Efficient Mixed Implementation Method for Audio Source Tracking

Issa Jaafar[1] and Jarvis Tirk[1]

[1]Communication Engineering Technology, Assiniboine Community College
Brandon, Manitoba-Canada
Issajaafari@Assiniboine.net, Tirkj@ Assiniboine.net

Abstract

It's known that most of the audio sources tracking techniques are complicated as they require high degree of computations and considerable long real time processing. As a result of this, only certain hardware architectures can meet these application requirements. This work is assessing a previously presented source detection algorithm which meant to be adaptive in real time spectrum searching targeting less workload. The described method in this paper is mixing the good features of high accuracy of Delay and Sum Beamforming (DSB) algorithm and low complexity of the General Cross Correlation (GCC) technique. The requirements and constraints were fairly analyzed to reach an optimal implementation that overcomes efficiently the limitations of the alternatives. The presented results show clearly an identical accuracy performance with faster real time implementation.

1. Introduction

Audio source detection and tracking is currently a field of great interest due to the large number of possible applications [1] [2] [3]. Based on an array of closely positioned microphones, beamforming algorithms [4] [5] [6] can filter noise, reverberation effects and interference audios coming from other directions, by focusing the beam towards the selected audio source location. Among possible applications, we can mention to increase the audio quality of handheld devices or security cameras able to target and follow an audio source.

Microphone array algorithms base on the Time Difference of Arrival (TDOA) of the audio signal to each microphone. There are two major groups of microphone-array processing algorithms: time-invariant and adaptive [5] [6]. Algorithms in the first group, such as the Generalized Cross-Correlation with Phase Transform (GCC-PHAT) or the Delay and Sum Beamformer (DSB), are fast and simpler, suitable for a real-time implementation. Adaptive algorithms are able to automatically adapt their response to different weightings or time-delays, though, they require more processing power and are complex to implement. Therefore this work focuses on the GCC and DSB algorithms.

The GCC-PHAT is used to find the TDOA and then to deduce the Direction of Arrival, or angle, of the audio source target. The DSB calculates the Steered Response Power (SRP) at different locations in a predefined area, called the Field of View (FOV), to yield an acoustic power map. Since the DSB still requires a lot of computation, i.e. processing power, we propose a mixed algorithm that uses the angle provided by GCC to reduce the FOV of DSB while preserving its accuracy [7]. Consider Fig. 1 (below-left), where there is an example of a FOV for a audio source located at 2.5 meters above the microphone array at 45 degrees from the center. By using our mixed algorithm the audio source is located faster since the FOV is reduced, as shown in the right. The scale at the right side of both figures is the Steered Response Power (SRP) which will be detailed later.

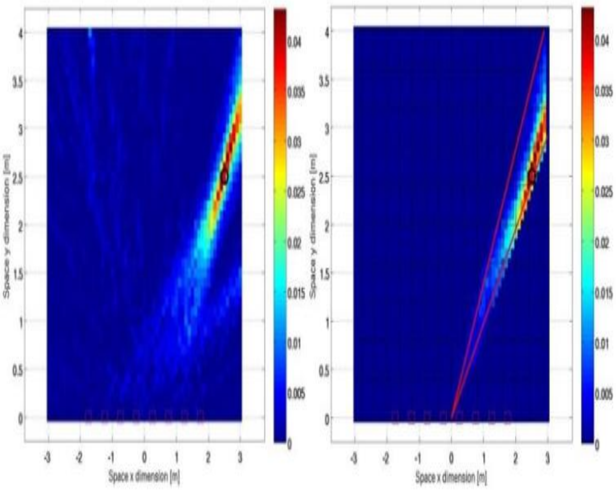


Fig. 1. Steered Power Response (SRP) of a audio source obtained with the DSB algorithm (left) and with the mixed one (right).

Pure software implementation of beamforming algorithms combined with a large number of microphones presents a significant challenge for real-time processing. Although they are more suitable to parallel processing, which guarantees the highest throughput at the shortest execution time, a parallel solution cannot be done without an additional cost in energy consumption and resources utilization [7] [8] [9] [10] [11]. Therefore, an additional contribution of this work is to evaluate hardware alternatives to achieve with an optimal solution.

The remainder of this paper will be organized as follows. In Section 2 we describe the principles of the algorithms considered. In Section 3 we describe our mixed algorithm proposal. In section 4 we analyse the algorithms' efficiency and complexity regarding hardware feasibility. In Section 5 we compare the computation load performance. In section 6 we present preliminary results about accuracy considerations. Finally, in section 7, we conclude this work and advice further expectation.

2. **Audio Source Tracking Algorithms**

This section describes the principles of the algorithms evaluated. For the reasons already mentioned, real-time execution and complexity, only time-invariant algorithms were considered: the *Generalized Cross-Correlation with Phase Transform* (GCC-PHAT) and the *Delay and Sum Beamformer* (DSB). For the sake of simplicity, some considerations where adopted [1]. Firstly, no multipath contribution is considered and the noise sources are stationary. Secondly, this work applies to a linear microphone-array and the audio source is far enough from the microphones so that the difference between the signals received by the *n-th* microphone located at *xn* and the one at the center of the array is a pure delay (*far-field approximation*) [6].

