Likelihood-Maximizing Beamforming for Robust Hands-Free Speech Recognition

Michael L. Seltzer, *Member, IEEE*, Bhiksha Raj, *Member, IEEE*, and Richard M. Stern, *Member, IEEE*

***Abstract*—Speech recognition performance degrades significantly in distant-talking environments, where the speech signals can be severely distorted by additive noise and reverberation. In such environments, the use of microphone arrays has been proposed as a means of improving the quality of captured speech signals. Currently, microphone-array-based speech recognition is performed in two independent stages: array processing and then recognition. Array processing algorithms, designed for signal enhancement, are applied in order to reduce the distortion in the speech waveform prior to feature extraction and recognition. This approach assumes that improving the quality of the speech waveform will necessarily result in improved recognition performance and ignores the manner in which speech recognition systems operate. In this paper a new approach to microphone-array processing is proposed in which the goal of the array processing is not to generate an enhanced output waveform but rather to generate a sequence of features which maximizes the likelihood of generating the correct hypothesis. In this approach, called likelihood-maximizing beamforming, information from the speech recognition system itself is used to optimize a filter-and-sum beamformer. Speech recognition experiments performed in a real distant-talking environment confirm the efficacy of the proposed approach.**

***Index Terms*—Adaptive filtering, beamforming, distant-talking environments, microphone array processing, robust speech recognition.**

I. Introduction

STATE-OF-THE-ART automatic speech recognition (ASR) systems are known to perform reasonably well when the speech signals are captured using a close-talking microphone worn near the mouth of the speaker. However, there are many environments where the use of such a microphone is undesirable for reasons of safety or convenience. In these settings, such as vehicles, meeting rooms, and information kiosks, a fixed microphone can be placed at some distance from the user. Unfortunately, as the distance between the user and the microphone grows, the speech signal becomes increasingly degraded by the effects of additive noise and reverberation, which in turn degrades speech recognition accuracy. The use of an array of microphones, rather than a single microphone, can compensate for this distortion in these distant-talking environments by providing spatial filtering to the sound field, effectively focusing attention in a desired direction.

Many microphone array processing techniques which improve the quality of the output signal and increase the signal-to-noise ratio (SNR) have been proposed in research. The simplest and most common method is called delay-and-sum beamforming [1]. In this approach, the signals received by the microphones in the array are time-aligned with respect to each other in order to adjust for the path-length differences between the speech source and each of the microphones. The time-aligned signals are then weighted and added together. Any interfering signals that are not coincident with the speech source remain misaligned and are thus attenuated when the signals are combined. A natural extension of delay-and-sum beamforming is filter-and-sum beamforming, in which each microphone signal has an associated filter and the captured signals are filtered before they are combined.

In adaptive beamforming schemes, such as the generalized sidelobe canceller (GSC) [2], the array parameters are updated on a sample-by-sample or frame-by-frame basis according to a specified criterion. Typical criteria used in adaptive beamforming include a distortionless response in the look direction and/or the minimization of the energy from all directions not considered the look direction. In some cases, the array parameters can be calibrated to a particular environment or user prior to use, e.g., [3].

Adaptive filtering methods such as these generally assume that the target and jammer signals are uncorrelated. When this assumption is violated, as is the case for speech in a reverberant environment, the methods suffer from signal cancellation because reflected copies of the target signal appear as unwanted jammer signals. While various methods have been proposed to mitigate this undesirable effect, e.g., [4] and [5], signal cancellation nevertheless still arises in reverberant environments. As a result, conventional adaptive filtering approaches have not gained widespread acceptance for most speech recognition applications.

A great deal of recent research has focused specifically on compensating for the effects of reverberation. One obvious way to perform dereverberation is to invert the room impulse response. However, methods based on this approach have largely been unsuccessful because room impulse responses are generally nonminimum phase which causes instability in the in-

Manuscript received September 1, 2003; revised March 6, 2004. This work was supported by the Space and Naval Warfare Systems Center, San Diego, CA, under Grant No. N66001-99-1-8905. The work of M. L. Seltzer was also supported by a Microsoft Research Graduate Fellowship. The content of the information in this publication does not necessarily reflect the position or the policy of the U. S. Government, and no official endorsement should be inferred. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Man Mohan Sondi.

M. L. Seltzer is with the Microsoft Research, Speech Technology Group, Redmond, WA 98052 USA (e-mail: mseltzer@microsoft.com).

B. Raj is with the Mitsubishi Electric Research, Cambridge Research Laboratory, Cambridge, MA 02139 USA (e-mail: bhiksha@merl.com).

R. M. Stern is with the Department of Electrical and Computer Engineering and School of Computer Science, Carnegic Mellon University, Pittsburgh, PA 15213 USA (e-mail: rms@cs.cmu.edu).

Digital Object Identifier 10.1109/TSA.2004.832988

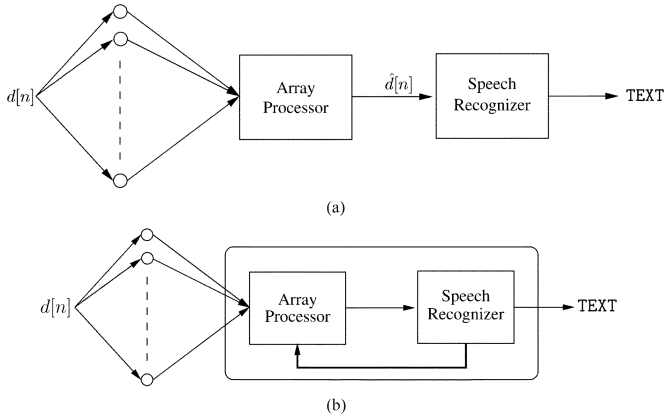


Fig. 1. (a) Conventional architecture used for speech recognition with a microphone array front-end. The objective of the array processor is to estimate the clean *waveform*. (b) An architecture for array processing optimized for speech recognition. The array processor and the speech recognizer are fully connected, allowing information from the recognizer to be used in the array processing. Note that the system no longer attempts to estimate the clean waveform.

verse filters [6]. Rather than performing deconvolution, some researchers take a matched filter approach to dereverberation, e.g., [7], [8]. While there are theoretical benefits to such an approach in terms of improved SNR, matched filtering has been shown to provide only minimal improvement in speech recognition accuracy over conventional delay-and-sum processing, even if the room impulse responses are known *a priori* [9].

All of these microphone array processing methods were designed for signal enhancement, and as such, process incoming signals according to various signal-level criteria, e.g., minimizing the signal error, maximizing the SNR, or improving the perceptual quality as judged by human listeners. Conventional microphone-array-based speech recognition is performed by utilizing one of these algorithms to generate the best output waveform possible, which then gets treated as a single-channel input to a recognition system. This approach, shown in Fig. 1(a), implicitly assumes that generating a higher quality output waveform will necessarily result in improved recognition performance. By making such an assumption, the manner in which speech recognition systems operate is ignored.

