Pseudo Pitch Synchronous Analysis of Speech With Applications to Speaker Recognition

Ran D. Zilca, *Senior Member, IEEE*, Brian Kingsbury, *Member, IEEE*, Jiří Navrátil, and Ganesh N. Ramaswamy, *Member, IEEE*

***Abstract*—The fine spectral structure related to pitch information is conveyed in Mel cepstral features, with variations in pitch causing variations in the features. For speaker recognition systems, this phenomenon, known as "pitch mismatch" between training and testing, can increase error rates. Likewise, pitch-related variability may potentially increase error rates in speech recognition systems for languages such as English in which pitch does not carry phonetic information. In addition, for both speech recognition and speaker recognition systems, the parsing of the raw speech signal into frames is traditionally performed using a constant frame size and a constant frame offset, without aligning the frames to the natural pitch cycles. As a result the power spectral estimation that is done as part of the Mel cepstral computation may include artifacts. Pitch synchronous methods have addressed this problem in the past, at the expense of adding some complexity by using a variable frame size and/or offset. This paper introduces *Pseudo Pitch Synchronous* (PPS) signal processing procedures that attempt to align each individual frame to its natural cycle and avoid truncation of pitch cycles while still using constant frame size and frame offset, in an effort to address the above problems. Text independent speaker recognition experiments performed on NIST speaker recognition tasks demonstrate a performance improvement when the scores produced by systems using PPS are fused with traditional speaker recognition scores. In addition, a better distribution of errors across trials may be obtained for similar error rates, and some insight regarding of role of the fundamental frequency in speaker recognition is revealed. Speech recognition experiments run on the Aurora-2 noisy digits task also show improved robustness and better accuracy for extremely low signal-to-noise ratio (SNR) data.**

***Index Terms*—Pitch mismatch, pseudo pitch synchronous, speaker recognition, speech recognition.**

I. Introduction

STATE-OF-THE-ART speech and speaker recognizers typically extract cepstral features such as Mel Frequency Cepstral Coefficients (MFCCs) from consecutive frames of the speech signal, and use the MFCCs and their time derivatives as feature vectors in conjunction with a statistical classifier, such as a Hidden Markov Model (HMM) or a Gaussian Mixture Model (GMM). The MFCC features are calculated from the power spectrum, and include some harmonic structure related to the fundamental frequency, especially for high-pitched female speakers [1], [2]. For speaker recognition tasks, this property of MFCC features has been found to have a dramatic effect on accuracy. In speaker recognition evaluations conducted by the National Institute of Standards and Technology (NIST) in recent years [3]–[5] it was found that speaker recognition systems perform considerably worse for high pitched speakers, and when the target speakers' pitch varies between enrollment and testing. Present speaker recognition systems exhibit extreme sensitivity to absolute pitch values and pitch mismatch. For example, speakers who exhibit a large pitch mismatch can experience an Equal Error Rate (EER) more than twice worse than pitch matched speakers [5]. The effect of pitch-induced mismatch and variability in the features on speech recognition systems has received much less attention than pitch mismatch in speaker recognition. Variability due to pitch may be implicitly addressed by training a speech recognition system on a corpus collected from a large, diverse collection of speakers. However, based on previous work using vocal tract length normalization and feature-space maximum-likelihood linear regression to reduce inter-speaker variability [6], we might expect that explicit reduction or elimination of pitch-induced feature variability could lead to better recognition performance.

Manuscript received August 13, 2004; revised February 14, 2005. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Li Deng.

The authors are with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: zilca@us.ibm.com; bedk@us.ibm.com; jiri@us.ibm.com; ganeshr@us.ibm.com).

Digital Object Identifier 10.1109/TSA.2005.857809

In this paper, we propose three Pseudo Pitch Synchronous (PPS) time domain processing algorithms. These algorithms accept individual raw speech frames, perform an adjustment of the fundamental frequency value on the LPC residual, and produce time domain pitch-adjusted frames. Different pitch adjustment operations are done by the three algorithms. The adjustment may attempt to completely suppress the presence of pitch information in the signal and produce a power spectrum free of pitch harmonics. Alternatively, the pitch adjustment operation may attempt to better align the frame boundaries to the raw signal, thus reducing the spectral estimation artifacts.

The three PPS analysis algorithms are described in detail in Section II. In Section III, we illustrate the effects of the three PPS analysis methods on speech power spectra. Section IV presents experiments with PPS analysis in both speaker recognition systems and speech recognition systems. Finally, we present conclusions and suggested future work in Section V.

II. Pseudo Pitch Synchronous (PPS) Algorithms

A general block diagram that describes the three PPS algorithms is shown in Fig. 1. The procedure is always performed independently for each individual frame. We term the three algorithms, *depitch, syncpitch, and padpitch*, and refer to all three as x*-pitch* for short. All x-pitch algorithms accept time domain speech and also produce time domain speech. However, since speech frames typically overlap, the resulting

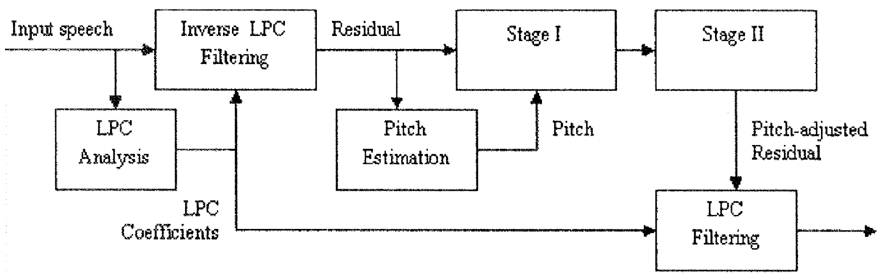


Fig. 1. Block diagram of the depitch and syncpitch methods.

