Real-Time Implementation of a Particle Filter with Integrated Voice Activity Detector for Acoustic Speaker Tracking

Anders M. Johansson, Eric A. Lehmann and Sven Nordholm
Western Australian Telecommunications Research Institute
35 Stirling Highway, Perth WA 6009, Australia
E-mail: ajh@watri.org.au, Eric.Lehmann@watri.org.au, sven@watri.org.au

***Abstract*— In noisy and reverberant environments, the problem of acoustic source localisation and tracking (ASLT) using an array of microphones presents a number of challenging difficulties. One of the main issues when considering real-world situations involving human speakers is the temporally discontinuous nature of speech signals: the presence of silence gaps in the speech can easily misguide the tracking algorithm, even in practical environments with low to moderate noise and reverberation levels. This work focuses on a real-time implementation of the ASLT algorithm proposed in [1], which circumvents this problem by integrating measurements from a voice activity detector (VAD) within the tracking algorithm framework. The algorithm is here optimized for low computational complexity, and is implemented on a PC based real-time system. The resulting computational load is calculated and is presented along with real measurements of the true execution speed for the considered algorithm implementation. The results show that the algorithm is suitable for implementation in currently existing low-power embedded systems.**

I. Introduction

The concept of speaker tracking using an array of acoustic sensors has become an increasingly important field of research over the last few years [2–5]. Typical applications such as teleconferencing, automated multi-media capture, smart meeting rooms and lecture theaters, etc., are fast becoming an engineering reality. This in turns requires the development of increasingly sophisticated algorithms to deal efficiently with problems related to background noise and acoustic reverberation during the speech acquisition process. Furthermore, these algorithms must also be computationally efficient in order to be suitable for real-time implementation.

One of the major difficulties in a practical implementation of ASLT for speech-based applications lies in the nonstationary character of typical speech signals, with potentially significant silence periods existing between separate utterances. During such silence gaps, currently available ASLT methods will usually keep updating the source location estimates as if the speaker was still active. The algorithm is therefore likely to momentarily lose track of the true source position since the updates are then based solely on disturbance sources such as reverberation and background noise, whose influence might be quite significant in practice. Consequently, existing works on speaker tracking implicitly rely on the fact that silence periods in the speech signal remain relatively short [2–5].

The work presented in [1] deals with this specific issue by fusing VAD observations within the statistical framework of a sequential Monte Carlo algorithm (particle filter, PF). Simulation results of this algorithm, denoted PF-VAD, are provided in [1] on the basis of synthetic audio data generated with the image method [6], and in [7] using samples of real audio data recorded in a reverberant room. These simulations show that the newly proposed ASLT algorithm has the potential to drastically outperform more basic PF implementations that do not integrate VAD data, such as those presented in [3]. While these initial off-line simulations are useful to gauge the algorithm's ability to deal with the considered ASLT problem, they give little insight into how suitable the algorithm actually is for a practical, real-world implementation.

This paper presents a real-time implementation of the ASLT method presented in [1]. Below is a brief review of the theoretical concepts behind the algorithm (as it appears in [7]), followed by an in-depth description of the implementation including an analysis of the computational complexity. The The paper concludes with an example of the real-time tracking results obtained with the developed system for the case of a human speaker walking across a reverberant and noisy room.

II. Bayesian Filtering for Target Tracking

Consider an array of M acoustic sensors distributed at known locations in a reverberant environment. Assuming a single sound source, the problem consists in estimating the location of this "target" based on the signals $f_m(t)$, $m \in \{1, \dots, M\}$, provided by the array. It is further assumed that the sensor signals are sampled in time, and subsequently decomposed into a series of successive frames $k = 1, 2, \dots,$ of equal length L before being processed.

