WIDEBAND RADAR ADAPTIVE BEAMFORMING USING FREQUENCY-DOMAIN DERIVATIVE BASED UPDATING

Ben Mathews (NAVSYS Corporation, Colorado Springs, Colorado, USA, benm@navsys.com)
Jacob Griesbach (SAIC, Colorado Springs, Colorado, USA, griesbachj@saic.com)
Alison Brown (NAVSYS Corporation, Colorado Springs, Colorado, USA, alison@navsys.com)

ABSTRACT

This paper details an advanced wideband adaptive beamforming algorithm for multi-channel radar systems that is currently in the process of being implemented in an FPGA. This approach models the entire frequency spectrum and linearly tracks statistical source covariance changes over frequency. achieved using a technique called derivative based updating (DBU). The novel aspect of this algorithm is its application in the frequency domain to track jammer source variations in angle as a function of frequency. This algorithm is compared with a traditional subband ECCM technique for multiple jamming scenarios. Front-end array hardware recommendations are also considered in this presentation. The baseline front-end architecture involves use of non-overlapping subarrays to reduce the number of digitized channels. An overlapping architecture is recommended that gives greater flexibility and control over spatial ambiguities.

With wider operating bandwidths come increased processing demands on the beamforming hardware. By taking advantage of the increasing processing power available in commercially available Field Programmable Gate Arrays (FPGA), and with the increasing speed of modern digital interfaces, it is becoming increasingly feasible to perform wideband adaptive processing in the digital domain, where advanced beamforming techniques can be exploited to realize performance and flexibility benefits of a portable firmware-based implementation.

1. INTRODUCTION

The need for high-resolution radar imagery and accurate target identification is in high demand. Both of these applications require high bandwidth radar waveforms to provide the sensor with required range resolution. As technology improves to allow increasing bandwidths, electronic counter-countermeasure (ECCM) technology must also follow to allow such sensors to function in hostile jamming environments. Adaptive beamforming (ABF) algorithms can counter jamming threats by greatly attenuating power from the jammers' directions. By effectively eliminating power from jamming, while still maintaining gain at the aimpoint, the radar is able to spatially focus its aperture and maintain functionality. Historically, radars have employed narrowband ABF technology, which appropriately nulls jamming interference for a specific frequency. However, as bandwidths increase, the narrowband assumption becomes invalid as wideband jammers will appear to spread in angle. This spreading allows the jammers to deny greater area to the sensor, limiting the sensor's usefulness, and is inherent with narrowband phase-based steering.

This paper presents a wideband derivative based updating (DBU) adaptive beamforming (ABF) algorithm and compares its performance with the more traditional Subband ABF (SABF) algorithm. The paper includes details relating non-adaptive or conventional beamforming (CBF) algorithms to the adaptive algorithms.

The DBU-ABF algorithm has several key advantages over SABF:

- Wideband beams vary smoothly over frequency reducing signal distortion
- Narrower jammer extents to minimize area denied by jamming
- Natural wideband modeling without forcing a narrowband solution to be wideband
- More accurate covariance modeling with more available training samples
- Works naturally with stretch processing for high bandwidth operation

A variety of features are discussed in detail, such as wideband beamforming, tapering and colored noise loading for sidelobe suppression, and white noise gain constraints for adaptive beam control. Finally, an approach is outlined for FPGA implementation.

2. SENSOR CONFIGURATION

The sensor configuration used in this paper was chosen to better control the grating lobes that occur with standard non-overlapped subarray configurations. This configuration is defined by the following parameters:

- 14 horizontal subarrays with 2/3 overlap
- $N_{se} = 48$ elements per subarray
- $d_{se} = 16$ element subarray phase center spacing

- $d_s = d_{se} \dfrac{c}{2 f_c} =$ subarray phase center spacing

With 2/3 overlapped horizontal subarrays, the subarrays overlap such that the majority of the elements participate in three subarrays.

Figure 1 shows pictorially how the individual array elements are combined to form subarrays. The combining process begins with analog time-delay beamsteering to align the element in time with respect to a selected AOI. Also guard channels are tapped and an array taper is applied. Next, a subarray taper is applied to the elements forming each subarray. The subarray taper is real valued, and the same subarray taper is applied to each subarray. Since each individual element participates in up to 3 subarrays, element signals are split up to 3 ways before the subarray tapers are applied. Following the subarray taper, the elements involved in each subarray taper are added together in the subarray formation step. Then each subarray formation signal is demodulated, IQ processed, and digitized to produce a complex data channel.