time domain signal is not continuous. The input speech frame is first windowed and undergoes linear prediction analysis to estimate Linear Prediction Coding (LPC) coefficients. In our experiments we used a standard Hamming window. The LPC coefficients are used to calculate the residual signal by filtering the windowed speech using the inverse LPC filter. Pitch estimation follows, performed on the residual signal, using a variant of the autocorrelation method [7]. The pitch estimation stage uses a nonlinear, center-clipping function prior to computing the autocorrelation function, in order to reduce the influence of noise. Then, the pitch cycle is estimated to be the time difference between the main autocorrelation peak and the closest secondary peak [7]. This pitch estimation method also results in a voicing score, given by the ratio of the secondary to main peak energies in the autocorrelation function. A more detailed description of the pitch and voicing detector follows. The residual signal is then processed in two stages (I and II) to perform some form of pitch adjustment, depending on the particular algorithm. Stages I and II are described below for depitch, syncpitch, and padpitch separately. Finally, the output of stage II (the "pitch adjusted" residual signal) is filtered using the LPC filter to synthesize a time domain speech frame.

For the depitch procedure [8], stages I and II are as follows.

```
Stage I: Extract a single pitch cycle con-
sisting of $p$ samples from the center of
the frame, where $p$ is the estimated pitch
cycle in samples.
Stage II: Interpolate the $p$ samples to fit
the size of one frame, $N$ (up-sample from
$p$ to $N$).
```

For the syncpitch procedure [9], stages I and II are as follows.

```
Stage I: Calculate the maximal number of
pitch cycles, $n_{\max}$, that would not exceed
the frame size $N$. Then, extract $n_{\max} \cdot p$
samples of the residual signal, where $p$ is
the estimated pitch cycle in samples.
Stage II: Interpolate the $n_{\max}$ samples to
fit the size of one frame (up-sample from
$n_{\max}$ to $N$). Perform a cyclic shift such
that the frames edges have the lowest en-
ergy.
```

And finally, for the 3rd algorithm, padpitch, stages I and II are:

```
Stage I: Calculate the maximal number of
pitch cycles, $n_{\max}$, that would not exceed
the frame size $N$. Then, extract $n_{\max} \cdot p$
samples of the residual signal, where $p$ is
the estimated pitch cycle in samples.
Stage II: Set the remaining $N - n_{\max} \cdot p$ sam-
ples to zero and perform a cyclic shift
such that the frame edges have the lowest
energy.
```

We note the following regarding the algorithms:

1) Stage II is lossy for depitch and syncpitch, as some samples of the residual signal are ignored in every frame. However, since we use significant overlap between frames very few samples are eventually ignored in the entire signal.
2) Interpolation involves low pass filtering of the residual signal in stage II, resulting in some spectral distortion in the output speech spectrum. Depitch will typically involve more aggressive filtering than syncpitch because $n_{\max}$ is closer to N than p is. Loosely speaking, syncpitch is less lossy than depitch. Padpitch attempts to address this issue by using zero padding instead of interpolation.
3) Voicing detection is performed for each frame and only frames classified as "voiced" undergo the procedure. The voicing detection is performed by thresholding the score generated by the autocorrelation-based pitch detector.
4) The cyclic shift is meant to provide better continuity, for example for the purpose of listening. It is not meant to have any effect on the resulting features, as it changes only the phase and not the power spectrum. However, since the resulting processed frame is typically windowed, the shift does have a small effect on the features.

A. *Rationale and Motivation for the* x-*Pitch Algorithms*

1) Depitch: For speaker recognition systems, it has been suggested in [8] that speakers who often get falsely rejected (sometimes referred to as "goat speakers" [10]) may experience this difficulty in part due to pitch mismatch and high average pitch. In practice, although a system can obtain high accuracy on average, the individual experience of these goat speakers will be far worse than the expected overall performance. The objective of applying the depitch procedure is therefore

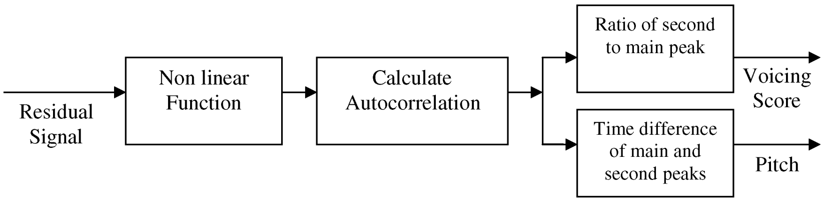

Residual Signal
Non linear Function
Calculate Autocorrelation
Ratio of second to main peak
Time difference of main and second peaks
Voicing Score
Pitch

Fig. 2. Pitch and voicing detection block diagram.

two-fold. First and foremost, it is targeted at improving the accuracy for the subset of high-pitched speakers and pitch mismatched trials that normally perform significantly worse than average. For the procedure to be useful, it should cause the individual errors to be more uniformly distributed between trials and speakers, but not at the expense of compromising the overall average accuracy. Second, the depitch method provides explanation and insight regarding the role of pitch information in automatic speaker recognition and characterization.