A. GCC-PHAT

The Generalized Cross Correlation (GCC) algorithm returns an angle *Φ* which is the audio source Direction of Arrival (DOA). To compute *Φ*, GCC uses the temporal shift between two microphones that leads to the maximum cross-correlation as in equation (1):

$$\Delta_{ij} = arg_k \max R(k) \tag{1}$$

The cross correlation is computed by taking the Inverse Fourier transform (IFFT) of the product between the Fast Fourier Transform (FFT) of one microphone and the FFT conjugate of the other, as shown in equation (2):

$$R(k) = IFFT\left(FFT(f(t)).FFT^*(g(t))\right) \tag{2}$$

FFT(f(t)) is the Fourier transform of the signal at the microphone *i*, and *FFT*(g(t))* is the Fourier transform conjugate of the signal at the microphone *j*. IFFT is required to go back to the time domain and extract the value corresponding to the index *k*. The *Phase Transform* (*PHAT*) is usually applied to equation (2) to correct the phase, which improves the robustness against noise and other adverse solution. Equation (3) defines the GCC-PHAT:

$$R(k) = IFFT\left(\frac{FFT(f(t)).FFT^*(g(t))}{\left|FFT(f(t)).FFT^*(g(t))\right|^{\beta}}\right) \tag{3}$$

β is a coefficient factor in the interval (*0, 1*). Considering the *Far Field Approximation*, the cosine of the arrival angle measured by microphones *i* and *j* can be computed as follows:

$$\cos(\emptyset_{ij}) = \frac{k v_s}{f_s d_{ij}} \tag{4}$$

With f_s as the sampling frequency, d_{ij} the distance between microphones *i* and *j*, and v_s the audio speed. Because every pair of microphones can provide one angle, the results from them can be combined (using a least squares solution) as:

$$\cos(\emptyset) = \frac{(\mathbf{d}\mathbf{d}^{\mathrm{T}})^{-1}\mathbf{d}k v_s}{f_s} \tag{5}$$

Where **d** is a vector containing the distances between all pairs of microphones and **k** is the vector containing the index k_{ij} extracted from their cross-correlation. Therefore the angle of arrival is defined by the *arccosine* of equation (5).

B. DSB-SRP

The *Delay and Sum Beamformer* (DSB) computes the *Steered Response Power* (SRP) of the audio source within a predefined region called *Field Of View* (FOV). The point with the greatest SRP is where the audio source is located.

The SRP is calculated for each point *p* in the FOV as in equation (6):

$$SRP_p = \sum_{k=1}^{N} \sum_{j=k+1}^{N} R(\tau_{ij}) \mid i \neq j \tag{6}$$

For each pair of microphones (i,j) is computed the *theoretical delay* τ_{ij} given by equation (7):

$$\tau_{ij} = \frac{d_{ij} f_s}{v_s} \tag{7}$$

d_{ij} is the difference between d_i and d_j which refer to the distances from the point *p* to microphones *i* and *j*, respectively. By using this index, the corresponding energy value in $R(\tau_{ij})$ the cross-spectrum output is extracted.

The size and shape of the FOV is application dependent. Fig.2 shows the example of an FOV of size 150x150 cm^2 split into 9 small squares of size 50x50 cm^2. The Number of Small Squares (NOSS) is related to the resolution as defined by equation (8):

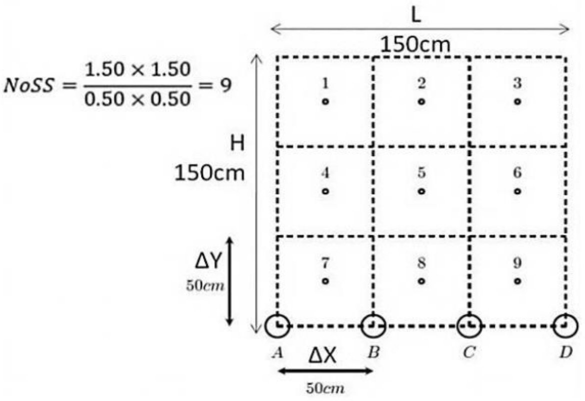


Fig. 2. 2D 3x3 FO V of size 150cm x 150cm a nd r esoluti on 50cm*50cm

$$NOSS = \frac{FOV}{a} = \frac{L \cdot H}{\Delta x \cdot \Delta y} \qquad (8)$$

3. Our Contribution

Our contribution accelerates the DSB computation by using the GCC, while preserving its accuracy. In addition, since the DSB still requires a lot of computation, i.e. processing power, the proposed mixed algorithm uses GCC to reduce the FOV of DSB, thus the number of points to be computed. To illustrate this mechanism, we will use a 4-microphone array (microphones A to D) and a FOV of size L*H with a resolution a=Δx* Δy , as shown in Fig. 3.