A. State-Space Filtering

Let $\mathbf{X}_k$ represent the state variable for time frame k, corresponding to the position and velocity of the target in the state space: $\mathbf{X}_k = [x_k \;\; y_k \;\; \dot{x}_k \;\; \dot{y}_k]^{\mathrm{T}}$. At any time step, each microphone in the array delivers a frame of audio signal which can be processed using some localization technique, such as steered beamforming (SBF). Let $\mathbf{Y}_k$ denote the observation variable (measurement), which here typically corresponds to the localization information resulting from the SBF processing of the audio signals. Using a Bayesian filtering approach and assuming Markovian dynamics, this system can be globally represented as follows:

$$\mathbf{X}_k = g(\mathbf{X}_{k-1}, \mathbf{u}_k)\,, \tag{1a}$$

$$\mathbf{Y}_k = h(\mathbf{X}_k, \mathbf{v}_k)\,, \tag{1b}$$

where $g(\cdot)$ and $h(\cdot)$ are possibly nonlinear functions, and $\mathbf{u}_k$ and $\mathbf{v}_k$ are possibly non-Gaussian noise variables. Ultimately, one would like to compute the so-called posterior probability density function (PDF) $p(\mathbf{X}_k|\mathbf{Y}_{1:k})$, where $\mathbf{Y}_{1:k} = \{\mathbf{Y}_1, \dots, \mathbf{Y}_k\}$ represents the concatenation of all measurements up to time k. This density contains all the statistical information available regarding the current condition of the state variable $\mathbf{X}_k$, and an estimate $\widehat{\mathbf{X}}_k$ of the state then follows, for instance, as the mean or the mode of $p(\mathbf{X}_k|\mathbf{Y}_{1:k})$.

B. Sequential Monte Carlo Approach

Particle filtering (PF) is an approximation technique that solves the above Bayesian filtering problem by representing the posterior

This work was supported by National ICT Australia (NICTA) and the Australian Research Council (ARC) under grant no. DP0451111.

density as a set of N samples of the state space $\mathbf{X}_k^{(n)}$ (particles) with associated weights $w_k^{(n)}$, $n \in \{1, \ldots, N\}$, see, e.g., [8]. The so-called bootstrap algorithm [9] is an attractive PF variant due to its simplicity and low computational demands. Assuming that the set of particles and weights $\{(\mathbf{X}_{k-1}^{(n)}, w_{k-1}^{(n)})\}_{n=1}^N$ is a discrete representation of the posterior density at time $k-1$, $p(\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1})$, and given the observation $\mathbf{Y}_k$ obtained at the current time k, the bootstrap PF algorithm forms a new set of particles and weights $\{(\mathbf{X}_k^{(n)}, w_k^{(n)})\}_{n=1}^N$, which is an approximate representation of the current posterior $p(\mathbf{X}_k|\mathbf{Y}_{1:k})$. An estimate $\hat{\boldsymbol{\ell}}_k$ of the source position for the current time step k can then be computed according to

$$\hat{\boldsymbol{\ell}}_k = \mathbb{E}\{\boldsymbol{\ell}_k\} \approx \sum_{n=1}^{N} w_k^{(n)} \boldsymbol{\ell}_k^{(n)} ,$$

where $\boldsymbol{\ell}_k^{(n)} = [x_k^{(n)} \;\; y_k^{(n)}]^{\mathrm{T}}$ corresponds to the location information in the n-th particle vector. A second output from the PF algorithm is a measure of the confidence level in the PF estimates, which can be obtained by computing the standard deviation of the particle set:

$$\varsigma_k = \sqrt{\sum_{n=1}^{N} w_k^{(n)} \left\| \boldsymbol{\ell}_k^{(n)} - \hat{\boldsymbol{\ell}}_k \right\|^2} ,$$

where $\|\cdot\|$ denotes the Euclidean norm. The parameter ς_k provides a direct assessment of how reliable the PF considers its current source position estimate to be.

III. PF for Acoustic Source Tracking

The bootstrap PF algorithm requires the definition of two important concepts [9]: the source dynamics (through the transition function $g(\cdot)$) and the so-called likelihood function $p(\mathbf{Y}_k|\mathbf{X}_k^{(n)})$, $n \in \{1, \ldots, N\}$.

A. Target Dynamics

In order to remain consistent with previous ASLT literature [3, 4], a Langevin process is used to model the dynamics equation (1a). This process is typically used to characterize' various types of stochastic motion, and it has proved to be a good choice for speaker tracking. With this model, the source motion in each of the Cartesian coordinates is assumed to be an independent first-order Markov process.

