Simulation of Beamforming Techniques for the Linear Array of Transducers

2 AUTHORS:

Grant Hampson
The Commonwealth Scientific and Industrial …
74 PUBLICATIONS 577 CITATIONS
SEE PROFILE

Andrew P Paplinski
Monash University (Australia)
92 PUBLICATIONS 576 CITATIONS
SEE PROFILE

Faculty of Computing and Information Technology

Department of Robotics and Digital Technology

Technical Report 95-3

Simulation of Beamforming Techniques for the Linear Array of Transducers

Grant Hampson, Student Member, IEEE
Andrew Papliński, Member, IEEE

March 13, 1995

Enquiries:-

Technical Report Coordinator
Robotics and Digital Technology
Monash University
Clayton VIC 3168
Australia

tr.coord@rdt.monash.edu.au +61 3 905 3402

Contents

List of Tables

List of Figures

Abstract

This report describes the fundamentals of various time and frequency domain beamforming techniques. Details of MATLAB code required for simulation are described in detail, and graphical results illustrated. Also, various sampling algorithms applicable to beamformers are described in detail.

Simulations permit the designer to easily determine the ability of each digital beamformer, before actual VLSI implementation. Underlying problems and design constraints can also be easily pre-calculated using simulation. Finally, it is possible to assess each type of beamformer, relative to each other, in terms of size and complexity.

Keywords

Linear Array, Time Domain Beamforming, Frequency Domain Beamforming, MATLAB

Chapter 1

Beamforming Techniques

Beamforming is the process of performing spatial filtering, ie., the response of an array of sensors is made sensitive to signals coming from a specific direction while signals from other directions are attenuated. There are various methods of implementation, including Time and Frequency domain, depending on the desired processing speed, and the type of signals to be processed.

Beamforming is typically referred to in SONAR (acoustic applications) [1, 2, 3] and RADAR (electro-magnetic applications) [4, 5], however its applications extend into seismic, medical ultrasonic imaging, and various other applications [6, 7].

Time and frequency domain techniques which aid in the design and implementation of a digital beamformers will be discussed and simulated in this chapter. The efficiency and complexity for each beamformer type will be discussed and the resulting advantages/disadvantages can then be resolved.

1.1 Medium Constraints

In the simplest form, beamforming can be considered to be the combining of omnidirectional sensor outputs, arranged in an arbitrary geometry, so as to spatially filter a required direction. This process can be implemented in a variety of ways, however it is important to consider the propagation medium of the operating environment [8].

The extremes in which a beamformer must process signals are the environments in which SONAR and RADAR systems operate. Table (1.1) illustrates the governing properties for each medium [1].

The property of most significance is the velocity of propagation which effects the frequency range, pulse durations and repetition rates. This indirectly places different requirements on a SONAR beamformer when compared to a beamformer used in a RADAR application.

It should be noted that this table was compiled in 1980 when high power SONAR transducers had to operate at low frequencies, however today much higher frequencies ($50 - 200$kHz) transducers are in common use [9].

Property	SONAR	RADAR
velocity of propagation	$1.5 \times 10^3 m/s$	$3 \times 10^8 m/s$
frequency range	$0.1 - 10$kHz	$300 - 30000$MHz
typical pulse duration	$0.01 - 1.0$s	$0.1 - 10\mu$s
typical pulse repetition rate	$0.2 - 0.01$s^{-1}	$2000 - 200$s^{-1}
typical wavelength	0.3m	0.03m
Doppler Shift	$\pm 2\%$	$\pm 0.0003\%$

Table 1.1: Comparison of SONAR and RADAR properties.

1.2 Sampling Methods

An implicit requirement for digital beamforming techniques is the availability of accurate, unaliased digital data stream. Various sampling methods can be employed to convert the analog signal from the transducer into a digital form via an A/D converter.

Firstly however, it is important to recognize the spectral frequencies commonly used in both RADAR and SONAR applications. Figure (1.1) illustrates the spectral coverage of lowpass, bandpass and narrowband signals. A narrow band signal is special class of bandpass signals when the bandwidth (BW) is very narrow, i.e., a single frequency.

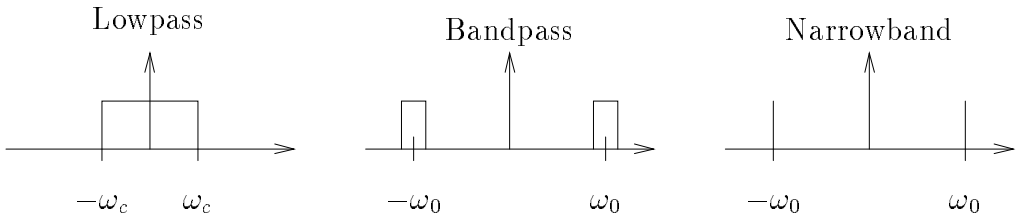

Figure 1.1: Spectral definitions.

For each of the spectral domains certain possibilities exist for sampling the signal. The Nyquist theorem [10] states that a bandlimited signal must be sampled at a rate that is atleast twice the highest frequency component to avoid aliasing. This is also known as as low-pass sampling [11].

Using time domain techniques, the sampling rate required for accurate beamforming is typically around 5-10 times the Nyquist rate, as fine time delays are required for accurate beamforming [12]. However, frequency domain beamforming only requires that the signal be sampled without aliasing. i.e., at the Nyquist frequency.

Typically high sampling rates are not achievable due to the expense of sampling a large number of sensor channels. A discrete time method for effectively increasing the sampling frequency is known as interpolation [13, 14]. Appendix A contains relevant background information to interpolation.

A method for sampling bandpass and narrow band signals is Quadrature sampling [15]. Also refer to Appendix A for relevant information.

1.3 Time Domain Beamforming

Beamforming or spatial filtering is a method of observing signals from a desired direction while attenuating the response of the array to signals from other directions. Beamforming can permit a two or three dimensional image to be formated from an array of sensors, depending on the array geometry.

Fundamentals will be considered firstly using the simple case of beamforming in the time domain using one-dimensional array or linear array. This theory can then be extended to include two-dimensional or planar array imaging.

1.3.1 A three-dimensional input space

To begin with, a three-dimensional coordinate system [16] as shown in Figure (1.2) is used where M sensors are located at $\mathbf{r}_m$, $(m \in [0, M-1])$, incoming wave direction $\mathbf{u}$ and a beam steering direction $\mathbf{u}_0$.

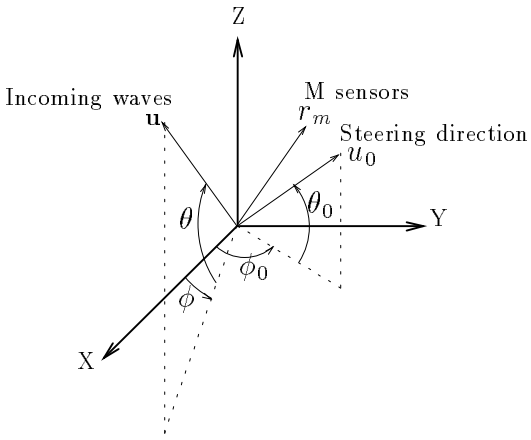

Figure 1.2: Coordinate systems for three dimensional input space

When considering a linear array of transducers, this is simplified to two-dimensions by reducing the azimuth and elevation angles (ϕ and θ respectively) into a conic angle ψ [12], such that $\sin\psi = \cos\phi\cos\theta$.

1.3.2 Beamforming using the linear array

A linear array of transducers is sufficient to map the two-dimensional input space as shown in Figure (1.3). A plane incoming wave with direction $\mathbf{u}$ makes contact with the evenly spaced linear array (spacing d) of transducer elements $\mathbf{r}_m$.

A signal $x_m(\omega t)$ arriving at the mth sensor is, in general, a sinusoidal pulse modulated signal with spatial information inscribed by reflections from objects to be detected. The phase of the incoming wave gives information about the spatial direction of the source.

To obtain spatial information from a direction $\mathbf{u}_b$, a beam is formed in the related conic direction ψ_b by performing a weighted summation of appropriately delayed signals $x_m(\omega t)$.

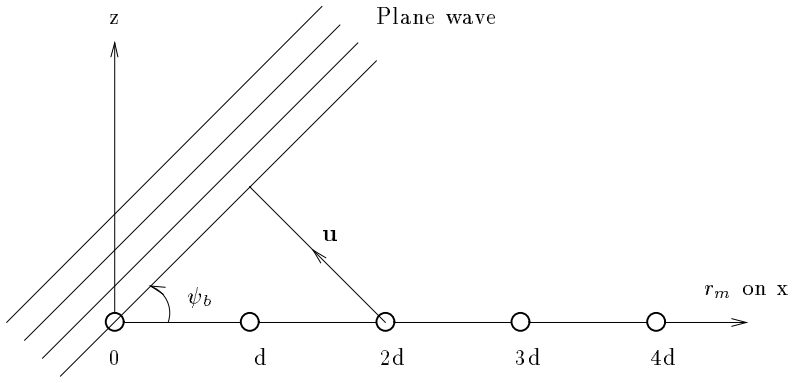

Figure 1.3: Projection of a plane wave on the linear array

To form a beam in the direction ψ_b, each sensor signal $x_m(\omega t)$ is delayed by

$$t_{mb} = \mathbf{r}_m \cdot \mathbf{u}_b / c = m \, \frac{d}{c} \, \sin \psi_b \tag{1.1}$$

and the bth beam output is formed by the general beamforming equation:

$$y_b(t) = \sum_{m=0}^{M-1} a_m \, x_m(\omega t - \omega t_{mb}) \tag{1.2}$$

where a_m is a spatial weighting function usually defined as a window function [14, 17] to control mainlobe width and side lobe magnitude. Figure (1.4) illustrates a beam steered 30° to the right using unity weighting.