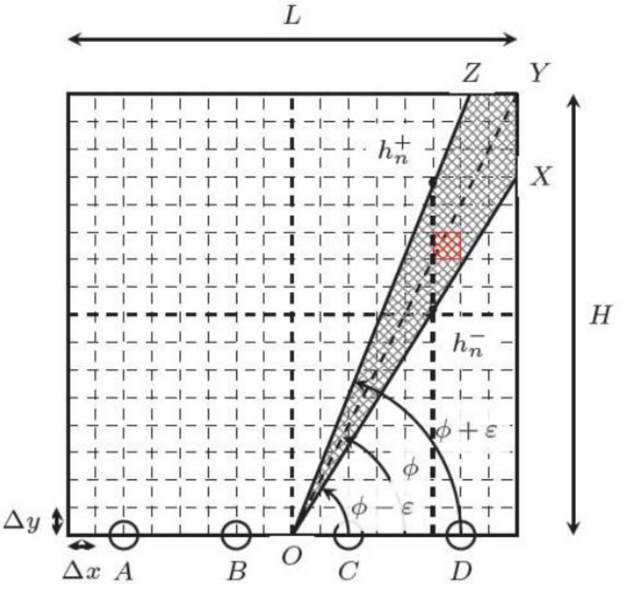


Fig .3. FOV of the GCC-DSB

The angle *Φ*, the direction of the audio source, is first provided by GCC (*0 ≤ Φ ≤ 180*) considering a linear microphone-array). The DSB search region is then limited by the angles *Φ* ± *ε*, where ε is a user defined parameter (its minimum value depends on the GCC angular precision). Our algorithm can be described as follows: The area of the cone encapsulated by the two lines (O-Z and O-X) is governed by Equations (9) and (10):

$$h_n^- = n \cdot \Delta x \cdot \tan(\phi - \varepsilon) \qquad (9)$$

$$h_n^+ = n \cdot \Delta x \cdot \tan(\phi + \varepsilon) \qquad (10)$$

The value of n represents the column, whilst both *h*, the associated minimum and maximum heights. The numbering of the small squares, the coordinates of the points where the SRP is to be computed, is highly important. Indeed, a sequential numbering, as shown in Fig.4 , greatly increases the difficulty of determining the enclosed region.