For speech recognition systems, depitch is expected to eliminate a good deal of cross-speaker variability by forcing all voiced frames to exhibit the same pitch frequency.

2) Syncpitch: Syncpitch is designed to be a milder version of depitch. Depitch involves aggressive filtering as part of the interpolation of the residual signal, since the interpolation from p samples to N samples involves applying a low pass filter with a normalized cutoff frequency of p/N to the residual signal. In contrast, syncpitch addresses mainly pitch cycle alignment, and maintains most of the fundamental frequency information. This is achieved by performing up-sampling to the closest possible integer number of pitch cycles rather than forcing a single pitch cycle in every frame. The resulting low pass filter in the residual domain will therefore have a normalized cutoff frequency of $n_{\max}/N$ instead of p/N.

3) Padpitch: Padpitch attempts to completely eliminate the lossy filtering involved in the residual domain interpolation altogether, and replace it by zero padding. Zero padding is theoretically expected to preserve the spectral estimation performed by applying the Fourier transform only on the first $n_{\max}$ samples.

B. Pitch and Voicing Detection Algorithm

This section describes the pitch and voicing detection algorithm used for the three PPS procedures. The algorithm is based on the autocorrelation method with some variants [7]. A block diagram of the pitch and voicing detection algorithm is shown in Fig. 2. The detection is performed separately for each frame. It is performed on the residual LPC signal, after applying the inverse LPC filter to the original speech frame. First, the residual frame goes through a nonlinear function performing center clipping and compression, as shown in Fig. 3. The clipping threshold value for the function is relative, and is recomputed for each frame. Let H be the maximum sample value in the residual frame to be processed, and L the minimum value. A single threshold TH is computed for each frame as a constant fraction, α, of the maximum absolute value in that particular frame, i.e., the full scale FS, as follows:

$$FS = \max\{|H|, |L|\} \tag{1}$$

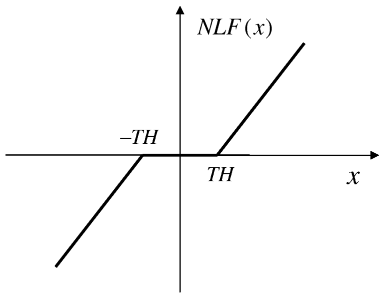


Fig. 3. Clipping function used in the voicing and pitch detection.

$$TH = \alpha \cdot FS. \tag{2}$$

The resulting nonlinear function is given by

$$NLF(x) = \begin{cases} 0, & |x| <= TH \\ x - TH, & |x| > TH; x > 0 \\ x + TH, & |x| > TH; x <= 0 \end{cases} \tag{3}$$

where x is the input sample, and $NLF(x)$ is the resulting nonlinear function.

In our experiments we set α to 7% based on a small subset of held out utterances. On this subset, the pitch detection algorithm did not demonstrate excess sensitivity to the exact value of this factor. After the nonlinear function is applied to each sample, the sample autocorrelation function is computed. The autocorrelation function has a main peak corresponding to the residual signal energy (sum of squares), and additional peaks, corresponding to different modes of periodicity, as depicted in Fig. 4. The pitch lag is estimated as the number of samples between the main peak and the second largest peak, where a peak is simply defined as a local maximum, i.e., an autocorrelation sample for which the derivative changes sign from positive to negative. The voicing score is estimated as the ratio between the autocorrelation function value at the second (pitch) peak and the main peak. The more random the residual frame, the more its autocorrelation will resemble an impulse function (nonzero value only at the main peak, hence a voicing score of zero), and the more periodic it is, the more it will resemble a cosine function (a voicing score of one). The accuracy of the pitch detector has not been formally assessed in this study. Since the pitch and voicing detection algorithm is run frame by frame with no smoothing, some doubling and halving will occur. However, PPS algorithms are not very sensitive to these errors since the frames overlap, and since the main objective is to capture an integer number of pitch cycles for analysis.

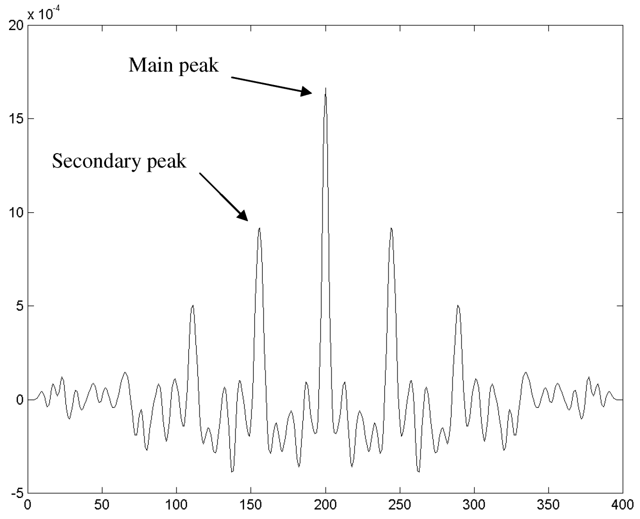


Fig. 4. Autocorrelation function in the voicing and pitch detector.