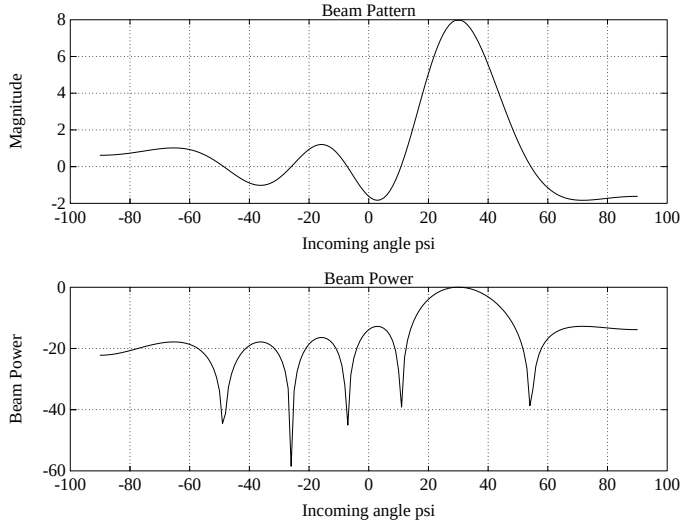

Figure 1.4: A simple Time domain-Delay sum beamformer pattern with M=8.

A discrete version [12] of this equation is obtained by assuming a sampling frequency $f_s = 1/t_s$, and $\Omega = \omega t_s$. Substituting $t = nt_s$, Equation (1.2) becomes

$$y_b(n) = \sum_{m=0}^{M-1} a_m x_m(\Omega(n - q_{mb})) \tag{1.3}$$

where $q_{mb} = t_{mb}/t_s$ represents an **integer** delay (number of delay units) for a given sensor input, m, and beam direction, b. To form a beam y_b in a steering direction ψ_b, the output of each sensor is delayed by an integer multiple of the sample spacing q_{mb}, where q_{mb} represents

$$q_{mb} = m\delta_s \sin\psi_b \qquad \text{where } \delta_s = \frac{d\, f_s}{c} \tag{1.4}$$

Since q_{mb} is integer delay, it is imperative that $\delta_s \sin\psi_b$ is an integer also. This obviously **quantises** the steering directions which can be realised.

A simple time domain beamforming structure is the **Delay-Sum** beamformer which implements Equation (1.3), and is shown in Figure (1.5). The structure is repeated B times, where B is the number of beams desired.

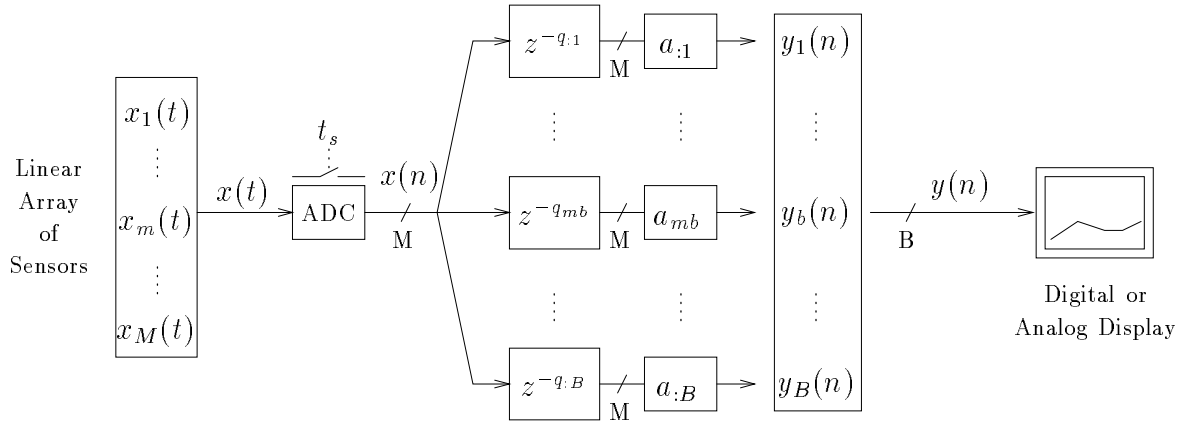

Figure 1.5: A simple Time domain-Delay sum beamformer.

The sensor signals from the linear array are converted to a digital signal at a frequency greater than the Nyquist rate to achieve acceptable beam pattern performance. The signals are then delayed appropriately and the weighting function applied and effectively summed. If same set of weights are to be used for all steering directions, as is often the case, the weights can be applied directly at the sensor outputs, before the delay-sum operation.

1.3.3 Beam Steering Directions

To form a beam in direction ψ_b, the delay required to bring the output of the M sensors into time alignment is equal to q_{mb}, which are integer multiplies of the sample spacing t_s. If the beam index b, is an integer

$$b = 0, \pm 1, \pm 2, \pm 3, \ldots$$

then the beam directions that we can realize are those for which

$$\delta_s \sin\psi_b = b \tag{1.5}$$

and effectively quantises the beam steering directions ψ_b, such that

$$\psi_b = \arcsin\left(b \cdot \frac{1}{\delta_s}\right) \tag{1.6}$$

Consider an average SONAR beamformer with sensor separation of $d = 0.003$ and $f_s = 600$kHz, then the discrete beamforming angles would be $0, \pm 56.4°$. It can be shown [18] that the beam separation is given by Equation (1.7), and showing in Figure (1.6) for $f_s = 2.5f$.

$$\text{Beam Separation} = \frac{2\,f}{f_s} \text{ in } \sin\psi \text{ space} \tag{1.7}$$

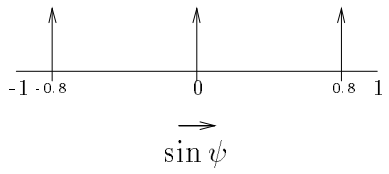

Figure 1.6: Directions of synchronous beams for $f_s = 2.5f$.

This is a major limitation of the simple beamformer, and it is only possible to get more beams by increasing the sensor spacing or increasing the sampling frequency. Over sampling can be achieved by using high-speed A/D converters, or by interpolating the sensor signals in the digital domain.

Simulation in MATLAB [19] is achieved by examining the response of the beamformer for all input directions, for a particular steering direction. Firstly, create the input waves:

```
fs = 3*f;                       % sampling frequency
Om = 2*pi*f/fs;                 % angular digital frequency

N = 50;
n = (0:N-1)';                   % discrete time samples

del_s = d*fs/c;                 % some constants
mv = 0:M-1;
a = ones(M, 1)/M;               % rectangular window

s_psi = sin((-90:90)*pi/180); % sine of incoming plane wave angle
```

Next, apply each input direction, and calculate the power of the beam. The beam direction is set by `b`, the beam index.

```
for id = 1:psi_range*2+1        % FOR each INPUT direction
   qb = del_s*s_psi(id) ;
   X = sin(Om*( n(:, ones(1, M)) + qb*mv(ones(N, 1), :) ));

   for m = 1:M                   % FOR each SENSOR
      qmb = eye(1, (m-1)*b+M);   % calculate delay (offset by M so +)
      Xqmb(:,m) = filter(fliplr(qmb),qmb, X(:,m));      % delay signal
   end;

   Y = Xqmb*a;   % formation of beam with weighting function
   P(id) = sum(Y.^2);
end;
```

Results for a beam formed at broadside ($b = 0$, $\psi_b = 0°$, $M = 8$) are shown in Figure (1.7). The plot illustrates that incoming wave angles other that the steering direction are attenuated.