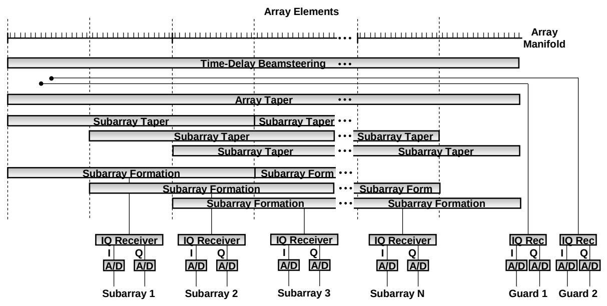

Figure 1: 2/3 overlapped subarray architecture

To demonstrate the motivation for selecting this configuration, the beampattern was calculated for it and for a non-overlapped configuration consisting of 16 subarrays, each consisting of 16 antenna elements. A full-array taper is applied to both configurations, as described in [1]. The beampattern plot for the standard non-overlapped configuration is shown in Figure 2, and beampattern plot for the overlapped configuration described in this section is shown in Figure 3.

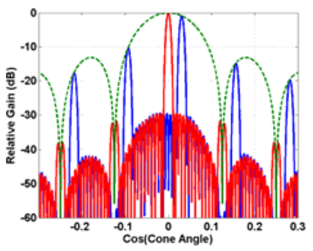

Figure 2: Beampattern of non-overlapped configuration

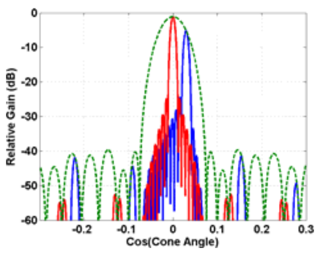

Figure 3: Beampattern of overlapped configuration

In these plots the red beampattern illustrates a very special case, where the subarray nulls directly counter the beampattern grating lobes that would normally be spaced at the same locations. However, when the beam is shifted slightly, the grating lobes emerge. This effect is even more prevalent in the absence of the full-array taper. Note how the subarray beampattern envelope is much tighter (3x actually) due to the subarray overlapping. In this configuration all of the grating lobes are controlled to remain below -40 dB, while the mainlobe is broadened to allow for simultaneous beam scanning.

3. CONVENTIONAL AND ADAPTIVE BEAMFORMING

3.1. Narrowband Conventional Beamforming

Most beamforming techniques are based on the computation of a replica signal, or a steering vector, for signals arriving from an angle θ with a center frequency f, given as

$$\mathbf{v}(\phi, f) = \exp\left\{\begin{bmatrix} \vdots \\ -j2\pi \frac{f}{c} d_e \sin(\theta) \\ 0 \\ j2\pi \frac{f}{c} d_e \sin(\theta) \\ j4\pi \frac{f}{c} d_e \sin(\theta) \\ \vdots \end{bmatrix}\right\} = \exp\left\{\begin{bmatrix} \vdots \\ -j2\pi \frac{f}{c} d_e \phi \\ 0 \\ j2\pi \frac{f}{c} d_e \phi \\ j4\pi \frac{f}{c} d_e \phi \\ \vdots \end{bmatrix}\right\}. \qquad (1)$$

Since $\sin(\theta)$ is nonlinear, we define $\phi = \sin(\theta)$ to be the cosine of the signal angle relative to the array manifold, or the signal cosine cone-angle. In the narrowband case or when irreverent, the frequency f is left out and the steering vector is written simply as $\tilde{\mathbf{v}}(\phi)$.

Due to the hardware subarray formation, the steering vectors presented in (1) do not represent the signal that is actually recorded. In fact, the recorded signal is a linear combination of the array elements and thus the recorded signal replica may be specified as

$$\tilde{\mathbf{v}}(\phi, f) = \mathbf{C}\mathbf{v}(\phi, f), \qquad (2)$$

where the $N + N_{aux} \times N_X N_Z$ subarray formation matrix, $\mathbf{C}$, linearly combines the array elements to form the output channels that are digitally sampled. The subarray formation matrix combines the combinatory effects of the full array and subarray tapers (if used) and the subarray formation steps. The hardware time-delay steering effects are not encompassed by this matrix.