III. Effect of Pseudo Pitch Synchronous (PPS) Analysis on the Speech Power Spectrum

The features used for speech recognition and speaker recognition systems such as MFCCs are typically based on the power spectrum of individual speech frames. Therefore, it is of interest to explain the effect of PPS analysis on the speech power spectrum. For depitch, we expect to have a power spectrum that is free from the fine structure related to the fundamental frequency. Fig. 5 shows the log-power spectra of both the original and depitch signals, for a single speech frame. The harmonic fluctuations present in the original spectrum have disappeared, and a smoother power spectrum is obtained. Therefore, when using depitch as a front-end processing stage in speaker recognition systems, we may expect an improvement in the false rejection rate, since speakers can alter their pitch between enrollment and testing with no effect on the feature vectors. However, we also expect a potential increase in false acceptance rate, since prospective imposters who previously had a different typical pitch and therefore were easily rejected may be accepted more easily now. When used in speech recognition, we expect the power spectrum to be more representative of the spectral envelope, and less sensitive to the glottal excitation frequency, implying a possible improvement in recognition accuracy.

For syncpitch analysis, the fine spectral structure related to the fundamental frequency is maintained, as seen in Figs. 6 and 7. Both Figures show selected examples of the log power spectra of the original signal and syncpitch processed one, for two different speech frames. We expect the spectral distortion resulting from having truncated pitch cycles to be reduced, since syncpitch processing forces an integer number of pitch cycles inside a single frame. The two Figures show that the power spectrum changes after syncpitch processing. As clearly seen in Fig. 7, the interpolation performed in stage II is lossy, and the signal is low-pass filtered. This is less evident for the frame shown in Fig. 6, where $n_{\max} \cdot p$ is close to N, resulting in a higher cut-off frequency of the interpolation low pass filter.

Finally, the log power spectrum of a padpitched speech frame is depicted in Fig. 8. For padpitch processing there is no loss resulting from the interpolation filter. The log power spectrum of the processed frames is closer to the original one compared to the other two processing methods. The spectral artifacts related to pitch cycle truncation are expected to be reduced, explaining the difference between the two spectral images.

IV. Experiments

A. Speaker Verification Experiments

1) Experiments Using Depitch: The experiments were conducted using the NIST 2002 speaker recognition evaluation corpus [4]. The task is text independent speaker verification on cellular telephony speech data. Previous NIST evaluation results demonstrated that pitch-related performance degradation is observed mostly for female speakers (and female trials, since the evaluation is gender matched) [3], [5]. We therefore chose to run the evaluation on the female subset. The subset includes 191 speakers and 23 309 trials using 2128 test sentences. A detailed description of the data and task may be found in [4].

We implemented two systems that were identical in terms of data used, feature extraction, and modeling. The baseline system, termed "nondep" herein, uses normal speech input, while the depitch system uses depitch pre-processing for all data (i.e., background, target, and test data). The two systems share the following common attributes. A diagonal covariance GMM with 2048 components was adapted for each speaker using MAP adaptation of the Gaussian means from a Universal Background Model (UBM), similarly to the system described in [16]. The UBM was trained using approximately 10 h of data consisting of NIST 2001 cellular data and landline data passed through cellular speech vocoders. A Maximum Likelihood Linear Transform (MLLT) was also computed from the UBM data [23]. At the front-end, a filter bank of 24 Mel filters was used to extract 19 MFCCs and their time derivatives from 25-ms frames with 60% overlap. No score normalization was applied. Further information about the system can be found in [11] and [16].

We measure the performance of the systems by their EER and Detection Cost Function (DCF) [5]. Both metrics represent points on the DET plot [12]. While the EER corresponds to the middle point where the false rejection and false acceptance errors are the same, the DCF corresponds to a point in the upper-left region of the DET plot, giving higher cost to false acceptance errors [4]. Increased EER and a reduced DCF therefore correspond to a counterclockwise tilted DET plot, and vice versa. For convenience, all the DCF values in this paper were multiplied by 10^3. The nondep system obtained an EER of 10.7% and DCF of 42.6. The depitch system obtained 16.7% EER and 66.5 DCF. The relative degradation is 56% in both EER and DCF. This result by itself simply reinforces the importance of pitch information for speaker recognition, as was previously illustrated by NIST evaluation results. However, we expected to gain an improvement by combining the two systems, since the errors from the two systems might be uncorrelated to some degree. We also expected the combined system to allow for better distribution of errors across speakers and trials and in particular to alleviate the problem of goat speakers.

An improvement resulting from applying depitch is expected to be more noticeable in any subset of the data that includes