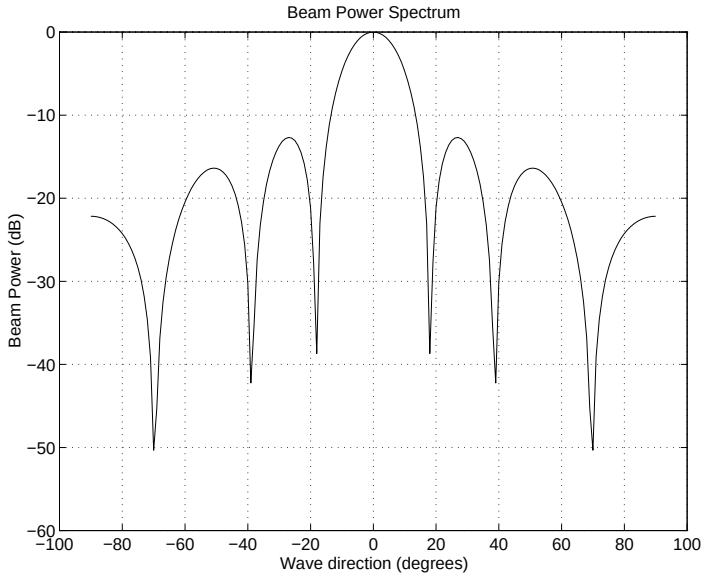

Figure 1.7: MATLAB simulation of the Simple Beamformer.

1.3.4 Interpolation Beamforming

Interpolation is the process of increasing the effective sampling frequency by an integer factor, thereby decreasing the sampling period. This decreased sampling period allows finer time delays to be implemented in the beamformer, and hence more steering directions. Refer to Appendix A for more information regarding interpolation.

An implementation of such an interpolation beamformer [20, 21, 22] is shown in Figure (1.8). Here the sensor signals are interpolated by a factor of D. A set of desired delays now is approximated more accurately by specifying integer multiples of the smaller sampling time t_s/D. Once again a different set of integers q_{mb} is specified for each beam direction,

$$q_{mb} = m\ D \cdot \delta_s \sin \psi_b \tag{1.8}$$

However, the **effective** sampling frequency $D \cdot f_s$ is significantly larger enabling more closely spaced beams to be formed.

$$\psi_b = \arcsin\left(b \cdot \frac{1}{D\delta_s}\right) \tag{1.9}$$

Figure (1.8) is refered to as the interpolation filter before beamforming structure. Here the linear array of sensors produces an analog signal, $x(t)$, which is digitised (as in the simple beamformer) and padded with a vector of $(D-1)$ zeros. The zero padded signals are then interpolated using M FIR low-pass filters (one for each sensor). The delayed signals are then decimated by D and summed to produced a beam output $y_b(n)$.

Simulation is MATLAB is easily achieved by a few extra commands to perform the filtering, such that:

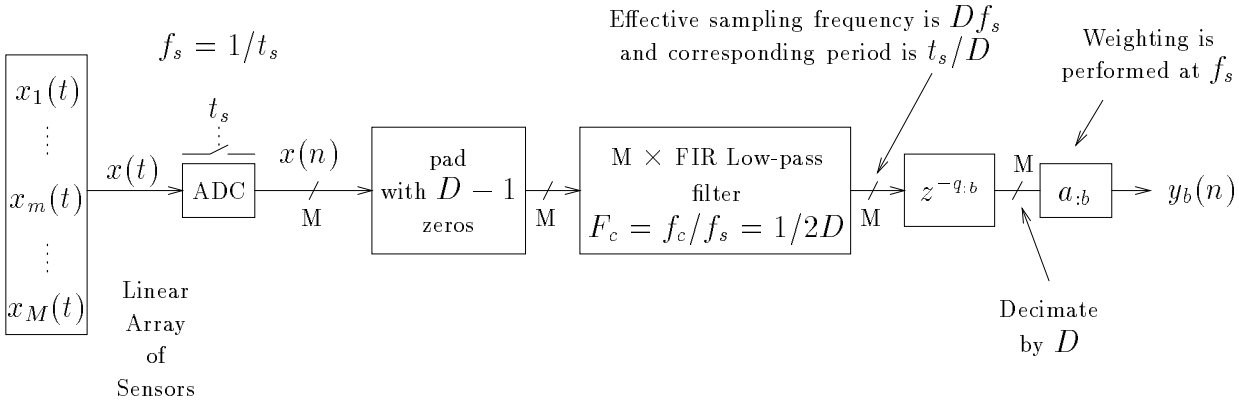

Figure 1.8: Interpolation filter before beamforming for one steering direction $y_b(n)$.

```
num = fir1(32, 1e-5);      % low pass filter co-eff
Xd = zeros(D*N,M);         % initialise

for id = 1:psi_range*2+1         % FOR each INPUT direction

   qb = del_s*s_psi(id) ;
   X = sin(Om*( n(:, ones(1, M)) + qb*mv(ones(N, 1), :) ));

   Xd([1:D:D*N],:) = X;          % and put X into Xd with zeros

   for m = 1:M                   % FOR each SENSOR
      qmb = eye(1, (m-1)*b+M);   % calculate delay (offset by M so +)

      Xdf(:,m) = filter(num, 1, Xd(:,m));     % interpolate data

      Xqmb(:,m) = filter(fliplr(qmb),qmb, Xdf(:,m));    % delay signal
   end;

   Y = Xqmb(1:D:D*N,:)*a;   % decimate and form beam
   P(id) = sum(Y.^2);
end;
```

Results for the interpolation beamformer indicate that the beam pattern degradation [12] increases as the interpolation factor D increases as shown by Figure (1.9).

For a large number of sensors this structure becomes computationally expensive since **each sensor** requires a low-pass filter. Alternatively a structure as shown in Figure (1.10) can be used when the number of beams formed is less than the number of sensors.

In this structure, the order in which the summation and filtering has been changed to optimise the number of FIR filters. Filtering and summation are both linear operations and can be interchanged. The figure illustrates that the sensor signals are zero padded, delayed, weighted and then summed at the increased sampling frequency $D\,f_s$. The interpolation filter follows the summation and is calculated at the increased sampling frequency, and then decimated.

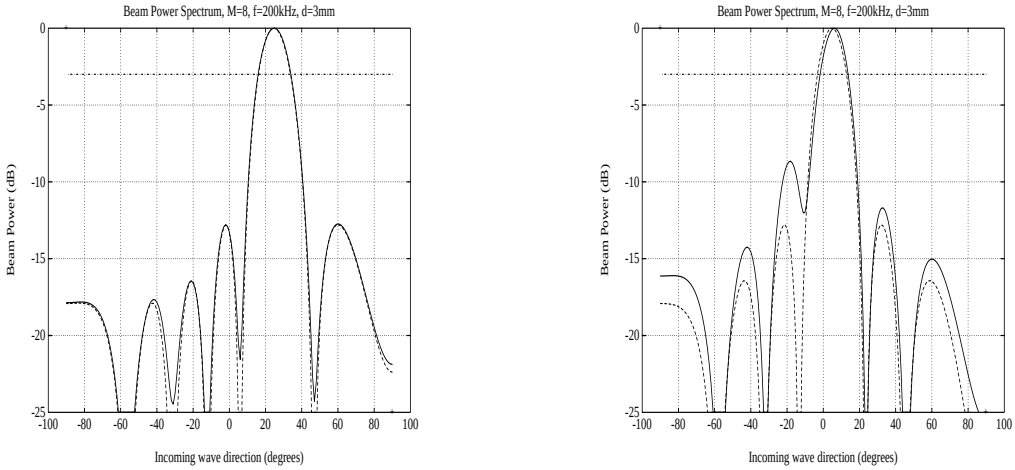

Figure 1.9: Power spectrums for the interpolation beamformer $D = 2$ and $D = 10$. Note that the steering direction magnitude changes with D.

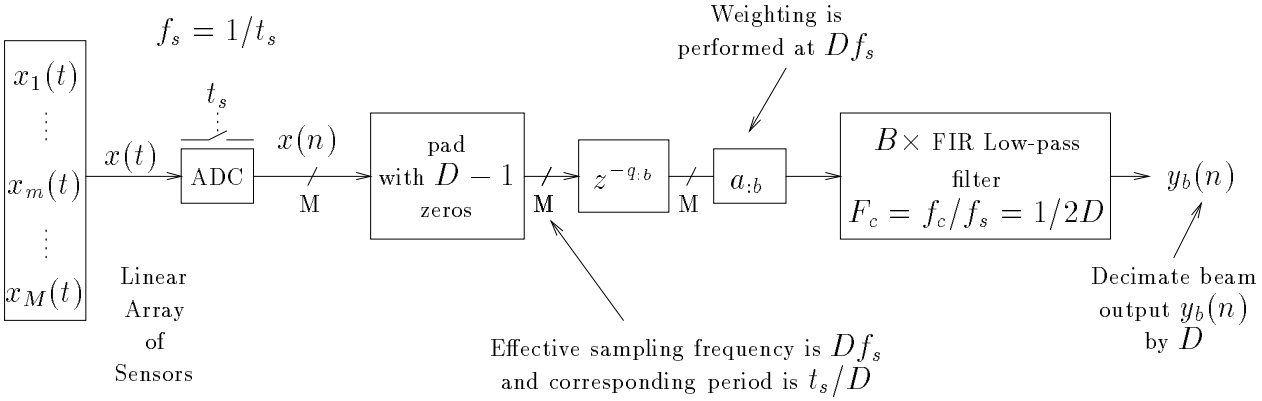

Figure 1.10: Interpolation filter after beamforming.

