THE JOURNAL OF THE ACOUSTICAL SOCIETY OF KOREA Vol.29, No.1E 2010. 3. pp. 45-53

Performance Improvement of Microphone Array Speech Recognition Using Features Weighted Mahalanobis Distance

Dinh Cuong Nguyen*, Hyun-Yeol Chung*
*Department of Information and Communication, Yeungnam University
(Received September 14, 2009; revised October 29, 2009; accepted November 12, 2009)

Abstract

In this paper, we present the use of the Features Weighted Mahalanobis Distance (FWMD) in improving the performance of Likelihood Maximizing Beamforming (Limabeam) algorithm in speech recognition for microphone array. The proposed approach is based on the replacement of the traditional distance measure in a Gaussian classifier with adding weight for different features in the Mahalanobis distance according to their distances after the variance normalization. By using Features Weighted Mahalanobis Distance for Limabeam algorithm (FWMD-Limabeam), we obtained correct word recognition rate of 90.26% for calibrate Limabeam and 87.23 % for unsupervised Limabeam, resulting in a higher rate of 3 % and 6 % respectively than those produced by the original Limabeam. By implementing a HM-Net speech recognition strategy alternatively, we could save memory and reduce computation complexity.

Keywords: *Speech recognition, Microphone array, Limabeam, FWMD-Limabeam, HM-Net, Features Weighted Mahalanobis Distance.*

I. Introduction

In recent years, using microphone array has been considered in improving the recognition performance of captured speech. Many methods have been proposed for microphone array speech recognition. For example, Delay and Sum (D&S) Beamformer [1], in which delays were inserted in each channel to compensate the difference in travel time between the desired sound source and the various sensors before summing them to give a single enhanced output channel. Post-filtering algorithms proposed by Zelinski [2], McCowan [3], and Leukimmiatis [4] use the input channel auto- and cross-spectral densities to estimate a Wiener post-filter being applied to the beamformer output. All microphone array processing approaches described above were designed for signal enhancement (by the improvement of Signal to Noise Ratio). However, most of the speech recognition systems do not interpret waveform-level information directly. It is the statistical pattern classifier that operates on a sequence of features derived from the waveform. Therefore, the techniques described above might not improve Word Error Rate (WER) significantly. Recent work of Michael L. Seltzer [5] shows that the WER can be significantly reduced in microphone array speech recognition by adapting the weights ofbeamformer for generating a sequence of features which maximizes the likelihood of the correct hypothesis. In this approach, called Likelihood Maximizing Beamforming algorithm (Limabeam), information from the recognition system is used for optimizing the beamformer weights. Limabeam has several advantages over classical methods in

Corresponding author: Hyun-Yeol Chung (hychung@yu.ac.kr)
Department of Information and Communication, Yeungnam University, 214-1 Dae-dong, Gyongsan, Gyongbuk, 714-749 Korea.

enhancing signal components which are important for accurate recognition. It requires no assumption for the interfering signals nor any priori knowledge of microphone array geometry, room configuration, speaker-to-receiver impulse responses, etc. Limabeam is mainly a data-driven approach [5]. But the problem is that the feature parameters used for calculating distance can be distorted by noise in this algorithm [6].

In the present study, we first investigated the recognition performance of Limabeam applied for Korean speech and then we proposed a new method to improve recognition performance of this algorithm by reducing the influence of features distorted by noise. We also studied the computational complexity of this system so that it could be implemented in real environment.