A speech recognition system does not interpret waveform-level information directly. It is a statistical pattern classifier that operates on a sequence of features derived from the waveform. We believe that this discrepancy between the waveform-based objective criteria used by conventional array processing algorithms and the feature-based objective criteria used by speech recognition systems is the key reason why sophisticated array processing methods fail to produce significant improvements in recognition accuracy over far simpler methods such as delay-and-sum beamforming. Speech recognition systems generate hypotheses by finding the word string that has the maximum-likelihood of generating the observed sequence of feature vectors, as measured by statistical models of speech sound units. Therefore, an array processing scheme can only be expected to improve recognition performance if it generates a sequence of features which maximizes, or at least increases, the likelihood of the correct transcription, relative to other hypotheses.

In this paper, we present a new array processing algorithm called *likelihood-maximizing beamforming* (LIMABEAM), in which the microphone array processing problem is recast as one of finding the set of array parameters that maximizes the likelihood of the correct recognition hypothesis. The array processor and the speech recognizer are no longer considered two independent entities cascaded together, but rather two interconnected components of a single system, with the common goal of improved speech recognition accuracy, as shown in Fig. 1(b). In LIMABEAM, the manner in which speech recognition systems process incoming speech is explicitly considered and pertinent information from the recognition engine itself is used to optimize the parameters of a filter-and-sum beamformer.

LIMABEAM has several advantages over current array processing methods. First, by incorporating the statistical models of the recognizer into the array processing stage, we ensure that the processing enhances those signal components important for recognition accuracy without undue emphasis on less important components. Second, in contrast to conventional adaptive filtering methods, no assumptions about the interfering signals are made. Third, the proposed approach requires no *a priori* knowledge of the room configuration, array geometry, or source-to-sensor room impulse responses. These properties enable us to overcome the drawbacks of previously-proposed array processing methods and achieve better recognition accuracy in distant-talking environments.

The remainder of this paper is organized as follows. In Section II, filter-and-sum beamforming is briefly reviewed. The LIMABEAM approach to microphone-array-based speech recognition is then described in detail in Section III. In Section IV, two implementations of LIMABEAM are presented, one for use in situations in which the environmental conditions are stationary or slowly varying and one for use in time-varying environments. The performance of these two algorithms is evaluated in Section V through a series of experiments performed using a microphone-array-equipped personal digital assistant (PDA). Some additional considerations for these algorithms are presented in Section VI. Finally, we present a summary of this work and some conclusions in Section VII.

II. Filter-and-Sum Beamforming

In this paper, we assume that filter-and-sum array processing can effectively compensate for the distortion induced by additive noise and reverberation. Assuming the filters have a finite impulse response (FIR), filter-and-sum processing is expressed mathematically as

$$y[n] = \sum_{m=0}^{M-1} \sum_{p=0}^{P-1} h_m[p] x_m[n - p - \tau_m] \tag{1}$$

where $h_m[p]$ is the pth tap of the filter associated with microphone m, $x_m[n]$ is the signal received by microphone m, τ_m is the steering delay induced in the signal received by microphone m to align it to the other array channels, and $y[n]$ is the output signal generated by the processing. M is the number of microphones in the array and P is the length of the FIR filters.

For notational convenience, we define $\boldsymbol{\xi}$ to be the vector of all filter coefficients for all microphones, as

$$\boldsymbol{\xi} = [h_0[0], h_0[1], \ldots, h_{M-1}[P-2], h_{M-1}[P-1]]^T. \quad (2)$$

III. Likelihood-Maximizing Beamforming (LIMABEAM)

Conventionally, parameters of a filter-and-sum beamformer are chosen according to criteria designed according to the notion of a *desired signal*. In contrast, we consider the output waveform $y[n]$ to be incidental and seek the filter parameters that optimize recognition accuracy. Therefore, we forgo the notion of a desired signal, and instead focus on a *desired hypothesis*. In order to so, we must consider both 1) the manner in which speech is input to the recognition system, i.e., the feature extraction process, and 2) the manner in which these features are processed by the recognizer in order to generate a hypothesis.

Speech recognition systems operate by finding the word string $\boldsymbol{w}$ most likely to generate the observed sequence of feature vectors $\mathcal{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_T\}$, as measured by the statistical models of the recognition system. When the speech is captured by a microphone array, the feature vectors are a function of both the incoming speech and the array processing parameters. Recognition hypotheses are generated according to Bayes optimal classification as

$$\hat{\boldsymbol{w}} = \underset{w}{\operatorname{argmax}}\, P(\mathcal{Z}(\boldsymbol{\xi})|\boldsymbol{w})P(\boldsymbol{w}) \quad (3)$$

where the dependence of the feature vectors $\mathcal{Z}$ on the array processing parameters $\boldsymbol{\xi}$ is explicitly shown. The acoustic score $P(\mathcal{Z}(\boldsymbol{\xi})|\boldsymbol{w})$ is computed using the statistical models of the recognizer and the language score $P(\boldsymbol{w})$ is computed from a language model.

Our goal is to find the parameter vector $\boldsymbol{\xi}$ for optimal recognition performance. One logical approach to doing so is to choose the array parameters that maximize the likelihood of the correct transcription of the utterance that was spoken. This will increase the difference between the likelihood score of the correct transcription and the scores of competing incorrect hypotheses, and thus, increase the probability that the correct transcription will be hypothesized.

For the time being, let us assume that the correct transcription of the utterance, which we notate as $\boldsymbol{w}_C$, is known. We can then maximize (3) for the array parameters $\boldsymbol{\xi}$. Because the transcription is assumed to be known *a priori*, the language score $P(\boldsymbol{w}_C)$ can be neglected. The maximum-likelihood (ML) estimate of the array parameters can now be defined as the vector that maximizes the acoustic log-likelihood of the given sequence of words, expressed as

$$\hat{\boldsymbol{\xi}} = \underset{\boldsymbol{\xi}}{\operatorname{argmax}} \log\left(P(\mathcal{Z}(\boldsymbol{\xi})|\boldsymbol{w}_C)\right). \quad (4)$$

In an HMM-based speech recognition system, the acoustic likelihood $P(\mathcal{Z}(\boldsymbol{\xi})|\boldsymbol{w}_C)$ is computed as the total likelihood of all possible state sequences through the HMM for the sequence of words in the transcription $\boldsymbol{w}_C$. However, many of these sequences are highly unlikely. For computational efficiency, we assume that the likelihood of a given transcription is largely represented by the single most likely HMM state sequence. If $\mathcal{S}_C$ represents the set of all possible state sequences through this HMM and $\boldsymbol{s}$ represents one such state sequence, then the ML estimate of $\boldsymbol{\xi}$ can be written as (5), shown at the bottom of the page.

According to (5), in order to find $\hat{\boldsymbol{\xi}}$, the likelihood of the correct transcription must be *jointly optimized* with respect to both the array parameters and the state sequence. This joint optimization can be performed by alternately optimizing the state sequence and the array processing parameters in an iterative manner.

