4

Localization and Tracking of Acoustical Sources

Gerhard Doblinger

The Institute of Communications and Radio-Frequency Engineering, Vienna University of Technology

Speaker localization and automatic tracking in a reverberant environment are challenging and often needed tasks in many audio-based applications including hands-free mobile phones, speech recognition, and teleconferencing. In this chapter, we present signal processing algorithms for reliable location estimation of audio sources. We discuss high-quality techniques based on time-delay estimation using only two microphones. These algorithms can be used to estimate directions of sound waves travelling to a one-dimensional microphone array. We focus on this basic situation because it frequently occurs in practice. Furthermore, a precise and robust algorithm for time-delay estimation is fundamental to multi-dimensional source localization tasks as well. We present an automatically steered microphone array for speaker tracking using an adaptive beamformer in connection with a direction estimation subsystem. This array is very well suited to adjust the main lobe of the beam pattern to the direction of a moving speaker while suppressing sounds from other directions. In addition, the system is capable to track speaker movements or to switch among speakers in rooms with modest reverberation. The automatically steered microphone array uses a computationally efficient multi-input FFT filterbank. MATLAB® programs are available to facilitate algorithm implementation and testing by interested readers.

4.1 Introduction

Acoustical source localization is a well developed feature of the human auditory system. Using only two sensors, this biological system has a remarkable precision in resolving the position of speakers and other acoustical sources. The human ears in conjunction with the brain can accurately localize and track sources in a sound field around the head except two small ambiguity regions (cones of confusion) [1]. In addition, noise and reverberation do not greatly influence the precision of source localization. Achieving such a performance using two microphones and digital signal processing is a rather chal-

lenging task. In this chapter, our primary goal is the presentation of robust acoustical source localization algorithms which can be used to steer adaptive microphone arrays. Multiple microphones in array configurations offer many advantages over systems with a single microphone. Due to miniature piezoelectric sensors and powerful digital signal processors, microphone arrays can now be built in a compact and inconspicuous design. This leads to a number of applications of automatically steered microphone arrays like voice communications in cars, hands-free mobile phones, speech recognition, and teleconferencing. With these applications in mind, we focus on one-dimensional source localization since knowledge of the angle of arrival (azimuth in the xy-plane of a Cartesian coordinate system) is sufficient to adjust one-dimensional microphone arrays. To determine the position of a speaker in a room, we can use a multi-dimensional array or separate one-dimensional arrays.

The two-microphone technique of delay estimation is fundamental to all multi-dimensional source localization algorithms because different delay measurements can be combined by refined procedures to estimate a speaker's position and movement. However, extensions to multiple microphones and localization of multiple sources will not be treated in this chapter. Further readings on multi-microphone techniques for multi-source localization can be found in recent books [2–4].

The basic setup using two microphones is sketched in Fig. 4.1. If we assume far-field conditions (plane wave propagation), the estimation of azimuth Φ can easily be carried out by measuring the Time Delay Difference (TDD) between the two microphone signals.

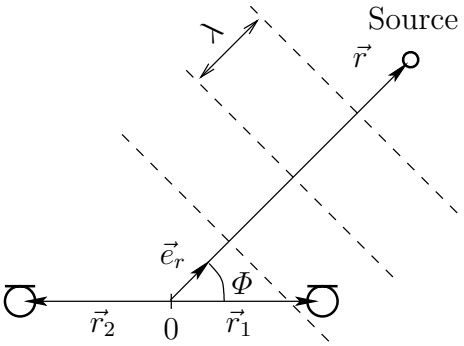

Fig. 4.1. Basic two-microphone layout for source localization (azimuth Φ of arrival direction, single frequency plane wave with wavelength λ).

Denoting microphone distance $d = \|\vec{r}_2 - \vec{r}_1\|$, sound velocity v_s, and TDD Δt, we get

$$\Phi = \arccos \frac{v_\mathrm{s} \Delta t}{d} \,. \tag{4.1}$$

Due to the nonlinear relationship, accuracy is poor for Φ near 0° and 180°. In addition, discrete-time processing of the microphone signals results in quantized TDD estimates. If we estimate azimuth Φ from TDDs with accuracy $\pm\frac{T}{2}$ (sampling interval $T = 1/f_\mathrm{s}$), we can expect an error behavior as shown in Fig. 4.2. Curves plotted in Fig. 4.2 obey the relationship

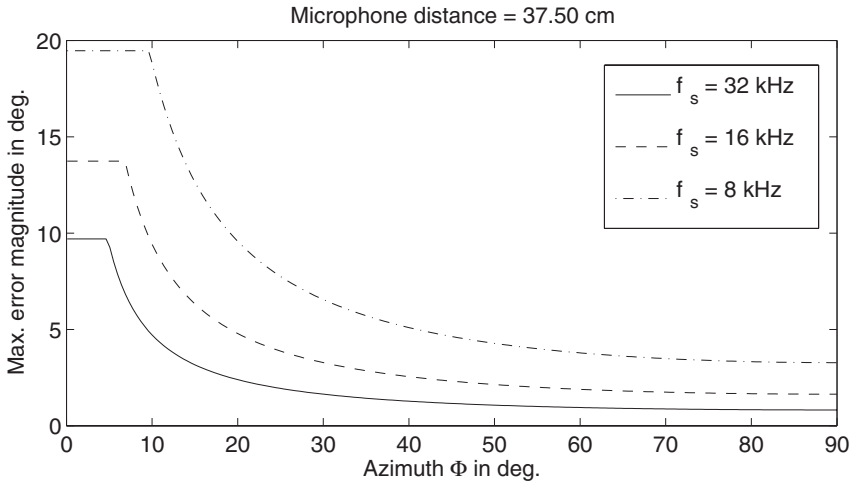

Fig. 4.2. Maximum azimuth error magnitude as a function of azimuth Φ and sampling frequency f_s.

$$\delta\Phi_{\max}(\Phi) \approx \min\left(\delta\Phi_0, \frac{\beta}{|\sin\Phi|}\right), \tag{4.2}$$

with $\beta = \frac{v_\mathrm{s}}{2df_\mathrm{s}} < 1$, and $\delta\Phi_0 = \arccos(1-\beta)$. As a consequence, we must use oversampling or a two-dimensional array (e.g. a quadratic array layout with 4 microphones) to reduce errors at $\Phi \approx 0°$ and $\Phi \approx 180°$. Later in this chapter, we will present an algorithm which exhibits an improved performance. It should be noted that Fig. 4.2 only shows the influence of delay quantization. In addition, errors resulting from TDD estimation must also be taken into account.

According to (4.2), the azimuth error at a given sampling frequency f_s can be reduced by increasing microphone distance d. For practical reasons, however, array size is limited in most situations like car cockpits. Additional problems affecting the performance of source localization algorithms are introduced by the specific nature of speech signals exhibiting speech pauses and segments with different spectral contents, and by noise and reverberation.

In the next sections, we will discuss algorithms which are rather robust in regard to these obstacles. We begin with a classical method using the Generalized Cross-Correlation (GCC) function [5]. The GCC method can efficiently be

implemented using the Fast Fourier Transform (FFT). Motivated by binaural signal processing, an algorithm based on Interaural Time Differences (ITD) is presented next. This method offers an azimuth estimation with high accuracy but requires more computational load [6]. Afterwards, two source localization algorithms involving adaptive filters are described. One technique uses an adaptive eigenvalue decomposition to estimate TDDs [7]. This promising technique employes a normalized Least Mean-Square (LMS) adaptive algorithm suitable for implementation using the FFT. We conclude with a presentation of an adaptive microphone array comprised of an FFT filterbank beamformer and a source localization subsystem to automatically steer the beam pattern towards a moving speaker.

In order to facilitate implementation, algorithm variables and equations are formulated in a discrete-time framework. We do not use continuous-time variables, as sometimes found in the literature on TDD estimation. In addition, MATLAB® programs and test data for all algorithms presented in this chapter are available at `www.nt.tuwien.ac.at/dspgroup/gdobling.html`. Testing and comparison of the algorithms can thus be carried out with minimal effort.

4.2 Source Localization Using the Generalized Cross-Correlation Function

If we assume an ideal wave propagation model and an array with two microphones (see Fig. 4.1), then the analog (continuous-time) sensor signals are given by

$$x_{\mathrm{a}1}(t) = s_{\mathrm{a}}(t) + v_{\mathrm{a}1}(t) \tag{4.3}$$

$$x_{\mathrm{a}2}(t) = s_{\mathrm{a}}(t - \tau_0) + v_{\mathrm{a}2}(t), \tag{4.4}$$

with source signal $s_{\mathrm{a}}(t)$ and noise disturbances $v_{\mathrm{a}1,2}(t)$. In (4.3), (4.4), we have neglected any signal attenuation and spreading (caused by room acoustics). The discrete-time representations of the bandlimited sensor signals are

$$x_1(n) = s(n) + v_1(n) \tag{4.5}$$

$$x_2(n) = \underbrace{s_{\mathrm{a}}(nT - \tau_0)}_{s_{\tau_0}(n)} + v_2(n), \tag{4.6}$$

with sampling interval T. In general, signal delay τ_0 is not an integer multiple of T. Therefore, $s_{\tau_0}(n)$ is not simply a delayed version of $s(n)$. Only if $\tau_0 = n_0 T$, then $s_{\tau_0}(n) = s(n - n_0)$. However, using the reconstruction property of a bandlimited analog signal

$$s_{\mathrm{a}}(t) = \sum_{k=-\infty}^{\infty} s(k) \, \frac{\sin \frac{\pi}{T}(t - kT)}{\frac{\pi}{T}(t - kT)}\,, \tag{4.7}$$

we obtain

$$s_{\tau_0}(n) = s_{\mathrm{a}}(nT - \tau_0) = \sum_{k=-\infty}^{\infty} s(k) \underbrace{h_{\mathrm{a}}\big((n-k)T - \tau_0\big)}_{h_{\tau_0}(n-k)}, \tag{4.8}$$

with $h_{\tau_0}(n) = \frac{\sin \pi(n-\tau_0/T)}{\pi(n-\tau_0/T)}$. Thus, the discrete-time representation of the delayed microphone signal is an interpolated version of the non-delayed signal. An ideal lowpass interpolation function with parameter τ_0/T is used. If we determine signal delays in time domain, we have to use a sufficiently high sampling frequency or some kind of signal interpolation. As an alternative, signal delays can be obtained in the frequency domain from the phase spectrum. Application of the Fourier Transform to (4.5), (4.6) results in

$$X_1\left(e^{j\Omega}\right) = S\left(e^{j\Omega}\right) + V_1\left(e^{j\Omega}\right) \tag{4.9}$$

$$X_2\left(e^{j\Omega}\right) = S\left(e^{j\Omega}\right) e^{-j\Omega\frac{\tau_0}{T}} + V_2\left(e^{j\Omega}\right). \tag{4.10}$$

Assuming zero-mean uncorrelated noise disturbances, the cross-power spectrum is

$$S_{x_1x_2}(\Omega) = E\big\{X_1\left(e^{j\Omega}\right) X_2^*\left(e^{j\Omega}\right)\big\} = S_{ss}(\Omega)\, e^{j\Omega\frac{\tau_0}{T}}, \tag{4.11}$$

where $E\{\cdot\}$ means expectation and $*$ denotes complex conjugate operation. A computation of signal delays τ_0 from (4.11) requires a robust phase unwrapping algorithm and a least-squares procedure involving phase measurements at a set of different frequencies. In the context of microphone arrays, robust phase unwrapping has been proposed in [8, 9]. However, these methods pose less robustness regarding room reverberation.