II. Speech Recognition Using Microphone Array

2.1. Time alignment

The speech signal received from a speaker in an acoustical environment is corrupted by additive noise as well as room reverberation. This condition obtains time-delay of signal on each channel in microphones system, so that pre-processing for the recognition system using microphone array is to reduce time delay on each channel. Most of the methods for time delay estimation are based on finding the time lag which maximizes the cross-correlation between filtered versions of the received signals. To estimate the time delay, one channel is chosen as a reference and the Time-Difference Of Arrival (TDOA) for the rest of the channels is estimated using Generalized Cross-Correlation (GCC) Phase Transform (PHAT) [7]. When the signals $x_1(n)$ and $x_2(n)$ are obtained by each of two microphones, the generalized cross-correlation between $x_1(n)$ and $x_2(n)$ can be obtained by the following equation

$$R_{12}(\tau)=\frac{1}{2\pi}\int_{-\infty}^{\infty}W(\omega)G_{x_1x_2}(\omega)e^{j\omega\tau}d\omega \tag{1}$$

where

$$G_{x_1x_2}(\omega)=X_1(e^{j\omega\tau}).X_2^*(e^{j\omega\tau}) \tag{2}$$

$$W(\omega)=\frac{1}{\left|X_1(e^{j\omega\tau}).X_2^*(e^{j\omega\tau})\right|} \tag{3}$$

Where $X_1(\omega)$ and $X_2(\omega)$ are the Fourier transforms of microphone signal $x_1(n)$ and $x_2(n)$ respectively, and $W(\omega)$ is the weight function.

It is the estimated TDOA $\hat{\tau}_1(t)$ that maximizes $R_{12}(\tau)$. This technique is seen as pre-processing for the signal in Limabeam.

2.2. The Limabeam Algorithm

In array processing, the goal is to produce a distortion-free waveform. On the other hand, the goal of speech recognizer is to hypothesize the correct transcription of the utterance that was spoken. Michael L. Seltzer overcame this problem by proposing Limabeam [5]. In this algorithm, Seltzer broke the pipeline structure of classical microphone array processing techniques by using the information from the speech recognition system itself to find the array parameters that improve the speech recognition performance. Figure 1 shows the structure of this approach.

Limabeam uses an adaptive Filter-and-Sum Beamformer. Assuming that the filters have a finite impulse response (FIR), Filter-and-Sum processing is expressed mathematically as

$$y[n]=\sum_{m=0}^{M-1}\sum_{p=0}^{P-1}h_m[p]x_m[n-p-\tau_m] \tag{4}$$

where $h_m[P]$ is the p th 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. P is the length of FIR filter. All filter coefficients for all microphones are presented by

$$\xi = [h_0[0], h_0[1], ..., h_{M-1}[P-2], h_{M-1}[P-1]]^T \quad (5)$$

We can see the flowchart of processing filter-and-sum process in Figure 2.

The feature vector of the filter-and-sum output $y[n]$, $Z = \{z_1, z_2, ..., z_T\}$, is the function of both incoming speech and array processing parameters. The recognizer chooses a hypothesis $\hat{\omega}$ according to Bayes optimal classification as

$$\hat{\omega} = \arg\max_{\omega} P(Z(\xi) \mid \omega) P(\omega) \quad (6)$$

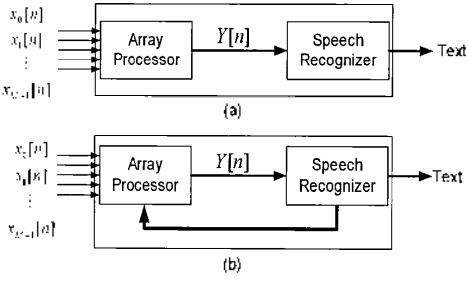

Fig. 1. Structure of microphone array processing for speech recognition techniques.

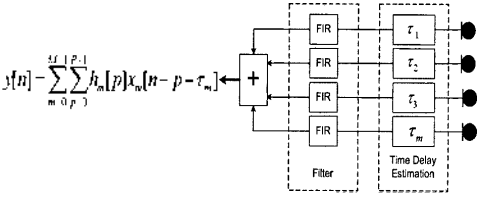

Fig. 2. Flowchart of Filter and Sum process.