B. Likelihood Function

The SBF principle is used here as a basis for the derivation of the likelihood function. With $F_m(\omega) = \mathcal{F}\{f_m(t)\}$ the Fourier transform of the signal data from the m-th sensor, the output $\mathcal{P}(\boldsymbol{\ell})$ of a delay-and-sum beamformer steered to the location $\boldsymbol{\ell} = [x \;\; y]^{\mathrm{T}}$ is

$$\mathcal{P}(\boldsymbol{\ell}) = \int_{\Omega} \Big| \sum_{m=1}^{M} W_m(\omega)\, F_m(\omega)\, \mathrm{e}^{j\omega\|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c} \Big|^2 \,\mathrm{d}\omega , \qquad (2)$$

where $c = 343\,\mathrm{m/s}$ represents the propagation speed of acoustic waves, $\boldsymbol{\ell}_m = [x_m \;\; y_m]^{\mathrm{T}}$ is the known position of the m-th microphone, and Ω corresponds to the frequency range of interest, typically defined as $\Omega = \{\omega \,|\, 2\pi \cdot 300\,\mathrm{Hz} \leqslant \omega \leqslant 2\pi \cdot 3000\,\mathrm{Hz}\}$ for speech processing applications. The frequency weighting term $W_m(\cdot)$ is computed according to the PHAT (phase transform) weighting, i.e., $W_m(\omega) = 1/|F_m(\omega)|$.

In the PF-VAD implementation, an approach based on the concept of a "pseudo-likelihood" is adopted, as introduced previously in [3]. This concept relies on the idea that the SBF output $\mathcal{P}(\cdot)$ itself can be used as a measure of likelihood. For the n-th particle, the likelihood PDF is therefore defined as

$$p(\mathbf{Y}_k|\mathbf{X}_k^{(n)}) = q_0 \cdot \mathcal{U}(\boldsymbol{\ell}_k^{(n)}) + \gamma\,(1-q_0) \cdot \left[\mathcal{P}(\boldsymbol{\ell}_k^{(n)})\right]^r , \qquad (3)$$

where $\mathcal{U}(\cdot)$ is the uniform PDF defined over the considered room boundaries, q_0 is the prior probability that an SBF measurement might originate from clutter, and the nonlinear exponent r is used to help shape the SBF output to make it more amenable to source tracking [3]. The parameter γ is a normalization constant ensuring that the two PDFs in the mixture likelihood definition of (3) are properly scaled with respect to each other [1].

IV. Fusion of VAD Measurements

A. Voice Activity Detection

The voice activity detector (VAD) employed in [1] is based on the work presented by Davis *et al.* in [10]. The VAD relies on an estimate of the instantaneous signal-to-noise ratio (SNR) in the current signal frame. It assumes that the data recorded at the microphones is an additive combination of the clean speech signal and noise.

The scheme works on the basis of the average noise power spectral density, which is estimated during nonspeech periods. The estimated noise level, which is assumed to vary slowly in relation to the speech power, is then used during periods of speech activity to estimate the SNR from the observed signal. The assumption is that the speaker is active when the frequency-averaged SNR level is higher than a given threshold, which is set in such a way as to minimize' the occurrence of false alarms.

The spectral resolution, i.e., the number D of considered subbands, is set to a value much lower than the frame length L in order to reduce the variance of the signal power estimates. The specific application considered here also makes it possible to further reduce the variance by averaging over multiple microphones.

B. VAD Fusion

The output of the VAD can be linked to the probability q_0 in (3) in an obvious manner. The probability $1-q_0$ corresponds to the likelihood of the acoustic source being active (non-clutter SBF measurement), an estimate of which is delivered by the VAD. Therefore, instead of setting the variable q_0 to a constant value in the design of the algorithm as done in [3, 4], the following time-varying definition of q_0 is used: $q_0(k) = 1 - \alpha(k)$, with $\alpha(k) \in [0, 1]$ the soft-decision output from the VAD algorithm (where 1 denotes speech and 0 nonspeech). In the current implementation, $\alpha(k)$ corresponds to the estimated speech signal level, derived from the SNR and noise power estimates delivered by the VAD.