A. Optimizing the State Sequence

Given a set of array parameters $\boldsymbol{\xi}$, the speech can be processed by the array and a sequence of feature vectors $\mathcal{Z}(\boldsymbol{\xi})$ produced. Using the features vectors and the transcription $\boldsymbol{w}_C$, we want to find the state sequence $\hat{\boldsymbol{s}} = \{\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_T\}$ [see (6), shown at the bottom of the page]. This state sequence $\hat{\boldsymbol{s}}$ can be easily determined by forced alignment using the Viterbi algorithm [10].

B. Optimizing the Array Parameters

Given a state sequence, $\hat{\boldsymbol{s}}$, we are interested in finding $\hat{\boldsymbol{\xi}}$ such that

$$\hat{\boldsymbol{\xi}} = \underset{\boldsymbol{\xi}}{\operatorname{argmax}} \sum_i \log(P(\boldsymbol{z}_i(\boldsymbol{\xi})|\hat{s}_i)). \quad (7)$$

This acoustic likelihood expression cannot be directly maximized with respect to the array parameters $\boldsymbol{\xi}$ for two reasons. First, the state distributions used in most HMM-based speech recognition systems are complicated density functions, i.e., mixtures of Gaussians. Second, the acoustic likelihood of an utterance and the parameter vector $\boldsymbol{\xi}$ are related through a series

$$\hat{\boldsymbol{\xi}} = \underset{\boldsymbol{\xi}, \boldsymbol{s} \in \mathcal{S}_C}{\operatorname{argmax}} \left\{ \sum_i \log(P(\boldsymbol{z}_i(\boldsymbol{\xi})|s_i)) + \sum_i \log(P(s_i|s_{i-1}, \boldsymbol{w}_C)) \right\}. \quad (5)$$

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s} \in \mathcal{S}_C}{\operatorname{argmax}} \left\{ \sum_i \log(P(\boldsymbol{z}_i(\boldsymbol{\xi})|s_i)) + \sum_i \log(P(s_i|s_{i-1}, \boldsymbol{w}_C)) \right\}. \quad (6)$$

of linear and nonlinear mathematical operations performed to convert a waveform into a sequence of feature vectors. Therefore, for a given HMM state sequence, no closed-form solution for the optimal value of $\boldsymbol{\xi}$ exists. As a result, nonlinear optimization methods must be used.

We employ a gradient-based approach to finding the optimal value of $\boldsymbol{\xi}$. For convenience, we define $\mathcal{L}(\boldsymbol{\xi})$ to be the total log likelihood of the observation vectors given an HMM state sequence. Thus

$$\mathcal{L}(\boldsymbol{\xi}) = \sum_i \log(P(\boldsymbol{z}_i(\boldsymbol{\xi})|s_i)). \tag{8}$$

Using the definition of $\boldsymbol{\xi}$ given by (2), we define the gradient vector $\nabla_{\boldsymbol{\xi}}\mathcal{L}(\boldsymbol{\xi})$ as

$$\nabla_{\boldsymbol{\xi}}\mathcal{L}(\boldsymbol{\xi}) = \left[\frac{\partial\mathcal{L}(\boldsymbol{\xi})}{\partial h_0[0]}, \frac{\partial\mathcal{L}(\boldsymbol{\xi})}{\partial h_0[1]}, \ldots, \frac{\partial\mathcal{L}(\boldsymbol{\xi})}{\partial h_{M-1}[P-1]}\right]^T. \tag{9}$$

Clearly, the computation of the gradient vector is dependent on the form of the HMM state distributions used by the recognition system and the features used for recognition. In Section III-C and D, we derive the gradient expressions when the state distributions are modeled as Gaussian distributions or mixtures of Gaussians. In both cases, the features are assumed to be mel frequency cepstral coefficients (MFCC) or log mel spectra.

1) Gaussian State Output Distributions: We now derive the expression for $\nabla_{\boldsymbol{\xi}}\mathcal{L}(\boldsymbol{\xi})$ for the case where the HMM state distributions are multivariate Gaussian distributions with diagonal covariance matrices. If we define $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ to be the mean vector and covariance matrix, respectively, of the pdf of the most likely HMM state at frame i, the total log likelihood for an utterance can be expressed as

$$\mathcal{L}(\boldsymbol{\xi}) = \sum_i \left\{-\frac{1}{2}\left(\boldsymbol{z}_i(\boldsymbol{\xi}) - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{z}_i(\boldsymbol{\xi}) - \boldsymbol{\mu}_i\right) + \kappa_i\right\} \tag{10}$$

where κ_i is a normalizing constant. Using the chain rule, the gradient of $\mathcal{L}(\boldsymbol{\xi})$ with respect to $\boldsymbol{\xi}$ can be expressed as

$$\nabla_{\boldsymbol{\xi}}\mathcal{L}(\boldsymbol{\xi}) = -\sum_i \frac{\partial \boldsymbol{z}_i(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{z}_i(\boldsymbol{\xi}) - \boldsymbol{\mu}_i\right) \tag{11}$$

where $\partial \boldsymbol{z}_i(\boldsymbol{\xi})/\partial\boldsymbol{\xi}$ is the Jacobian matrix, composed of the partial derivatives of each element of the feature vector at frame i with respect to each of the array parameters. The Jacobian is of dimension $MP \times L$ where M is the number of microphones, P is the number of parameters per microphone, and L is the dimension of the feature vector.

It can be shown that for log mel spectral feature vectors, the elements of the Jacobian matrix can be expressed as

$$\frac{\partial z_i^l}{\partial h_m[p]} = 2\frac{1}{M_i^l}\sum_{k=0}^{N/2} V^l[k]\Re\left(X_i^{mp}[k]Y_i^*[k]\right) \tag{12}$$

where $Y_i[k]$ is the discrete fourier transform (DFT) of frame i of the output signal $y[n]$, $X_i^{mp}[k]$ is the DFT of the signal captured by microphone m, beginning p samples prior to the start of frame i, $V^l[k]$ is the value of the lth mel filter applied to DFT bin k, and M_i^l is the lth mel spectral component in frame i. The size of the DFT is N and * denotes complex conjugation. Note that in (12), we have assumed that time-delay compensation (TDC) has already been performed and that the microphone signals $x_m[n]$ have already been time-aligned.

If optimization of the filter parameters is performed using MFCC features rather than log mel spectra, (12) must be modified slightly to account for the additional discrete cosine transform (DCT) operation. The full derivation of the Jacobian matrix for log mel spectral or cepstral features can be found in [11].