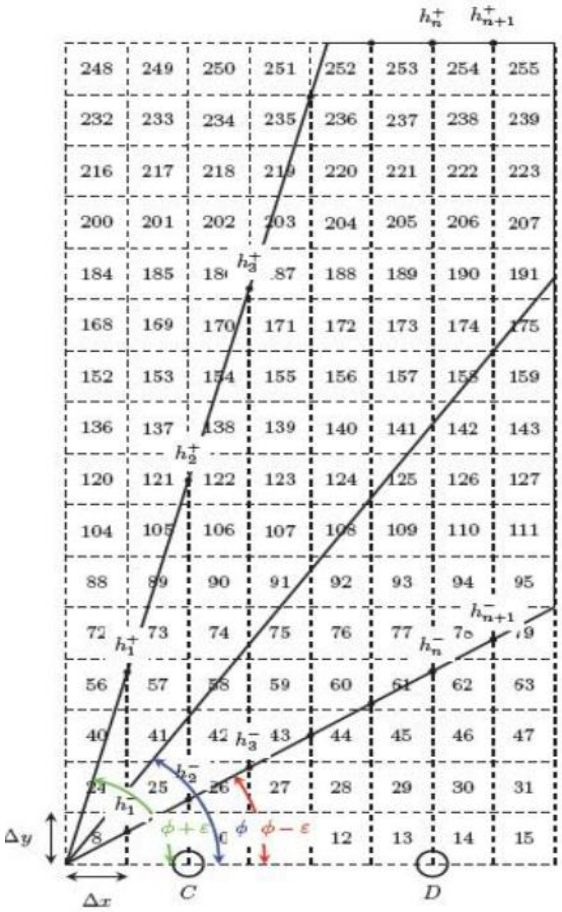


Fig .4. Numbered small squares in t he FOV

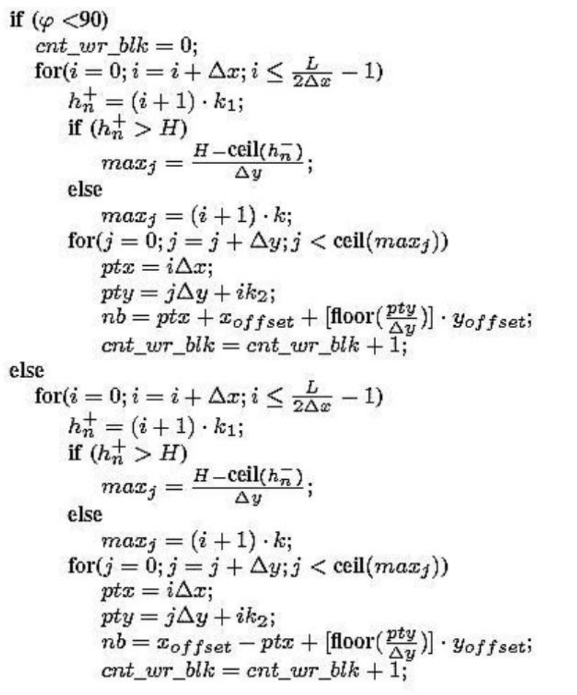
In Fig.4, the region between these two points h^-_n and h^+_n represents the addresses that need to be computed by Algorithm 1 by using equation (9) and (10). Every small square of the FOV has a number assigned to it. With that number additional information such as his weight and his shift can be retrieved. In our approach the GCC is computed in middle of the microphone array FOV. Besides the GCC computation, two correctives factors, *Xoffset* and *Yoffset* are required. Note that Algorithm 1 shows only the steps required when *Φ* < 90°. The Division by Δx and Δy in Algorithm 1 shows that the unitary resolution or a power of 2 will be a better choice to avoid costly division implementation.

4. Algorithms Computation Load

In this section we analyze the efficiency of GCC and DSB regarding hardware feasibility. We analyzed the MATLAB model of each algorithm and added all operations in order to compare in a purely sequential implementation. Since our final aim is a hardware implementation, a way to measure their computational load was developed. We decided to measure all operations in terms of multiply-accumulate (MA) units. MA operation is defined as in (11):

$$u = u + v * w \qquad (11)$$

The most complex operation for both algorithms is the FFT [12]. At the hardware level, this is usually implemented with the decimated in time *Cooley-Tukey* algorithm [13]. The reduction rate reported for an *N* samples *Discrete Fourier Transform* (DFT), is from $O(N^2)$ to $O(NLog_2N)$.

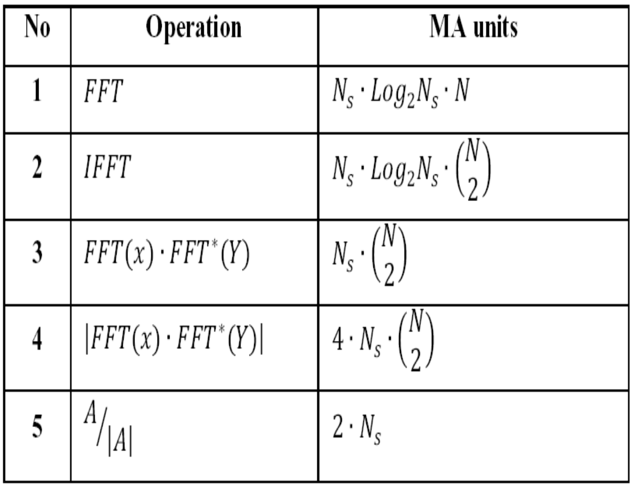


```
if ($\varphi$ <90)
  $cnt\_wr\_blk = 0$;
  for($i = 0; i = i + \Delta x; i \leq \frac{L}{2\Delta x} - 1$)
    $h_n^+ = (i+1) \cdot k_1$;
    if ($h_n^+ > H$)
      $max_j = \frac{H - \text{ceil}(h_n^-)}{\Delta y}$;
    else
      $max_j = (i+1) \cdot k$;
    for($j = 0; j = j + \Delta y; j < \text{ceil}(max_j)$)
      $ptx = i\Delta x$;
      $pty = j\Delta y + ik_2$;
      $nb = ptx + x_{offset} + [\text{floor}(\frac{pty}{\Delta y})] \cdot y_{offset}$;
      $cnt\_wr\_blk = cnt\_wr\_blk + 1$;
else
  for($i = 0; i = i + \Delta x; i \leq \frac{L}{2\Delta x} - 1$)
    $h_n^+ = (i+1) \cdot k_1$;
    if ($h_n^+ > H$)
      $max_j = \frac{H - \text{ceil}(h_n^-)}{\Delta y}$;
    else
      $max_j = (i+1) \cdot k$;
    for($j = 0; j = j + \Delta y; j < \text{ceil}(max_j)$)
      $ptx = i\Delta x$;
      $pty = j\Delta y + ik_2$;
      $nb = x_{offset} - ptx + [\text{floor}(\frac{pty}{\Delta y})] \cdot y_{offset}$;
      $cnt\_wr\_blk = cnt\_wr\_blk + 1$;
```

Algorithm1 . Algorithm to interface t he GCC with the DSB.

The GCC-PHAT algorithm load depends on the *number of microphones N* and *frame size Ns* (number of samples per data-frame) as shown in TABLE 1.