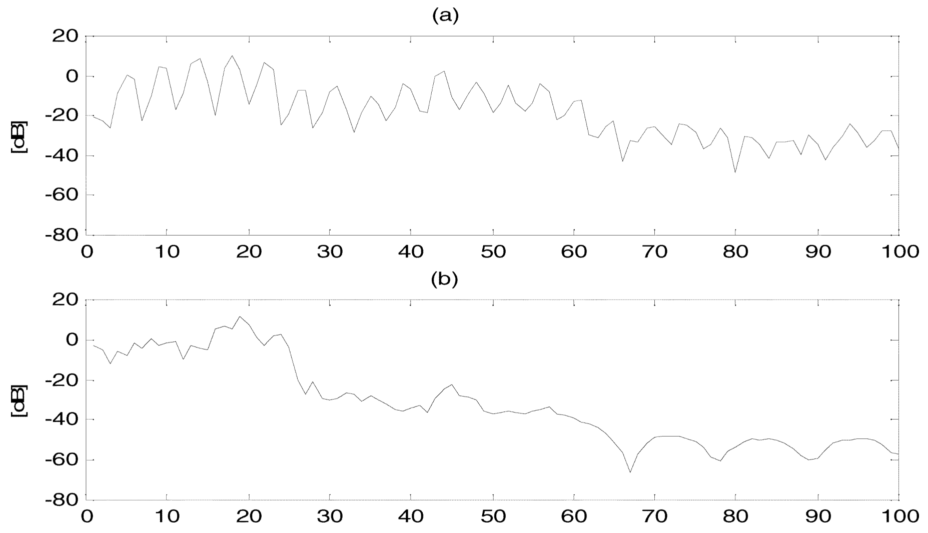


Fig. 5. Effect of depitch on the power spectrum. (a) Original and (b) depitched.

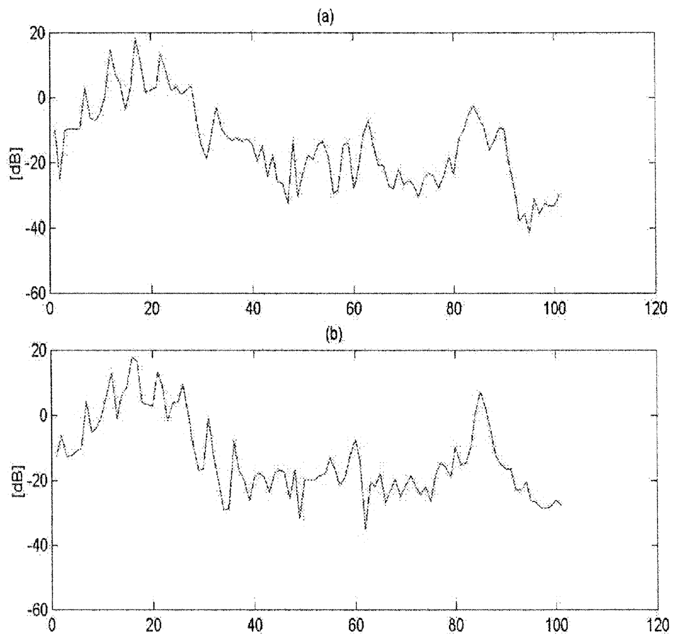


Fig. 6. Effect of syncpitch on spectrum example number 1. (a) Original and (b) syncpitched.

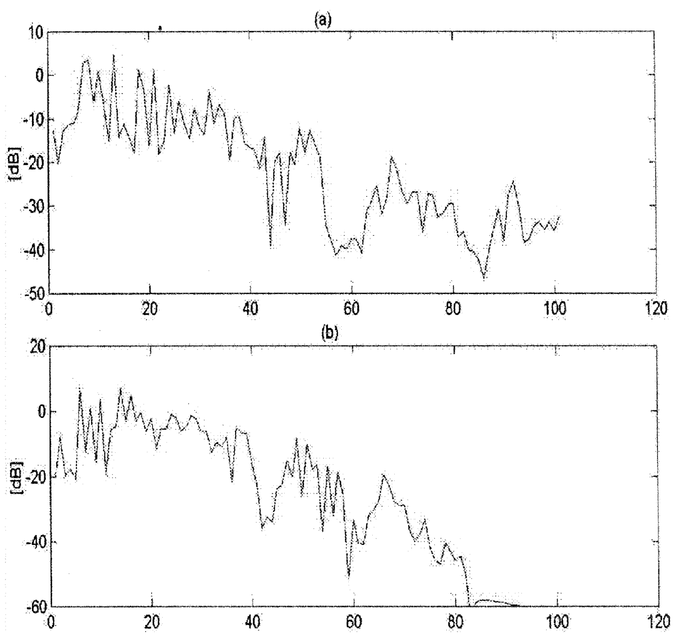


Fig. 7. Effect of syncpitch on spectrum example number 2. (a) Original and (b) syncpitched.

high pitched speakers and pitch mismatched trials, where the nondep system is expected to perform far worse than average. We refer to these subsets of the data as *error-prone subsets*. We first evaluated the performance of both individual systems and a simple average score-fusion system for the error-prone subsets. The subsets were chosen from the data by thresholding the following three indicators:

$$\begin{aligned} a_1 &= \mu_t \\ a_2 &= \sigma_t \\ a_3 &= \frac{\mu_t - \mu_e}{\mu_t} \end{aligned} \tag{4}$$

where μ_t is the average pitch of the training data, μ_e is the average pitch of the test data, and σ_t is the standard deviation of training pitch (indicating how consistently high the pitch was during training). a_3 indicates the pitch mismatch (relative change in average pitch between training and test).