1.3.5 Quadrature Beamforming

Quadrature beamformers [23, 24, 25] employ the quadrature sampling method to obtain a complex envelope of a bandpass signal. Refer to Appendix A for more information about Quadrature Sampling.

Consider the case of demodulating frequency being equal to the centre frequency of the bandpass signal. This implements a Base-Band-Beamformer and the complex envelope is centered around DC. Remembering that the signal $x_m(t)$ can be retrieved from the complex envelope using

$$x_m(t) = \Re\left[\tilde{x}_m(n)e^{j\Omega_0 n}\right] \tag{1.10}$$

A similar expression for the beamformer output can also be written:

$$y_b(n) = \Re\left[\tilde{y}_b(n)e^{j\Omega_0 n}\right] \tag{1.11}$$

Replacing n by $(n - q_{mb})$ in (1.10) and using the general beamforming Equation (1.2) with (1.11) results in an expression for a quadrature beamformer:

$$\tilde{y}_b(n)\ e^{j\Omega_0 n} = \sum_{m=0}^{M-1} a_m \tilde{x}_m(n - q_{mb}) e^{j\Omega_0 (n - q_{mb})} \tag{1.12}$$

$$\tilde{y}_b(n) = \sum_{m=0}^{M-1} a_m \tilde{x}_m(n - q_{mb}) e^{-j\Omega_0 q_{mb}} \tag{1.13}$$

The required operations for beamforming using the baseband sensor data are channel weighting of both by a_m and $e^{-j\Omega_0 q_{mb}}$. Interpolation may still be used in conjunction with the base-band-beamformer to achieve greater number of beams, as shown in Figure (1.11).

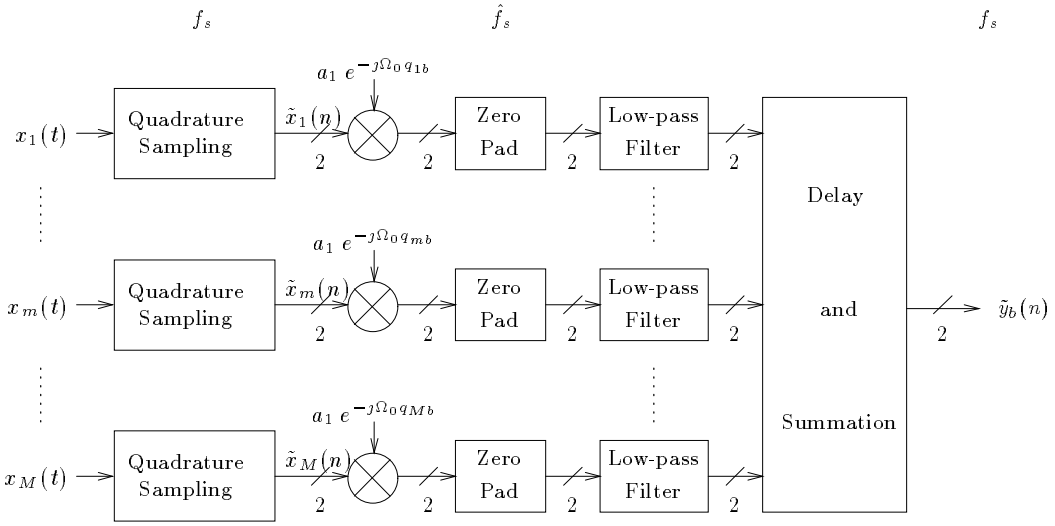

Figure 1.11: Quadrature beamformer with interpolation

As shown previously with the interpolation beamformer implementations, the interpolation can be performed prior to, or subsequent to beamforming, in order to minimise hardware requirements.

The base-band-beamformer has one main draw back - time delaying DC levels cannot achieve spatial discrimination. i.e., base banding a signal removes any phase information and hence beamforming cannot be achieved. To avoid the DC levels, the complex demodulating frequency is chosen to be:

$$0 < f_1 < f_0 - BW/2 \tag{1.14}$$

which ensures that the lower band edge of the down shifted spectrum doesn't overlap DC. This is appropriately given the name shifted-sideband-beamformer and is identical to the base-band-beamformer except ω_1 replaces ω_0.

The requirements imposed on t_s by the shifted sideband beamformer are now dependent on an increased sampling frequency which is atleast twice $f_0 - f_1 + BW/2$.

Simulation in MATLAB is slightly more complex since signals must firstly be quadrature sampled, however the theme remains basically the same.

```
num = fir1(16,1/fs);

df = .9*f;  % a fraction of the original frequency
Omd  = 2*pi*df/fs; % angular digital demodulation frequency

rt = exp(-j*Omd*n);  % demodulation factor

%********************* Do beamforming  *****************************
for id = 1:length(s_psi)        % FOR each INPUT direction

   % ******* form input waves with phase shift due to dir *************
   % *********** and also demodulate the waves with df ****************
   qb = del_s*s_psi(id) ;
   X = sin(Omf*( n(:, ones(1, M)) + qb*mv(ones(N, 1), :) )).*rt(:, ones(1,M));

   for m = 1:M                    % FOR each SENSOR

      Xf = [filter(num, 1, real(X(:, m))) , filter(num, 1, imag(X(:, m))) ];

      % *********** multiply and then delay signals ***************
      csf = [cos(Omd* (m-1)*b ) ; -sin(Omd* (m-1)*b ) ]; % rotational factors
      b_in = Xf * csf ;  % rotate signals

      qmb = eye(1, (m-1)*b+M);   % calculate delay (offset by M so +)
      b_in_del(:,m) = filter(fliplr(qmb),qmb, b_in);    % delay signals
   end;

   Y = b_in_del * a;   % formation of beam with weighting function

   P(id) = 0.5*sum(Y.^2) ;
end;
```

The results of the script are illustrated in Figure (1.12) for a beamformed at broadside.

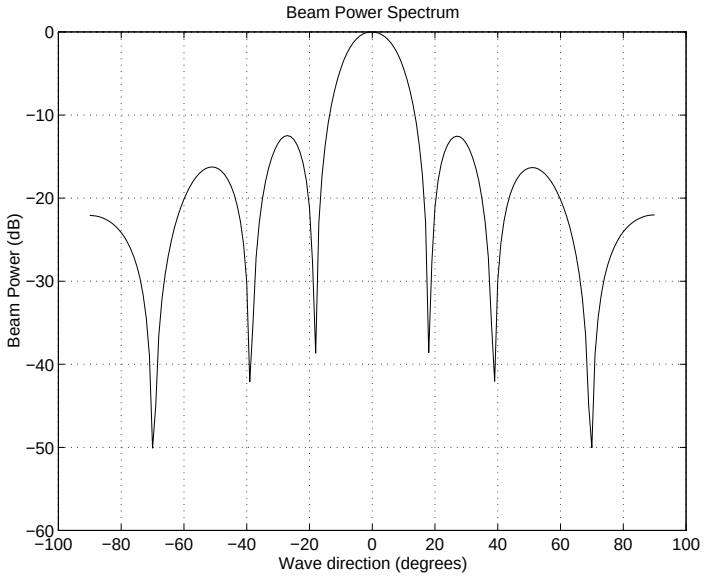

Figure 1.12: MATLAB simulation of the Quadrature Beamformer.

1.4 Frequency Domain Beamforming

Frequency (or transform) domain beamforming has numerous advantages and disadvantages when compared with time domain techniques. The most significant difference is that a time delay transforms to a phase shift in the frequency domain [26].

$$f(t - t_0) \leftrightarrow \mathcal{F}(\omega)e^{-j\omega t_0} \tag{1.15}$$

Frequency domain beamforming concepts are the result of the applying the Fourier transform to the beamforming equation. i.e., the Fourier transform of the beam output is:

$$\begin{aligned} Y(f, \psi_b) &= \mathcal{F}\{y(t, \psi_b)\} \\ &= \mathcal{F}\left\{\sum_{m=0}^{M-1} a_m x_m(t - t_{mb})\right\} \\ &= \sum_{m=0}^{M-1} a_m X_m(f) e^{-j\omega t_{mb}} \end{aligned} \tag{1.16}$$

where $X_m(f)$ is the Fourier transform of each transducer output. Each beam is a weighted linear combination of the Fourier transform coefficients of the received waveforms. This is illustrated in Figure (1.13) where it is important to note that the complex weights are linear with frequency.

An inverse Fourier transform maybe required to obtain the time domain representation, however some useful features can be extracted from the transform data.

However, the Fourier transform cannot be calculated for all time and alternative methods are adopted. The first uses Discrete Fourier transform (DFT) for low-pass and band-pass signals and the second approach, known as phase shift beamforming can only be used for well defined narrow band signals.