Limabeam finds the parameter vector $\hat{\xi}$ for optimal recognition performance. Let us assume that the correct transcription of the utterance w_c is known. The optimal parameter vector ξ can be calculated as

$$\hat{\xi} = \arg\max_{\xi} \log(P(Z(\xi) \mid \omega_C)) \quad (7)$$

Let S_c be the set of all possible HMM state sequences and s be one such state sequence, maximum likelihood estimate of ξ that can be written as

$$\hat{\xi} = \arg\max_{\xi, s \in S_C} \left\{ \sum_i \log(P(z_i(\xi) \mid s_i)) + \sum_i \log(P(s_i \mid s_{i-1}, \omega_C)) \right\} \quad (8)$$

According to (8), in order to find $\hat{\xi}$, the likelihood of the correct transcription must be jointly optimized with respect to both array parameters and state sequence. This joint optimization can be performed by alternately optimizing the state sequence and the array processing parameters.

2.2.1. Optimizing the State Sequence

Given a set of parameter ξ, the speech can be processed by the array and the sequence of feature vector $Z(\xi)$. Using the features vectors and the transcription w_c, we can find the state sequence of $\hat{s} = \{\hat{s}_1, \hat{s}_2, ..., \hat{s}_T\}$ as in the following formula

$$\hat{s} = \arg\max_{s} \sum_i \log(P(s_i \mid s_{i-1}, \omega_C, Z(\xi))) \quad (9)$$

This state sequence $\hat{S}$ can be easily determined by forced alignment using the Viterbi algorithm [8].

2.2.2. Optimizing the Array Parameters

Given a state sequence $\hat{S}$, we can find $\hat{\xi}$ as in the following

$$\hat{\xi} = \arg\max_{\xi} \sum_i \log(P(z_i(\xi) \mid \hat{s}_i)) \quad (10)$$

This acoustic likelihood expression can be maximized with respect to the array parameter ξ via gradient-based approach. Let $L(\xi)$ represent the total log-likelihood the observation vector given an HMM state sequence.

$$L(\xi)=\sum_i \log(P(z_i(\xi)\,|\,\tilde{s}_i)) \tag{11}$$

Using the definition of ξ in (3), $\nabla L(\xi)$ can be calculated as

$$\nabla_\xi L(\xi)=\left[\frac{\partial L(\xi)}{\partial h_0[0]},\frac{\partial L(\xi)}{\partial h_0[1]},\ldots,\frac{\partial L(\xi)}{\partial h_{M-1}[P-1]}\right]^T \tag{12}$$

The computation of the gradient vector is dependent on the form of the HMM state distribution. We can derive the gradient expression when the state distributions are modeled as Gaussian distribution or mixtures of Gaussian. In both cases, the features are assumed to be mel-frequency coefficients (MFCC) or log-mel spectra. The total log likelihood for an utterance can now be expressed as

$$L(\xi)=\sum_i\left\{-\frac{1}{2}(z_i(\xi)-\mu_i)^T\Sigma_i^{-1}(z_i(\xi)-\mu_i)+\kappa_i\right\} \tag{13}$$

Where μ_i and Σ_i are the mean vector and covariance matrix respectively, the pdf of the most likely HMM state at frame i, k_i is a normalizing constant. Using the chain rule, the gradient of $\nabla L(\xi)$ with respect to ξ can be expressed as

$$\nabla_\xi L(\xi)=-\sum_i\frac{\partial z_i(\xi)}{\partial \xi}\Sigma_i^{-1}(z_i(\xi)-\mu_i) \tag{14}$$

Equation (11) can be written in Gaussian mixture model as

$$L(\xi)=\sum_i\log\left\{\sum_{k=1}^{K}\exp(-\frac{1}{2}(z_i(\xi)-\mu_{ik})^T\Sigma_{ik}^{-1}(z_i(\xi)-\mu_{ik})+\log(\alpha_{ik}\kappa_{ik}))\right\} \tag{15}$$

The gradient can be replaced as

$$\nabla L(\xi)=-\sum_i\sum_{k=1}^{K}\gamma_{ik}(\xi)\frac{\partial z_i}{\partial \xi}\Sigma_{ik}^{-1}(z_i(\xi)-\mu_{ik}) \tag{16}$$

Where μ_{ik} and Σ_{ik} are the mean vector and covariance matrix, α_{ik} is the mixture weight and k_{ik} is a normalizing constant at frame i and k_{th} mixture component. $\gamma_{ik}(\xi)$ represents the a posteriori probability of the k_{th} mixture component, $\partial z_i(\xi)/\partial\xi$ is the Jacobin matrix.