An example for the results obtained on such a subset is shown in Fig. 9. The subset is $a_1 = 40$ samples, $a_2 = 6.5$ samples, and $a_3 = 0.1$. For this subset depitch obtains 75.1 DCF and 17.1% EER, nondep obtains 55.3 DCF and 16.7% EER, and the fusion 56.1 DCF and 13.2% EER. For this subset the fusion actually increases the DCF a little, yet the relative improvement in EER is 21%. This improvement in EER is consistent with the intuition of depitch tilting the DET plot clockwise, which is also apparent from the fact that the EER of the two systems is closer compared with their DCF. Choosing different subsets by modifying the values of a_1, a_2, and a_3 produced similar results. The overall performance (i.e., fusing the subset and taking nondep scores for the rest) is 44.4 DCF and 10.9% EER, which is slightly worse than

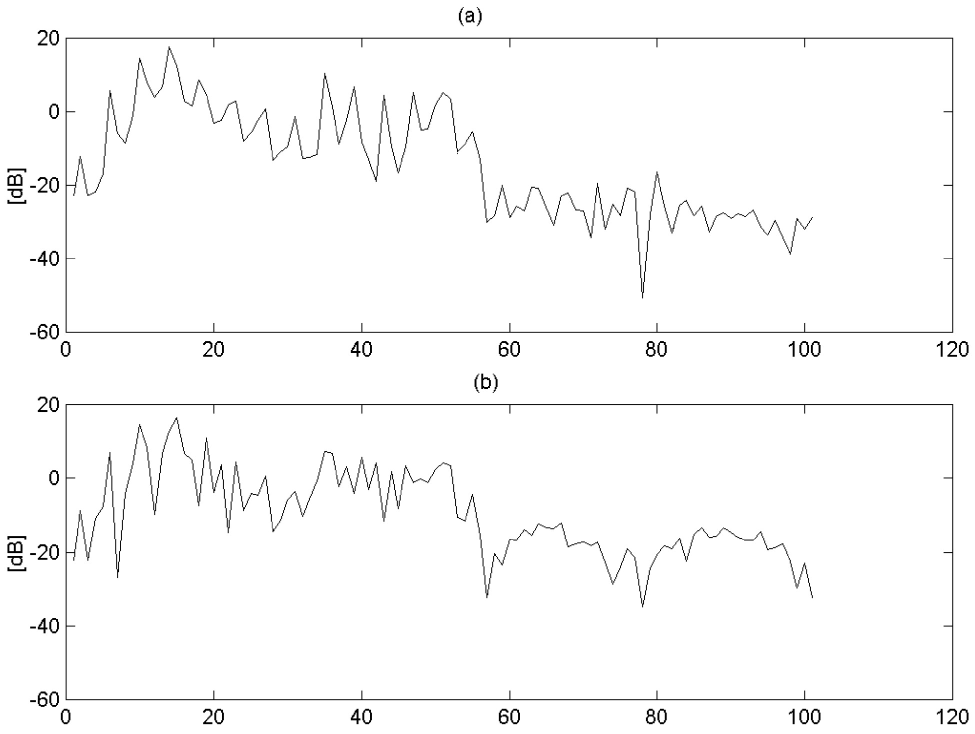


Fig. 8. Example of padpitch spectrum. (a) Original and (b) padpitched.

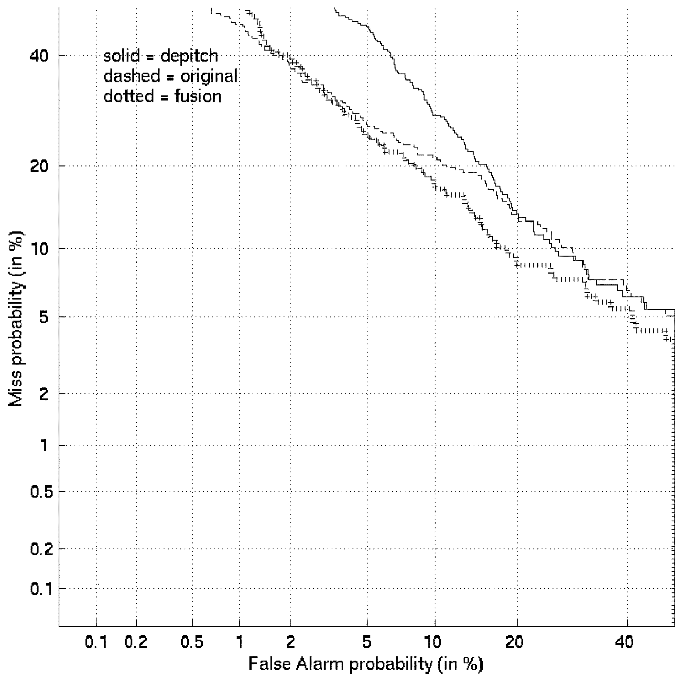


Fig. 9. DET plot for error prone subset.

the baseline nondep accuracy. In practice, this means that the amount of errors almost did not change; however, the distribution of errors between speakers and trials is more uniform.

To illustrate this, we calculated the standard deviation of true scores and imposter scores per speaker. Fig. 10 depicts the difference between nondep and the fused system in standard deviation of true and imposter scores per speaker. It is shown that the target scores' standard deviation was reduced, implying again a clockwise tilt of the DET plot, under the Gaussianity assumption [13]. Furthermore, a lower standard deviation of target scores for a particular speaker indicates that this particular speaker experiences a more consistent false rejection rate across trials. Similarly, a lower standard deviation of imposter scores per speaker indicates that imposters are experiencing a more consistent false acceptance rate between trials for that speaker. It is clear from Fig. 10 that while the imposter experience changes differently for different speakers, the false rejection experience of most individual speakers becomes more consistent as a result of the score fusion. We were not able to explain the anomaly in Fig. 10 where the standard deviation for one speaker increased significantly.