2) Mixture of Gaussians State Output Distributions: Most state-of-the-art recognizers do not model the state output distributions as single Gaussians but rather as mixtures of Gaussians. It can be shown [11] that when the HMM state distributions are modeled as mixtures of Gaussians, the gradient expression can be expressed as

$$\nabla_{\boldsymbol{\xi}}\mathcal{L}(\boldsymbol{\xi}) = -\sum_i \sum_{k=1}^{K} \gamma_{ik}(\boldsymbol{\xi}) \frac{\partial \boldsymbol{z}_i(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} \boldsymbol{\Sigma}_{ik}^{-1}\left(\boldsymbol{z}_i(\boldsymbol{\xi}) - \boldsymbol{\mu}_{ik}\right) \tag{13}$$

where $\gamma_{ik}(\boldsymbol{\xi})$ represents the *a posteriori* probability of the kth mixture component of state s_i, given $\boldsymbol{z}_i(\boldsymbol{\xi})$.

Comparing (11) and (13), it is clear that the gradient expression in the Gaussian mixture case is simply a weighted sum of the gradients of each of the Gaussian components in the mixture, where the weight on each mixture component represents its *a posteriori* probability of generating the observed feature vector.

C. Optimizing Log Mel Spectra Versus Cepstra

Array parameter optimization is performed in the log mel spectral domain, rather than the cepstral domain. Because the log mel spectra are derived from the energy using a series of triangular weighting functions of unit area, all components of the vectors have approximately the same magnitude. In contrast, the magnitude of cepstral coefficients decreases significantly with increasing cepstral order. When there is a large disparity in the magnitudes of the components of a vector, the larger components dominate the objective function and tend to be optimized at the expense of smaller components in gradient-descent-based optimization methods. Using log mel spectra avoids this potential problem.

In order to perform the array parameter optimization in the log mel spectral domain but still perform decoding using mel frequency cepstral coefficients (MFCC), we employ a parallel set of HMMs trained on log mel spectra, rather than cepstra. To obtain parallel models, we employed the statistical re-estimation (STAR) algorithm [12], which ensures that the two sets of models have identical frame-to-state alignments.

These parallel log mel spectral models were trained without feature mean normalization, since mean normalization is not incorporated into the optimization framework (we will revisit this issue in Section VI).

D. Gradient-Based Array Parameter Optimization

Using the gradient vector defined in either (11) or (13), the array parameters can be optimized using conventional gradient descent [13]. However, improved convergence performance can

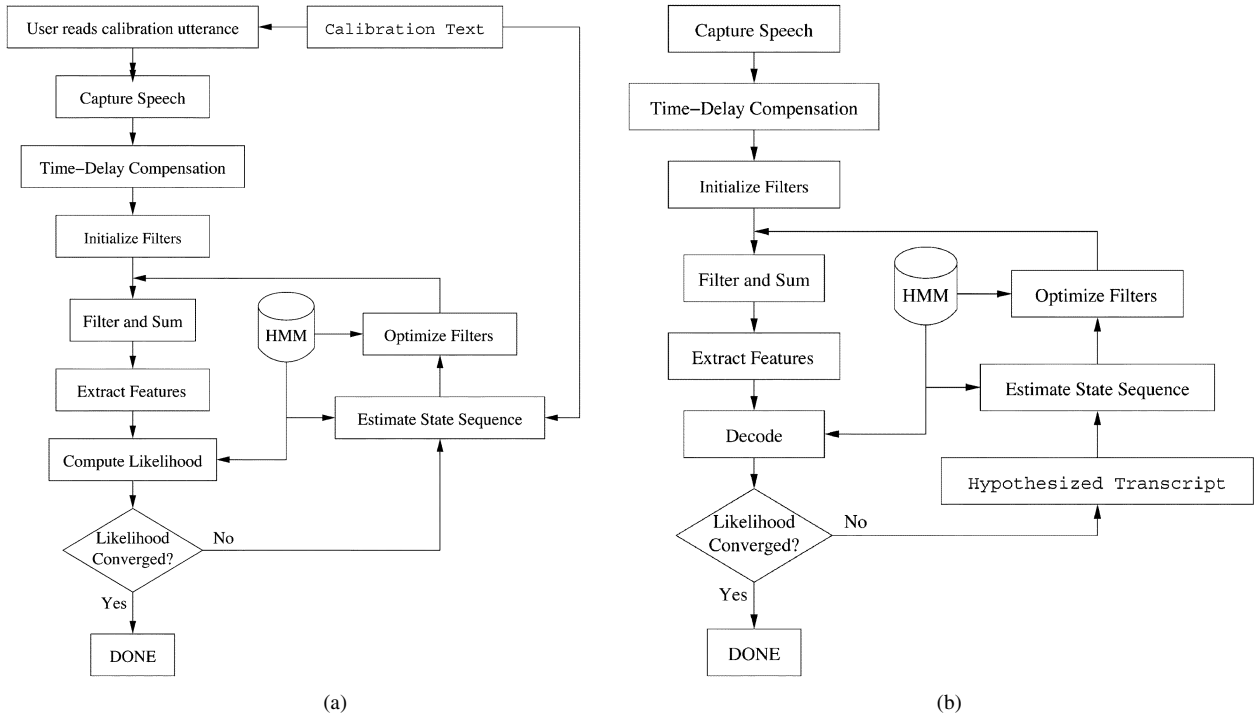


Fig. 2. Flowcharts of (a) Calibrated LIMABEAM and (b) Unsupervised LIMABEAM. In Calibrated LIMABEAM, the parameters of the filter-and-sum beamformer are optimized using a calibration utterance with a known transcription and then fixed for future processing. In Unsupervised LIMABEAM, the parameters are optimized for each utterance independently using hypothesized transcriptions.

be achieved by other methods, e.g., those which utilize estimates of the Hessian. In this work, we perform optimization using the method of conjugate gradients [14], using the software package found in [15]. In this method, the step size varies with each iteration and is determined by the optimization algorithm itself.

IV. LIMABEAM in Practice

In Section III, a new approach to microphone array processing was presented in which the array processing parameters are optimized specifically for speech recognition performance using information from the speech recognition system itself. Specifically, we showed how the parameters of a filter-and-sum beamformer can be optimized to maximize the likelihood of a known transcription. Clearly, we are faced with a paradox: prior knowledge of the correct transcription, is required in order to maximize its likelihood, but if we had such knowledge, there would be no need for recognition in the first place. In this section we present two different implementations of LIMABEAM as solutions to this paradox. The first method is appropriate for situations in which the environment and the user's position do not vary significantly over time, such as in a vehicle or in front of a desktop computer terminal, while the second method is more appropriate for time-varying environments.

A. Calibrated LIMABEAM

In this approach, the LIMABEAM algorithm is cast as a method of microphone array calibration. In the calibration scenario, the user is asked to speak an enrollment utterance with a *known* transcription. An estimate of the most likely state sequence corresponding to the enrollment transcription is made via forced alignment using the features derived from the array signals. These features can be generated using an initial set of filters, e.g., from a previous calibration session or a simple delay-and-sum configuration.