An alternative to phase unwrapping is delay estimation from the generalized cross-correlation (GCC) $R_{x_1x_2}(n)$:

$$\frac{\tau_0}{T} \approx n_0 = \arg\max_n R_{x_1x_2}(n), \tag{4.12}$$

with

$$R_{x_1x_2}(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \psi_{12}\left(e^{j\Omega}\right) S_{x_1x_2}(\Omega)\, e^{j\Omega n} d\Omega. \tag{4.13}$$

Non-integer delays τ_0/T can only be approximately obtained from (4.12). To increase accuracy of delay estimation, an interpolation must be applied to $R_{x_1x_2}(n)$ prior to maximum detection. If we omit the weighting function $\psi_{12}\left(e^{j\Omega}\right)$ in (4.13), we obtain the classical cross-correlation between the sensor signals as the inverse Fourier Transform of the cross-power spectrum.

The benefits of using a weighting function $\psi_{12}\left(e^{j\Omega}\right) \not\equiv 1$ are discussed in detail in [5]. The main idea is to create a sharp dominant peak and to reduce spurious peaks in $R_{x_1x_2}(n)$ caused by room reverberation and colored source signal spectra. A single sharp peak in the GCC function requires a flat

cross-power spectrum magnitude. As a result, the weighting function must act as a pre-whitening filter. This leads to the SCOT (Smoothed Coherence Transform) algorithm with a weighting function

$$\psi_{12}\left(e^{j\Omega}\right) = \psi_S\left(e^{j\Omega}\right) = \frac{1}{\sqrt{S_{x_1x_1}(\Omega)\, S_{x_2x_2}(\Omega)}}\,. \tag{4.14}$$

Alternatively, we obtain the PHAT (Phase Transform) algorithm with the weighting function

$$\psi_{12}\left(e^{j\Omega}\right) = \psi_P\left(e^{j\Omega}\right) = \frac{1}{\left|S_{x_1x_2}(\Omega)\right|}\,. \tag{4.15}$$

Under ideal conditions as given in (4.11), the PHAT weighting function delivers an ideal GCC

$$R_{x_1x_2}(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\Omega\frac{\tau_0}{T}}\, e^{j\Omega n} d\Omega = \frac{\sin\pi(n+\frac{\tau_0}{T})}{\pi(n+\frac{\tau_0}{T})}\,. \tag{4.16}$$

The PHAT weighting function has the computational advantage that only the cross-power spectrum is needed. Both the SCOT-GCC and the PHAT-GCC algorithm perform very well in practical situations with modest room reverberation, like medium-size office rooms, and car cabins. Furthermore, these GCC algorithms are robust against environmental noise and the specific nature of speech spectra. As shown by a comprehensive statistical analysis in [10], the PHAT-GCC is optimal among the class of GCC functions when used in reverberant environments. The GCC principle can be extended to more than one microphone pair, yielding better precision of source position estimates, especially in larger rooms [11].

Speech signals require an estimation of power spectra on a short-time basis. Therefore, the expectation operator in (4.11) will be replaced by a suitable time-average. Power spectra can be estimated from windowed signal frames of N samples (e.g. $N = 512$ at $f_\mathrm{s} = 16$ kHz). Frames may overlap by some extend (typically $N/2$ to $3N/4$ samples). We use an exponential weighting of past frames resulting in the following cross-power spectrum estimate:

$$\widehat{S}_{x_1x_2}(m,k) = \alpha\widehat{S}_{x_1x_2}(m-1,k) + (1-\alpha)X_1(m,k)X_2^*(m,k), \tag{4.17}$$

with $\alpha = 0.7\ldots0.8$ to accommodate for the short-time stationarity of speech signals (m is the frame index, k the index of the discrete frequency axis, respectively). The Discrete Fourier Transforms (DFTs) of the windowed microphone signal frames are

$$X_i(m,k) = \sum_{n=0}^{N-1} x_i(mM+n)w(n)e^{-j\frac{2\pi}{N}nk}, \quad i = 1,2 \tag{4.18}$$

(frame index $m = 0, 1, 2, \ldots$, frequency index $k = 0, 1, \ldots, N-1$). The frame hop size M determines frame overlapping (no overlapping if $M \geq N$). A bell-shaped function $w(n)$ like Hann or Hamming windows may be used for time-windowing.

By means of the inverse DFT (IDFT), the cross-power spectrum estimate (4.17) can now be used to estimate the PHAT-GCC of the m^{th} signal frame:

$$\widehat{R}_{x_1x_2}(m,n) = \frac{1}{N} \sum_{k=0}^{N-1} \frac{\widehat{S}_{x_1x_2}(m,k)}{\left|\widehat{S}_{x_1x_2}(m,k)\right|} e^{j\frac{2\pi}{N}nk}, \quad n = 0, 1, \ldots, N-1. \tag{4.19}$$

Finding the maximum location of $\widehat{R}_{x_1x_2}(m,n)$ in order to determine the TDD must be done with care. First, TDDs may be positive or negative depending on the azimuth of the sound wave (see Fig. 4.1). Therefore, indices $N-n$ must be used instead of $-n$ according to the periodicy of the DFT. Secondly, we do not need to carry out maximum search over the whole interval $n \in [0, N-1]$ because the maximum delay $\tau_{0\max}$ is limited by the microphone distance d ($\tau_{0\max} = d/v_{\text{s}}$). Third, and most important: In order to resolve fractional signal delays, we must use an interpolation of $\widehat{R}_{x_1x_2}(m,n)$ before finding the maximum location. This can conveniently be done in the frequency domain by increasing the length (e.g. $N' = 4N$) of the IDFT in combination with proper zero-padding. Alternatively, GCC interpolation can efficiently be carried out in the time domain since the relevant GCC length is rather short.

Fig. 4.3 and Fig. 4.4 show a typical example of a PHAT-GCC azimuth estimation using a 50 seconds speech record of a moving speaker in a room with modest reverberation and noise. The initial speaker position is at azimuth 90°. After 16 seconds, the speaker moves towards 0°, and finally to 180°. Azimuth estimates are held constant during speech pauses detected by comparing the maximum of $\widehat{R}_{x_1x_2}(m,n)$ with a threshold value. This speech activity detection is very robust at virtually no additional cost. Frame size is set to $N = 512$ samples with a frame hop size $M = 128$. FFT length is increased by a factor of 4 when calculating $\widehat{R}_{x_1x_2}(m,n)$ in (4.19). In (4.18), however, an $N = 512$ point FFT is applied to compute the DFTs of the two microphone signals.

4.3 Source Localization Based on Interaural Time Differences

As briefly discussed in the introduction, human beings have an astonishing precise sound localization ability based on interaural differences in time delay and intensity between sound pressure signals at the two ears. Processing of these interaural differences is carried out to a great extend in the human brain. Several binaural models exist to describe numerous experimental data (see [12] for a detailed review). One of these models is the basis of the source localization algorithm presented in this section [6]. Basically, we create a set

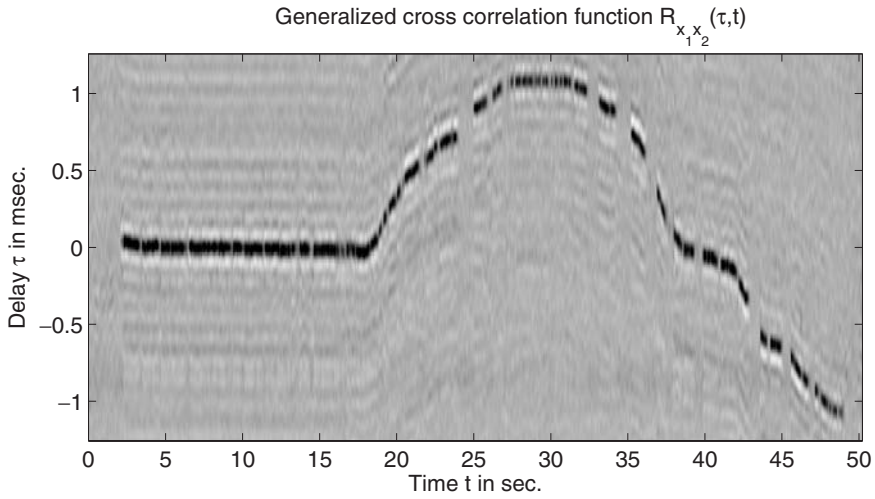

Fig. 4.3. PHAT-GCC map of a speaker movement in a medium-size office environment (Speech pauses are clearly visible as discontinuities).

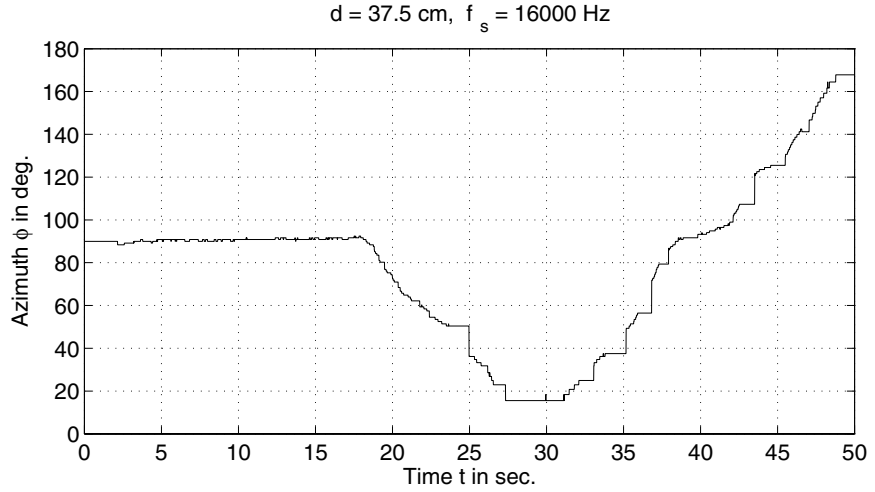

Fig. 4.4. Azimuth estimation using maximum search on the PHAT-GCC of Fig. 4.3 (Estimates are held constant during speech pauses).

of all relevant delays between the two microphone signals needed to estimate azimuth Φ to a given resolution. This set is searched for the optimum delay value resulting in the best coincidence of the two microphone signals. However, the matching procedure is implemented in the frequency domain to obtain fractional delays in an easy way.