V. Implementation

The algorithm is implemented in software executed on a standard PC. The implementation uses single precision floating point arithmetic and is written in the C programming language. The microphone array is connected to the PC using a multi-channel analog input/output (I/O) card.

A. Hardware Configuration

The PC is a standard IBM-PC, equipped with a 1.8 GHz AMD Athlon processor and 512 MB of memory. The operating system on the PC is Debian GNU/Linux version 3.0. The kernel version is 2.6.8, and is compiled with preemptive multitasking.

The microphone array consists of eight elements which are mounted on a metal fixture in an octagonal pattern (see Figure 2).

TABLE I
FLOATING POINT OPERATIONS PER DATA FRAME.

Operation	FFT	VAD	PF
Real divisions		$4D + 10$	ML_r
Real additions		$25D + L(M + 1)/2$	$ML_r + N(2L_r + 2MR + 4R + 2)$
Real multiplications		$9D + 3L(M + 1)/2$	$2ML_r + N(2L_r + MR + 5M + 4R + 8)$
Complex to real multiplications			ML_r
Complex additions	$ML/2\log_2(L/2)$		ML_rN
Complex multiplications	$ML/2\log_2(L/2)$		$2ML_rN$
Real square root		$D + 4$	$M(L_r + 2N)$
Complex exponential			$N(2M + 1)$
Pseudo random number			$N(R + 2)$

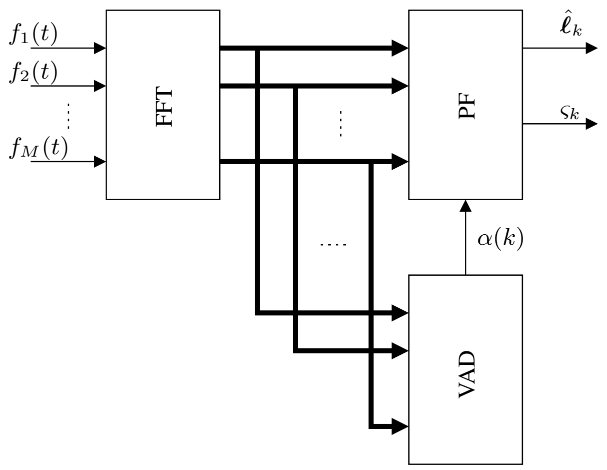

Fig. 1. Block diagram showing the integration of the PF and the VAD.

The microphones are connected to a preamplifier which in turn is connected to the I/O card. The microphone elements, model 2541/PRM902, and the preamplifier, model 2210, are from Larson Davis. The I/O card has 24-bit analog-to-digital converters with built-in anti-aliasing filters and is operated at a sample frequency of 16 kHz. The card is an M-Audio Delta-1010LT.

B. Software

Both the beamformer in the PF and the power spectrum estimation in the VAD can be implemented using an FFT. This makes it possible to further integrate the two sub-algorithms and thereby reduce the computational complexity, see Figure 1. The delay-and-sum beamformer in (2) is the most critical part in terms of computational complexity, since it has to be executed N times per frame for the computation of the likelihood function in (3). Here, it is implemented according to

$$\mathcal{P}(\boldsymbol{\ell}) = \sum_{l=L_0}^{L_1} \Big| \sum_{m=1}^{M} G_m(\omega_l)\, \mathrm{e}^{j\omega_l \|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c} \Big|^2, \tag{4}$$

where $\omega_l = 2\pi l/L$, L_0 and L_1 define the frequency range of interest, and $G_m(\omega) = W_m(\omega) \cdot F_m(\omega)$. The computational complexity in the inner loop is reduced by rewriting the complex exponential in (4) according to

$$\begin{aligned} \mathrm{e}^{j\omega_{l+1}\|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c} &= \mathrm{e}^{j(\omega_l+\omega_1)\|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c} \\ &= \mathrm{e}^{j\omega_l\|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c} \cdot \mathrm{e}^{j\omega_1\|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c}. \end{aligned} \tag{5}$$

Thus, the exponential term for frequency band l in (4) can be computed recursively using the corresponding term for band $l - 1$ multiplied by the constant $\mathrm{e}^{j\omega_1\|\boldsymbol{\ell}-\boldsymbol{\ell}_m\|/c}$, which is calculated only once for each particle. Experiments showed that this code optimization reduced the computational complexity by a factor 10.