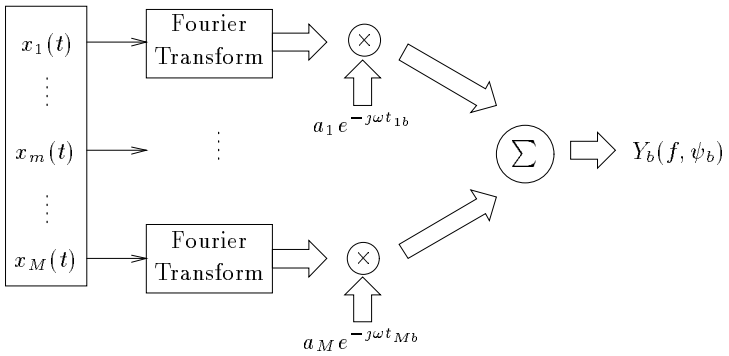

Figure 1.13: Basic Frequency domain configuration.

1.4.1 Discrete Fourier Transform

Application of a N-point DFT to each sensor output, $x_m(n)$, results in frequency transformed data, $X_m(k)$ where $f_k = kf_s/N$, such that

$$X_m(k) = \sum_{n=0}^{N-1} x_m(n) e^{j2\pi nk/N} \qquad \text{where } 0 \leq k \leq N-1 \tag{1.17}$$

where $X_m(k)$ is an estimate of $X_m(f)$. The frequency domain representation of the beamforming equation is then, and illustrated in Figure (1.14).

$$Y(k, \psi_b) = \sum_{m=0}^{M-1} a_m X_m(k) e^{-j\Omega_k t_{mb}} \tag{1.18}$$

where

$$\begin{aligned} \omega_k t_{mb} &= \frac{2\pi k}{N} \cdot m\delta_s \sin\psi_b \\ &= \phi_b mk \qquad \text{where } \phi_b = 2\pi\delta_s \sin\psi_b/N \end{aligned} \tag{1.19}$$

Finally, we have

$$Y(k, \psi_b) = \sum_{m=0}^{M-1} a_m X_m(k) e^{-j\phi_b mk} \tag{1.20}$$

An advantage of frequency domain beamforming is that the sampling frequency, f_s, doesn't effect the beam steering resolution. The beam steering directions can be of any magnitude, unlike time domain beamformers where it is quantised. Another obvious advantage is that the DFT can be computed efficiently using the Fast Fourier Transform [27].

Simulation in MATLAB is relatively simple using the `fft()` function to generate the frequency transformed data. However, only a single sided spectrum is required for the beamforming process. The direction is set by the variable `dir`. The power of the beam pattern is calculated from the time domain sequence, however it could be calculated from the magnitude of the transform data.

```
jfi_b = -j*2*pi*del_s*sin(dir*pi/180)/N ; % beam direction parameter
```

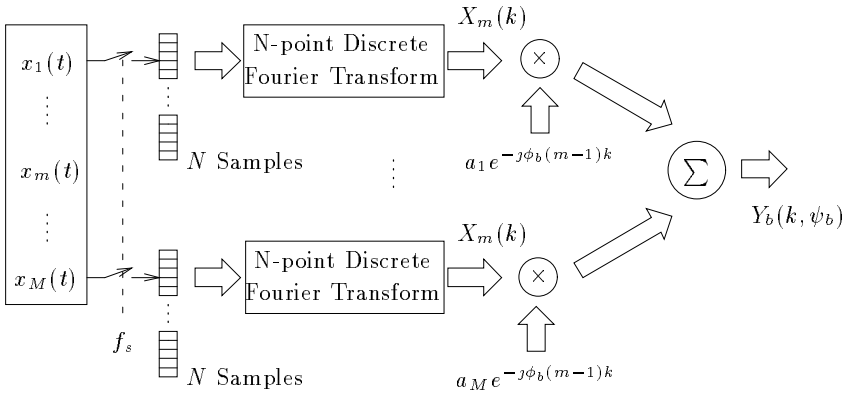

Figure 1.14: DFT based frequency domain beamforming configuration.

```
% Rotational ( Fourier Transform ) operator
XFT = exp(jfi_b*(0:(N/2)-1)'*(0:M-1)) ;

for id = 1:length(s_psi)        % FOR each INPUT direction

   q = del_s*s_psi(id) ;
   x = sin(Om*( n(:, ones(1, M)) + q*mv(ones(N, 1), :) ));

   X = fft(x);           % get the discrete fourier transform
   X = X(1:N/2,:);       % only interested in the single sided spectrum

   Y_b = (X.*XFT)*a ;       % Fourier transform of the beam "dir"

   y_b = abs(ifft(Y_b));     % get back y_b(n)
   P(id) = sum(y_b.^2);      % get average power value of y_b
end;
```

Results from the MATLAB script are shown in Figure (1.15), where a steering direction of 42° has been used.

1.4.2 Multiple-beam Beamformer

If **multiple** beams are required, then a different set of complex multipliers is required, however there is a more efficient method which involves further application of a FFT [28, 29]. If the beam outputs, index b, were given by [30]

$$Y_b = \sum_{m=0}^{M-1} X_m \; e^{-j\frac{2\pi}{M} \cdot bm} \qquad -\frac{M}{2} \leq b \leq \frac{M}{2} \tag{1.21}$$

which is in the form of a FFT, then simultaneous beams are formed at fix angles. Note that the discrete frequency index k has been omitted for simplicity. The number of beams formed is equal to the number of sensors in the array.

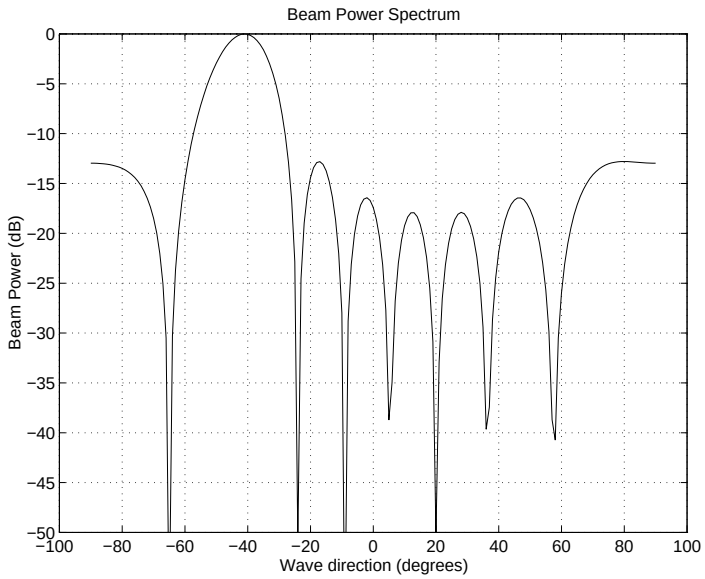

Figure 1.15: MATLAB simulation of the Frequency domain Beamformer.

To determine the directions of the beams formed, Equations (1.18) and (1.21) are compared, i.e.,

$$\begin{aligned} 2\pi \cdot t_{mb} \cdot f_k &= 2\pi \frac{mb}{M} \\ m \, \frac{d}{c} \, \sin\psi_b &= \frac{mb}{M f_k} \\ \psi_b &= \arcsin\left(\frac{cb}{f_k M d}\right) \end{aligned} \tag{1.22}$$

Note that the beams formed are linear within the $\sin\psi_b$ domain, and it is important to note that they vary with the frequency used in the array. At the design frequency $f_k = c/2d$, all beams except the $-90°$ beam (at which stage spatial aliasing occurs [12, 31]) are useful. However, at frequencies below the upper design frequency, some beams are formed at angles which spatial aliasing will occur.

This is the main drawback of the beamformer, as a bandpass signal will be steered in different directions depending on the frequency. A few methods exist [30, 32] for ensuring that beams are formed at the same angle regardless of frequency.

The resulting transform sequence is thus defined by Equation (1.23) and illustrated in Figure (1.16).

$$x(n,m) \overset{FFT}{\longrightarrow} X(k,m) \overset{FFT}{\longrightarrow} B(k,b) \overset{IFFT}{\longrightarrow} b(n,b) \tag{1.23}$$

Once again, simulation in MATLAB is very straight forward, and the simulation results for M=8 are shown in Figure (1.17).

```
for id = 1:length(s_psi)        % FOR each INPUT direction

   qb = del_s*s_psi(id) ;
   x_tm = sin(Om*( n(:, ones(1, M)) + qb*mv(ones(N, 1), :) ));
```