The whole azimuth range $\Phi \in [0, \pi]$ is subdivided into an odd number I of equally spaced sectors. Using the array geometry of Fig. 4.1, each sector corresponds to a TDD[1]

$$\tau_i = \frac{d}{2v_\mathrm{s}} \sin\left(\frac{i-1}{I-1}\pi - \frac{\pi}{2}\right), \quad i = 1, 2, \ldots, I, \tag{4.20}$$

with microphone distance d and sound velocity v_s. As an example, we need a set of $I = 73$ values τ_i to obtain an azimuth resolution of 2.5°. If we use an N-point DFT to represent the microphone signals in the frequency domain, this set of delays corresponds to phase factors

$$p_k(i) = e^{-j\frac{2\pi}{N}kf_\mathrm{s}\tau_i}, \quad k = 0, 1, \ldots, \frac{N}{2}, \quad i = 1, 2, \ldots, I, \tag{4.21}$$

with sampling frequency f_s and τ_i from (4.20). The N-point DFTs $X_{1,2}(m, k)$ of the microphone signals are computed on a frame by frame basis as in (4.18). To find the optimum delay for each frequency index k, we can use the system shown in Fig. 4.5. The DFTs $X_{1,2}(m, k)$ are multiplied by phase factors

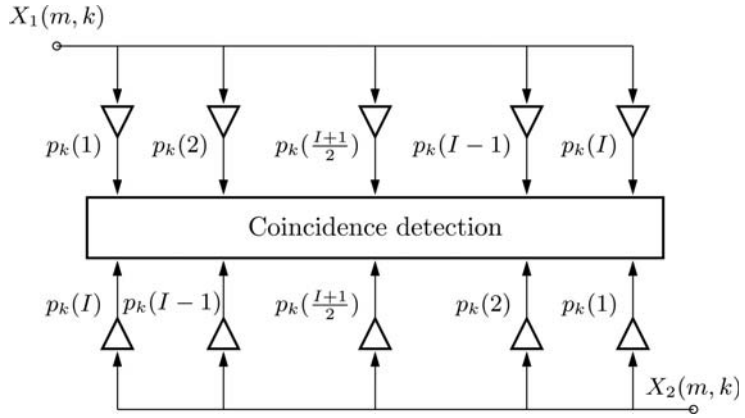

Fig. 4.5. Delay (phase) matching in frequency domain for each frequency index k (frame index m).

from (4.21) and compared in the coincidence detection box. Comparison is performed on each vertically aligned pair only, since the delays of the two microphone signals are coupled due to the array geometry and phase factors are properly arranged in Fig. 4.5. The coincidence detection is carried out according to the simple matching rule

[1] Delays τ_i are measured here with respect to the origin in Fig. 4.1.

$$i_{\text{opt}}(m,k) = \arg\min_i \Delta_i(m,k), \quad k = 0,1,\ldots,\frac{N}{2} \tag{4.22}$$

$$\Delta_i(m,k) = \Big| p_k(i)X_1(m,k) - p_k(I-i+1)X_2(m,k) \Big|^2, \quad i = 1,2,\ldots,I \tag{4.23}$$

(frame index $m = 0,1,2,\ldots$). With optimum delay indices from (4.22), optimum delays τ_i can be found for each frequency point k and frame m according to (4.20). To obtain the TDD, and thus the azimuth of the sound source from this set of data, we first build a histogram map $P_k(\tau_i, m)$ by counting τ_i values for each frequency point in several consecutive signal frames. τ_i values will gather around the actual delay corresponding to the azimuth of the signal source. In a similar manner as in [6], we use the following histogram averaging procedure in case of speech signals:

$$\begin{aligned} P_k(\tau_i,m) = {} & \alpha P_k(\tau_i, m-1) + \delta\big(i - i_{\text{opt}}(m,k)\big), \\ & i = 1,2,\ldots,I \\ & k = 0,1,\ldots,\frac{N}{2} \\ & m = 0,1,2,\ldots, \end{aligned} \tag{4.24}$$

where $\delta(\cdot)$ is the unit impulse and τ_i is the set of delays in (4.20). Forgetting factor α is chosen between 0.85 and 0.95.

An illustrative example of a histogram map is shown in Fig. 4.6 wherein delay values τ_i are replaced by corresponding azimuth values. A stationary broadband noise source emitting from azimuth direction 60° is used. In the frequency range below 2 kHz, a prominent population of azimuth values along a vertical line is observed. An additional curved pattern stems from phase ambiguity. Spatial aliasing occurs for signals with frequency contents above $f_{\max} = \frac{v_s}{2d}$ due to $\frac{\lambda}{2} < d$. With a microphone distance $d = 37.5$ cm, we get $f_{\max} \approx 450$ Hz.

To reduce the influence of phase ambiguity, we sum up histogram data over all frequency indices k for each azimuth (or τ_i, respectively). The optimum delay is then obtained by searching for the maximal sum. As a result, the azimuth of the source location is given by

$$\tau_{\text{opt}}(m) = \arg\max_{\tau_i} \sum_{k=0}^{\frac{N}{2}} P_k(\tau_i, m), \tag{4.25}$$

for each signal frame m. Despite the presence of phase ambiguity, the maximum in (4.25) is rather sharp. Further improvements, especially in case of multiple sources, are discussed in [6]. However, a high computationally effort is needed which is not justified in case of a single speaker or even for multiple speakers not talking at the same time. Our investigations show that no significant improvements by the refinements proposed in [6] are obtained in real acoustic environments.

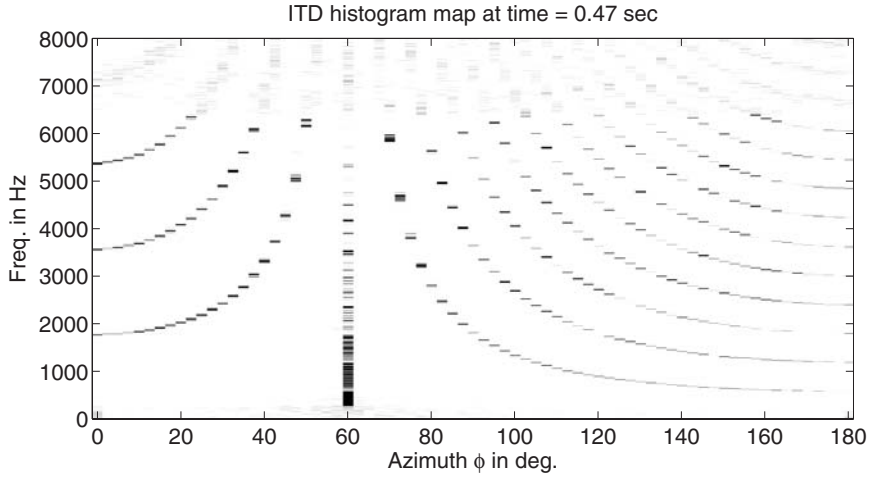

Fig. 4.6. Histogram map of a stationary white noise source, bandlimited between 300 Hz and 6400 Hz, emitting from azimuth direction $\Phi = 60°$, (FFT length $N = 512$, $\alpha = 0.9$, azimuth resolution 2.5°).

Using the same source signal as in Fig. 4.3, a representative example of a histogram map summed up over frequency is shown in Fig. 4.7. The result of azimuth estimation by searching for maxima locations in the ITD histogram map of Fig. 4.7 is presented in Fig. 4.8. Performance differences between the PHAT-GCC and ITD algorithm can barely be derived from these example figures. However, they can be better detected by using artificial broadband noise from known directions as test signals. The ITD method offers the advantage that the angular resolution can be selected by choosing the size I of the delay set in (4.20). In comparison with the PHAT-GCC algorithm, the accuracy is better for azimuths near 0° and 180°. Obviously, this is an advantage if two microphone pairs are used to find a speaker's position by calculating the cross point of the two azimuth estimates. Furthermore, there is no need for signal oversampling or increasing the FFT size because phase matching is done in the frequency domain. On the other hand, substantially more search algorithms are required for minima and maxima detections.

Our experiments with speech signals indicate less robustness against environmental noise and reverberation as compared to the PHAT-GCC method. The increased sensitivity with respect to room acoustics is due to the influence of sound reflections that smear maxima locations in the ITD histogram map. In [6] the authors suggest to set $P_k(\tau_i, m)$ to zero for values below a certain threshold. According to our experience, however, this does not improve the performance in reverberant rooms. Therefore, application of the ITD algorithm is limited to situations where accurate source localization under moderate environmental noise is needed. For automatic steering of microphone

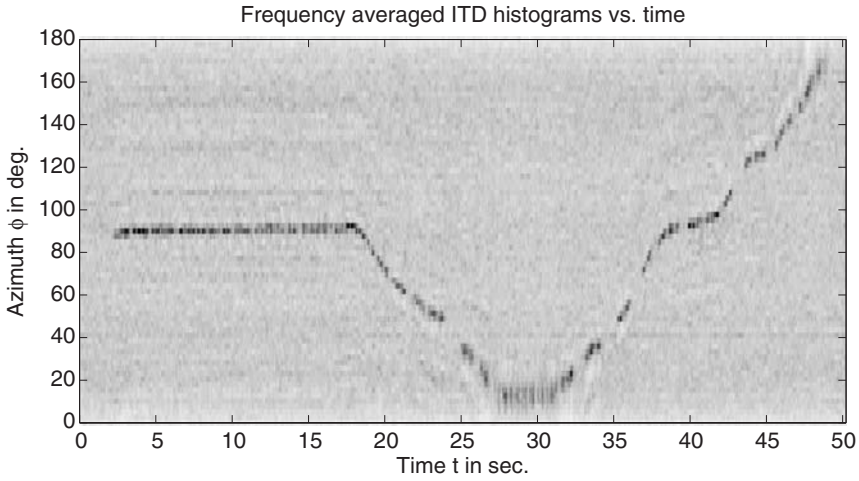

Fig. 4.7. ITD histogram map summed over frequency of a moving speaker (same acoustical environment as in Fig. 4.3, azimuth instead of delay values on vertical axis).

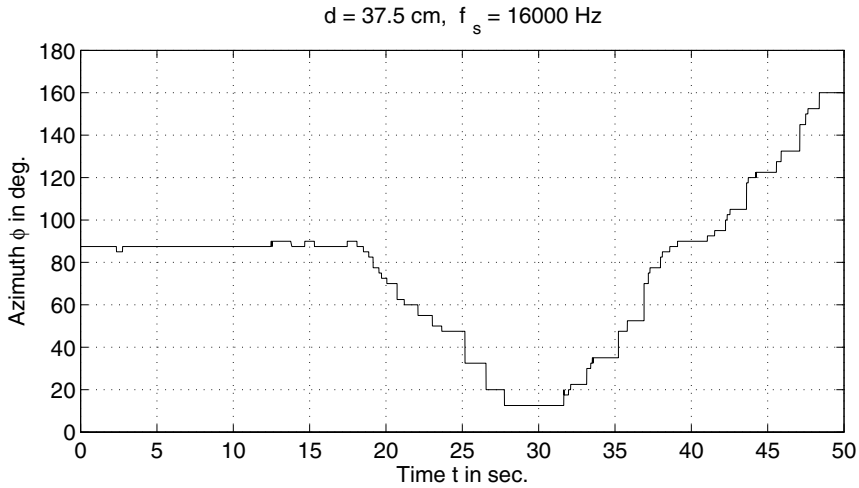

Fig. 4.8. Azimuth estimation by maximum search on the ITD histogram map from Fig. 4.7 (Estimates are held constant during speech pauses).

arrays, we prefer to use the PHAT-GCC method because of its robustness. Arrays of up to 8 microphones exhibit relatively broad main lobes in their array patterns. As a consequence, there is no need for an azimuth estimation accuracy less then $3^\circ \ldots 5^\circ$.

4.4 Source Localization Using Adaptive Filters

In the derivations of source localization algorithms, we have assumed an ideal wave propagation model so far. In such an environment with no sound reflections, the two microphone signals in Fig. 4.1 are simply delayed versions of the source signal. Although this model works remarkably well in real acoustic environments too, a more realistic approach is to find the signal delay from the actual impulse responses between source and microphones. In this section, two different adaptive systems for delay estimation are presented. The first system models the time delay between the two microphones. It is assumed that the direct path of sound propagation dominates. In the second method, we estimate the impulse responses by an adaptive eigenvalue decomposition. This method is more robust if strong reverberation is present. Both algorithms can efficiently be implemented by frequency-domain adaptive filters.

The first adaptive filtering technique is straight forward and shown in Fig. 4.9.[2] A detailed performance analysis can be found in [13]. We denote

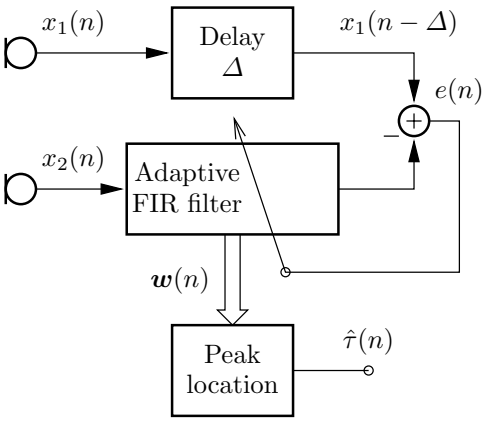

Fig. 4.9. Time delay estimation using an adaptive FIR filter (length L, coefficient vector $\boldsymbol{w}(n)$, delay $\Delta = \lfloor \frac{L-1}{2} \rfloor$).