TABLE II
CLOCK CYCLES PER INPUT SAMPLE FOR EACH SUB-ALGORITHM. THE RIGHT MOST COLUMN LISTS THE NUMBER OF FLOATING POINT OPERATIONS PER SECOND (FLOPS) AT 16 KHZ SAMPLE FREQUENCY.

	FFT	VAD	PF	Total	FLOPS
Theoretical	256	21.2	3.34k	3.62k	57.9M
Measured	225	53.7	4.19k	4.47k	71.5M

The computational load for the different parts of the implementation is found in Table I, where $L_r = L_1 - L_0 + 1$, and R denotes the considered number of room dimensions (typically 2 or 3, for either a two or three-dimensional problem definition). The table is calculated considering a worst case scenario (e.g., when considering the execution of conditional algorithm sections). The computations required for the data fusion is included in the computational complexity for the VAD. The table shows that the bulk of the computations is in the particle filter, signifying that the new algorithm is not much more computationally complex compared to a traditional PF.

The actual number of clock cycles spent in each sub-algorithm has been measured by reading the time stamp counter (TSC) in the CPU. The results are presented in Table II along with theoretical number of floating point operations. The theoretical values have been obtained by inserting actual numerical values in the equations given in Table I. The discrepancy between the measured and real values lies in the function calls overhead, integer operations, pipeline utilization, cache faults and a large number of branch instructions for the VAD. The parameters used for the implementation are $N = 100$, $D = 8$, $L = 512$, $R = 2$, $L_r = 86$ and $M = 8$. The total CPU usage for the algorithm process during execution was around 5%, which is consistent with the 71.5 MFLOPS computational load plus overhead for data acquisition. The results shows that the considered algorithm has a rather low overall computational complexity and can run comfortably on a system made up of widely available and low-cost hardware.

VI. EXPERIMENTAL RESULTS

A. Environmental Setup

An array of $M = 8$ omnidirectional microphones was set up at a constant height of 1.51 m in a room with dimensions 3.5 m × 3.1 m × 2.2 m, in a octagonal pattern with one sensor pair on each wall, as shown in Figure 2. The distance between the sensors in each pair is 0.8 m, and the total area spanned by the array is $2.52\,\mathrm{m} \times 2.52\,\mathrm{m}$. The acoustic of the room was controlled by padding the walls of the room with sound absorbing material to achieve a reverberation time $T_{60} \approx 270\,\mathrm{ms}$ (frequency-averaged up to $24\,\mathrm{kHz}$).

B. Source Position Measurements

Ground-truth measurements of the real speaker trajectory over time were extracted from the microphone signals using the method described in [7], which works on the basis of a high-accuracy beamformer scanning the considered enclosure. The microphone signals are split into successive frames of $32\,\mathrm{ms}$ of data, with a 50% overlapping factor. Each frame is processed using the SBF formula in (2), at first computed on a relatively coarse grid across the entire search space. A coarse estimate of the current source position is obtained as the location maximizing this SBF output, this estimate is then refined by considering a high-resolution grid (uniform $1\,\mathrm{mm}$ spacing between grid points) centered around the region of interest. An approximate knowledge of the overall source path across the room, combined with the use of some voice activity detection scheme, allows the easy discrimination of outliers and yields a series of two-dimensional location data points vs. time. Finally, a polynomial approximation is fitted to the SBF localization data in order to obtain an estimate of the true source trajectory over the entire audio sample length.

C. Real-Time Tracking

A male speaker, moving randomly across the room while uttering a series of sentences, was tracked using the implemented real-time algorithm. Background noise was added with a signal-to-noise ratio of approximately $20\,\mathrm{dB}$ by means of a number of loudspeakers emitting white Gaussian noise.

The software program saves the audio data along with the estimated source position to disk during execution. A typical tracking result obtained with the above setup is depicted in Figure 2, along with the sound picked up by one of the microphones. The plot shows that a successful and accurate tracking is achieved during periods of speech activity. A typical example of temporary track loss resulting from a nonspeech period can be observed in Figure 2 towards the end of the simulation, where the estimates slightly deviate from the true source trajectory. The considered algorithm is however able to successfully resume tracking when the speaker becomes active again.