2.2.3. Implementation of Limabeam Algorithm

There are two ways to implement Limabeam. The first one, called calibrated Limabeam, is appropriate for situations in which the environment and the user's position do not vary significantly over time. The second one, called unsupervised Limabeam, is more appropriate for time-varying environments. The entire algorithm for estimating filter parameters ξ can be stated as follows:

- Calibrated Limabeam

1. Time-align the signal from the M microphones
2. Initialize ξ as $h_i[0]=1/M; h_i[k]=0, k\neq 0$
3. Process the signal using ξ to generate an output signal
4. Determine optimal state sequence through the calibrated transcription, array output signal, and HMM from speech recognizer.
5. Use optimal state sequence and (10) to estimate ξ
6. Go to step 3 if $L(\xi)$ has not converged.

- Unsupervised Limabeam

1. Time-align the signals from the M microphones
2. Initialize ξ as $h_i[0]=1/M; h_i[k]=0, k\neq 0$
3. Process the signals using ξ to generate an output signal

4. Perform speech recognition on the array output to obtain a word sequence
5. Determine optimal state sequence through the hypothesized transcription, array output signal, and HMM form speech recognition
6. Use optimal state sequence and (10) to estimate ξ
7. Go to step 3 if $L(\xi)$ has not converged.

III. Feature Weighted Mahalanobis Distance

From (13), the Mahalanobis distance in original Limabeam is used to estimate the distance for the features when optimizing the parameters of filters. This distance measure shows good performance for clean features, but loses performance significantly in case of noise features [6]. To improve the noise robustness of the Mahalanobis distance in Limabeam, we replaced it with our proposed FWMD. The aim is to give less weight to noise features and higher weight to noise free features which are more reliable.

The Mahalanobis distance is represented by

$$D_i^{Mahalanobis} = (x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i) \tag{17}$$

where μ_i represents the mean vector of class (I), x represents the sample vector to classify and Σ_i^{-1} represents the inverse of the covariance matrix of class (I).

To find the features which have the strongest influence on the distance, solving the Mahalanobis distance equation for every single feature c over input samples i and class j, we have the following equation

$$\forall c,i,j : D_{i,j}[c] = (x_i[c]-\mu_j[c])^T \Sigma_j[c,c]^{-1} (x_i[c]-\mu_j[c]) \tag{18}$$

Now, the goal is to give less weight to the features with big distance to avoid the masking of the features with small distances. Weighting for the feature parameters FWMD is written as

$$\forall i,j : D_{i,j}^{weighted} = \sum_{c=1}^{N} w_{i,j} D_{i,j}[c] \tag{19}$$

where N is the number of features for i sample. To calculate the individual weights

$$\forall c,i,j : w_{i,j}[c] = \frac{\sum_{a=1}^{N} d_{i,j}[a]}{N.d_{i,j}[c]} \tag{20}$$

The difference $d[c]$ for features can be calculated as in the following

$$\forall c,i,j : d_{i,j}[c] = \sum_{a=1}^{N} \left| D_{i,j}[c] - D_{i,j}[a] \right| \tag{21}$$

IV. Speech Recognition System

Limabeam algorithm was originally investigated with Sphinx3, an HMM-based large-vocabulary speech recognition system, and English database [5]. In this algorithm, array parameter optimization is performed in the log-mel spectral domain, rather than the cepstral domain. For this reason, M. Seltzer employed a parallel set of HMMs trained on log-mel spectra, rather than cepstral by using STAtistical Re-estimation (STAR) algorithm [5] so that the two sets of models could have identical frame-to-state alignments. The general flowchart of Limabeam algorithm is shown in Figure 3.