FIR filter state as vector $\boldsymbol{x}_2(n)$ and coefficients as vector $\boldsymbol{w}(n)$ according to

$$\boldsymbol{x}_2(n) = \big[x_2(n)\; x_2(n-1)\; \cdots\; x_2(n-L+1)\big]^{\mathrm{T}} \tag{4.26}$$

$$\boldsymbol{w}(n) = \big[w_0(n)\; w_1(n)\; \cdots\; w_{L-1}(n)\big]^{\mathrm{T}}, \tag{4.27}$$

("T" denotes vector transpose). The error signal $e(n)$ is then given by

$$e(n) = x_1(n-\Delta) - \boldsymbol{w}^{\mathrm{T}}(n)\boldsymbol{x}_2(n), \tag{4.28}$$

[2] FIR = Finite Impulse Response Duration

($\Delta = \lfloor \frac{L-1}{2} \rfloor$). The Least Mean-Square (LMS) algorithm can be used to update the weight vector:

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu_{\text{LMS}}\, e(n)\boldsymbol{x}_2(n). \tag{4.29}$$

In general, however, a better performance is achieved with the normalized LMS algorithm

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \frac{\mu_{\text{NLMS}}}{\|\boldsymbol{x}_2(n)\|^2}\, e(n)\boldsymbol{x}_2(n), \tag{4.30}$$

with $\|\boldsymbol{x}_2(n)\|^2 = \boldsymbol{x}_2^{\text{T}}(n)\boldsymbol{x}_2(n)$. In order to improve the convergence behavior, a pre-emphasis filter with impulse response $h_{pre}(n) = \delta(n) - 0.9\delta(n-1)$ (unit impulse $\delta(n)$) can be used for simple pre-whitening of speech signals. Such a pre-filter is not required if we use the following frequency-domain adaptive filter. Only three FFTs per frame plus one FFT every M samples ($M = 2000$, typically) are needed. In addition, convergence is superior in case of speech signals due to a frequency dependent adaptive filter step size. The algorithm is based on the fast block LMS adaptive filter as proposed in [14], combined with a frequency dependent step size as suggested in [15]. To implement the LMS adaptive filter in the frequency domain by means of the FFT, samples are grouped into frames and coefficients are held constant till the next frame is processed. The update of the adaptive filter coefficients in frequency-domain at each frame index m can be summarized as follows:

$$X_2(m,k) = \sum_{n=0}^{N-1} x_2(mL+n)e^{-j\frac{2\pi}{N}nk}, \quad k = 0,1,\ldots,N-1 \tag{4.31}$$

$$y(m,n) = \frac{1}{N}\sum_{k=0}^{N-1} W(m,k)X_2(m,k)e^{j\frac{2\pi}{N}nk}, \quad n = 0,1,\ldots,N-1 \tag{4.32}$$

$$\tilde{e}(m,n) = \begin{cases} 0 & n = 0,1,\ldots,L-1 \\ x_1(mL+n-\Delta) - y(m,n) & n = L, L+1,\ldots,N-1 \end{cases} \tag{4.33}$$

$$E(m,k) = \sum_{n=0}^{N-1} \tilde{e}(m,n)e^{-j\frac{2\pi}{N}nk}, \quad k = 0,1,\ldots,N-1 \tag{4.34}$$

$$S_{x_2x_2}(m,k) = \alpha S_{x_2x_2}(m-1,k) + (1-\alpha)|X_2(m,k)|^2, \quad k = 0,1,\ldots,N-1 \tag{4.35}$$

$$W(m+1,k) = W(m,k) + \frac{\mu}{S_{x_2x_2}(m,k)+\varepsilon}\, X_2^*(m,k)E(m,k) \quad k = 0,1,\ldots,N-1. \tag{4.36}$$

The frame length is $N = 2L$, with a frame hop size equal to the adaptive filter length L. An overlap-save method with an N point DFT/IDFT is used

to perform the linear convolution needed in (4.28). Note that the step size of the weight update (4.36) is normalized by an estimate of the spectral power at each frequency point.[3] As a consequence, the convergence behavior of the adaptive algorithm is nearly independent on the signal spectrum.

Delay estimates are computed every M' frames (i.e. every $M = M'L$ samples) by finding peak locations of the adaptive filter coefficients

$$w(m', n) = \frac{1}{N} \sum_{k=0}^{N-1} W(m', k) e^{j \frac{2\pi}{N} nk}, \quad n = 0, 1, \ldots, N-1. \tag{4.37}$$

Due to the overlap-save method, the last L values of $w(m', n)$ are the valid filter coefficients to be searched to find the peak location. In addition, the search range can be further reduced because peak positions are limited to $[\Delta - N_d, \Delta + N_d]$, where $N_d = \lceil \frac{d}{v_s} f_s \rceil$ is the maximum delay between the microphone signals.

A typical example using the same microphone signals as before is shown in Fig. 4.10 and Fig. 4.11. The proposed frequency-domain adaptive filter is

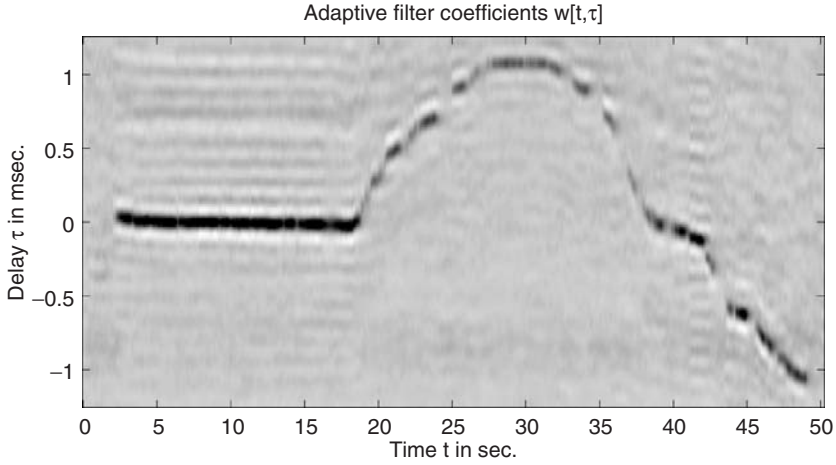

Fig. 4.10. Adaptive filter coefficient map of a moving speaker (same acoustical environment as in Fig. 4.3).

applied with length $L = 512$, step size $\mu = 0.2$, and $\alpha = 0.2$. The coefficient map is updated every $M = 2048$ samples to allow for sufficient convergence of the adaptive filter. Delay estimation is performed every M samples too by maximum detection using the coefficient map. Coefficients are oversampled by a factor of 4 to determine the peak location with sufficient accuracy.

[3] ε avoids division by zero during speech pauses.

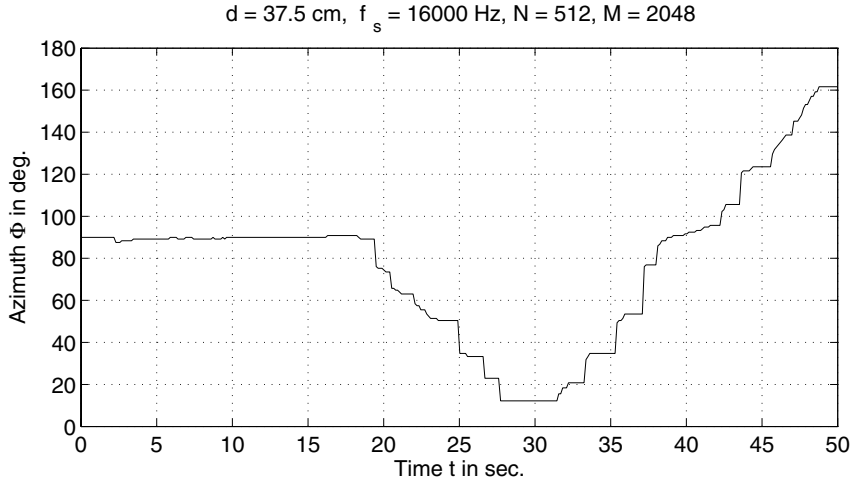

Fig. 4.11. Azimuth estimation by maximum search on the coefficient map from Fig. 4.10 (Estimates are held constant during speech pauses).

A different adaptive system showing a better performance in environments with strong reverberation is proposed in [7]. In principle, the impulse responses between source and microphones are estimated by means of an eigenvalue decomposition. Denoting $h_1(n)$ and $h_2(n)$ as impulse response from source to microphone 1, and microphone 2, respectively, we get the following discrete-time model:

$$x_1(n) = \sum_{k=-\infty}^{\infty} h_1(k)s(n-k) + v_1(n) \tag{4.38}$$

$$x_2(n) = \sum_{k=-\infty}^{\infty} h_2(k)s(n-k) + v_2(n), \tag{4.39}$$

(source signal $s(n)$, noise disturbances $v_{1,2}(n)$). At the moment, we assume a linear environment with time-invariant impulse responses. Later on, we will relax the time-invariance property by estimating $h_{1,2}(n)$ on a frame by frame basis. This allows for adaptation to sufficiently slow changes in the room acoustics, and for speaker movements. For the estimation of the impulse responses, we further assume that $h_{1,2}(n)$ can be approximated by filters with finite impulse response length L. Additionally, the noise signals $v_{1,2}(n)$ are neglected at first. This leads to the relation

$$(x_1 * h_2)(n) = (s * h_1 * h_2)(n) = (x_2 * h_1)(n) \tag{4.40}$$

between the convolutions since the order in which two stable sequences are convolved is unimportant (see Fig. 4.12). Equation (4.40) is the basis of an adaptive algorithm to estimate the impulse responses.

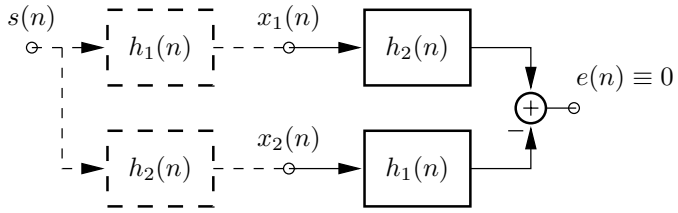

Fig. 4.12. Relationship between impulse responses according to (4.40) (signal model left, perfect estimation of impulse responses right).

If the impulse responses are approximated by length L filters, all data can be grouped in $L \times 1$ vectors

$$\boldsymbol{x}_i(n) = \left[x_i(n)\ x_i(n-1)\ \cdots\ x_i(n-L+1)\right]^{\mathrm{T}}, \quad i = 1, 2 \tag{4.41}$$

$$\boldsymbol{h}_i = \left[h_i(0)\ h_i(1)\ \cdots\ h_i(L-1)\right]^{\mathrm{T}}. \tag{4.42}$$

Equation (4.40) can now be rewritten as

$$\boldsymbol{x}_1^{\mathrm{T}}(n)\boldsymbol{h}_2 = \boldsymbol{x}_2^{\mathrm{T}}(n)\boldsymbol{h}_1. \tag{4.43}$$

Following the derivation outlined in [7], we introduce $2L \times 1$ vectors

$$\boldsymbol{x}(n) = \left[\boldsymbol{x}_1^{\mathrm{T}}(n)\ \ \boldsymbol{x}_2^{\mathrm{T}}(n)\right]^{\mathrm{T}} \tag{4.44}$$

$$\boldsymbol{u} = \left[\boldsymbol{h}_2^{T}\ \ -\boldsymbol{h}_1^{T}\right]^{T} \tag{4.45}$$

to rewrite (4.40):

$$\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{u} = \boldsymbol{x}_1^{\mathrm{T}}(n)\boldsymbol{h}_2 - \boldsymbol{x}_2^{\mathrm{T}}(n)\boldsymbol{h}_1 = 0. \tag{4.46}$$

Left multiplying (4.46) by $\boldsymbol{x}(n)$ and taking expectation yields

$$\boldsymbol{R}_{\boldsymbol{xx}}(n)\boldsymbol{u} = \boldsymbol{0}. \tag{4.47}$$

$\boldsymbol{R}_{\boldsymbol{xx}}(n) = E\{\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)\}$ is the $2L \times 2L$ covariance matrix of the two microphone signals. Note that $\boldsymbol{R}_{\boldsymbol{xx}}(n)$ contains both temporal and spatial correlations of the microphone signals. Equation (4.47) indicates that $\boldsymbol{u}$ is the eigenvector of $\boldsymbol{R}_{\boldsymbol{xx}}(n)$ corresponding to eigenvalue 0. Therefore, both impulse responses can be found by determining this eigenvector.