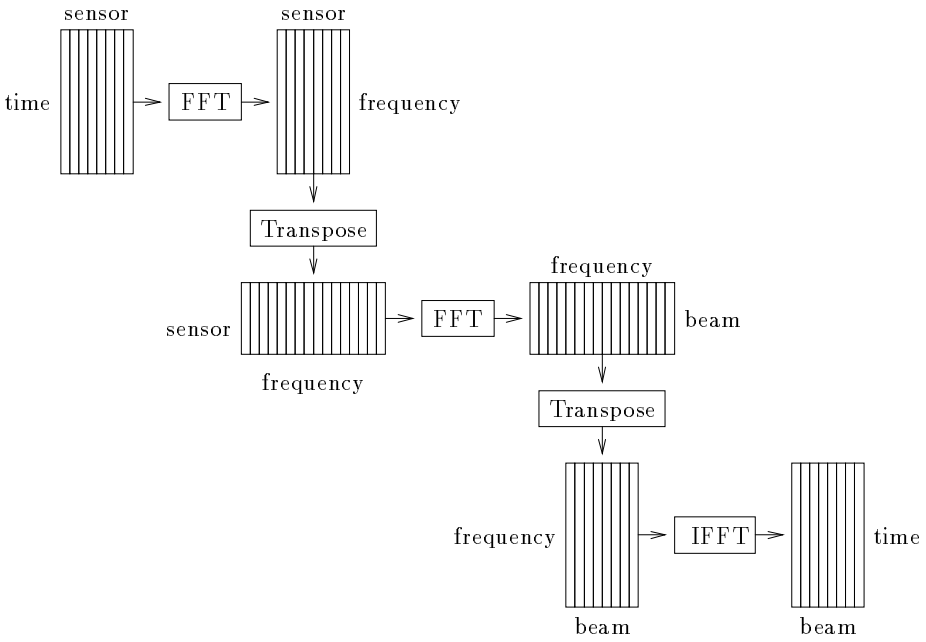

Figure 1.16: Multiple-beam Beamforming using the FFT transform graph.

```
   X_fm = fft(x_tm);          % get the discrete Fourier transform

   X_mf = X_fm(1:N/2,:)';     % need single sided spectrum + transpose

   B_bf = fft(X_mf);          % multiple beam formation of B(w,psi_k)

   B_fb = B_bf(:,1:N/4)';     % need single sided spectrum + transpose

   b_tb = abs(ifft(B_fb));   % get back b(t,b)

   for bb = 1:M
      P(bb,id) = sum(b_tb(:,bb).^2);    % get average power for each beam
   end;
end;
```

1.4.3 Phase Shift Beamformer

The phase shift beamformer is a specialised case of the discrete Fourier transform beamformer and is only applicable to narrow band signals. Since the method is based on frequency domain concepts, steering delays are realised by phase shifts and consequently not dependent on the sampling frequency [33].

The phase shift beamformer makes the approximation that the linear phase shifts in the DFT beamformer are replaced by a constant phase shift,

$$2\pi f t_{mb} \approx 2\pi f_0 t_{mb} \tag{1.24}$$

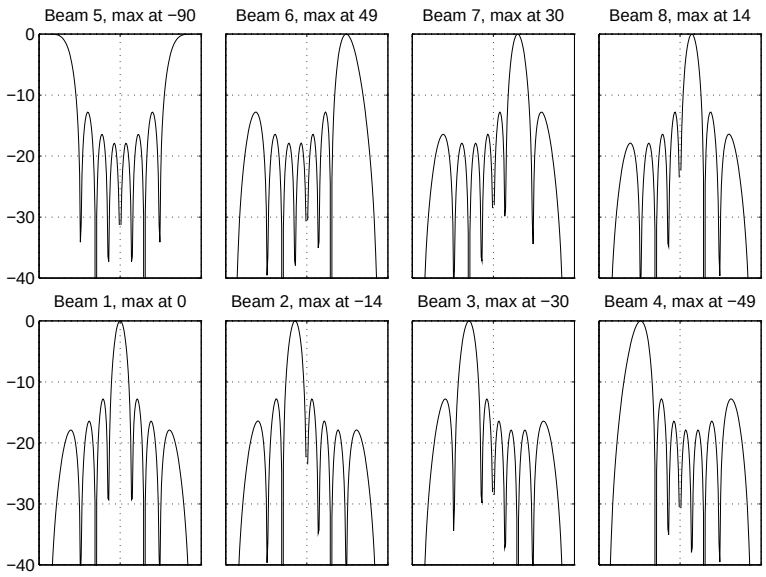

Figure 1.17: MATLAB simulation of the Multi-beam Frequency domain Beamformer.

where f_0 is the centre frequency of the narrow band signal. This is only valid for narrowband signals, otherwise errors occur in the beam patterns [34]. Note that phase shift beamforming is related to the DFT beamformer if only a single frequency bin in the Fourier transformed data is used.

The frequency domain beamforming equation, Equation (1.16), becomes:

$$Y(f,\psi_b) = \sum_{m=1}^{M} a_m X_m(f) e^{-j\omega_0 t_{mb}} \tag{1.25}$$

and the time domain equivalent is obtained by taking the inverse Fourier transform of $Y(f,\psi_b)$, resulting in

$$\begin{aligned} Y(t,\psi_b) &= \mathcal{F}^{-1}\left\{Y(f,\psi_b)\right\} &(1.26)\\ &= \sum_{m=0}^{M-1} a_m x_m(t) e^{-j\omega_0 t_{mb}} &(1.27)\end{aligned}$$

As a result, an implementation of the phase shift beamformer [35] is relatively simply; quadrature samples can be obtained at a rate consistent with the bandwidth, phase shifted and summed. This relatively simple structure is illustrated in Figure (1.18), where a narrow band filter has been used to reduce the spectrum to a single frequency.

Although the phase shift beamformer requires very little circuitry to implement, good performance is only achieved for well defined narrow band signals. The following MATLAB code is used to simulate the phase shift beamformer.

```
num = fir1(32,.000001);
rt = exp(-j*0md*n);          % demodulation frequency
rc = -j*2*pi*f * (d/c)*sin(dir*pi/180);   % constant phase factor
Xf = zeros(N,M);
```

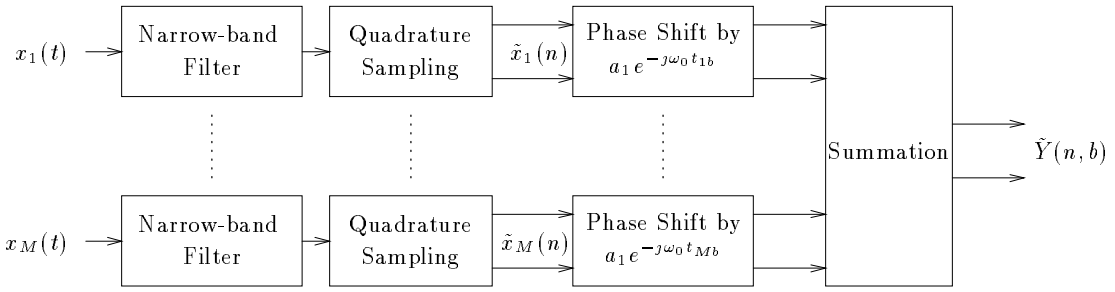

Figure 1.18: Phase Shift Beamformer.

```
for id = 1:length(s_psi)        % FOR each INPUT direction

   % *********** form input wave with phase shift due to dir *************
   qb = del_s*s_psi(id) ;
   X = sin(0mf*( n(:, ones(1, M)) + qb*mv(ones(N, 1), :))).*rt(:,ones(1,M));

   % *********** for each sensor rotate ****************
   for m = 1:M                     % FOR each SENSOR
      Xf(:,m) = [filter(num, 1, real(X(:,m)))+j*filter(num, 1, imag(X(:,m)))];

      % *********** phase shift signals ****************
      Xp(:,m) = Xf(:,m) * exp(m*rc);    % rotate signals
   end;

   Y = Xp*a;   % formation of beam with weighting function
   P(id) = sum(real(Y).^2);
end;
```

The results for a beam steered to $+30°$ are shown in Figure (1.19).

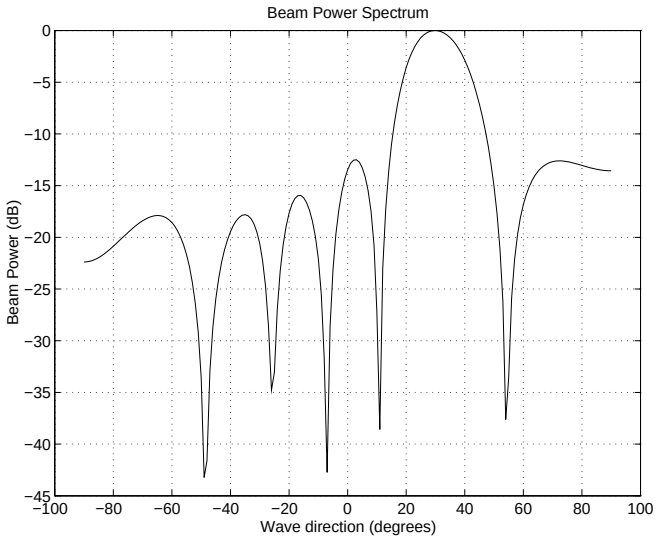

Figure 1.19: MATLAB simulation of the Phase Shift Beamformer.

Chapter 2

Summary and Conclusions

Table (2.1) is a summary of all the beamforming techniques discussed in the previous chapter. For each beamforming technique discussed, a relative index on the bandwidth of the A/D converters, data storage and additional complexity is given.