Herein, we refer to non-adaptive beamforming as conventional beamforming (CBF). Conventional beamforming consists of computing steering vectors for a narrowband (tone) signal (at the radar center frequency) arriving from a set of hypothetical directions.

For CBF beamforming, the normalized full array taper weights are applied to the data to restore the taper effect to the data. These denormalizing values may be realized as a diagonal matrix that may be directly applied to the steering vectors as

$$\mathbf{v}_{CBF}(\phi) = \mathbf{D}_{norm}\tilde{\mathbf{v}}(\phi), \quad (3)$$

Finally, the unit gain response may be computed as

$$y(\phi,t) = \frac{\mathbf{v}_{CBF}(\phi)^H \mathbf{x}_t}{\sqrt{\mathbf{v}(\phi)^H \mathbf{v}(\phi)}}, \quad (4)$$

where $\mathbf{x}_t$ denotes the channel data vector for time t. This equation is repeated for all hypothesis cosine cone-angles, ϕ, and time instances, t.

Computing a unit gain response may be desired for some applications where further adaptive processing is to be performed to mitigate clutter. However, the noise floor level varies over ϕ due to the subarray beampattern. For signal detection, a noise balanced or deshaded response may be desirable. This response may be computed as

$$z(\phi,t) = \frac{\mathbf{v}_{CBF}(\phi)^H \mathbf{x}_t}{\sqrt{\mathbf{v}_{CBF}(\phi)^H \mathbf{C}\mathbf{C}^H \mathbf{v}_{CBF}(\phi)}}. \quad (5)$$

3.2. Wideband Conventional Beamforming / Time-Delay CBF (TDCBF)

Wideband beamforming may be considered by forming steering vectors as a function of frequency, or equivalently, performing an element specific time-delay for each desired beam, as

$$\mathbf{v}_{TDCBF}(\phi, f) = \mathbf{D}_{norm}\tilde{\mathbf{v}}(\phi, f), \quad (6)$$

$$y(\phi, f) = \frac{\mathbf{v}_{TDCBF}(\phi, f)^H \mathbf{X}_f}{\sqrt{\mathbf{v}(\phi, f)^H \mathbf{v}(\phi, f)}}, \quad (7)$$

where $\mathbf{X}_f$ denotes the data vector in the frequency domain for frequency f. Here, the spatial combination is done in the frequency domain by applying a fast-time FFT to the data. When applied in the frequency (FFT) domain, the time-delays correspond to circular time-delay shifts in fast-time. This equation is again repeated for all hypothesis cosine cone-angles, ϕ, and frequencies, f.

A time-domain version of (7) may be written as

$$y(\phi,t) = \frac{\mathbf{v}_{TDCBF}(0)^H \mathbf{x}_{t,\frac{d_s}{c}\phi}}{\sqrt{\mathbf{v}(0)^H \mathbf{v}(0)}}, \quad (8)$$

where $\mathbf{x}_{t,\delta} = [\cdots \quad x_{n-1,t+\delta} \quad x_{n,t} \quad x_{n+1,t-\delta} \quad x_{n+2,t-2\delta} \quad \cdots]^T$ and $x_{n,t}$ is a complex data sample from channel n and time t.

3.2. Narrowband Adaptive Beamforming

The standard approach to adaptive beamforming is to utilize the minimum variance distortionless response (MVDR) beamforming result [2]. Given time snapshots of the data from each channel stacked in a vector, $\mathbf{x}_t = [x_{1,t} \quad x_{2,t} \quad \cdots \quad x_{N+N_{aux},t}]^T$, where $x_{n,i}$ is the complex data sample from channel n and time index t, a covariance matrix is constructed as

$$\mathbf{R} = \frac{1}{N_t}\sum_t \mathbf{x}_t\mathbf{x}_t^H + \delta\mathbf{I}, \quad (11)$$

where N_t equals the number of time snapshots summed, δ is the system noise power level and $\mathbf{I}$ is an $(N+N_{aux})\times(N+N_{aux})$ identity matrix.

A steering vector, $\mathbf{v}(\phi)$, is necessary to provide the desired phase response of the array to the particular direction of interest the beam should be directed. The steering vector must be combined as the elements were combined into subarrays, thus the appropriate channel steering vector is given by $\tilde{\mathbf{v}}(\phi)$ from (2). Note that for adaptive beamforming, the taper denormalization matrix, $\mathbf{D}_{norm}$, is not applied, such that the steering vector matches the normalization of the data.