If noise signals $v_{1,2}(n)$ are present, $\boldsymbol{u}$ may be estimated by minimizing $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}_{\boldsymbol{xx}}(n)\boldsymbol{u}$ with constraint $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} = 1$ [7]. Consequently, we get $\boldsymbol{u}$ by computing the normalized eigenvector of $\boldsymbol{R}_{\boldsymbol{xx}}(n)$ corresponding to the smallest eigenvalue. There exist several efficient algorithms to find the smallest eigenvalue and the associated eigenvector of a correlation matrix. Since the dimension of matrix $\boldsymbol{R}_{\boldsymbol{xx}}(n)$ is quite large, an adaptive algorithm will be used. As a main advantage, we need only a few iterations because the TDD between

the microphone signals is of interest. There is no need to estimate the actual shapes of the impulse responses. According to (4.46) and Fig. 4.12, the error signal

$$e(n) = \boldsymbol{u}^{\mathrm{T}}(n)\boldsymbol{x}(n) \tag{4.48}$$

should be zero under ideal conditions. Actually, the cost function

$$J(n) = \frac{1}{2}E\left\{e^2(n)\right\} = \frac{1}{2}\boldsymbol{u}^{\mathrm{T}}(n)\boldsymbol{R}_{\boldsymbol{xx}}(n)\boldsymbol{u}(n) \tag{4.49}$$

can be minimized with the gradient-based adaptive algorithm

$$\boldsymbol{u}(n+1) = \boldsymbol{u}(n) - \mu_{\mathrm{LMS}}\,\boldsymbol{\nabla}_{\boldsymbol{u}}J(n) = \boldsymbol{u}(n) - \mu_{\mathrm{LMS}}\,\boldsymbol{R}_{\boldsymbol{xx}}(n)\boldsymbol{u}(n). \tag{4.50}$$

$\boldsymbol{\nabla}_{\boldsymbol{u}}J(n)$ is the cost function gradient with respect to vector $\boldsymbol{u}$. With the approximation $\boldsymbol{R}_{\boldsymbol{xx}}(n) = E\{\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)\} \approx \boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)$, we get the LMS algorithm

$$\boldsymbol{u}(n+1) = \boldsymbol{u}(n) - \mu_{\mathrm{LMS}}\,e(n)\boldsymbol{x}(n). \tag{4.51}$$

The constraint $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} = 1$ can be taken into account by normalization [7]:

$$\boldsymbol{v}(n) = \boldsymbol{u}(n) - \mu_{\mathrm{NLMS}}\,e(n)\boldsymbol{x}(n) \tag{4.52}$$

$$\boldsymbol{u}(n+1) = \frac{\boldsymbol{v}(n)}{\sqrt{\boldsymbol{v}^{\mathrm{T}}(n)\boldsymbol{v}(n)}}\,. \tag{4.53}$$

As mentioned above, only the delay between the two microphone signals is of interest. If we initialize the elements $u_i(n)$ of vector $\boldsymbol{u}(n)$ at $n = 0$ by

$$u_i(0) = \begin{cases} 0 & 0 \le i \le \lfloor \frac{L}{2} \rfloor - 1 \\ 1 & i = \lfloor \frac{L}{2} \rfloor \\ 0 & \lfloor \frac{L}{2} \rfloor + 1 \le i \le 2L-1 \end{cases}, \tag{4.54}$$

then a negative peak will evolve in $\boldsymbol{u}(n)$ during adaptation. This peak corresponds to the direct path in the impulse response $\boldsymbol{h}_1$ (see (4.45)). The positive peak will remain at the initial position $i = \lfloor \frac{L}{2} \rfloor$. The index difference of these two peaks in $\boldsymbol{u}(n)$ determines the delay between the microphone signals. Since the position of the positive peak is fixed, we need to find the index of the negative peak only by searching vector elements $u_i(n)$, $\lfloor \frac{L}{2} \rfloor + 1 \le i \le 2L-1$. In a practical implementation, we will interpolate $\boldsymbol{u}(n)$ before peak position finding. Additionally, in case of a moving speaker we have to reset the adaptive algorithm periodically to allow tracking. Otherwise, peaks will stick at the first estimated positions, particularly for small step size values μ_{LMS}. Setting $u_i(nK) = u_i(0)$ for some period K removes all old negative peaks and allows the adaptive algorithm to adjust to the new delay position. Period K determines the tracking speed and is set to some 1000 samples, typically. During this period, the adaptive algorithm has plenty of time to converge.

The adaptive source localization algorithm can easily be implemented in the time domain. However, a significantly greater computational efficiency can

be achieved by using a frequency-domain adaptive filter. As opposed to [7], an FFT based algorithm can be devised requiring only 4 FFTs per frame (plus one FFT at every initialization period) instead of 7 FFTs. This saving is obtained by eliminating the normalization of vector $\boldsymbol{u}$ in (4.53). It is argued in [7] that the normalization may avoid an error propagation in (4.51) if the algorithm runs over a long period of time. However, in order to ensure tracking, we have to periodically reset the adaptive algorithm. Thus, an eventual error propagation will efficiently be eliminated too.

The algorithm has a similar structure as the fast LMS algorithm (4.31) - (4.36):

$$X_1(m,k) = \sum_{n=0}^{N-1} x_1(mL+n)e^{-j\frac{2\pi}{N}nk}, \quad k = 0,1,\ldots,N-1 \tag{4.55}$$

$$X_2(m,k) = \sum_{n=0}^{N-1} x_2(mL+n)e^{-j\frac{2\pi}{N}nk}, \quad k = 0,1,\ldots,N-1 \tag{4.56}$$

$$e(m,n) = \frac{1}{N}\sum_{k=0}^{N-1}\Big[U_1(m,k)X_1(m,k) + U_2(m,k)X_2(m,k)\Big]e^{j\frac{2\pi}{N}nk}, \quad n = 0,1,\ldots,N-1 \tag{4.57}$$

$$\tilde{e}(m,n) = \begin{cases} 0 & n = 0,1,\ldots,L-1 \\ e(m,n) & n = L,L+1,\ldots,N-1 \end{cases} \tag{4.58}$$

$$E(m,k) = \sum_{n=0}^{N-1} \tilde{e}(m,n)e^{-j\frac{2\pi}{N}nk}, \quad k = 0,1,\ldots,N-1 \tag{4.59}$$

$$S_{x_1x_1}(m,k) = \alpha S_{x_1x_1}(m-1,k) + (1-\alpha)|X_1(m,k)|^2, \quad k = 0,1,\ldots,N-1 \tag{4.60}$$

$$S_{x_2x_2}(m,k) = \alpha S_{x_2x_2}(m-1,k) + (1-\alpha)|X_2(m,k)|^2, \quad k = 0,1,\ldots,N-1 \tag{4.61}$$

$$U_1(m+1,k) = U_1(m,k) - \frac{\mu}{S_{x_1x_1}(m,k)+\varepsilon}X_1^*(m,k)E(m,k), \quad k = 0,1,\ldots,N-1 \tag{4.62}$$

$$U_2(m+1,k) = U_2(m,k) - \frac{\mu}{S_{x_2x_2}(m,k)+\varepsilon}X_2^*(m,k)E(m,k), \quad k = 0,1,\ldots,N-1. \tag{4.63}$$

Similarly to the fast LMS algorithm, the DFT length is set to $N = 2L$, with impulse response length L. Vector $\boldsymbol{u}$ (see (4.45)) is split into two length L sub-vectors, i.e. $\boldsymbol{u} = [\boldsymbol{u}_1^{\mathrm{T}}\ \boldsymbol{u}_2^{\mathrm{T}}]^{\mathrm{T}}$. The updates of these sub-vectors are performed in the frequency domain. Delay estimates are computed every M' frames (i.e. every $M = M'L$ samples) by finding the dominant negative peak in $\boldsymbol{u}_2$. Likewise to (4.37), the elements of $\boldsymbol{u}_2$ are obtained by the IDFT

$$u_2(m', n) = \frac{1}{N} \sum_{k=0}^{N-1} U_2(m', k) e^{j\frac{2\pi}{N}nk}, \quad n = 0, 1, \ldots, N-1. \tag{4.64}$$

Nearly the same results as in Fig. 4.10, 4.11 are obtained if we use the same microphone signals and algorithm parameters $L = 512$, $\alpha = 0.2$, and $\mu = 0.2$.

4.5 Some Remarks on Algorithm Selection

Deciding which algorithm to choose depends on the specific area of application. In a car cabin, with no speaker movement, little reverberation, and heavy disturbing noise, the PHAT-GCC and the frequency-domain adaptive filter perform best. Both algorithms also exhibit the lowest computational demand. In situations with modest reverberation, the two adaptive source localization algorithms show the same performance. However, according to a detailed experimental comparison of algorithms in [7], the adaptive eigenvalue decomposition offers a better performance in rooms with strong reverberation and moderate noise. The best accuracy in azimuth estimation can be expected by the ITD based algorithm if nearly ideal sound propagation is present. However, the prize to be payed is the relatively high computational cost and memory demand.

If we compare the arithmetic operations per frame interval required by each algorithm, we get the coarse result listed in Tab. 4.1. The FFT length is equal to frame length N in case of PHAT-GCC and ITD algorithm. All FFTs use real-valued input data. The fast LMS algorithm (FLMS) and the adaptive eigenvalue decomposition (AEVD) require length $N = 2L$ FFTs (impulse response length L). One FFT is needed every M' frames only. Oversampling is not considered in Tab. 4.1. If we apply e.g. an oversampling (factor R) to find the GCC peak, one FFT must have a length RN. The IDT-algorithm requires only 2 real-input FFTs and no oversampling. However, the numbers of additions and multiplications depend on the azimuth resolution $\Delta\Phi \approx 180°/I$. In addition, $\frac{N}{2} + 1$ maximum/minimum search operations are needed.

Table 4.1. Comparison of computational requirements per frame of length N

Algorithm	FFT	Add.	Mult.	Div.	Sqrt.	Search
PHAT	3	$\frac{5}{2}N$	$8N$	$\frac{N}{2}$	$\frac{N}{2}$	1
ITD	2	$\left(4I + \frac{1}{2}\right)N$	$\left(\frac{11}{2}I + 2\right)N$	-	-	$\frac{N}{2} + 1$
FLMS	4	$\frac{9}{2}N$	$7N$	$\frac{N}{2}$	-	1
AEVD	5	$9N$	$14N$	N	-	1

4.6 Frequency-Domain Adaptive Beamformer with Speaker Tracking

In this section, we present an adaptive beamformer combined with source localization. The system automatically adjusts the main lobe of the array pattern to a speaker and suppresses sounds from all other directions. This behavior is preserved if the speaker moves. Applications include teleconferencing, hands-free telecommunications in cars, etc. The adaptive beamformer is based on the Frost constrained LMS algorithm [16]. However, as opposed to the original Frost beamformer, the adaptive algorithm is formulated in the frequency domain.