An alternative method for making parallel HMM model is to use Invert Discrete Cosine Transform (IDCT) and model adaptation Maximum a Posteriori (MAP) and/or Maximum Likelihood Linear Regression (MLLR). This process, presented in Figure 4, works

as follows. First, LogMelSpec HMM models were generated by employing IDCT on MFCC HMM models. These LogMelSpec HMM models were then adapted to make them suitable to LogMelSpec features of training database through MAP and/or MLLR. Although this process does not employ STAR algorithm as in M. Seltzer' s approach, it still warrants the property of parallel models - identical frame-to-state alignment of two HMM models.

In our study, we used HM-Net recognition system [9] [10] to implement Limabeam algorithm as shown in Figure 5. Here, an optimal state sequence was estimated on cepstral domain and the LogMelSpec HMM Model was calculated directly from MFCC HMM model through IDCT.

For feature extraction, the relation between MFCC and Log Mel Spectral (LogMelSpec) is

$$Z_{MFCC} = DCT(Z_{LogMelSpec}) \tag{22}$$

where Z_{MFCC} is MFCC feature, $Z_{LogMelSpec}$ is the Log-Mel Spectral feature, and DCT is the Discrete Cosine Transform. LogMelSpec feature can be calculated in reverse way

$$Z_{LogMelSpec} = IDCT(Z_{MFCC}) \tag{23}$$

where $IDCT$ is the Inverse Discrete Cosine Transform.

Compared with Limabeam implementation, this scheme has two main advantages.

1. *Reducing the required memory for storing HMM model*: Usually, the size of LogMelSpec HMMs is about 2 times larger than that of MFCC HMMs. Thus, this method could significantly reduce the memory required to store HMMs used for optimization. This can potentially enable the array process to be performed on limited memory devices such as cell-phone, PDA.
2. *Reducing complexity*: Generating a LogMelSpec parallel set of HMMs requires converting speech

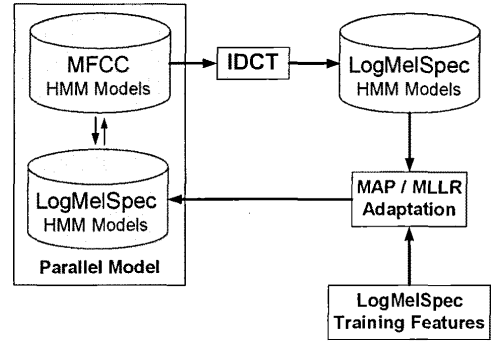

Fig. 4. Flowchart of making parallel HMM model via IDCT and MAP/MLLR Adaptation.

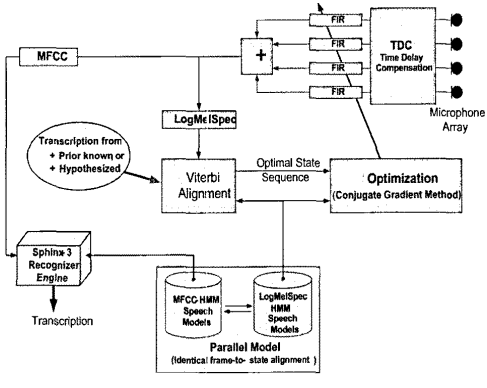

Fig. 3. General flow-chart of Limabeam algorithm, transcription for finding optimal state sequence can be taken from prior known (calibrated Limabeam) or from hypothesized (unsupervised Limabeam).

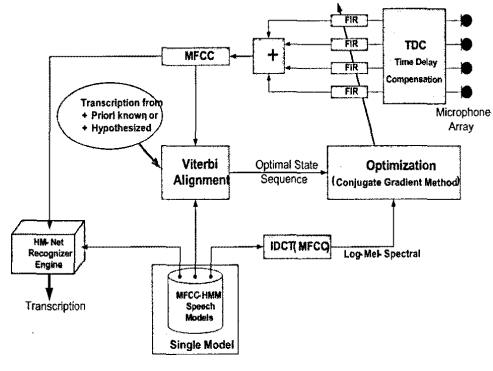

Fig. 5. Flowchart of proposed scheme to implement Limabeam algorithm. Optimal state sequence was estimated on cepstral domain and the LogMelSpec HMM Model was calculated directly from MFCC HMM model through IDCT.

signals to log-mel spectral features process and employing STAR algorithm to generate parallel models. By removing these processes, our scheme reduces complexity.