Using this estimated state sequence, the filter parameters can be optimized. Using the optimized filter parameters, a second iteration of calibration can be performed. An improved set of features for the calibration utterance is generated and used to re-estimate the state sequence. The filter optimization process can then be repeated using the updated state sequence. The calibration process continues in an iterative manner until the overall likelihood converges. Once convergence occurs, the calibration process is complete. The resulting filters are now fixed and used to process future incoming speech to the array. Because the array parameters are calibrated to maximize the likelihood of the enrollment utterance, we refer to this method as *Calibrated LIMABEAM*. A flowchart of the calibration algorithm is shown in Fig. 2(a).

B. Unsupervised LIMABEAM

In order for the proposed calibration algorithm to be effective, the array parameters learned during calibration must be valid for future incoming speech. This implies that there will not be any significant changes over time to the environment or the user's position. While this is a reasonable assumption for several situations, there are several applications in which either the environment or the position of the user do vary over time. In these cases, filters obtained from calibration may no longer be valid. Furthermore, there may be situations in which requiring the user to speak a calibration utterance is undesirable. For example, a typical interaction at an information kiosk is relatively brief and requiring the user to calibrate the system will significantly increase the time it takes for the user to complete a task.

Fig. 3. Four-microphone PDA mockup used to record the CMU WSJ PDA corpus.

In these situations, it is more appropriate to optimize the array parameters more frequently, i.e., on an utterance-by-utterance basis. However, we are again faced with the paradox discussed earlier. In order to maximize the likelihood of the correct transcription of the test utterances, we require *a priori* knowledge of the very transcriptions that we desire to recognize. In this case, where the use of a calibration utterance is no longer appropriate, we solve this dilemma by estimating the transcriptions and using them in an *Unsupervised* manner to perform the array parameter optimization.

In *Unsupervised LIMABEAM*, the filter parameters are optimized on the basis of a hypothesized transcription, generated from an initial estimate of the filter parameters. Thus, this algorithm is a multi-pass algorithm. For each utterance or series of utterances, the current set of filter parameters are used to generate a set of features for recognition which in turn, are used to generate a hypothesized transcription. Using the hypothesized transcription and the associated feature vectors, the most likely state sequence is estimated using Viterbi alignment as before. The filters are then optimized using the estimated state sequence, and a second pass of recognition is performed. This process can be iterated until the likelihood converges. A flowchart of the algorithm is shown in Fig. 2(b).

V. Experimental Evaluation

In order to evaluate the proposed Calibrated LIMABEAM and Unsupervised LIMABEAM algorithms, we employed the CMU WSJ PDA corpus, recorded at CMU. This corpus was recorded using a PDA mockup, created with a Compaq iPaq outfitted with four microphones using a custom-made frame attached to the PDA. The microphones were placed in a 5.5 cm $\times$ 14.6 cm rectangular configuration, as shown in Fig. 3. The four microphones plus a close-talking microphone worn by the user were connected to a digital audio multitrack recorder. The speech data was recorded at a sampling rate of 16 kHz.

Recordings were made in a room approximately 6.0 m $\times$ 3.7 m $\times$ 2.8 m. The room contained several desks, computers and a printer. The reverberation time of the room was measured to be approximately 270 ms. Users read utterances from the Wall Street Journal (WSJ0) test set [16] which were displayed on the PDA screen. All users sat in a chair in the same location in the room and held the PDA in whichever hand was most comfortable. No instructions were given to the user about how to hold the PDA. Depending on the preference or habits of the user, the position of the PDA could vary from utterance-to-utterance or during a single utterance.

Two separate recordings of the WSJ0 test set were made with 8 different speakers in each set. In the first set, referred to as *PDA-A*, the average SNR of the array channels is approximately 21 dB. In the second recording session, a humidifier was placed near the user to create a noisier environment. The SNR of the second set, referred to as *PDA-B*, is approximately 13 dB.

Speech recognition was performed using Sphinx-3, a large-vocabulary HMM-based speech recognition system [17]. Context-dependent three-state left-to-right HMM's with no skips (8 Gaussians/state) were trained using the speaker-independent WSJ training set, consisting of 7000 utterances. The system was trained with 39-dimensional feature vectors consisting of 13-dimensional MFCC parameters, along with their delta and delta-delta parameters. A 25-ms window length and a 10-ms frame shift were used. Cepstral mean normalization (CMN) was performed in both training and testing.

A. Experiments Using Calibrated LIMABEAM

The first series of experiments were performed to evaluate the performance of Calibrated LIMABEAM algorithm. In these experiments, a single calibration utterance for each speaker was chosen at random from utterances at least 10 s in duration. For each speaker, delay-and-sum beamforming was performed on the calibration utterance and recognition features were generated from the delay-and-sum output signal. These features and the known transcription of the calibration utterance were used to estimate the most likely state sequence via forced alignment. Using this state sequence, a filter-and-sum beamformer with 20 taps per filter was optimized. In all cases, the steering delays were estimated using the PHAT method [18] and the filters were initialized to a delay-and-sum configuration for optimization. The filters obtained were then used to process all remaining utterances for that speaker.

Experiments were performed using both 1 Gaussian per state and 8 Gaussians per state in the log-likelihood expression used for filter optimization. The results of these experiments are shown in Fig. 4(a) and (b) for the PDA-A and PDA-B test sets, respectively. For comparison, the results obtained using only a single microphone from the array and using conventional delay-and-sum beamforming are also shown. The GSC algorithm ([2]) with parameter adaptation during the nonspeech regions only (as per [4]) was also performed on the PDA data. The recognition performance was significantly worse than delay-and-sum beamforming and, therefore, the results are not reported here.

As the figures show, the calibration approach is, in general, successful at improving the recognition accuracy over delay-and-sum beamforming. On the less noisy PDA-A test data, using mixtures of Gaussians in the likelihood expression to be optimized resulted in a significant improvement over conventional delay-and-sum processing, whereas the improvement using single Gaussians is negligible. On the other hand, the improvements obtained in the noisier PDA-B set are substantial in both cases and the performance is basically the same. While it is difficult to compare results across the two test sets directly because the speakers are different in each set,

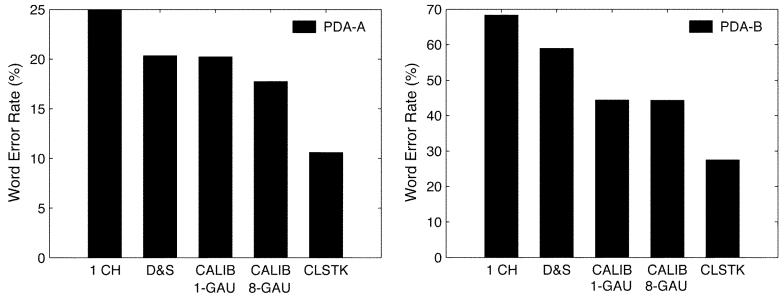


Fig. 4. Word error rate obtained using Calibrated LIMABEAM on the CMU WSJ (a) PDA-A and (b) PDA-B corpora. The figures show the performance obtained using a single microphone, delay-and-sum beamforming, and the proposed Calibrated LIMABEAM method with 1 Gaussian/state or 8 Gaussians/state in the optimization. The performance obtained using a close-talking microphone is also shown.