The main advantages of this approach are the possibility to use an efficient multi-input overlap-add FFT filterbank, the avoidance of variable fractional delay filters, and the inclusion of more constraints like nulls in the array pattern. In addition, the FFT filterbank beamformer can easily be combined with an adaptive post-filter for speech enhancement purposes [17–19]. Disadvantages are the signal delay introduced by the FFT block processing and a higher storage demand as compared with the time domain approach. However, signal delays are within usual tolerance limits if the frame size is properly chosen (e.g. 512 at 16 kHz sampling frequency). Additionally, memory requirements are no limiting factors with modern hardware.

The basic structure of the adaptive beamformer is shown in Fig. 4.13. Single channel overlap-add FFT filterbanks are used in many audio-based ap-

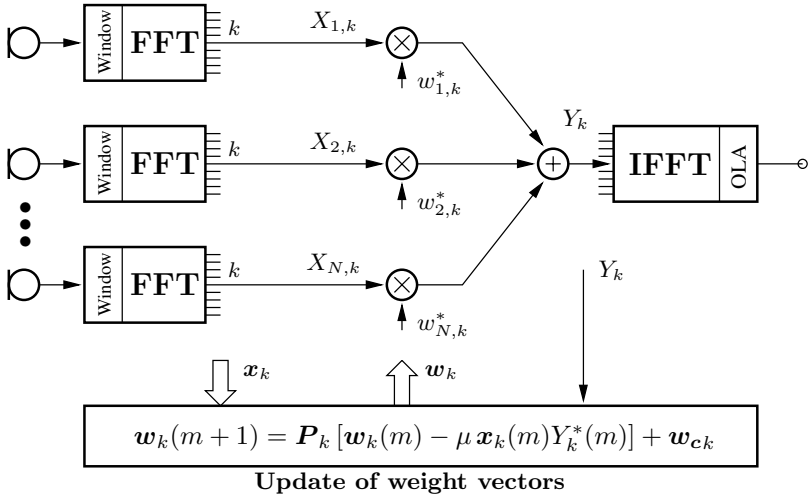

Fig. 4.13. Adaptive beamformer with N-channel overlap-add FFT filterbank and constrained LMS algorithm to compute weights $\boldsymbol{w}_k(m)$ (frequency index k, frame index m).

plications [20]. Such a multirate filterbank structure is highly efficient and offers a nearly perfect signal reconstruction property. In our extended filterbank system with multiple input channels, FFT spectra are modified by complex-valued weights on a frame by frame basis. For each frequency index k and frame index m the N-dimensional weight vectors $\boldsymbol{w}_k(m)$ are updated according to a constrained LMS algorithm. This algorithm will be derived in the sequel. Algorithm parameters $\boldsymbol{P}_k$ and $\boldsymbol{w}_{ck}$ depend on the desired direction which is supplied by a source localization algorithm. The source localization algorithm makes use of the already available FFTs of the out-most two microphone signals of the array.

In the following derivation of the frequency-domain adaptive algorithm, the frame index m is omitted for clarity. The beamformer optimization problem to be solved by means of an adaptive algorithm may be defined by the minimization of a quadratic cost function under linear constraints:

$$\boldsymbol{w}_k = \arg\min_{\boldsymbol{w}_k} \boldsymbol{w}_k^H \boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k} \boldsymbol{w}_k, \quad \boldsymbol{C}_k^H \boldsymbol{w}_k = \boldsymbol{f} \tag{4.65}$$

(for each frequency index k). Superscript H denotes Hermitian transposition, i. e. transposition combined with complex conjugation. The minimization of the quadratic form stems from the desired minimization of the power of Y_k given by

$$E\{Y_k^2\} = \boldsymbol{w}_k^{\mathrm{H}} E\{\boldsymbol{x}_k \boldsymbol{x}_k^{\mathrm{H}}\} \boldsymbol{w}_k = \boldsymbol{w}_k^{\mathrm{H}} \boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k} \boldsymbol{w}_k, \tag{4.66}$$

with $\boldsymbol{w}_k = [w_{1,k}\, w_{2,k} \cdots w_{N,k}]^{\mathrm{T}}$, $\boldsymbol{x}_k = [X_{1,k}\, X_{2,k} \cdots X_{N,k}]^{\mathrm{T}}$. Matrix $\boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k}$ is the $N \times N$ spatio-spectral correlation matrix at frequency index k. This matrix depends on array geometry and sound field, and will be estimated by the adaptive algorithm. Minimization of $E\{Y_k^2\}$ has to be done with constraints. At least, signals from the desired direction must not be attenuated. In addition, signals from certain other directions may be suppressed by imposing nulls in the array pattern. These constraints are collected in (4.65) as a set of equations with matrix $\boldsymbol{C}_k$. The structure of this matrix is determined by the wave propagation model. If we assume plane waves and far field conditions, $\boldsymbol{C}_k$ is composed of steering vectors of the form

$$\boldsymbol{d}_k(\Phi) = \left[e^{j\Omega_k \tau_1(\Phi)}\, e^{j\Omega_k \tau_2(\Phi)}\, \cdots\, e^{j\Omega_k \tau_N(\Phi)} \right]^{\mathrm{T}}, \tag{4.67}$$

with $\Omega_k = 2\pi f_{\mathrm{s}} \frac{k}{N_f}$ (sampling frequency f_{s}, FFT lengths N_f). Microphone signal delays τ_i depend on the direction (azimuth Φ) of the impinging wave. For simplicity, we are using a one-dimensional array with a coordinate system as shown in Fig. 4.1. This is not a restriction in general because delays τ_i can easily be calculated in a 3-dimensional coordinate system. In addition, more complicated steering vectors can be used if we apply other wave propagation models like those covering near field conditions. The structure of the optimization problem remains the same. We have to use different steering vectors only. Actually, knowledge of the sound propagation is very incomplete. Therefore, the simple steering vectors offer a convenient way to overcome this lag

of information. However, a better beamformer performance can be achieved with more realistic steering vectors.

Suppose that the desired speaker direction has azimuth Φ_d and we want an array pattern null at azimuth Φ_s. Then $\boldsymbol{d}_k(\Phi_d)^{\mathrm{H}}\boldsymbol{w}_k = 1$ is the beamformer response in desired direction and $\boldsymbol{d}_k(\Phi_s)^{\mathrm{H}}\boldsymbol{w}_k = 0$ is the response in the unwanted direction. Therefore, matrix $\boldsymbol{C}_k$ is given by $\boldsymbol{C}_k = [\boldsymbol{d}_k(\Phi_d)\, \boldsymbol{d}_k(\Phi_s)]$ and vector $\boldsymbol{f}$ must be set to $\boldsymbol{f} = [1\,0]^{\mathrm{T}}$ in order to get the constraints in (4.65). We can include more array pattern nulls and extend the row dimension of matrix $\boldsymbol{C}_k$. To avoid an over-determined set of equations, the number of constraints must be less than the number N of microphones. In practice, only a few constraints should be used to obtain a good beamforming pattern with a strong main lobe and small side lobes.

We can solve the constrained optimization problem (4.65) with Lagrange multipliers by defining the cost function

$$L(\boldsymbol{w}_k, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{w}_k^{\mathrm{H}}\boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k}\boldsymbol{w}_k + \boldsymbol{\lambda}^{\mathrm{H}}\left(\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{w}_k - \boldsymbol{f}\right). \tag{4.68}$$

Evaluation of the gradient of this cost function yields

$$\boldsymbol{\nabla}_{\boldsymbol{w}_k} L(\boldsymbol{w}_k, \boldsymbol{\lambda}) = \boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k}\boldsymbol{w}_k + \boldsymbol{C}_k\boldsymbol{\lambda}. \tag{4.69}$$

Using the gradient relationship, an iterative solution of the optimization problem on a frame by frame basis is given by

$$\boldsymbol{w}_k(m+1) = \boldsymbol{w}_k(m) - \mu_{\mathrm{LMS}}\, \boldsymbol{\nabla}_{\boldsymbol{w}_k} L(\boldsymbol{w}_k, \boldsymbol{\lambda}). \tag{4.70}$$

Lagrange multiplier $\boldsymbol{\lambda}$ is obtained from (4.69) and (4.70) combined with the constraints $\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{w}_k(m+1) = \boldsymbol{f}$ (see (4.65)) according to

$$\begin{aligned}\boldsymbol{\lambda} = {} & \frac{1}{\mu_{\mathrm{LMS}}}\left(\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{w}_k(m) - \left(\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k}\boldsymbol{w}_k(m) \\ & - \frac{1}{\mu_{\mathrm{LMS}}}\left(\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{f}.\end{aligned} \tag{4.71}$$

Using this relationship in (4.69), we get from (4.70)

$$\boldsymbol{w}_k(m+1) = \boldsymbol{P}_k\Big[\boldsymbol{w}_k(m) - \mu_{\mathrm{LMS}}\, \boldsymbol{S}_{\boldsymbol{x}_k\boldsymbol{x}_k}\boldsymbol{w}_k(m)\Big] + \boldsymbol{w}_{\boldsymbol{c}k}, \tag{4.72}$$

with $N \times N$ matrix

$$\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{C}_k\left(\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{C}_k^{\mathrm{H}}, \tag{4.73}$$

and $N \times 1$ vector

$$\boldsymbol{w}_{\boldsymbol{c}k} = \boldsymbol{C}_k\left(\boldsymbol{C}_k^{\mathrm{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{f}. \tag{4.74}$$

We finally arrive at the constrained LMS algorithm by replacing the unknown spatio-spectral correlation matrix by the basic estimate $\widetilde{\boldsymbol{S}}_{\boldsymbol{x}_k\boldsymbol{x}_k} = \boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{H}}$ and applying $Y_k(m) = \boldsymbol{w}_k^{\mathrm{H}}(m)\boldsymbol{x}_k(m)$ (see Fig. 4.13):

$$\boldsymbol{w}_k(m+1) = \boldsymbol{P}_k\Big[\boldsymbol{w}_k(m) - \mu_{\text{LMS}}\,\boldsymbol{x}_k(m)Y_k^*(m)\Big] + \boldsymbol{w}_{\boldsymbol{c}k}. \tag{4.75}$$

Although the constrained LMS algorithm can easily be implemented, the basic form given by (4.75) exhibits a suppression of the desired signal in real environments. The constraint $\boldsymbol{d}_k(\Phi_d)^{\text{H}}\boldsymbol{w}_k = 1$ can hardly be met in practical situations due to microphone tolerances, microphone position errors, and most important, errors of the desired direction. If we modify the adaptive algorithm in order to achieve a large robustness against these influences, suppression of the desired signal can be avoided. By modeling the errors as uncorrelated white noise signals at the microphone inputs, we observe that the variances of these errors are amplified by $\boldsymbol{w}_k^{\text{H}}\boldsymbol{w}_k$. Thus, limiting $\boldsymbol{w}_k^{\text{H}}\boldsymbol{w}_k = \|\boldsymbol{w}_k\|^2$ will reduce the influence of these errors. A detailed discussion on making the Frost beamformer more robust can be found in [21].