Once $\mathbf{R}$ and $\tilde{\mathbf{v}}(\phi)$ are computed, the resulting MVDR weight vector is obtained as

$$\mathbf{w}_{MVDR}(\phi) = \frac{\mathbf{R}^{-1}\tilde{\mathbf{v}}(\phi)}{\tilde{\mathbf{v}}(\phi)^H \mathbf{R}^{-1}\tilde{\mathbf{v}}(\phi)}. \quad (12)$$

Finally, the ABF output at time index t is computed by multiplying the weight vector against the data as

$$\mathbf{y}_t = \mathbf{w}_{MVDR}{}^H \mathbf{x}_t. \quad (13)$$

As (12) normalizes the weight vector by the steering vector response, it provides the unit gain response equation that may be compared with (4).

The deshaded version of the weight vector is given as

$$\mathbf{w}_{AMF}(\phi) = \frac{\mathbf{R}^{-1}\tilde{\mathbf{v}}(\phi)}{\sqrt{\tilde{\mathbf{v}}(\phi)^H \mathbf{R}^{-1}\tilde{\mathbf{v}}(\phi)}}, \quad (14)$$

where the noise power is adaptively estimated in the denominator using the covariance matrix. This is commonly called the adaptive matched filter solution and is technically no longer MVDR. A deterministically deshaded weight vector may be specified as

$$\mathbf{w}_{DMF}(\phi) = \frac{\mathbf{R}^{-1}\tilde{\mathbf{v}}(\phi)}{\sqrt{\tilde{\mathbf{v}}(\phi)^H (\mathbf{C}\mathbf{C}^H)^{-1}\tilde{\mathbf{v}}(\phi)}}, \quad (15)$$

where the matrix in the denominator is substituted by the deterministic covariance noise matrix $\mathbf{C}\mathbf{I}\mathbf{C}^H$.

3.2. Wideband (Subband) Adaptive Beamforming

Wideband adaptive beamforming is more complex than wideband CBF processing, since a covariance matrix must be computed for each frequency. Since it is very difficult to obtain accurate covariance matrices for individual frequencies, the frequency spectrum is separated into subbands, where the assumption is that the subbands are narrow enough such that narrowband ABF may be applied separately in each band.

Thus the first step is to create a filterbank of M analysis and synthesis filters that may be used to break apart the individual subbands and combine them back together to reform a fullband signal. There are a number of techniques to do so, but they consist of two categories, perfect reconstruction (PR) and near-perfect reconstruction (NPR). PR filters exactly recreate the original signal, while NPR filters sacrifice perfect reconstruction to obtain properties such as low-sidelobes or better frequency containment.

Subband adaptive beamforming has a fundamental difficulty in that as tighter subbands are required, more data samples are required to obtain accurate covariance estimates. Thus, the number of subbands is really limited by the number of samples in a pulse repetition interval (PRI), the number of adaptive data channels, and the desired accuracy of the covariance matrix.

Figure 4 illustrates the frequency response associated with each of the filterbank filters used in this paper. The magnitude responses are identical between the analysis and synthesis filterbanks, since the filters were generated using the pseudo-quadrature mirror filter technique.

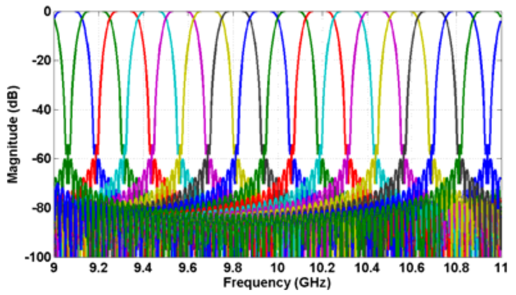

Figure 4: 16-Band NPR filterbank

4. TIME-DELAY DBU ADAPTIVE BEAMFORMING

4.1. Motivation

This algorithm uses a technique called derivative based updating (DBU) to effectively compute a wideband covariance matrix.

This technique involves an FFT operation so that the ABF algorithm may be applied in the frequency domain. Furthermore, it involves applying time-delays to each individual channel, as with the TDCBF algorithm outlined in Section 3.2, to pre-steer a non-adaptive wideband beam in the direction of interest before ABF is utilized. Thus, if the ABF algorithm were replaced with a channel sum, the result would be a non-adaptive beam output.