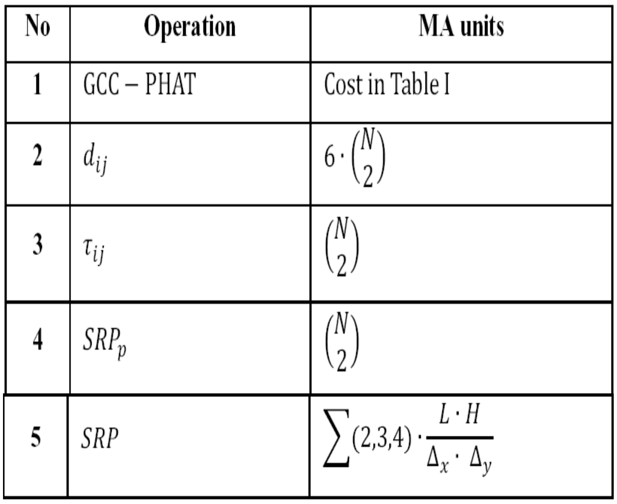
TABLE 1. Gcc-Phat load in multiply-Accumulate operations



No	Operation	MA units
1	FFT	$N_s \cdot Log_2N_s \cdot N$
2	$IFFT$	$N_s \cdot Log_2N_s \cdot \binom{N}{2}$
3	$FFT(x) \cdot FFT^*(Y)$	$N_s \cdot \binom{N}{2}$
4	$\lvert FFT(x) \cdot FFT^*(Y) \rvert$	$4 \cdot N_s \cdot \binom{N}{2}$
5	$A/_{\lvert A \rvert}$	$2 \cdot N_s$

The MATLAB analysis shows a big number of IFFTs compared to the FFTs. In Operation 4, the absolute value implies computing 2 products, an addition and a square-root; therefore, we assumed an equivalent cost of 4 MA units. Moreover, in operation number 5, the division of the vector *A* by its magnitude is performed twice, for real and imaginary parts; we then included a constant factor of two in the equation.
Table 2 shows the DSB-SRP algorithm analysis. Under the assumption that the cross-correlation between signals is provided by the GCC-PHAT, the remaining of the algorithm becomes, at least, as complex as the previous one in terms of number of MA units (Operation 1 on TABLE 2).

TABLE 2. DSB-SRP load in multiply-Accumulate operations



No	Operation	MA units
1	GCC – PHAT	Cost in Table I
2	d_{ij}	$6 \cdot \binom{N}{2}$
3	τ_{ij}	$\binom{N}{2}$
4	SRP_p	$\binom{N}{2}$
5	SRP	$\sum(2{,}3{,}4) \cdot \frac{L \cdot H}{\Delta_x \cdot \Delta_y}$

According to Equation (12) the distance computation (Operation 1 on TABLE II) requires 3 additions, a square-root and 2 square powers:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \qquad (12)$$

We arbitrarily established the cost of each single operation to be equivalent to a single MA, therefore 6 operations for each pair of microphones. According to Equation (7), the *theoretical delay* τ_{ij} (Operation 3 on TABLE II) only implies a constant multiplication performed once per pair of microphones, i.e. the distance d_{ij} times the constant: f_s / v_s Based on Equation (6), each individual SRP (Operation 4 on TABLE II) requires *N* combined 2 additions and the total SRP implies to multiply costs 3, 4 and 5, by the number of NOSS defined by Equation (8).
The main difference between the pure DSB compared to the proposed mixed GCC-DSB algorithm is in the number of points it needs to evaluate (NOSS), and that depends on the user defined parameter that limits the DSB search region. The NOSS is plotted in Fig.5 for three different values of . For a particular , NOSS is maximum around =70°. As expected, a higher is a greater number of points to evaluate.

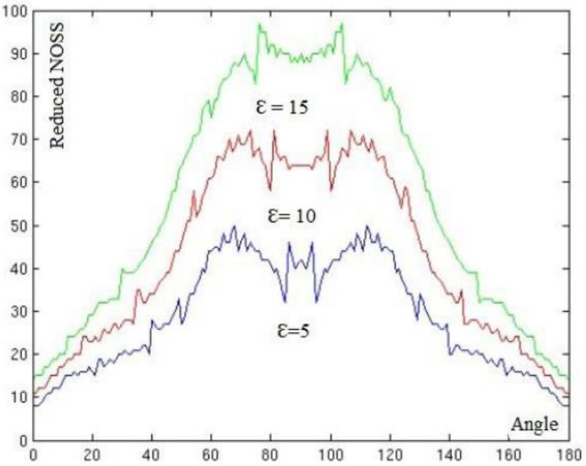


Fig. 5. Number of small squares NOSS for different values of Φ and ε

More optimizations can be performed to both, the pure DSB-SRP algorithm and the mixed one; if the distances and time delays between each pair of microphones are pre-computed, Operations 2 and 3 in Table II, can be reduced to single memory accesses.