The weight vector norm constraint can conveniently be included in the adaptive algorithm, if we split the weight vector into $\boldsymbol{w}_k(m) = \boldsymbol{v}_k(m) + \boldsymbol{w}_{\boldsymbol{c}k}$ and recognize $\boldsymbol{P}_k\boldsymbol{w}_{\boldsymbol{c}k} = \boldsymbol{0}$ (see (4.73), (4.74)). With upper bound B_k, the norm constraint can be expressed as

$$\|\boldsymbol{w}_k(m)\|^2 = \|\boldsymbol{v}_k(m)\|^2 + \|\boldsymbol{w}_{\boldsymbol{c}k}\|^2 \le B_k. \tag{4.76}$$

It follows that the norm of the variable component $\boldsymbol{v}_k(m)$ of $\boldsymbol{w}_k(m)$ must be limited by

$$\|\boldsymbol{v}_k(m)\| \le \sqrt{B_k - \|\boldsymbol{w}_{\boldsymbol{c}k}\|^2} = b_k. \tag{4.77}$$

Parameter b_k does not depend on frame index m and can be pre-computed for every frequency index k. Therefore, we get the final adaptive algorithm:

$$\text{Initialization:}\quad \boldsymbol{w}_{\boldsymbol{c}k} = \boldsymbol{C}_k\left(\boldsymbol{C}_k^{\text{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{f} \tag{4.78}$$

$$\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{C}_k\left(\boldsymbol{C}_k^{\text{H}}\boldsymbol{C}_k\right)^{-1}\boldsymbol{C}_k^{\text{H}} \tag{4.79}$$

$$b_k = \sqrt{B_k - \|\boldsymbol{w}_{\boldsymbol{c}k}\|^2} \tag{4.80}$$

$$\boldsymbol{v}_k(0) = \boldsymbol{0}. \tag{4.81}$$

$$\text{For each frame index } m: \tag{4.82}$$

$$\tilde{\boldsymbol{v}}_k(m+1) = \boldsymbol{P}_k\Big[\boldsymbol{v}_k(m) - \mu_{\text{LMS}}\,\boldsymbol{x}_k(m)Y_k^*(m)\Big] \tag{4.83}$$

$$\boldsymbol{v}_k(m+1) = \begin{cases} \tilde{\boldsymbol{v}}_k(m+1) & \text{if } \|\tilde{\boldsymbol{v}}_k(m+1)\| \le b_k \\ \dfrac{b_k\tilde{\boldsymbol{v}}_k(m+1)}{\|\tilde{\boldsymbol{v}}_k(m+1)\|} & \text{if } \|\tilde{\boldsymbol{v}}_k(m+1)\| > b_k \end{cases} \tag{4.84}$$

$$\boldsymbol{w}_k(m+1) = \boldsymbol{v}_k(m+1) + \boldsymbol{w}_{\boldsymbol{c}k} \tag{4.85}$$

$$k = 0, 1, \ldots, N_f.$$

In general, this adaptive algorithm requires a substantial amount of memory due to storage of matrix $\boldsymbol{P}_k$ and vector $\boldsymbol{w}_{\boldsymbol{c}k}$ for each FFT frequency index. However, for special cases like broadside arrays (azimuth $\Phi = 90°$),

all of these vectors and matrices are equal. In addition, memory savings are also possible in case of symmetries regarding the location of specified nulls in the array pattern. If no specified null is present, matrix inversion in (4.73) and (4.74) reduces to scalar division because constraint matrix $\boldsymbol{C}_k$ is equal to the steering vector $\boldsymbol{d}_k(\Phi_d)$. This important case occurs at arrays for speaker tracking where fixed nulls in the beamformer pattern are not desired.

The step size μ_{LMS} of the adaptive algorithm must be selected with some care. As shown in [16], convergence of the constrained LMS algorithm is ensured if

$$0 < \mu_{\text{LMS}} < \frac{2}{3\,E\{\boldsymbol{x}_k^{\text{H}}\boldsymbol{x}_k\}}\,. \tag{4.86}$$

Therefore, a proper normalization of the step size μ_{LMS} will improve the convergence behavior of the adaptive algorithm. With such a modification, the convergence speed is independent on the signal magnitudes. In accordance to the normalized LMS algorithm, the modified weight vector update is then given by

$$\tilde{\boldsymbol{v}}_k(m+1) = \boldsymbol{P}_k \left[\boldsymbol{v}_k(m) - \frac{\mu}{\|\boldsymbol{x}_k(m)\|^2 + \varepsilon}\,\boldsymbol{x}_k(m)Y_k^*(m)\right]. \tag{4.87}$$

Typically, the new step size μ should be chosen between 0.001 and 0.02 to ensure a stable convergence of the adaptive algorithm.

Another important design parameter of the constrained LMS algorithm is the upper bound B_k. We get a sensitive superdirective array with $B_k > 10$. On the other hand, a robust delay-and-sum beamformer is obtained with small values ($B_k < 1$). In addition, B_k must be frequency dependent in order to achieve a flat beamformer frequency response not only in the exact desired direction but also at small deviations thereof. In principle, the frequency dependency of B_k can be optimized to obtain a flat frequency response. However, a tolerance analysis of perturbated arrays shows that the following set of limits works very well at a sampling frequency of $f_{\text{s}} = 16$ kHz [22]:

$$10\log_{10} B_k = \begin{cases} 10\,\text{dB} & 0 < f \le 250\,\text{Hz} \\ 8\,\text{dB} & 250\,\text{Hz} < f \le 450\,\text{Hz} \\ 2\,\text{dB} & 450\,\text{Hz} < f \le 700\,\text{Hz} \\ -2\,\text{dB} & 700\,\text{Hz} < f \le 1000\,\text{Hz} \\ -4\,\text{dB} & 1000\,\text{Hz} < f \le 2000\,\text{Hz} \\ -6\,\text{dB} & 2000\,\text{Hz} < f \le 4000\,\text{Hz} \\ -7.5\,\text{dB} & 4000\,\text{Hz} < f \le 8000\,\text{Hz}. \end{cases} \tag{4.88}$$

Note that the frequency index k of the N_f-point FFT is given by $k = \text{round}\left(N_f \frac{f}{f_{\text{s}}}\right)$.

We can combine the adaptive filterbank beamformer with a source localization subsystem as shown in Fig. 4.14. This augmented system is capable to

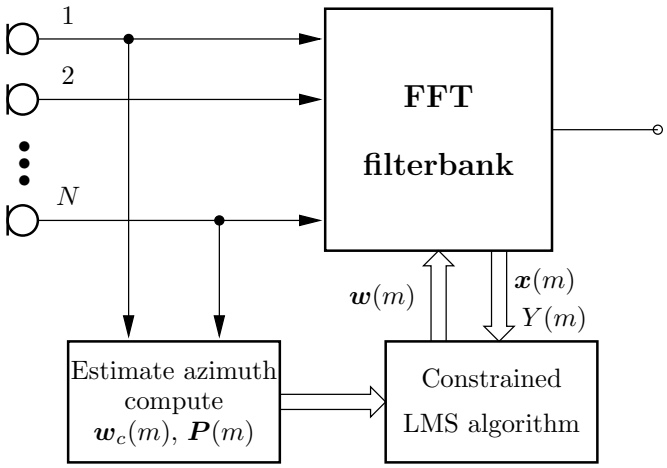

Fig. 4.14. Adaptive FFT filterbank beamformer combined with source localization to be used for automatic speaker tracking (frame index m).

focus the main lobe of the beam pattern to a moving speaker by re-computing the parameters $\boldsymbol{P}_k$, $\boldsymbol{w}_{\boldsymbol{c}k}$, b_k at multiples of the frame index. With a typical frame length of 512, frames are processed every $512/4 = 128$ samples, i. e. every 8 ms at 16 kHz sampling frequency. This is the minimum time period to re-compute $\boldsymbol{P}_k$, $\boldsymbol{w}_{\boldsymbol{c}k}$, b_k based on azimuth estimation. It can barely be used because the adaptive filter typically needs several 100 ms to converge. The starting solution $\boldsymbol{w}_k(0) = \boldsymbol{w}_{\boldsymbol{c}k}$ corresponds to a delay-and-sum beamformer and offers an adequate beam pattern during fast movements of the speaker. Adaptation will begin after the speaker position has been settled. It should be noted, however, that there is no need to reset the adaptive filter weight vectors $\boldsymbol{w}_k$ at new azimuth estimates.

For azimuth estimation, all of the previously presented source localization algorithms can efficiently be implemented in the frequency domain. Therefore, we can directly use the already available FFTs of the microphone signals (and not the signals themselves, as shown in Fig. 4.14). We have implemented the adaptive beamformer using an array of 8 microphones, a sampling frequency of 16 kHz, and an FFT length of 512 with Hann windowing of input frames. With a frame hop size of $512/4 = 128$ samples, we obtain a filterbank oversampling by a factor of 4. This oversampling factor guarantees that distortions due to multirate filterbank processing are not audible.

Both uniform and non-uniform array geometries have been investigated. As an example, the layout of a non-uniform microphone array is sketched in Fig. 4.15. This configuration requires fewer sensors than a comparable uniform array and offers a good tradeoff between main lobe width and side lobe amplitudes over the whole frequency range. Due to the use of a linear array,

Fig. 4.15. Microphone array geometry in cm (total size 37.5 cm).

the azimuth range is confined to a 180° field-of-view. If a 360° field-of-view is required, a circular array geometry should be preferred [23].

To avoid spatial aliasing, the input signal must be bandlimited to 6400 Hz. There is no need for additional low pass filters in the microphone channels, if we set the respective frequency bins of the FFTs to zero. This will also reduce the size of vectors and matrices needed by the adaptive algorithm.

The PHAT-GCC algorithm is used for automatic speaker tracking. To provide sufficient time for convergence of the adaptive algorithm, parameters $\boldsymbol{P}_k$, $\boldsymbol{w}_{\boldsymbol{c}k}$ are held constant during speech pauses and during speaker movements with changes in azimuth less than 2°.

In order to visualize the functioning of the adaptive beamformer with speaker tracking, we show a representative array pattern in Fig. 4.16 and Fig. 4.17 at a frequency of 1 kHz. We use the same speaker movement as in the source localization experiments. The speaker's position starts at broadside

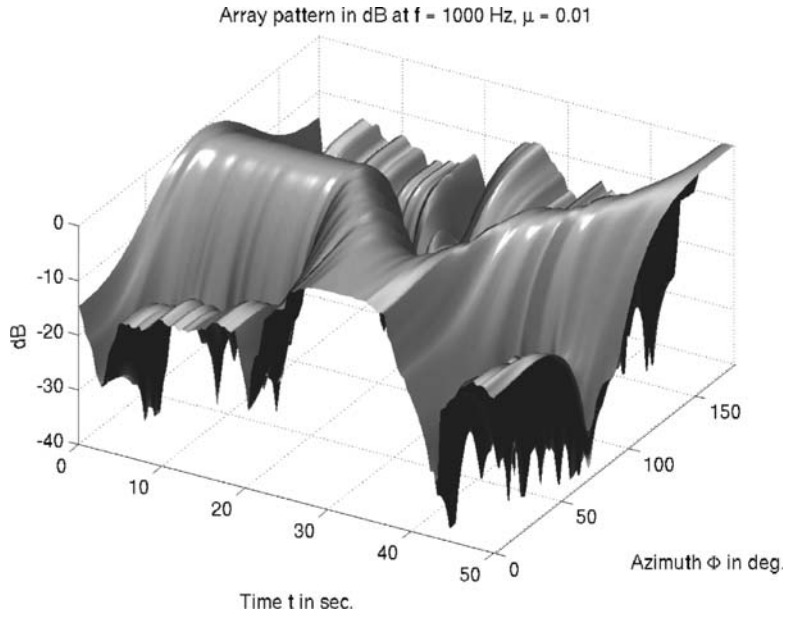

Fig. 4.16. Log-scale array pattern at $f = 1$ kHz of the adaptive beamformer automatically steered to a moving speaker.

($\Phi = 90°$), moves on towards $\Phi = 0°$, and continues to move back to $\Phi = 90°$, and finally $\Phi = 180°$. The main lobe of the array pattern follows this movement. The estimated azimuth trace is overlayed in the image plots shown in Fig. 4.17 at a frequency of 1 kHz, and in Fig. 4.18 at 3 kHz, respectively.