Since this technique is applied in the frequency domain with pre-steered data, it may be likened to applying ABF in a subband architecture with a very large number of subbands. However, to avoid ABF discrepancies over frequency (or subband-to-subband) that may lead to spectral or range distortion, derivative based updating (DBU) is applied, which smoothes the adaptive weight vectors over frequency and allows the ABF algorithm to train with nonstationary data [3]. Also, a white noise gain constraint (WNGC) is employed for beam control.

4.2. Derivative Based Updating (DBU)

Derivative based updating (DBU) is a technique that allows the covariance matrix to be trained and applied with nonstationary data. DBU has been applied in the past for rapidly moving jamming scenarios [4,5]. In the case of wideband adaptive beamforming, the data will be nonstationary if a jammer is present at an angle other than the beam's aimpoint angle. Even though the data has been pre-steered via time-delays to the direction of interest with (6), signals arriving at other angles will still vary in phase over frequency. This causes the jammer direction to apparently change as a function of frequency. DBU can account for this by tracking the jammer's angle derivative.

While the time-delays have been applied to each channel, the channels are not yet summed together, as this is done adaptively to mitigate jamming. To accomplish this, the standard covariance matrix in (11) is augmented as

$$\mathbf{R}_{dbu} = \frac{1}{N_f}\sum_f \begin{bmatrix} \mathbf{X}_f\mathbf{X}_f^H + \delta\mathbf{I} & k\mathbf{X}_f\mathbf{X}_f^H \\ k\mathbf{X}_f\mathbf{X}_f^H & k^2\mathbf{X}_f\mathbf{X}_f^H + \delta\mathbf{I} \end{bmatrix}, \quad (16)$$

where $k = 2\frac{f - f_{\min}}{f_{\max} - f_{\min}} - 1$, a normalized frequency index that maps f between -1 and +1.

Next, the DBU weight vector and its derivative is computed as

$$\begin{bmatrix} \mathbf{w}_0(\phi) \\ \dot{\mathbf{w}}(\phi) \end{bmatrix} = \mathbf{R}_{dbu}^{-1} \begin{bmatrix} \tilde{\mathbf{v}}(0) \\ \mathbf{0} \end{bmatrix}, \quad (17)$$

where $\mathbf{w}_0(\phi)$ denotes the DBU weight vector solution at $k = 0$ and $\dot{\mathbf{w}}(\phi)$ denotes its derivative. The zero angle steering vector, $\tilde{\mathbf{v}}(0)$, is used because the data has been already pre-steered via time-delays. Thus, the weight vector appropriate for other frequency indices is provided by

$$\mathbf{w}_k(\phi) = \mathbf{w}_0(\phi) + k\dot{\mathbf{w}}(\phi). \quad (18)$$

Thus, a different weight vector is computed for each FFT frequency. The weight vectors are multiplied and channel summed against the pre-steered data to form the output beam. A final IFFT may be used to convert the data back into the time-domain, if necessary.

This allows the ABF weight vector to linearly change over frequency to track the jammer's apparent moving angle as a function of frequency, providing a truly wideband ABF.

4.3. DBU with WNGC and Colored Noise Loading

Since the DBU weights vary linearly with respect to frequency in (18), it can be shown that if the weight vectors at the frequency extremes satisfy the WNGC [6], then all the intermediate frequency weight vectors also satisfy the WNGC. Thus, only the weight vectors associated with the frequency extremities need be checked to satisfy the WNGC. As previously mentioned, if either extremity fails the WNGC, then additional diagonal loading is added to (16) until the extremity weight vectors satisfy the WNGC.

Colored noise loading is again desirable to control the near-in sidelobes [6]. As before, the identity matrices in (16) may be replaced with the colored noise loading matrix for initial loading and for WNGC loading to control the sidelobes.

5. SIMULATION RESULTS

5.1. Simuation 1: Strong Wideband Jamming

In this simulation the environment is configured as shown in Table 1, where the jamming environment is defined in terms of jammer-to-noise ratio per element (JNR), cosine cone-angle (CCA), beamwidths from broadside or time-delay steered direction (BmWdths), degrees from broadside (deg), source bandwidth in GHz (BW), source center frequency in GHz (CF), and category (Cat), which can either be mainbeam (MB), or sidelobe (SB).