5. Computational Load Comparison

For simulation purposes, all algorithms were described in MATLAB. However, it was not possible to use the MATLAB's profiler to evaluate the computation load since most complex functions such as FFT, matrix multiplication and trigonometric computation are optimized, thus only valid for simulation. For that reason, the algorithms were modelled in terms of the amount of *Multiply-Accumulate* (MA) operations required to execute.

The MA load for the DSB algorithm compared to the mixed GCC-DSB for the same *FOV=9m*^{2}, according to the number of microphones *(N)* and resolution (Δx), is shown in Fig.6. In that figure we can see we can increase *N*, the resolution (by reducing Δx), or both, and the computation cost of GCC-DSB will be still

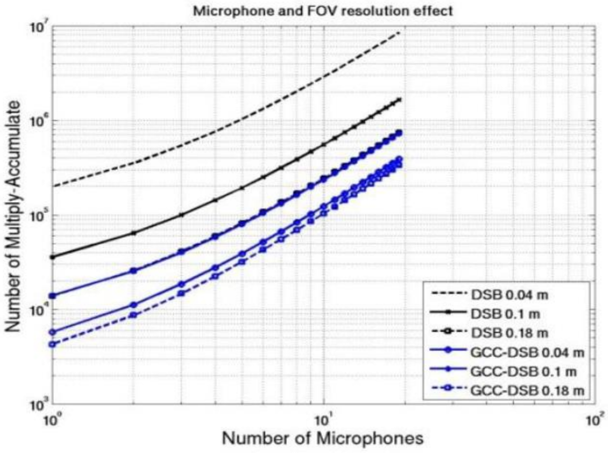


Fig. 6. DSB vs. GCC-DSB multiply-accumulate (MA) load lower than for the DSB.

Fig.7 shows, for *N=16* microphones, the amount of MA operations vs. the *Frame Size* (*Ns*) for the DSB compared to the GCC-DSB with different resolution values (Δx). When *Ns* is small, there is a difference in MA operations of around 1 order of magnitude between the DSB and the GCC-DSB. However, as *Ns* increases, this difference reduces significantly. Therefore, even when the GCC-DSB is still faster, its performance, compared to the DSB, will be barely perceptible for bigger frames.

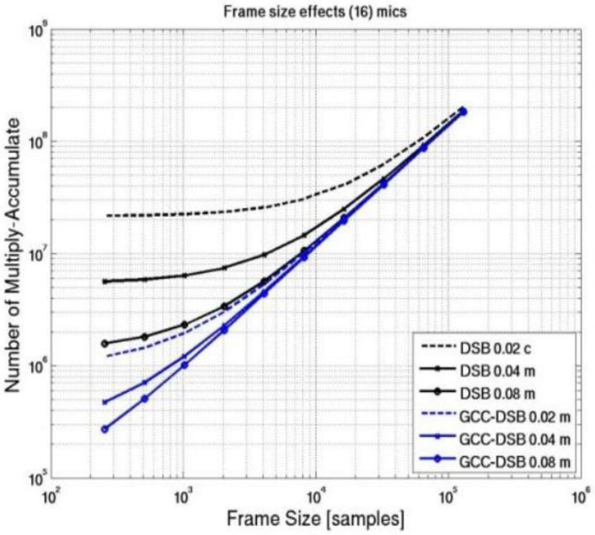


Fig. 7. DSB vs GCC -DSB for different Frame Sizes Ns

Finally, Fig.8 shows a comparison of the MA load between the DSB and the GCC-DSB for 3 difference values of ε. The audio source is located 90° from the centre of a *N=16* microphones array. The figure shows an increase of around one order of magnitude in terms of MA units when increases from 5 to 50 degrees. As expected, the processing speed increases with lower resolutions (bigger small-squares) but that increase is less important for ∘ values between 5 to 10 degrees.

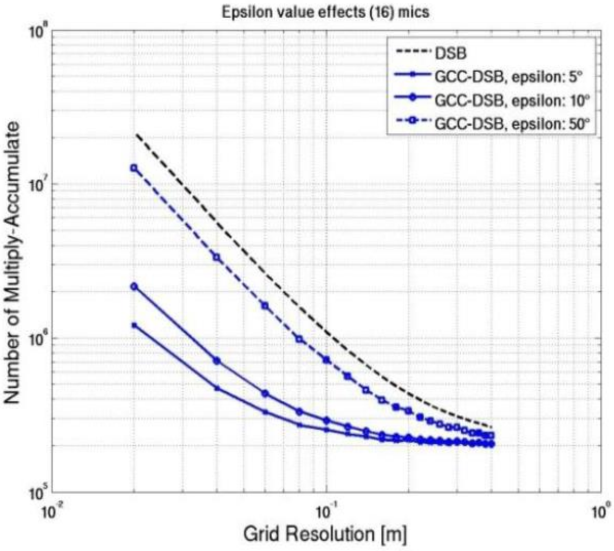


Fig. 8. Multiply-Accumulate load for different values of epsilon

6. Accuracy Considerations

Although, according to our previous analysis, there are improvements at the computation level, it is necessary to establish how accurate, this new approach, compared to the base one, really is. We present initial measurements of some quantities that can be used to start studying the subject.

