle structure which allows the other three functions to access the necessary GUI elements. It is up to the developer which information should be stored within the handle structure.
- Formatting of the filter parameters: This function takes the handle structure created during the UI creation as an argument. It is responsible for reading out the user input and creating a formatted string which stores the filter parameters. This string is displayed in the filter step listboxes and is later used during the image processing to determine the filter parameters. The string must not contain round brackets since these are used as delimiters in the listboxes. Usually the formatted string is a comma separated list of all filter options.
- Editing of existing filter: When the user clicks on the edit button the values of the filter to be edited must be copied to the appropriate UI elements in the filter options panel. This function gets the handle

structure created during UI creation as well as the formatted argument string of the filter to be edited. It is responsible for parsing the argument string and setting the correct values in the UI components.

- Image processing: This function carries out the image processing. It gets the input image, the argument string and the handle structure of the main GUI as arguments. The handle structure of the main GUI is necessary for some filters to access information on the RF data and the probe. The function returns the processed image. The function must first parse the argument string in order to extract the filter parameters.

In order to facilitate the UI design, the developer can use the GUIDE tool provided by MATLAB. The easiest way is to copy the template figure and UI creation script and to modify the files.
If the developer wants to build the filter from scratch following method proved to be easy and convenient:

- Create a new figure using GUIDE.
- Resize the figure to have the same size as the filter options panel in the main GUI (pay attention to the unit of measurement)
- Place the UI components setting Units and FontUnits to normalized.
- Add the following lines at the beginning of the UI creation script:
  ```
  h = hgload('./filters/template/template_createGui.fig');
  copyobj(get(h,'Children'),panel);
  close(h);
  ```
  where 'panel' is the filter options panels. Of course the path to the figure must be changed.
  This step copies the UI components defined in the figure to the filter options panel.
- To access and save handles to the UI elements the following expression can be used:
  ```
  guiHandle.uiComponent1 = findobj('Tag','nameOfUiComponent1','Parent'
  ```
  where 'panel' is panel which contains the UI components. Usually this is the filter options panel.

5.2.2 Registering the Filters

Before the filter modules can be used by the main program they have to be registered within the initFilters.m script located in the filter subdirectory.

- Add all necessary paths to the MATLAB search path; usually this is the folder containing the filter scripts

- Register all four basic functions using the registerFilter function. The function creates maps which associate the filter names with the corresponding functions. This maps are returned to the main GUI.
- The functions must be given to the registerFunction in the order:
  UI creation
  argument formatting
  image processing
  filter editing

5.2.3 Naming Conventions

In order to keep a clean filter module structure all filter modules should have a own subdirectory in the filter directory. All files belonging to the filter module should be placed there. The four basic scripts should be named as shown by the template example in order to easily identify them.

5.2.4 Some Examples

- **Simple standard filter module**: template: Intended for usage as a filter module template; implements a threshold filter with two edit text fields
- **Callbacks and new windows**: fir_parks_mcclellan: This filter is a good example on how callbacks within the filter options panel can be used and how to create a new figure from within the filter options panel.
- **Radio buttons and panels**: mucro: This filter uses a radio button group and deals with UI components whose direct parent is not the filter options panel.
  NOTE: To prevent MATLAB from throwing warnings concerning the callbacks of the radio buttons being overwritten simply delete all callbacks of the radio buttons. To do so, set the size of the callbacks-array in the property inspector to 0x0.

Chapter 6

Outlook

The objective of this project was to understand the processing pipeline of the Sonix RP system and to develop a modular framework which should help to reconstruct the methods used in the processing pipeline by the Sonix RP system. The project succeeded in developing a tool which provides the researcher with a customizable processing pipeline which can process RF data to B-Mode images. Besides allowing the user to add different filter and processing methods in the different processing steps the tool provides the user with additional information on the RF data and the probe. The tool has a modular structure which allows new filter modules to be integrated easily. However the project did not succeed in fully reconstructing the exact single processing methods.
Following are some open issues that need to be dealt with in the future. The Ultrasonix research forum is a good starting point and provides lots of useful hints on how to solve these problems.