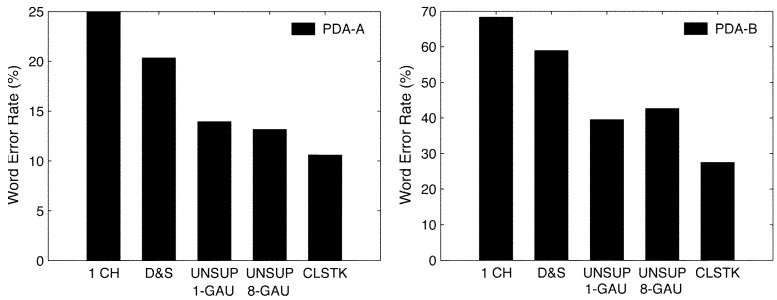


Fig. 5. Word error rate obtained using Unsupervised LIMABEAM on the CMU WSJ (a) PDA-A and (b) PDA-B corpora. The figures show the performance obtained using a single microphone, delay-and-sum beamforming, and the proposed Unsupervised LIMABEAM method with 1 Gaussian per state or 8 Gaussians per state in the optimization. The performance obtained using a close-talking microphone is also shown.

the results obtained using single Gaussians versus mixtures of Gaussians generally agree with intuition. When the test data are well matched to the training data, i.e., same domain and distortion, using more descriptive models is beneficial. As the mismatch between the training and test data increases, e.g., the SNR decreases, more general models give better performance.

Comparing Fig. 4(a) and (b), there is a significant disparity in the relative improvement obtained on the PDA-A test set compared with the PDA-B test set. A 12.7% relative improvement over delay-and-sum beamforming was obtained in the PDA-A test set, while the improvement on PDA-B was 24.6%. As described above, the users were not told to keep the PDA in the same position from utterance to utterance. The users in this corpus each read approximately 40 utterances while holding the PDA in their hand. Therefore, we can expect some movement will naturally occur. As a result, the filter parameters obtained from calibration using an utterance chosen at random may not be valid for many of the utterances from that user. We re-examine this hypothesis in Section V.B, where the parameters are adjusted for each utterance individually using the unsupervised approach.

Finally, the experimental procedure described constitutes a single iteration of the calibrated LIMABEAM algorithm. Performing additional iterations did not result in any further improvement.

B. Experiments Using Unsupervised LIMABEAM

A second series of experiments was performed to evaluate the performance of the Unsupervised LIMABEAM algorithm. In this case, the filter parameters were optimized for each utterance individually in the following manner. Delay-and-sum beamforming was used to process the array signals in order to generate an initial hypothesized transcription. Using this hypothesized transcription and the features derived from the delay-and-sum output, the state sequence was estimated via forced alignment. Using this state sequence, the filter parameters were optimized. As in the calibrated case, 20 taps were estimated per filter and the filters were initialized to a delay-and-sum configuration. We again compared the recognition accuracy obtained when optimization is performed using HMM state output distributions modeled as Gaussians or mixtures of Gaussians. The results are shown in Fig. 5(a) and (b) for PDA-A and PDA-B, respectively.

There is sizable improvement in recognition accuracy over conventional delay-and-sum beamforming in both test sets. Using Unsupervised LIMABEAM, an average relative improvement of 31.4% was obtained over delay-and-sum beamforming over both test sets. It is interesting to note that by comparing Fig. 4(a) and 5(a), we can see a dramatic improvement in performance using the unsupervised method, compared to that obtained using the calibration algorithm. This confirms our earlier conjecture that the utterance used for calibration was not representative of the data in the rest of the test set, possibly because the position of the PDA with respect to the user varied over the course of the test set.

Additionally, we can also see that the effect of optimizing using Gaussian mixtures versus single Gaussians in the unsupervised case is similar to that seen in the calibration experiments. As the mismatch between training and testing conditions increases, better performance is obtained from the more general single Gaussian models. As in the calibration case, these

TABLE I
WER OBTAINED ON PDA-A USING UNSUPERVISED LIMABEAM AND AN OPTIMIZED SINGLE-CHANNEL POST-FILTER APPLIED TO THE OUTPUT OF A DELAY-AND-SUM BEAMFORMER

Processing Method	Filter Length	WER (%)
Delay-and-sum	–	20.3
Unsupervised ML post-filter	20	20.1
Unsupervised ML post-filter	80	18.4
Unsupervised LIMABEAM	20	13.9

TABLE II
WER OBTAINED ON PDA-B USING BOTH LIMABEAM METHODS WITH AND WITHOUT TIME-DELAY COMPENSATION (TDC) PRIOR TO BEAMFORMER OPTIMIZATION

Array Processing Method	TDC	WER (%)
Delay-and-sum	Yes	58.9
Calibrated LIMABEAM	Yes	44.4
Calibrated LIMABEAM	No	45.7
Unsupervised LIMABEAM	Yes	39.5
Unsupervised LIMABEAM	No	39.8

results were obtained from only a single iteration of Unsupervised LIMABEAM, and additional iterations did not improve the performance further.

C. LIMABEAM Versus Sum-and-Filter Processing

There is another class of methods for microphone array processing which can be referred to as *sum-and-filter* methods. In such methods, the array signals are processed using conventional delay-and-sum beamforming or another array processing algorithm and the single-channel output signal is then passed through a *post-filter* for additional spectral shaping and noise removal [19], [20].

We performed a series of experiments to compare the performance of the proposed maximum-likelihood filter-and-sum beamformer to that of a single-channel post-filter optimized according to the same maximum-likelihood criterion and applied to the output of a delay-and-sum beamformer. For these experiments, the parameters of both the filter-and-sum beamformer and the single-channel post-filter were optimized using Unsupervised LIMABEAM with single-Gaussian HMM state output distributions. For the filter-and-sum beamformer, 20-tap filters were estimated as before. For the post-filtering, filters with 20 taps and 80 taps were estimated, the latter being the same number of total parameters used in the filter-and-sum case.

The results of these experiments are shown in Table I. As the table shows, jointly optimizing the parameters of a filter-and-sum beamformer provides significantly better speech recognition performance compared to optimizing the parameters of a single-channel post-filter.

D. Incorporating TDC Into LIMABEAM

In the filter-and-sum equation shown in (1), the steering delays were shown explicitly as τ_m and in the experiments performed thus far, we estimated those delays and performed TDC prior to optimizing the filter parameters of the beamformer. Thus, at the start of LIMABEAM, the microphone array signals are all in phase. However, because TDC is simply a time-shift of the input signals, it can theoretically be incorporated into the filter optimization process. Therefore it is possible that we can do away with the TDC step and simply let the LIMABEAM algorithm implicitly learn the steering delays as part of the filter optimization process.