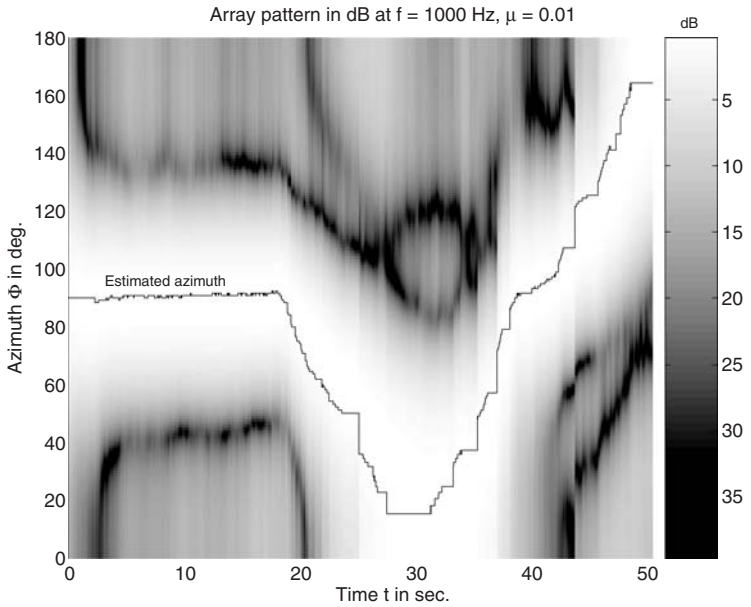

Fig. 4.17. Array pattern at $f = 1$ kHz of the automatically steered adaptive beamformer with superimposed estimated azimuth trace of a moving speaker.

The main lobe is clearly visible as a white region following the trace of the estimated azimuth. The settling period of the adaptive algorithm can be observed at the beginning where the speaker position remains constant at $\Phi = 90°$. A sharper main lobe but larger side lobe maxima are present in the array pattern at $f = 3$ kHz, as compared with the pattern at $f = 1$ kHz. This reflects the behavior of a delay-and-sum beamformer which is used as the starting solution of the adaptive algorithm. It should be noted that the beamformer shows a unity gain frequency response in desired direction. Only main lobe width and side lobe patterns change with frequency. The chopped texture of the array pattern in Fig. 4.18 is due to the step-like azimuth changes after hold operations during speech pauses.

The behavior of the adaptive beamformer depends on the input signals. The array patterns shown in Fig. 4.16, 4.17, 4.18 are computed using a single moving speaker. If we use a fixed desired direction, i.e. switch off speaker tracking, the adaptive algorithm will automatically suppress interfering sounds from other directions than the desired one. This build-in feature is due to

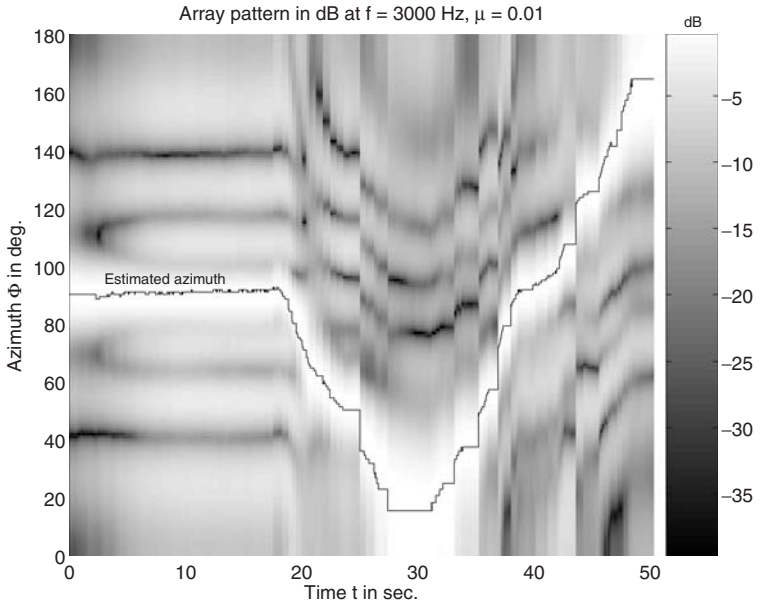

Fig. 4.18. Array pattern at $f = 3$ kHz of the automatically steered adaptive beamformer with superimposed estimated azimuth trace of a moving speaker.

the adaptive algorithm constraints by which the desired signal is emphasized. To illustrate this behavior, we show array patterns using random noise band-limited from 300 Hz to 6400 Hz as a desired source signal. The beamformer output signal power is calculated as a function of the noise signal direction. Typical results are shown in Fig. 4.19. Four different desired directions are given. The steady-state output power is computed after the settling period of the adaptive beamformer. A sharp main lobe can be observed, especially at desired direction $\Phi = 90°$. At $\Phi = 0°$ the array is less sensitive regarding changes in the desired direction. This behavior is common to broadband adaptive beamformers based on the constrained LMS algorithm because the optimization constraint is defined for a single desired direction only. A sharp main lobe is not a disadvantage of our adaptive beamformer because the desired direction is automatically adjusted using speaker tracking.

The entire system has been simulated using a MATLAB® program which can be downloaded from the authors home page.[4] An implementation written in the C programming language runs in real-time at 16 kHz sampling frequency on any modern PC equipped with an 8 channel analog input sound system (like Terratec® EWS88MT, M-Audio® Delta 1010, or RME® Hammerfall® DSP). With CPU clock frequencies at 2 GHz, 16 microphone channels can be processed in real-time at 16 kHz sampling frequency.

[4] `www.nt.tuwien.ac.at/dspgroup/gdobling.html`

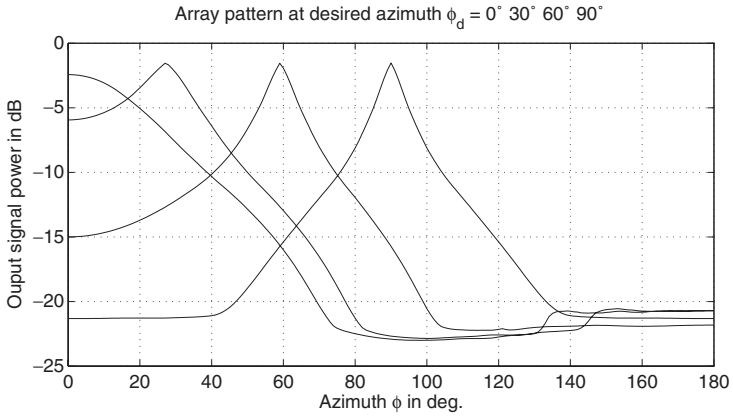

Fig. 4.19. Output signal power vs. azimuth of the adaptive array excited with random noise bandlimited from 300 Hz to 6400 Hz, and with four different desired directions.

4.7 Conclusions

We have presented an overview on different source localization techniques based on time-delay estimation using only two microphones. These algorithms are well suited for direction (azimuth) estimation and speaker tracking in real environments with moderate reverberation. The main purpose of source localization covered in this chapter is the application to speaker tracking with automatically steered microphone arrays. An efficient adaptive beamformer has been described in detail combining a multi-input overlap-add FFT filterbank, a constrained LMS algorithm, and a GCC-PHAT based source localization algorithm.

Acknowledgements

An industrial cooperation with AKG Acoustics Austria triggered my interest in acoustical beamforming and source localization. It also gave me the opportunity to consider both theoretical aspects and real-world problems in the context of microphone arrays. I particularly thank J. Granser, G. Stöbich, J. Schreiner, and G. Zach for their excellent contributions and valuable discussions.

References

[1] Brian C. J. Moore: *An Introduction to the Psychology of Hearing,* London, Great Britain: Academic Press, 2001.

[2] M. Brandstein, D. Ward (eds.): *Microphone Arrays – Signal Processing Techniques and Applications,* Berlin, Germany: Springer, 2001.

[3] S. L. Gay, J. Benesty (eds.): *Acoustic Signal Processing for Telecommunications,* Boston, MA: Kluwer, 2001.

[4] Y. Huang, J. Benesty (eds.): *Audio Signal Processing for Next-generation Multimedia Communication Systems,* Boston, MA: Kluwer, 2004.

[5] C. H. Knapp, G. C. Carter: The generalized correlation method for estimation of time delay, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-24**(4), 320–327, 1976.

[6] Chen Liu, et al.: Localization of multiple sound sources with two microphones, *J. Acoust. Soc. Am.*, **108**(4), 1888–1905, 2000.

[7] Jacob Benesty: Adaptive eigenvalue decomposition algorithm for passive acoustic source localization, *J. Acoust. Soc. Am.*, **107**(1), 384–391, 2000.

[8] D. Li, S. E. Levinson: A linear phase unwrapping method for binaural sound source localization on a robot, in *Proc. 2002 IEEE Conf. on Robotics and Automation*, 19–23, Washington, DC, USA, 2002.

[9] I. Potamitis, H. Chen, G. Tremoulis: Tracking of multiple moving speakers with multiple microphone arrays, *IEEE Trans. Speech Audio Signal Process.*, **T-SA-12**(5), 520–529, 2004.

[10] T. Gustafsson, B. D. Rao, M. Trivedi: Source localization in reverberant environments: modeling and statistical analysis, *IEEE Trans. Speech Audio Signal Process.*, **T-SA-11**(6), 791–803, 2003.

[11] M. Omologo, P. Svaizer: Use of the crosspower-spectrum phase in acoustic event location, *IEEE Trans. Speech Audio Process.*, **T-SA-5**(3), 288–292, 1997.

[12] Jens Blauert: *Spatial hearing - revised edition, the psychophysics of human sound localization,* Cambridge MA: The MIT Press, 1996.

[13] F. A. Reed, P. L. Feintuch, N. J. Bershad: Time delay estimation using the LMS adaptive filter – static behavior, *IEEE. Trans. Acoust. Speech Signal Process.*, **ASSP-29**(3), 561–571, 1981.

[14] E. A. Ferrara: Fast implementation of LMS adaptive filters, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-28**(4), 474–475, 1980.

[15] D. Mansour, A. H. Gray, Jr.: Unconstrained frequency-domain adaptive filter, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-30**(5), 726–734, 1982.

[16] O. L. Frost, III: An algorithm for linearly constrained adaptive array processing, *Proc. IEEE*, **60**(8), 926–935, 1972.

[17] C. Marro, Y. Mahieux, K. U. Simmer: Analysis of noise reduction and dereverberation techniques based on microphone arrays with postfiltering, *IEEE Trans. Speech Audio Process.*, **T-SA-6**(3), 240–259, 1998.

[18] K. U. Simmer, J. Bitzer, C. Marro: Post-filtering techniques, in M. Brandstein, D. Ward (eds.), *Microphone Arrays – Signal Processing Techniques and Applications*, 39–60, Berlin, Germany: Springer, 2001.

[19] I. A. McCowan, H. Bourlard: Microphone array post-filter based on noise field coherence, *IEEE Trans. Speech Audio Process.*, **T-SA-11**(6), 709–716, 2003.

[20] R. E. Crochiere, L. R. Rabiner: *Multirate Digital Signal Processing,* Englewood Cliffs, NJ: Prentice Hall, 1983.

[21] H. Cox, R. M. Zeskind, M. M. Owen: Robust adaptive beamforming, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-35**(10), 1365–1376, 1987.

[22] G. Stöbich: *Entwurf und Simulation eines adaptiven, zweidimensionalen Mikrofonarrays,* Vienna, Austria: Diploma Thesis, Vienna University of Technology, 2001 (in German).

[23] H. Teutsch, W. Kellermann: EB-ESPRIT: 2D localization of multiple wideband acoustic sources using eigen-beams, *Proc. ICASSP '05*, **3**, 89–92, Philadelphia, PA, USA, 2005.