6.1 Ideas for Future Projects

6.1.1 RF Filtering

The tool implements a simple frequency compounding allowing up to three separate filtering pipelines. However the user cannot select the weights for the different signals when summing them up. It is also not known how the weights should be selected optimally.
Another issue is the determining of the optimal bandwidths. The Sonix RP system has a file called filter.txt which contains a list of the FIR filters used by the system. However the information on the filters is sparse and the filter parameters are very specific to the system. It is not documented why this filter parameters were chosen and thus it is difficult to adjust the given filters. Since RF filtering is very important concerning the image quality future projects which work on the RF Processing tool should deal with this.

6.1.2 Compression

The simple logarithmic function implemented so far is not sufficient to compress the signal properly. The Sonix RP system uses a dynamic lookup-table which has logarithmic characteristics but is not purely logarithmic. Since the compression is also a very important step future projects should find out how the table is calculated and implement a filter module which allows the usage of a lookup-table.

6.1.3 Post-processing and Scan Conversion

So far the tool does not implement many post-processing filters. It is difficult to determine appropriate filters and the optimal parameters for enhancing the image. All other processing steps are relatively well documented but the post-processing and the interpolation methods of the scan conversion are kept secret. The main purpose of this tool is to allow researchers to experiment with different filters and parameters in order to reconstruct the post-processing steps. What needs to be done is therefore the implementation of various filter modules for image post-processing. To help future developers, the tool offers filter modules for the Mucro and Pando libraries. They can serve as black box reference filters.

Appendix

Quotes from the Ultrasonix research forum

Ultrasonix has an active forum to support SonixTOUCH Research, Sonix RP and ES500 RP users. It contains a lot of information on the system and the libraries used by the system. Since it was a major information source during the project following are some useful forum quotes:

- **Transducer specification sheet**
  http://www.research.ultrasonix.com/download/file.php?id=227
  Contains specifications for several probe models.

- **Mucro Wrapper for MATLAB**
  http://www.research.ultrasonix.com/viewtopic.php?f=5&t=152
  http://www.research.ultrasonix.com/download/file.php?id=62
  MATLAB wrapper for the Mucro SDK

- **Another RF processing tool**
  http://www.research.ultrasonix.com/viewtopic.php?f=5&t=10
  Hi,
  here is the matlab toolbox with graphic interface. The software is capable of reconstructing the B image out of raw RF data for linear probes (not convex probes). Here is what you can do with this software:
  Read Raw Data
  - enter the file name (copy and past)/ line length / line count / the frame that you are interested in.
  - press ”Read raw data” to read *.dmp file (Those that come from the remote control and don’t have any header).
  - see the center line RF A line and its power spectrum.
  Generate B image
  - select the range of freq. you want to filter the data (i.e 5MHz-9MHz). you can select main frequency or harmonic components or any specific frequency component that you want.
  - select the envelop detection technique (i.e. Hilbert)
  - select the down sampling factor (1 means no down sampling)

- select the compression algorithm (nth root)
- press the ”generate”
The software also includes a dmp file reader which can be used to read dmp file into matlab memory.
Good Luck
Reza

This tool by Reza performs RF processing implementing the basic processing pipeline steps. However it is not modular and allows no post-processing of the image.
In the beginning phase of the project Reza's tool was a first guide in developing the RF Processing tool.

- **Probe angle**
http://www.research.ultrasonix.com/viewtopic.php?f=2&t=387
Post by corina » Mon Jul 06, 2009 5:11 pm
Hi Ping,
To calculate the angle do the following:
probeSurfaceLength = (numElements - (2 * transmitoffset)) * pitch (probe surface length in microns)
probeAngle = probeSurfaceLength / probeRadius * 180 / PI (probe angle in radians)
You can get the transmitoffset parameter from setup.exe under ”Tx Offset”.
Corina Leung, Ultrasound Software Engineer, Ultrasonix Medical Corporation

- **Sonix RP pipeline**
http://www.research.ultrasonix.com/viewtopic.php?f=2&t=250
Information on the schematic processing pipeline of the Sonix RP system.