	Spectral Characterisation			Hardware considerations		
Beamforming Technique	Lowpass	Bandpass	Narrowband	Analog to Digital Converter	Data Storage	Additional Computational Complexity
Simple Delay-sum	*	*		High	High	None
Interpolation	*			Low	Low-Med	Low-Med
Complex Sampling with Interpolation		*		Low	Low	Med
Shifted Sideband		*	*	Low-med	Low-Med	Low-Med
Discrete Fourier Transform	*	*		Low	Low	Med
FFT Multibeam	*	*		Low	Low	Med
Phase Shift			*	Low	Low	Low

Table 2.1: Beamforming techniques and associated hardware considerations and spectral area of application.

In Table (2.1) a low sampling frequency corresponds to the Nyquist rate, and a high rate, several times the Nyquist rate. The amount of data storage is also consistent with the sampling rate, and beamforming using interpolation will require more data storage. The amount of additional circuitry required for the beamformer is relative to the simple beamformer.

Many authors [30, 32, 33] compare the number of operations required to calculate a particular algorithm to identify the most efficient algorithm. However, a VLSI implementation may be more efficient in processing data than an array processor, which may turn the tables in the opposite direction. The table should only be treated as a guide, however it does reveal a few important ideas. Firstly, complex sampling is required for high frequency bandpass applications, and secondly, interpolation is necessary in the time domain to realise the small sample time necessary for high angular resolution.

2.1 Conclusions

It was shown that beamformers could be expected to operate on signals in a wide frequency range, and it is therefor important to consider the nature of the signals to be processed. Lowpass sampling is sufficient for low-frequency signals, however for high frequency bandpass and narrowband signals, bandpass sampling techniques must be adopted. It was also shown that interpolation could be used to increase the effective sampling frequency.

Beamforming was introduced using the simple time domain beamformer and later extended using interpolation and quadrature sampling. Beamforming in the frequency domain was also discussed, and in some cases maybe more efficient method of forming simultaneous beams. MATLAB simulations were given for each beamformer to supplement the understanding of the operations required in the beamformer. The simulations also give an insight into design considerations and specifications of a real implementation.

It is now possible to assess each type of beamformer, relative to each other, in terms of size and complexity.

Bibliography

[1] T. Curtis and R. Ward, "Digital beamforming for SONAR systems," *IEE Proceedings*, vol. 127, August 1980.

[2] M. Okino and Y. Higashi, "Measurement of seabed topography by multibeam sonar using cfft," *IEEE Journal of Oceanic Engineering*, vol. OE-11, October 1986.

[3] R. B. Mitson, "Review of high-speed sector-scanning sonar and its application to fisheries research," *IEE Proceedings*, vol. 131, June 1984.

[4] P. Barton, "Digital beamforming for RADAR," *IEE Proceedings*, vol. 127, August 1980.

[5] B. Edde, *RADAR - Principles, Technology, Applications*. Prentice Hall, 1993.

[6] M. O'Donnell, "Applications of VLSI circuits to medical imaging," *Proceedings of the IEEE*, vol. 76, September 1988.

[7] B. D. V. Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE Acoustics, Speech, and Signal Processing Magazine*, pp. 4–24, April 1988.

[8] J. F. Dix, "Initial SONAR system considerations," *IEE Colloquium Digest*, November 1978.

[9] J. Sotelo, *Matrix Circuit Descriptions and Test Information*. Interphase Technologies, 1201 Shaffer Road, Santa Cruz, California, 95060, October 1991.

[10] C. E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, pp. 10–21, January 1949.

[11] D. A. Linden, "A discussion of sampling theorems," *Proceedings of the IRE*, 1959.

[12] G. Hampson and A. Papliński, "Beamforming by interpolation," Tech. Rep. 93-12, Monash University, 1993.

[13] R. W. Schafer and L. R. Rabiner, "A digital signal processing approach to Interpolation," *Proceedings of the IEEE*, vol. 61, June 1973.

[14] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing*. Prentice Hall, 1989.

[15] J. L. Brown, "On quadrature sampling of bandpass signals," *IEEE Trnasactions on Aerospace and Electronic Systems*, vol. AES-15, May 1979.

[16] R. O. Nielsen, *Sonar Signal Processing.* Artech House, 1991.

[17] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," *Proceedings of the IEEE*, vol. 66, January 1978.

[18] G. Hampson, "Investigation into threee dimensional depthsounders/fishfinders." Monash University Honours Thesis, 1993.

[19] The MathWorks Inc., *PRO-MATLAB.*

[20] R. G. Pridham and R. A. Mucci, "A novel approach to digital beamforming," *Journal on Acoustical Society of America*, vol. 63, pp. 425–434, February 1978.

[21] R. G. Pridham and R. A. Mucci, "Digital interpolation beamforming for low-pass and bandpass signals," *Proceedings of the IEEE*, vol. 67, June 1979.

[22] P. M. Pierre Da Sylva and D. Roy, "A Reconfigurable Real-Time Interpolation Beamformer," *IEEE Journal of Oceanic Engineering*, vol. OE-11, January 1986.

[23] R. G. Pridham and R. A. Mucci, "Shifted Sideband Beamformer," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-27, December 1979.

[24] M. O'Donnell *et al.*, "Real-time phased array imaging using digital beam forming and autonomous channel control," *Ultrasonics Symposium*, pp. 1499–1502, 1990.

[25] S. M. K. W. C. Knight, R. G. Pridham, "Digital signal processing for sonar," *Proceedings of the IEEE*, vol. 69, November 1981.

[26] H. P. Hsu, *Applied Fourier Analysis.* Harcourt Brace Jovanovich, 1984.

[27] J. W. Cooley, P. A. W. Lewis, and P. D. Welch, "Applications of the Fast Fourier Transform to computation of Fourier integrals, Fourier series, and convolution integrals," *IEEE Transactions Audio and Electroacoustics*, vol. AU-15, June 1967.

[28] J. R. Williams, "Fast beam-forming algorithm," *Journal on Acoustical Society of America*, vol. 44, no. 5, pp. 1454–1455, 1968.

[29] P. Rudnick, "Digital beamforming in the frequency domain," *Journal on Acoustical Society of America*, vol. 46, no. 5, pp. 1089–1090, 1969.

[30] B. Maranda, "Efficient digital beamforming in the frequency domain," *Journal Acoustical Society of America*, vol. 86, November 1989.

[31] M. J. Hinich, "Processsing spatially aliased arrays," *Journal on Acoustical Society of America*, vol. 64, pp. 792–794, September 1978.

[32] M. E. Weber and R. Heisler, "A frequency domain beamforming alogorithm for wideband, coherent signal processing," *Journal Acoustical Society of America*, vol. 76, October 1984.

[33] R. A. Mucci, "A comparison of efficient beamforming techniques," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-32, June 1984.

[34] R. A. Mucci and R. G. Pridham, "Impact of beam steering errors on shifted sideband and phase shift beamforming techniques," *Journal Acoustical Society of America*, vol. 69, May 1981.

[35] J. K. V. S. P. Pitt, T. Adams, "Design and implementation of a digital phase shift beamfomrer," *Journal of Acoustical Society of America*, vol. 64, September 1978.

[36] R. E. Crochiere and L. R. Rabiner, "Optimum FIR digital filter implementations for decimation, interpolation, and narrow-band filtering," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-23, October 1975.

Appendix A

Sampling Procedures

An implicit requirement for digital beamforming techniques is the availability of accurate, unaliased digital data stream. Various sampling methods can be employed to convert the analog signal from the transducer into a digital form via an A/D converter.

This appendix contains information regarding the sampling procedures relevant to beamforming, which include Interpolation and Quadrature sampling methods.

A.1 Interpolation

Interpolation is the process of increasing the initial sampling frequency f_s to an increased sampling frequency of $\hat{f}_s$ by an integer factor D. ie., the new sampling frequency is $\hat{f}_s = D \cdot f_s$ and the time separation of samples is $\hat{t}_s = t_s/D$ where $t_s = 1/f_s$. Thus a sampling frequency of f_s satisfies the Nyquist criterion for the data, whereas $\hat{f}_s$ is required for accurate beamforming.

To interpolate a sequence $x(n)$, where the samples are separated by t_s seconds, zero padding is the first process:

$$\hat{x}(k) = \begin{cases} x(k/D), & \text{if } k = 0, \pm D, \pm 2D, \ldots \\ 0, & \text{otherwise.} \end{cases} \tag{A.1}$$

Note that $D-1$ zeros are inserted between each pair of terms in the original sequence $x(n)$. An operation is then performed to interpolate between the nonzero samples in the sequence $\hat{x}(k)$.

An intuitive approach to interpolating between pairs of samples is simply to use linear interpolation. It can be shown that linear interpolation is equivalent to filtering with a symmetric FIR filter [36] and exactly two input samples are used for each calculation.

It is expected that linear interpolation would perform poorly in the presence of high frequency signals and therefore the highest frequency in the original signal $x(n)$ must be considerably smaller than $f_s/2$ for linear interpolation to provide adequate performance. Figure (A.1) illustrates the process of interpolation to increase the effective sampling frequency.