JNR	CCA	BmWdths	Deg	BW	CF	Cat
10 dB	0	0	0	2	10	MB
10 dB	-0.04	-4	-2	2	10	MB
10 dB	0.148	19	8.5	2	10	SB

Table 1: Strong Wideband Jamming

As shown in Figure 5, the strong jamming overwhelms the sidelobes of the conventional beamformer. Thus, the jamming consumes nearly the entire angle space available to the radar.

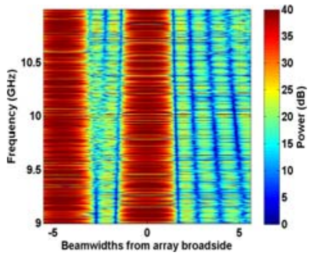

Figure 5 TDCBF Response 1

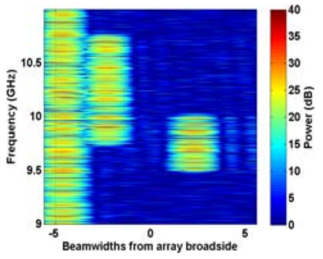

Figure 6 TDCBF Response 2

Figure 7 and Figure 8 show the Subband ABF and DBU-ABF results. These algorithms adaptively null the jamming unless a beam is pointing directly at (or very near) to a jammer. The DBU-ABF has a narrower jammer extent, especially for the jammer located 4 beamwidths left of broadside. Traces of grating lobe leakage for the sidelobe jammer are evident in both figures. However, it is more prominent with the Subband ABF algorithm.

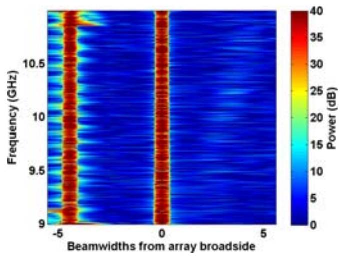

Figure 7 SABF Response 1

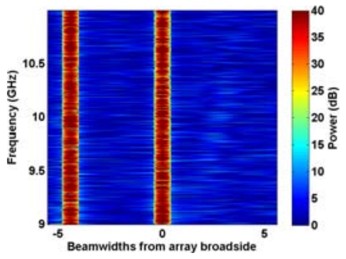

Figure 8 DBU-ABF Response 1

5.1. Simulation 2: Mixed Bandwidth Jamming

This simulation mixes together jammers with various bandwidths and center frequencies. Here, strong wideband, halfband, and quarter-band jammers are present with various center frequencies. Also, a strong CW (tone) jammer is also present, as outlined in Table 3.

JNR	CCA	BmWdths	Deg	BW	CF	Cat
-10 dB	-0.035	-4	-2	2	10	MB
-10 dB	-0.017	-2	-1	1	10.25	MB
-10 dB	0.017	2	1	0.5	9.75	MB
-20 dB	0.035	-4	2	0	9.5	MB

Table 2: Mixed Bandwidth Jamming

Figure 6 shows the TDCBF response to the mixed jamming scenario. Each of the 4 jammers are visible in the figure, although the CW jammer is hard to see because of its narrow bandwidth.

Figure 9 and Figure 10 compare the Subband ABF and DBU-ABF results. The Subband ABF algorithm has good frequency containment but worse spatial extent. Furthermore, the ends of the frequency portions of each jammer are distorted due to the subband band edges.

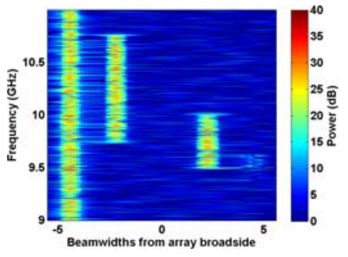

Figure 9 SABF Response 2

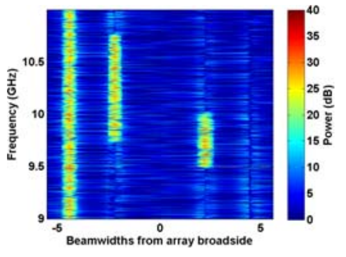

Figure 10 DBU-ABF Response 2

The DBU-ABF algorithm has better spatial extent, but each of the colored jammers leaves behind a "wake" of frequency. This is caused by the colored noise loading trying to compensate for being close to the jammer angle. The algorithm is trading off sidelobe suppression for a close-in null, and some sidelobe leakage from the other jamming sources is occurring. This may be corrected by adjusting the colored noise loading parameters.