Performing score fusion on a subset of the data provides good insight, but mandates some sensitivity to the choice of indicator thresholds that define the subsets. It is therefore preferable to perform score fusion on all scores. The pitch value and mismatch can still be accounted for, for example by calculating the fused score as a weighted average of the two scores, with weights that depend on the above indicator values. Table I lists the DCF and EER obtained for different choices of constant and indicator-dependent weights w_1 and w_2. The last entry in Table I is for linear score fusion using constant weights obtained using the pDETAC optimization method described in [13]. Table I demonstrates how minor clockwise tilts of the DET plot are obtained by fusing all scores. For example for the pDETAC score fusion the EER improved by 4% (relative) and the DCF is worse by 3%. It should be noted however, that for the subsets of interest a significant improvement in EER will be present, as described above. This implies both robustness to pitch mismatched target trials and a more uniform distribution of the errors across

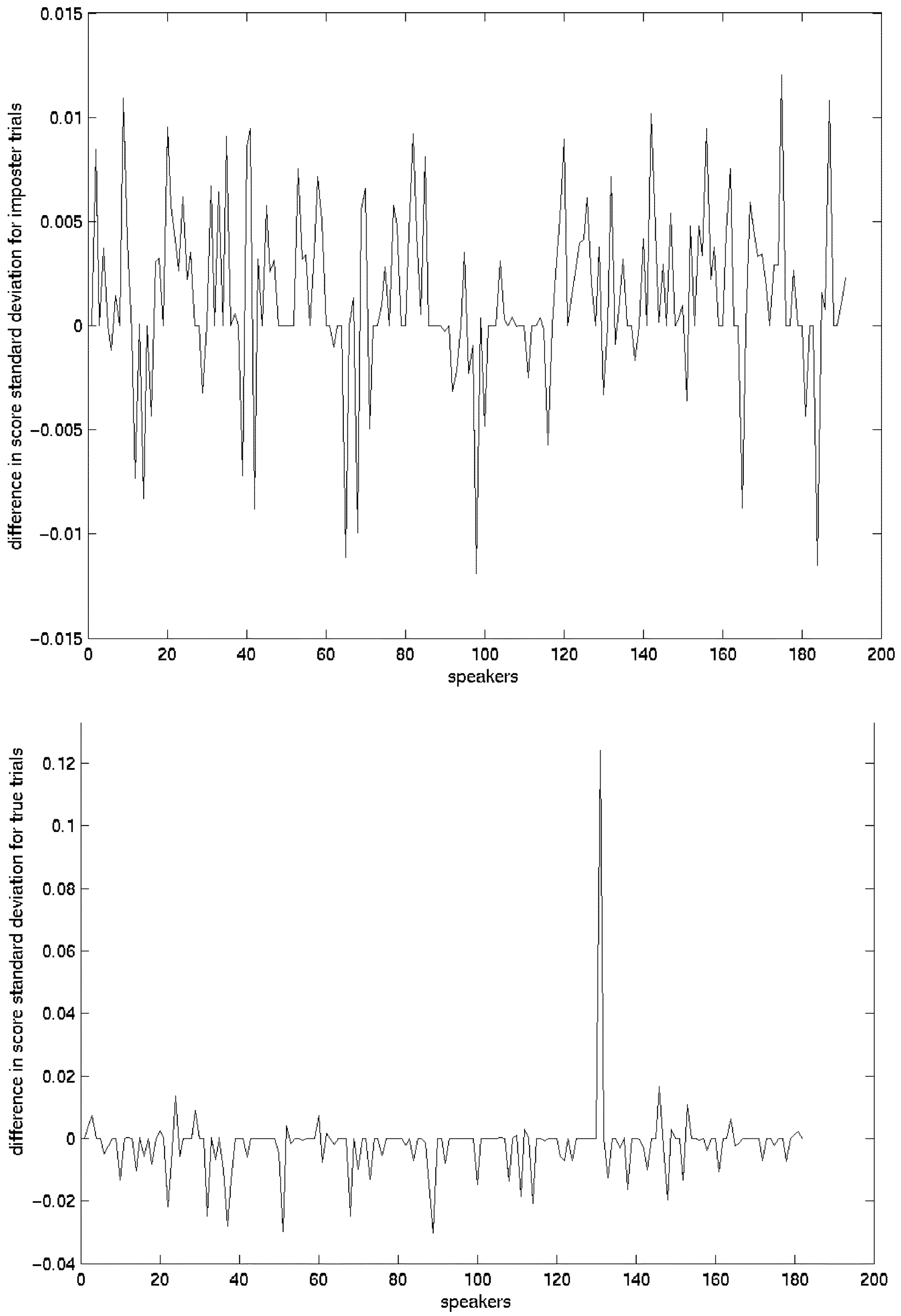


Fig. 10. Difference in score standard deviation per speaker for imposter trials.

speakers. For example, for the pDETAC score fusion the standard deviation of the EER across speakers drops from 8.6% to 8.1% versus the original nondep system.

For completeness, we also used a simple system based on the utterance level pitch statistics, where the score is computed as the exponent of the symmetric Kullback–Liebler divergence; see (5), shown at the bottom of the next page, where μ_{utt} and σ^2_{utt} are the mean and variance of the pitch cycle of the test utterance, and $\mu_{speaker}$ and $\sigma^2_{speaker}$ are the mean and variance of the claimant speaker's pitch in the training data. This trivial system performs better than random (36.9% EER), and when fused with depitch and nondep using the optimal pDETAC weights obtains 42.8 DCF and 10.3% EER – again a clockwise DET plot tilt.