D. Tracking Movies

In order to test the implemented algorithm under various conditions, a series of experiments were conducted with different speakers and source trajectories. The data recorded by the real-time system described in this paper was used to generate multi-media files of the tracking results. These movies demonstrate the effectiveness of the real-time implementation with a clarity that would be difficult to match with figures and/or tables. The movies also include the standard deviation output ς_k from the algorithm, drawn as an ellipse around the estimated source position. These files can be downloaded from `http://www.watri.org.au/~ajh/research/`.

VII. Conclusions

The ASLT algorithm presented in [1] has been implemented and evaluated in a real-time system. The results from the evaluation

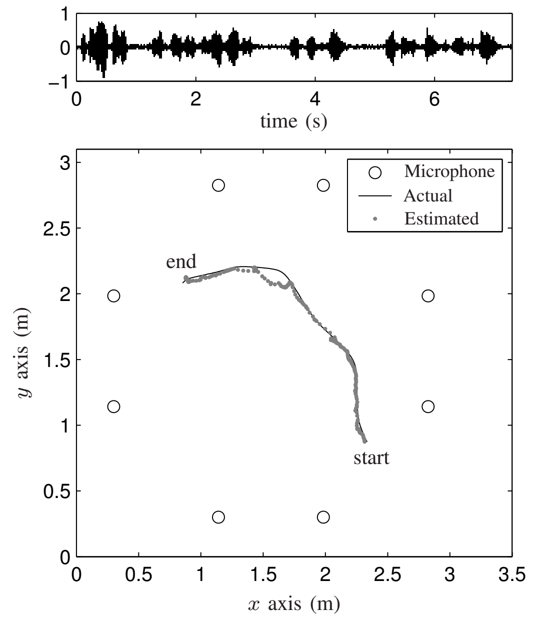

Fig. 2. Real-time tracking of human speaker walking in noisy room. The top plot shows an example of signal recorded by one of the microphones.

show that the algorithm is suitable for real-world implementations. The computational complexity is low enough for a real-time processing on low-power embedded systems using currently existing hardware, and the performance is good enough to successfully track a moving speaker in an acoustically adverse environment.

References

[1] E. Lehmann and A. Johansson, "Particle filter with integrated voice activity detection for acoustic source tracking," to appear in *EURASIP J. Applied Signal Processing*.

[2] T. Dvorkind and S. Gannot, "Speaker localization exploiting spatial-temporal information," in *Proc. IWAENC*, Kyoto, Japan, Sept. 2003, pp. 295–298.

[3] D. Ward, E. Lehmann, and R. Williamson, "Particle filtering algorithms for tracking an acoustic source in a reverberant environment," *IEEE Trans. Speech Audio Processing*, vol. 11, no. 6, pp. 826–836, Nov. 2003.

[4] J. Vermaak and A. Blake, "Nonlinear filtering for speaker tracking in noisy and reverberant environments," in *Proc. IEEE ICASSP*, vol. 5, Salt Lake City, USA, May 2001, pp. 3021–3024.

[5] I. Potamitis, H. Chen, and G. Tremoulis, "Tracking of multiple moving speakers with multiple microphone arrays," *IEEE Trans. Speech Audio Processing*, vol. 12, no. 5, pp. 520–529, 2004.

[6] J. Allen and D. Berkeley, "Image method for efficiently simulating small-room acoustics," *J. Acoust. Soc. Am.*, vol. 65, no. 4, pp. 943–950, Apr. 1979.

[7] E. Lehmann and A. Johansson, "Experimental performance assessment of a particle filter with voice activity data fusion for acoustic speaker tracking," in *Proc. IEEE Nordic Signal Processing Symposium*, Reykjavik, Iceland, June 2006.

[8] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[9] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc. F*, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[10] A. Davis, S. Nordholm, and R. Togneri, "Statistical voice activity detection using low-variance spectrum estimation and an adaptive threshold," *IEEE Trans. Speech Audio Processing*, vol. 14, no. 2, pp. 412–424, Mar. 2006.