V. Experiments

5.1. Implementing the recognition system

To implement Limabeam algorithm with Features Weighted Mahalanobis Distance, we used the HM-Net speech recognition system including 1000 states (8 Gaussians/state). HM-Net system was trained using TRADE database, a speaker-independent database which consisted of 8892 utterances spoken by 90 speakers. Each utterance was made up of a string of Korean words, such as "건평 육 하기 전 에 충분히 생각 하셔야 합니다". The sampling frequency of this database was 16 kHz. The system was trained, using 39-dimensional feature vectors consisting of 13 Mel Frequency Cepstral Coefficients (MFCC) parameters, along with their delta and delta-delta parameters. A 25-ms window length and a 10-ms frame shift were used.

In order to investigate the performance of speech recognition with microphone array, we employed 4 microphones. The space between each element was 20 cm. The distance between the loud-speaker and the center was 150 cm. The data were recorded in a room where the noise came from sources which included computer fans, air conditioner, human voice, footsteps and slams outside. A total of 596 utterances produced by six speakers recorded in above environment were used for testing. Recording conditions are shown in Figure 6.

For implementation of calibrate Limabeam, one utterance from each speaker was used to estimate filter parameters. This constituted first iteration of calibration. The second iteration of calibration was performed by using estimated filter parameters to initialize in step 2. The calibration process continued in an iterative manner until the overall likelihood converged. The optimal filter parameters were now fixed and used to process the remaining utterances of that speaker.

For unsupervised Limabeam, filter parameters were optimized afresh for each utterance 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, based on this state sequence. A second iteration could be performed by generating the hypothesized transcription from these optimal filters. This process could be iterated until the like likelihood converged.

To improve the recognition performance in two cases of Limabeam, calibrate Limabeam and unsupervised Limabeam, we used FWMD calibrate Limabeam and FWMD-unsupervised Limabeam. When optimizing parameters of FIR filter, we replaced Mahalanobis distance measure with FWMD measure.

5.2. Results and discussion

As shown in the results provided in Table 1, we can see that FWMD-Limabeam had an increase of up to 3 % compared to calibrate Limabeam and of 6% for unsupervised Limabeam in case of 40 taps of the filter. The results showed that recognition performance of FWMD-Limabeam was better than distance measure in original Limabeam algorithm because the drawback of the Mahalanobis distance in original

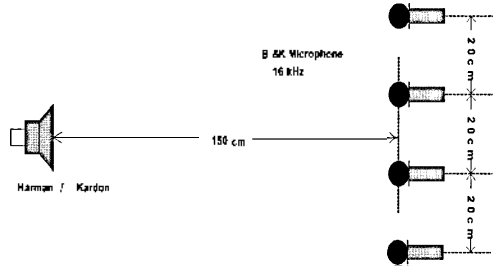

Fig. 6. Recording and recognition condition.

Table 1. Compared word recognition rate (in %) of FWMD-Limabeam and Limabeam for different number of taps.

Algorithm	Number of taps							
	5	10	15	20	25	30	35	40
Calibrate Limabeam	80.58	85.19	85.34	85.48	86.58	87.92	86.38	86.76
FWMD-Calibrate Limabeam	86.29	89.31	89.47	89.91	90.11	89.81	89.68	90.26
Unsupervised Limabeam	78.06	76.50	77.20	81.87	81.98	82.10	81.59	81.12
FWMD-unsupervised Limabeam	78.88	82.05	80.64	86.92	85.85	85.32	86.98	87.23

Limabeam was the equal adding up of the variance normalized squared distances of the features. If the feature was distorted by noise, due to the squaring of the distances, a single feature could have such a high value that it covered the information proved by the other features and it could lead to a misclassification. Therefore, it was necessary to give weight for the features.