To test this, we repeated the Calibrated LIMABEAM and Unsupervised LIMABEAM experiments on the PDA-B test set. We compared the performance of both algorithms with and without TDC performed prior to filter parameter optimization. In the case where TDC was not performed, the initial set of features required by LIMABEAM for state-sequence estimation was obtained by simply averaging the array signals together without any time alignment. The results of these experiments are shown in Table II.

As the results in the table show, there is very little degradation in performance when the TDC is incorporated into the filter optimization process. It should be noted that in these experiments, because the users held the PDA, they were never significantly off-axis to the array. Therefore, there was not a significant difference between the initial features obtained from delay-and-sum and those obtained from averaging the unaligned signals together. In situations where the user is significantly off-axis, initial features obtained from simple averaging without TDC may be noisier than those obtained after TDC. This may degrade the quality of the state-sequence estimation, which may, in turn, degrade the performance of the algorithm. In these situations, performing TDC prior to filter parameter optimization is preferable.

VI. OTHER CONSIDERATIONS

A. Combining the LIMABEAM Implementations

In situations where Calibrated LIMABEAM is expected to generate improved recognition accuracy, the overall performance can be improved further by performing Calibrated LIMABEAM and Unsupervised LIMABEAM sequentially. As is the case with all unsupervised processing algorithms, the performance of Unsupervised LIMABEAM is dependent on the accuracy of the data used for adaptation. By performing Calibrated LIMABEAM prior to Unsupervised LIMABEAM, we can use the calibration method as a means of obtaining more accurate state sequences to use in the unsupervised optimization.

To demonstrate the efficacy of this approach, we performed Unsupervised LIMABEAM on the PDA-B test set using state sequences estimated from the features and transcriptions produced by Calibrated LIMABEAM, rather than by delay-and-sum beamforming as before. Recalling Fig. 4(b), Calibrated LIMABEAM generated a 24.6% relative improvement over delay-and-sum processing on the PDA-B test set. By performing Unsupervised LIMABEAM using the transcriptions generated by the calibrated beamformer rather than by delay-and-sum beamforming, the word error rate (WER) was reduced from 42.8% to 37.9%. For comparison, the WER obtained from delay-and-sum beamforming was 58.9%.

B. Data Sufficiency for LIMABEAM

One important factor to consider when using either of the two LIMABEAM implementations described is the amount of speech data used in the filter optimization process. If too little data are used for optimization or the data are unreliable, then the filters produced by the optimization process will be sub-optimal and could potentially degrade recognition accuracy.

In Calibrated LIMABEAM, we are attempting to obtain filters that generalize to future utterances using a very small amount of data (only a single utterance). As a result, if the beamformer contains too many parameters, the likelihood of overfitting is quite high. For example, experiments performed on an eight-channel microphone array in [11] showed that a 20-tap filter-and-sum beamformer can be reliably calibrated with only 3–4 s of speech. However, when the filter length is increased to 50 taps, overfitting occurs and recognition performance degrades. When 8–10 s of speech data are used, the 50-tap filters can be calibrated successfully and generate better performance than the 20-tap filters calibrated on the same amount of data.

For Unsupervised LIMABEAM to be successful, there has to be a sufficient number of correctly labeled frames in the utterance. Performing unsupervised optimization on an utterance with too few correctly hypothesized labels will only degrade performance, propagating the recognition errors further.

C. Incorporating Feature Mean Normalization

Speech recognition systems usually perform better when mean normalization is performed on the features prior to being processed by the recognizer, both in training and decoding. Mean normalization can easily be incorporated into the filter parameter optimization scheme by performing mean normalization on both the features and the Jacobian matrix in the likelihood expression and its gradient.

However, we found no additional benefit to incorporating feature mean normalization into the array parameter optimization process. We believe this is because the array processing algorithm is already attempting to perform some degree of channel compensation for both the room response *and* the microphone channel, as it is impossible to separate the two.

D. Applying Additional Robustness Techniques

There is a vast literature of techniques designed to improve speech recognition accuracy under adverse conditions, such as additive noise and/or channel distortion. These algorithms typically operate in the feature space, e.g., codeword-dependent cepstral normalization (CDCN) [21], or the model space, e.g., maximum-likelihood linear regression (MLLR) [22].

We have found that applying such techniques after LIMABEAM results in further improvements in performance. For example, Table III shows the WER obtained when batch-mode unsupervised MLLR with a single regression class is applied after delay-and-sum beamforming and after Calibrated LIMABEAM for the PDA-B test set.

As the table shows, performing unsupervised MLLR after delay-and-sum beamforming results in recognition accuracy that is almost as good as Calibrated LIMABEAM alone. However, when MLLR is applied after Calibrated LIMABEAM, an additional 10% reduction in WER is obtained. It should also be noted that in this experiment, the MLLR parameters were estimated using the entire test set, while the parameters estimated by Calibrated LIMABEAM were estimated from only a single utterance. Furthermore, by comparing these results to those shown in Table II, we can see that the performance obtained by applying MLLR to the output of delay-and-sum beamforming is still significantly worse than that obtained by Unsupervised LIMABEAM alone.

TABLE III
WER OBTAINED BY APPLYING UNSUPERVISED MLLR AFTER ARRAY PROCESSING ON THE PDA-B TEST SET

Processing Method	WER (%)
Delay-and-sum	58.9
Delay-and-sum + MLLR	44.7
Calibrated LIMABEAM	44.4
Calibrated LIMABEAM + MLLR	40.8

VII. SUMMARY AND CONCLUSIONS

In this paper, we introduced LIMABEAM, a novel approach to microphone array processing designed specifically for improved speech recognition performance. This method differs from previous array processing algorithms in that no waveform-level criteria are used to optimize the array parameters. Instead, the array parameters are chosen to maximize the likelihood of the correct transcription of the utterance, as measured by the statistical models used by the recognizer itself. We showed that finding a solution to this problem involves jointly optimizing the array parameters and the most likely state sequence for the given transcription and described a method for doing so.

We then developed two implementations of LIMABEAM which optimized the parameters of a filter-and-sum beamformer. In the first method, called Calibrated LIMABEAM, an enrollment utterance with a known transcription is spoken by the user and used to optimize the filter parameters. These filter parameters are then fixed and used to process future utterances. This algorithm is appropriate for situations in which the environment and the user's position do not vary significantly over time. For time-varying environments, we developed an algorithm for optimizing the filter parameters in an unsupervised manner. In Unsupervised LIMABEAM, the optimization is performed on each utterance independently using a hypothesized transcription obtained from an initial pass of recognition.

The performance of these two LIMABEAM methods was demonstrated using a microphone-array-equipped PDA. In the Calibrated LIMABEAM method, we were able to obtain an average relative improvement of 18.6% over conventional beamforming, while the average relative improvement obtained using Unsupervised LIMABEAM was 31.4%. We were able to improve performance further still by performing Calibrated LIMABEAM and Unsupervised LIMABEAM in succession, and also by applying HMM adaptation after LIMABEAM.