- **Pre-Scan Image vs Post-Scan Image**
http://www.research.ultrasonix.com/viewtopic.php?f=2&t=157
Post by mgstauffer » Fri Sep 21, 2007 11:53 pm
Hi,
If I understand correctly, the pre-scan images output by Exam (*.bpr files) have been envelope detected, compressed and scaled to 8-bit. The post-scan (*.b8) images have then been scan converted and had image enhancement applied, such as MRU. Correct?
So, if I take a pre-scan image from Exam, and apply Pando and Mucro, should I be able to recreate the associated post-scan image precisely as it would look if it had been saved from Exam, assuming it was created with type 1 filter in Exam, and I use the right settings for Pando and Mucro? If not, what are the other steps?

Thanks,
Michael

Post by kris » Mon Sep 24, 2007 4:48 pm
This is correct, the .bpr files are pre-scan-converted, envelope detected, compressed 8 bit B image data, which have no processing applied. The images are sent directly from the cine buffer.
The .b8 files are post-scan converted and have processing applied, ie. MRU filter, greyscale map parameters (brightness, contrast, gamma). Our internal processing steps, however, process the MRU (Mucro) filter and also apply the greyscale map to the pre-scan converted data, rather than the post-scan converted data, except when filter type=4, then MRU is applied to post-scan converted data. We do most processing on the raw data because the amount of data is typically less than the post-scan image. After numerous tests, it is almost impossible to tell the difference between MRU and map applied to pre and post scan-converted data, results are pretty much identical.

- **Processing pipeline**
  http://www.research.ultrasonix.com/viewtopic.php?f=2&t=59
  Post by kris » Fri Oct 13, 2006 5:06 pm
  There have been comments on how to obtain good B-mode image quality when creating a B image from RF data, as some Sonix users notice that the clinical software produces very nice images compared to their own reconstruction.
  There are five key elements:
  #1 - RF Filtering
  It is important to filter the RF data properly once you obtain it, depending on what frequency your transmit was, and what type of probe you used to collect the data. Ultrasonix uses pretty basic FIR bandpass filters. You can find the corresponding ID's in the file filters.txt on the system.
  #2 - Envelope Detection
  We use a simple Hilbert, shift by 90 degree type filter for this. However you can choose your favorite method.
  #3 - Compression Table
  When going from 16 bits down to 8 bits, there has to be a nice logarithmic curve with some additional adjustable parameters. One way to produce a nice curve is to obtain or write some simulation software to show the changes of curve manipulation in real-time of a constructed B-mode image. For your convenience, I have attached a typical compression table used in the Sonix clinical application.
  #4 - Post Processing
  No ultrasound system is without its post processing algorithms used

to enhance the image. MRU (maximum resolution ultrasound) is a filtering technique that we use on our pre-scan converted data (can also be applied to post-scan converted data). This algorithm does a combination of edge enhancement and smoothing. To see the difference in the clinical software, toggle Filter Type from 0/1 on the post processing menu (on the right). There is an SDK called mucro to run the MRU algorithm on data, it can be found at:
http://www.ultrasonix.com/updates/sdk/mucro/
Please read the documentation carefully.
#5 - Scan Conversion
Without scan conversion, the B-mode data is never that nice to look at (especially on a curved or angular image). Therefore it is a good idea to interpolate to get the nicest image after steps 1 and 2. I will post a scan conversion library on the SDK site in the future which will make it convenient to interpolate data.