After some analysis, we established three main quantities that can affect accuracy. The first one is discrimination, defined as the number of points whose SRP value is at least 90 % the maximum detected value, divided by the total number of points. Secondly, we computed the mean distance between each of the detected points and the real maximum; finally, in order to easily establish spatial dispersion statistics, we measured the standard deviation of the *x* and *y* coordinates between all detected points and the real one.

Since the acoustic energy plot created by the mixed algorithm depends on the angle yielded by the GCC-PHAT one, we performed some tests, inducing some errors in this variable. The distance between microphones was changed from 20 cm up to 50 cm and the value of epsilon was fixed to 10°.

We ran several simulations for different angles using the Kentucky Toolbox [14] for audio sources located at different angles, with 2 coherent noise sources. Results regarding discrimination can be found in Figure 9. On the *x* axis, we plotted the ratio between the GCC-PHAT angle error and the value of epsilon, thus, since epsilon equals 10°, the whole range comprises an error from -30° up to 30°. From Fig.9 it can be seen that when the angle error is less than the value of epsilon i.e. the range between -1 and 1, the discrimination is highest, meaning that more points are recognized as having the biggest energy response. However, as the separation between microphones increases, the line tends to be flatter because the detection capacity improves, so fewer maximums are identified. When the spacing is 50 cm, the line is almost flat and even sometimes, fewer points are recognized in the interval [-1, 1]; nonetheless, even though the amount of points influences the accuracy, it does not hold that, the less points, the more accurate. The horizontal line shown is the output from the DSB-SRP algorithm.

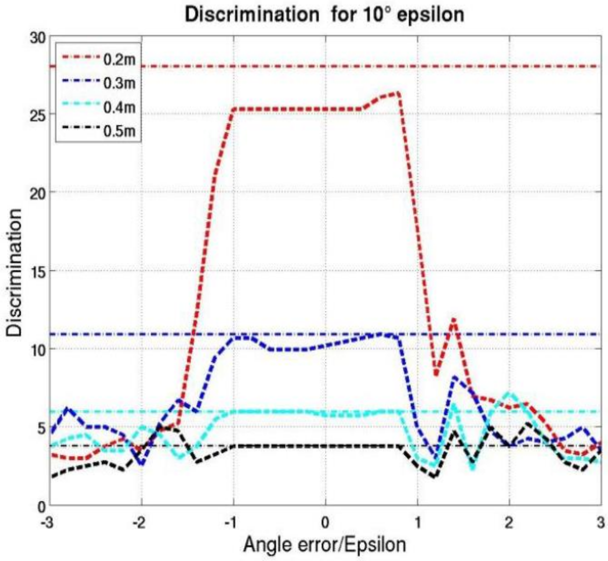


Fig. 9. GCC-DSB discrimination for microphone spacings between 20 and 50 cm.

The second variable measured is the mean distance between all maxima and the real source location. Fig.10 shows that, in effect, when the spacing between microphones increases, the selected point is closer to the actual source; changing the spacing from 30 to 50 cm does not represent a big increase in accuracy. This graph was also created by taking the average for different source locations to be more general. Again, the horizontal line is the output from the DSB-SRP algorithm, which is also the best result for the mixed one.

Measurements for the standard deviation of the *x* and *y* coordinate of all identified maxima, yielded rather unstable behavior when the angle error was outside the range of epsilon. As example, Fig.11 clearly shows that, when this is the case, no assumptions can be made about the accuracy based on this quantity, even when Fig.11 shows an almost stable trend throughout the graph. All previous graphs can be helpful to analyze the accuracy of the proposed algorithm; it is necessary to find a way to integrate them all and establish a robust criteria to define accuracy.

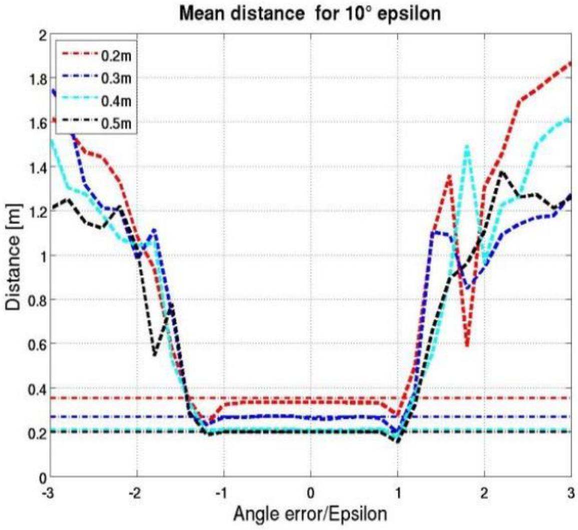


Fig.10. GCC-DSB mean distance of identified points and real target source for different microphone spacing (20 - 50 cm).