The experiments performed in this paper showed that we can obtain significant improvements in recognition accuracy over conventional microphone array processing approaches in environments with moderate reverberation over a range of SNRs. However, in highly reverberant environments, an increased number of parameters is needed in the filter-and-sum beamformer to effectively compensate for the reverberation. As the number of parameters to optimize increases, the data insufficiency issues discussed in Section VI begin to emerge more significantly, and the performance of LIMABEAM suffers. To address these issues and improve speech recognition accuracy in highly reverberant environments, we have begun developing a subband filtering implementation of LIMABEAM.

ACKNOWLEDGMENT

The authors wish to thank Y. Obuchi of the Hitachi Central Research Laboratory, for recording and preparing the PDA speech data used in this work, and the reviewers for their comments and suggestions which improved this manuscript.

REFERENCES

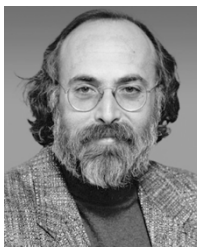
[1] D. H. Johnson and D. E. Dudgeon, *Array Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1993.
[2] L. J. Griffiths and C. W. Jim, "An alternative approach to linearly constrained adaptive beamforming," *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 27–34, Jan. 1982.
[3] S. Nordholm, I. Claesson, and M. Dahl, "Adaptive microphone array employing calibration signals," *IEEE Trans. Speech Audio Processing*, vol. 7, pp. 241–252, May 1999.
[4] D. V. Compernolle, "Switching adaptive filters for enhancing noisy and reverberant speech from microphone array recordings," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 2, Albequerque, NM, Apr. 1990, pp. 833–836.
[5] O. Hoshuyama, A. Sugiyama, and A. Hirano, "A robust adaptive beamformer for microphone arrays with a blocking matrix using constrained adaptive filters," *IEEE Trans. Signal Processing*, vol. 47, pp. 2677–2684, Oct. 1999.
[6] S. Neely and J. Allen, "Invertibility of a room impulse response," *J. Acoust. Soc. Amer.*, vol. 66, no. 1, pp. 165–169, July 1979.
[7] J. L. Flanagan, A. C. Surendran, and E. E. Jan, "Spatially selective sound capture for speech and audio processing," *Speech Commun.*, vol. 13, no. 1–2, pp. 207–222, Oct. 1993.
[8] S. Affes and Y. Grenier, "A signal subspace tracking algorithm for microphone array processing of speech," *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 425–437, Sept. 1997.
[9] B. Gillespie and L. E. Atlas, "Acoustic diversity for improved speech recognition in reverberant environments," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 1, Orlando, FL, May 2002, pp. 557–560.
[10] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.
[11] M. L. Seltzer, "Microphone array processing for robust speech recognition," Ph.D. dissertation, Dept. Elec. Comput. Eng., Carnegie Mellon University, Pittsburgh, PA, 2003.
[12] P. Moreno, B. Raj, and R. M. Stern, "A unified approach for robust speech recognition," in *Proc. Eurospeech*, vol. 1, Madrid, Spain, Sept. 1995, pp. 481–485.
[13] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 2002.
[14] J. Nocedal and S. Wright, *Numerical Optimization*. New York: Springer-Verlag, 1999.
[15] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1988.
[16] D. B. Paul and J. M. Baker, "The design of the wall street journal-based CSR corpus," in *Proc. ARPA Speech Natural Language Workshop*, Harriman, NY, Feb. 1992, pp. 357–362.
[17] P. Placeway, S. Chen, M. Eskenazi, U. Jain, V. Parikh, B. Raj, M. Ravishankar, R. Rosenfeld, K. Seymore, M. Siegler, R. Stern, and E. Thayer, "The 1996 hub-4 sphinx-3 system," in *Proc. DARPA Speech Recognition Workshop*, DARPA, Feb. 1997.
[18] C. H. Knapp and C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-24, pp. 320–327, Aug. 1976.
[19] R. Zelinkski, "A microphone array with adaptive post-filtering for noise reduction in reverberant rooms," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 5, New York, May 1988, pp. 2578–2581.
[20] C. Marro, Y. Mahieux, and K. U. Simmer, "Analysis of noise reduction and dereverberation techniques based on microphone arrays with post-filtering," *IEEE Trans. Speech Audio Processing*, vol. 6, pp. 240–259, May 1998.
[21] A. Acero, *Acoustical and Environmental Robustness in Automatic Speech Recognition*. Norwell, MA: Kluwer, 1993.
[22] C. J. Leggetter and P. C. Woodland, "Speaker Adaptation of HMM's Using Linear Regression," Cambridge University, Cambridge, U.K., Tech. Rep. CUED/F-INFENG/ TR. 181, 1994.

Michael L. Seltzer received the Sc.B. degree with honors from Brown University, Providence, RI in 1996, and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University (CMU), Pittsburgh, PA, in 2000 and 2003, respectively.

From 1996 to 1998, he was an Applications Engineer at Teradyne, Inc., Boston, MA, working on semiconductor test solutions for mixed-signal devices. From 1998 to 2003, he was a Member of the Robust Speech Recognition group at CMU. In 2003, he joined the Speech Technology Group at Microsoft Research, Redmond, WA. His current research interests include speech recognition in adverse acoustical environments, acoustic modeling, microphone array processing, and machine learning for speech and audio applications.

Bhiksha Raj received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in May 2000.

Since 2001, he has been working at Mistubishi Electric Research Laboratories, Cambridge, MA. He works mainly on algorithmic aspects of speech recognition, with special emphasis on improving the robustness of speech recognition systems to environmental noise. His latest work was on the use of statistical information encoded by speech recognition systems for various signal processing tasks.



Richard M. Stern (M'00) received the S.B. degree from the Massachusetts Institute of Technology (MIT), Cambridge, the M.S. degree from the University of California, Berkeley, and the Ph.D. degree from MIT in electrical engineering, in 1970 and 1976, respectively.

He has been a Member of the Faculty at Carnegie Mellon University (CMU), Pittsburgh, PA, since 1977, where he is currently Professor of electrical and computer engineering, and Professor by Courtesy of computer science, language technologies, and biomedical engineering. Much of his current research is in spoken language systems, where he is particularly concerned with the development of techniques with which automatic speech recognition systems can be made more robust with respect to changes of environment and acoustical ambience. He has also developed sentence parsing and speaker adaptation algorithms in earlier CMU speech systems. In addition to his work in speech recognition, he has also done active research in psychoacoustics, where he is best known for theoretical work in binaural perception.

Dr. Stern has served on many technical and advisory committees for the DARPA program in spoken language research, and for the IEEE Signal Processing Society's technical committees on speech and audio processing. He was a co-recipient of CMU's Allen Newell Medal for Research Excellence in 1992. He is a member of the Acoustical Society of America.