Simulation using MATLAB [19] is achieved by a few simple MATLAB commands:

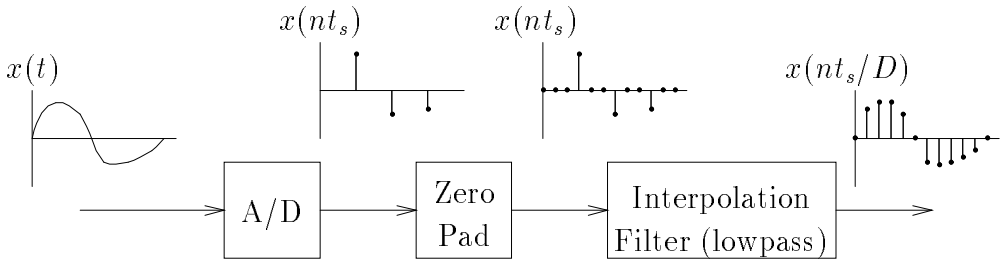

Figure A.1: The interpolation process.

```
x = sin(n*2*pi*f/fs);          % input waveform

x_d = x(1:D:length(x));        % decimate the input waveform by D

x_i = kron(x_d, eye(I, 1));    % use the tensor product to zero pad

fc = ((f/D)*I)/fs;             % normalised cut-off frequency
b = fir1(32,fc);               % filter co-efficients

x_r = filter(b, 1, x_i);       % filter the zero-padded waveform
```

Figure (A.2) illustrates an example where a original wave form (sampled at f_s) is decimated such that there are 4 samples a period. This is then interpolated and filtered by a factor of 9, ie., 9 times the sample rate of the decimated waveform. Note that the interpolated waveform has smaller amplitude due to the averaging nature of the filter. i.e., the higher the interpolation factor, the smaller the amplitude.

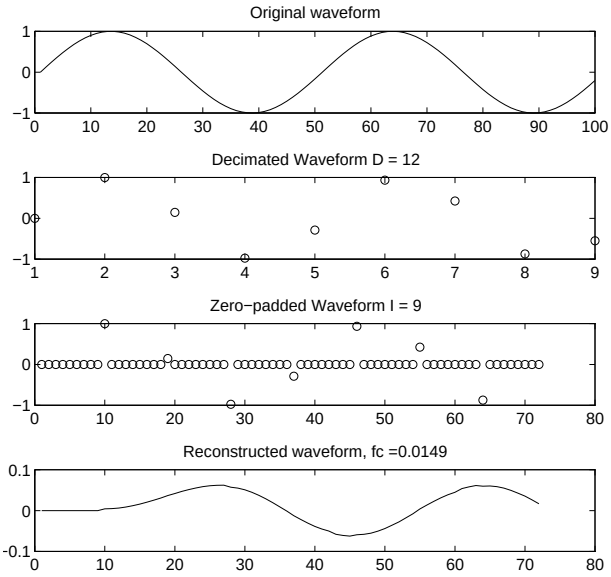

Figure A.2: MATLAB simulation of Interpolation.

A.2 Quadrature Sampling

A/D conversion can become expensive when considering lowpass sampling a 1-10MHz bandpass signal used in ultrasound medical imaging [6]. However, the nature of a bandpass or narrow band signals can be exploited [15] by band-pass sampling the signal.

Bandpass sampling methods include analytic, second order and quadrature sampling [21]. Utilizing these procedures, the data need only be sampled at a rate consistent with the signal bandwidth, rather than the highest frequency component. The increased complexity of these bandpass sampling methods is compensated by the efficient data representation and reduced data rates through the beamformer.

Using quadrature sampling, a complex envelope is calculated by shifting the bandpass spectrum of $x(t)$ by f_0 Hz to zero frequency, low-pass filtering and sampling, as illustrated in Figure (A.3). The output of the A/D converters is termed the in-phase and quadrature components ($x_I(n)$ and $x_Q(n)$ respectively) and forms a complex envelope defined by

$$\tilde{x}(t) = LP[\, x(t)e^{-j\omega_0 t}\,] \tag{A.2}$$

$$\tilde{x}(n) = x_I(n) + jx_Q(n) \qquad \text{where } n = t/t_s \tag{A.3}$$

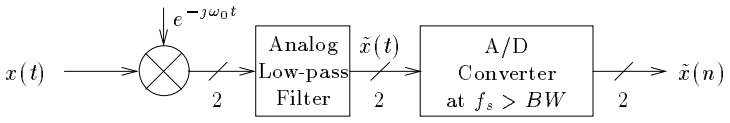

Figure A.3: Calculation of the complex envelope.

Because all the information of the bandpass signal is contained in the complex envelope $\tilde{x}(n)$, it can be sampled at a frequency f_s that is greater than only twice the highest frequency $B/2$ Hz (rather than greater than twice $f_0 + B/2$ Hz). It is also important to note that the complex envelope contains all the information of the original bandpass process $x(t)$, and hence all processing can be performed on the complex envelope.

The original signal $x(t)$ can be reconstructed by the inverse shift in the frequency domain, zero-order hold operation and low pass filtering:

$$x(n) = \Re\left(\tilde{x}(n) \cdot e^{j\Omega_0 n}\right) \tag{A.4}$$

$$= x_I(n) \cdot \cos\Omega_0 n - x_Q(n) \cdot \sin\Omega_0 n \tag{A.5}$$

$$x(t) = LP\,[ZOH[x(n)]] \tag{A.6}$$

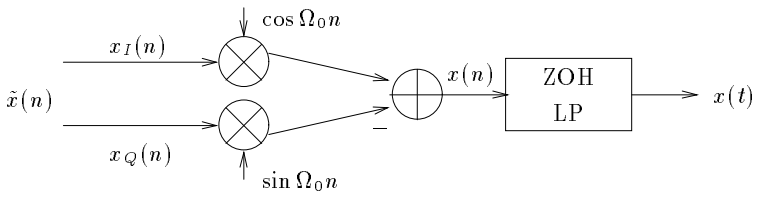

Figure A.4: Calculation of the original signal.

Simulation is MATLAB is easily achieved using the following MATLAB code framents:

```
x = [sin(n1*Om1); 0.5*sin(n1*Om2)]; % input waveform

demod = exp(-j*Omd*n);
xd = mod.* xdf;          % calculate complex envelope

b = fir1(16,(BW/2)/(fs/2));          % 16 tap filter coefficients
xdf = filter(b, 1, xd);              % filter quadrature samples

mod = exp(j*Omd*n);
xr = mod.* xdf;          % calculate complex envelope
```

Figure (A.5) illustrates an example of quadrature sampling procedure, where it is observed that the quadrature components are low frequency waveforms when compared to the input waveforms.

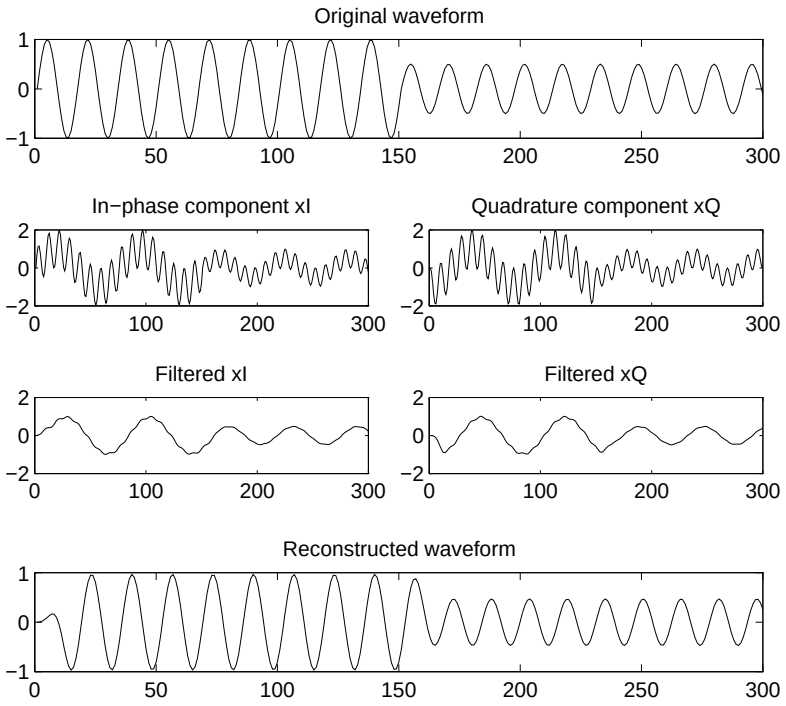

Figure A.5: MATLAB simulation of Quadrature Sampling.

Commonly the bandpass signals are interpolated to increase the effective sampling frequency at the beamformer if the original signal was not sampled at a sufficiently high rate for accurate beamforming. Using a large number transducers, the lowest A/D conversion rate is desirable to minimise hardware expense.

A modification of the quadrature sampling is obtained by demodulating the signal to a frequency above dc, referred to as shifted side band. More will be said about this at a later stage when beamforming fundamentals have been given.