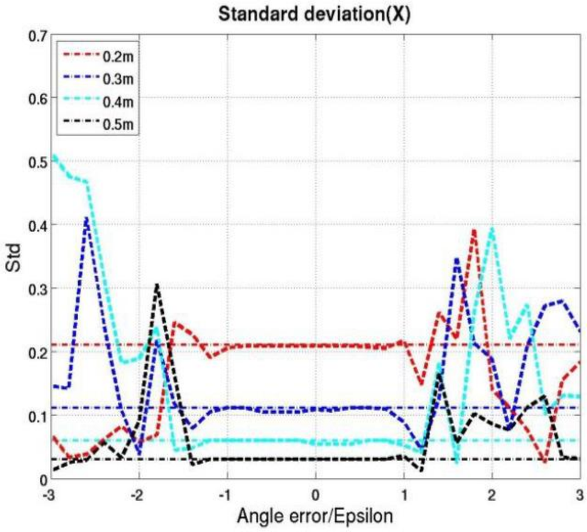


Fig.11. GCC-DSB standard deviation of the x-coordinate of all detected maximums. Microphone spacing measured in meters.

7. Conclusions and Future Work

This work demonstrates that our mixed GCC-DSB algorithm can improve the speed of audio source tracking compared to the DSB. Moreover, for an identical accuracy, it becomes a good option for real time applications. The results also showed several improvements with variables such as the number of microphones, samples or grid resolution.

Since the DSB computation reuses the GCC-PHAT output values, the area cost of a hardware implementation is expected to be small. However, specialized operators to compute tangents and arccosines will be required to compute the reduced FOV region. Finally, as the optimal hardware implementation implies a compromise between area, parallelism, and power consumption, different implementation alternatives must be explored and evaluated. Since not all platforms execute all functions in a sequential way, specific analysis, related to the target architecture, can help to compare the proposed algorithm's execution speed.

As previously stated, a way to combine all accuracy and precision variables is required to fully establish the outcome from the mixed algorithm. More studies regarding the influence of the epsilon value, signal distance, threshold value used for locating a maximum, and GCC-PHAT accuracy, are also required for better analysis.

8. References

[1] Q. Li, M. Zhu et W. Li, "A portable USB-Based Microphone Array Device For Robust Speech Recognition," chez IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP'09, Taipei, 2009.

[2] J. M. Valin, F. Michaud et J. Rouat, "Robust Detection and Tracking of Simultaneous Moving Audio Sources Using Beamforming and Particle Filtering," Robotics and Autonomous Systems, vol. 55, n° %13, pp. 216-228, 31 March 2007.

[3] E. Zwyssig, M.Lincoln et S. Renals, "A Digital Microphone Array for Distant Speech Recognition," chez IEEE INternational Conference on Acoustics Speech and Signal Processing ICASSP'10, March 2010.

[4] I. Tashev, Audio Capture and Processing: Practical Approaches, Wiley, 2009.

[5] J. Benesty, J. Chen, Y.Huang et J.Dmochowski, "On Microphone Array Beamforming From a MIMO Acoustic Signal Processing Perspective," IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, n° %13, pp. 1053-1065 , March 2007.

[6] I. McCowan, "Robust Speech Recognition using Microphone Arrays," Australia, 2001.

[7] P. Sinha, A. D. George et K. Kim, "Parallel algorithms for robust broadband MVDR beamforming," Journal of Computational Acoustics, vol. 10 , n° %11, pp. 69-96, 2002.

[8] W. C. Cave et R.-E. Wasser, "Estimating Parallel Processing Speed Multipliers," 2007.

[9] E. S. Chung, P.-A. Milder, J. C. Hoe et K. Mai, "Single-Chip Heterogeneous Computing: Does the Future Include Custom Logic, FPGAs, and GPGPUs," chez Proceedings of 43rd Annual IEEE/ACM International Symposium on Microarchictecture MICRO'43, Atlanta, GA, December 4-8 2010.

[10] S. Thatte et J. Blaine, "Xcell Journal Online," Fall 2002.

[11] H. Belhadj, V. Aggrawal, A. Pradhan et A.Zerrouki, "Power-Aware FPGA Design - Actel," Feb. 2009, Available: http://www.actel.com/documents/Power_Aware_WP.pdf.

[12] F. Qureshi, S. A. Alam et O. Gustafsson, "4K-Point FFT Algorithms based on optimized twiddle factor multiplication for FPGAs," chez The Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics PrimeAsia'10, Sanghai, Sept. 22-28 2010.

[13] J. W. Cooley et J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Journal of Mathematics of Computation MCOM, vol. 19, pp. 297-301, 1965.

[14] U. of Kentucky. Performance analysis of a scrp image based audio source detection algorithms, 2010.