8. HARDWARE IMPLEMENTATION DESIGN

A design for a firmware-based implementation of the DBU-ABF algorithm has been created and is under active development as the next stage of this project. Figure 11 illustrates the FPGA design architecture for a single beam. The design is based on having the input and output data stream on and off the FPGA. Thus, the input data streams onto the chip one range gate at a time, simultaneously for each of the 16 proposed complex channels. Thus 16 real and 16 complex 8-bit values will stream onto the device each clock cycle. The output will be a single complex value for each beam that streams off the device each clock cycle. With this streaming architecture, the required matrix operations are much simpler and require fewer resources.

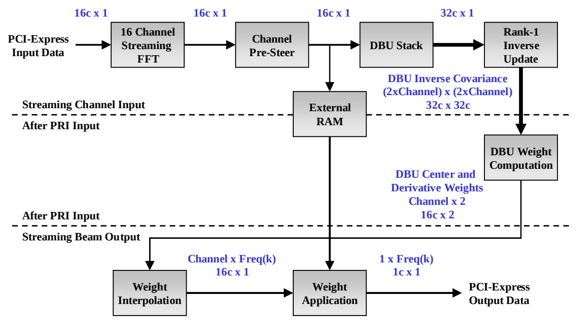

Figure 11 FPGA firmware block diagram

The first step in the diagram is to convert the incoming data into the frequency domain. A frequency dependent pre-steering vector is applied to the incoming data. Next, the data streams to an external dual-port RAM that stores the pre-steered channel data until the ABF weights have been computed. The RAM is dual-ported such that data may be stored while data elsewhere in the RAM maybe read. This allows concurrent input and output data streams.

The next step in the input chain is to perform the DBU stacking operation. This simply stacks the 16 channel data on top of itself creating a 32 channel vector, where the bottom portion is multiplied by it's frequency index number, k. The final step in the input chain is to accumulate the inverse covariance matrix for the current PRI of input data. Once all the samples in a PRI have streamed onto the device the DBU center and derivative weight vectors are computed using (17). The data from the PRI then begins to stream out from the external RAM storage. Simultaneously, the weight vectors are interpolated using (18) for the current frequency index.

Finally, the current frequency dependent weight vector is multiplied against the data and summed over the channel dimension forming the final scalar output value, and data is streamed off of the device in the frequency domain. If desired, a final streaming IFFT may be inserted to convert the data back to the time-domain.

9. CONCLUSIONS

The wideband DBU-ABF algorithm is a robust algorithm for wideband beamforming. In this paper we have demonstrated that it offers several key benefits over Subband adaptive beamforming in that it truly models the wideband system, allowing it to linearly vary its beamforming characteristics over frequency. Most importantly, it provides narrower jammer extents than Subband ABF with the same WNGC parameters. This minimizes the area denied to the radar by interference or jamming. Furthermore, it more accurately recovers the jamming signal for potential follow-on use with COMINT/SIGINT processing. A COTS FPGA implementation design has been presented and is under active development. Test results from this implementation with real data will be presented in subsequent papers.

10. REFERENCES

[1] J.D. Griesbach, "Optimal Taper Design for Overlapped Subarray Formation", Proceedings of the 40th Asilomar conference on signals, systems, and computers, Pacific Grove, CA, Oct. 29 – Nov. 1, 2006
[2] H.L. Van Trees, "Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing," Wiley, New York, NY, April 2002.
[3] S.D. Hayward, "Adaptive beamforming for rapidly moving arrays", CIE International Conference Proceedings, IEEE, Beijing, China, , pp. 480 - 483, Oct. 1996
[4] J.D. Griesbach, "Adaptive Beamforming Techniques for Sidelobe Control and Mitigation of Nonstationary Interference", in Proceedings of the Adaptive Sensor Array Processing (ASAP) Workshop, MIT, Lexington, MA, Jun. 2005
[5] M. Zatman, "Performance Analysis of the Derivative Based Updating Method", in Proceedings of the Adaptive Sensor Array Processing (ASAP) Workshop, MIT, Lexington, MA, Mar. 2001
[6] J.D. Griesbach, "Frequency-Domain Derivative-Based Updating for Wideband Radar Adaptive Beamforming", in Proceedings of the Adaptive Sensor Array Processing (ASAP) Workshop, MIT, Lexington, MA, May 2007