In the tests, calibrate Limabeam gave better recognition performance than unsupervised Limabeam. However, as we know, calibrate Limabeam need to have utterances to adapt the FIR parameters to every recognition environment but unsupervised Limabeam does not need them. Calibrated Limabeam can not be employed if the test environment is changed, whereas unsupervised Limabeam can improve performance in any situation. Also, our results indicated that recognition performance could be improved when the filter length was increased.

Examining unsupervised Limabeam, we observed that the performance of optimization strongly depended on the transcription output of the first recognition step. Therefore, performing unsupervised Limabeam on an utterance with too few correctly hypothesized labels may only degrade performance. Thus, in order to improve performance of unsupervised Limabeam, we can also employ some post-filter techniques to obtain more correct transcription output of the first step but this will be part of our future work.

VI. Conclusions

In this paper, wepresented the results of our investigation on the recognition performance of Limabeam applied for Korean speech. By implementing a HM-Net speech recognition system alternatively, we not only could save the memory but also reduce computation complexity. When we applied our proposed FWMD to both calibrated and unsupervised Limabeam, we obtained increased recognition accuracy, with a correct word recognition rate of 90.26 % for FWMD-calibrate Limabeam and of 87.23 % for FWMD-unsupervised Limabeam respectively, resulting in approximately an increase rate of 3 % and 6 % higher than that of conventional Limabeam.This outcome was achieved because the FWMD reflected noisy speech characteristics well by giving less weight to the noise features and more weight to the clean features.

Acknowledgements

This research was supported by the Yeungnam University research grants in 2008.

References

1. L. J. Griffiths and C. W. Jim, "An alternative approach to linearly constrained adaptive beamforming," *IEEE Transaction on Antennas and Propagation*, vol. AP-30, no.1, pp. 27-34, January 1982.
2. R. Zelinski, "A microphone array with adaptive post-filtering for noise reduction in reverberant room," *ICASSP-88*, vol. 5, pp. 2578-258', 1988.
3. I. A. McCowan and H. Bourlard, "Microphone array post-filter for diffuse noise filed," *ICASSP 2002*, vol. 1, pp.905-908, 2002.

4. S. Leukimmiats, "An Optimum Microphone Array Post-Filter for Speech Application," *ICSLP INTERSPEECH*, pp.2142-2145, Sep. 2006.
5. M. Seltzer, "Microphone array processing for robust speech recognition," *Doctoral dissertation, Carnegie Mellon University*, Pittsburgh, PA, 2003
6. M. Wolfel and H. K. Ekenel, "Feature weighted Mahalanobis distance: improved robustness for Gaussian classifiers," *13th European Signal Processing conference EUSIPCO2006*, Antalya, Turkey, Sep. 2005.
7. C. H. Knapp and C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Information Theory*, vol.13, no .2, pp.260-269, April 1967.
8. A. J. Viterbi, "Error bounds for convolution codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, Vol."3, issue 2, pp.260-267, April 1967.
9. M. Suzuki, S. Makino, A. Ito, H. Aso and H. Shimodaira, "A new HMnet construction algorithm requiring no contextual factors," *IFICF Trans, On Information System*, Vol. E78-D, No 6, pp. 662-669, 1995.
10. N. T. Hieu, "Robust Speech Recognition using Microphone Array in Adverse Environment," *M.S Thesis*, Yeungnam University, 2006.

[Profile]

• Dinh Cuong Nguyen

He received the Bachelor degree in Computer Science, Ha Noi University of Technology, Viet Nam (2003). Master degree from Yeungnam University (2008). Currently, he is a Ph.D Candidate in the Department of Information and Communication, Yeungnam University, Korea. His research interests in Artificial ty Intelligence, Computer Vision and Speech Recognition.

• Hyun-Yeol Chung

He received the Ph.D degree in the Information Engineering from Tohoku University, Japan in 1989. He is currently a professor in the Department of Information and Communication, Yeungnam University, Korea. Both his teaching and research interests include digital signal processing, speech recognition and voice synthesis.