- **MATLAB wrapper for Pando**
  http://www.research.ultrasonix.com/viewtopic.php?f=11&t=443
  http://www.research.ultrasonix.com/download/file.php?id=237
  Post by Thor » Fri Oct 16, 2009 1:57 pm
  Attached is a matlab mex wrapper for pando. See the scan_conversion.m file for documentation.
  To build the mex file go to the folder where you have the pandowrapper files and type
  mex -I"." -L"." -lpando scan_conversion.cpp
  The function can compute the output scale based on the input parameters or the user can specify one. Probe information can be retrieved from the probes.lst file or the probe parameters can be supplied in a struct. There is also a matlab function for retrieving probe information from the probes.lst file, see get_probe_info.m.
  Steering and extension angle parameter must be retrieved from the system. See
  http://research.ultrasonix.com/viewtopic.php?f=6&t=437
  for a tool for storing parameters together with a recording.
  If you have any questions or encounter any errors, please let me know.
  Thor Andreas

- **Image depth and probe angle**
  http://www.research.ultrasonix.com/viewtopic.php?f=11&t=321
  Post by pjq27 » Thu Jan 22, 2009 9:41 pm
  Hi Kris,
  In a previous post you showed us how to calculate the axial sample size in microns as shown:
  sample_size_in_microns = (1000000 * SPEED_OF_SOUND) / (2.0 * sampling_freq)
  depth = number_of_samples * sample_size_in_microns
  where,
  SPEED_OF_SOUND = 1540
  sampling_freq = header.sf
  number_of_samples = hdr.h

  ...

  Post by kris » Thu Apr 09, 2009 10:40 pm
  Here's some calculations from our software for convex probes:
  double ProbeSurfaceLength = ((NumElements-1.0)*(double)pitch);
  double TotalProbeAngle = ProbeSurfaceLength / ProbeRadius;
  int NumVirtualTransducers = (linedensity-1)*16+1;
  double AngleBetweenVirtualTransducers = TotalProbeAngle / (NumVirtualTransducers-1.0);

Bibliography

General Sources

- **Ultrasonix Forum** http://www.research.ultrasonix.com/
- **Sonix Wiki** http://ultrasonix.com/wikisonix
- **MATLAB Function Reference** http://www.mathworks.com/access/helpdesk/help/techdoc/ref/
- **MATLAB Central** http://www.mathworks.com/matlabcentral/
- **Wikipedia** http://www.wikipedia.org/
- **W. R. Hedrick, D. L. Hykes, D. E. Starchman, T.A. Wilson: Ultrasound physics and instrumentation.** , Mosby St. Louis 1995, ISBN: 0-8151-4246-3

The following lists contain additional sources used for some processing steps. However the general sources were used at all times.

FIR Filter

- http://www.dspguru.com/dsp/faqs/fir
- http://www.mathworks.com/access/helpdesk/help/toolbox/signal/firpm.html
- http://www.mathworks.com/access/helpdesk/help/toolbox/signal/firpmord.html
- http://www.mathworks.com/access/helpdesk/help/techdoc/ref/fft.html
- http://www.mathworks.com/access/helpdesk/help/toolbox/signal/f9-131178c.html#f9-136613
- http://de.wikipedia.org/wiki/Filter_mit_endlicher_Impulsantwort
- http://en.wikipedia.org/wiki/Finite_impulse_response
- http://cnx.org/content/m16889/latest/

Envelope Detection

- http://www.mathworks.com/products/demos/shipping/dspblks/dspenvdet.html
- http://en.wikipedia.org/wiki/Envelope_detector
- http://en.wikipedia.org/wiki/Hilbert_transform
- http://en.wikipedia.org/wiki/Analytic_signal
- http://www.mathworks.com/access/helpdesk/help/toolbox/signal/hilbert.html

Post-processing

- http://www.mathworks.co.jp/matlabcentral/fileexchange/14995-anisotropic-diffusion-perona-malik
- http://en.wikipedia.org/wiki/Gaussian_filter
- http://en.wikipedia.org/wiki/Gaussian_blur
- http://www.mathworks.com/access/helpdesk/help/toolbox/images/fspecial.html