Finally, in an attempt to bring the depitch performance closer to the nondep system we tried improving the depitch performance by applying the TNORM technique to the depitch scores [14], using 234 independent speakers. TNORM uses a set of held-out speakers to calculate a set of imposter scores for every trial, used for performing mean and variance normalization of the trial score. It was shown to provide accuracy improvement

TABLE I
RESULTS FOR DEPITCH SCORE FUSION ON THE ENTIRE TEST DATA

w_1	w_2	DCFx103	EER [%]
0.5	$1-w_1$	49.5	10.9
0.25	$1-w_1$	44	10.2
$\lvert a_3 \rvert$	$1-w_1$	42	10.9
$(a_3+\frac{a_1}{a_2})^{-1}$	$1-w_1$	43.1	10.3
0.1383	0.5043	43.9	10.3

TABLE II
WORD ERROR RATES (%) FOR THE BASELINE AND x-PITCH DIGIT RECOGNIZERS ON TEST SET C OF THE AURORA-2 NOISY DIGITS TASK

System	-5 dB	0 dB	5 dB	10 dB	15 dB	20 dB	Clean
Baseline	64.6	26.0	9.1	3.9	2.1	1.5	0.7
Baseline, 5-frame Δ	75.8	32.4	11.0	4.4	2.3	1.7	0.7
Depitch	77.9	38.5	15.5	8.0	5.1	3.5	1.6
Syncpitch	60.5	26.9	10.4	5.2	2.7	1.9	0.9
Padpitch	60.7	25.1	9.2	4.3	2.4	1.8	0.8
ROVER	61.6	24.7	9.2	4.0	2.1	1.7	0.6

and a counterclockwise tilt of the DET curve [15]. The TNORM depitch system obtains 54.9 DCF and 16.0% EER. Combining this system with the nondep system using the optimal pDETAC weights obtains 40.1 DCF and 10.2% EER, a relative 5% accuracy improvement for both DCF and EER. We also ran TNORM on the nondep scores, which produced 39.3 DCF and 10.2% EER. Combining with the depitch TNORM system using several different constant weights produced no performance improvement.

2) Experiments Using Syncpitch: The syncpitch experiments were conducted using the complete NIST 2002 Speaker Recognition Evaluation corpus: 330 target speakers and 39 105 test trials (male and female). All the systems that were used have the same attributes as described in Section IV-A1, and vary only by the type of data used. All of the systems use TNORM score normalization with a set of 234 imposter models.

Four systems were implemented and tested for the purpose of testing the usefulness of the syncpitch method. Two systems serve as a baseline system, while the other two are identical except that all of the speech data is preprocessed using syncpitch (background, target, and testing).

Baseline1: UBM trained on the following.

1) NIST 2001 cellular development set (consisting mostly of GSM cellular calls recorded in the United States).
2) NIST 1996 evaluation set, consisting of landline telephone speech, run through three different GSM coders (full rate, half rate, enhanced half rate). The use of cellular coders on landline telephone speech aims at matching the background data to the expected training and test data (e.g., [16], [17]).

Syncpitch1: Same as baseline 1, only syncpitched data (background, target, and test data).

Baseline2: UBM trained on the following:

1) NIST 2001 cellular development set, consisting mostly of GSM U.S. cellular calls.
2) NIST 1996 landline evaluation set, run through a CDMA cellular coder (U.S. IS-95 standard).

Syncpitch2: Same as baseline 2, only syncpitched data (background, target, and test data).

An initial experiment on the 2002 NIST evaluation dataset (with a configuration that does not use TNORM) found syncpitch to be worse than a corresponding baseline system (DCF and EER of 50.1 and 11.7% versus 42.2 and 10.7%). Adding TNORM score normalization to both baseline and syncpitch systems reduces this performance gap, as described in detail below. The results are shown in Fig. 11. Baseline1 obtains a DCF of 32.7 and an EER of 8.8%, while syncpitch1 performs slightly worse: 36.7 and 9.6%. However, the combination of the two systems (obtained by simple averaging the utterance level scores) provides a DCF of 31.4 and an EER of 8.6%, a 4% relative improvement in DCF and 3% in EER. It is important to note that this improvement is achieved with no additional data, but simply by processing all the existing data using syncpitch. The baseline2 system obtains a DCF of 35.24, and EER of 9.2%. Syncpitch 2 has a slightly better EER, 9.0%, and a slightly worse DCF, 36.3. The simple average combination yields a relative improvement of 6% in DCF and 10% in EER (33.2, 8.3%). Again, the improvement is obtained using the combination without adding data. The fact that the baseline systems outperform the syncpitch systems may be attributed to the loss of high frequency energy in Stage II of the process (interpolation). However, a simple average combination was found to be beneficial, indicating that for some trials the syncpitch system outperforms the baseline system, and that the errors are relatively uncorrelated. It is probable that many errors of the syncpitch system are related to the interpolation loss described in Section II and that for those same trials the baseline systems perform better. On the other hand, it is also probable that many errors of the baseline systems are related to the spectral distortion that syncpitch is designed to reduce, resulting in error decorrelation between the baseline and syncpitch systems.

3) Experiments Using Padpitch: Padpitch experiments were run on the complete NIST 2002 evaluation data set. The systems that were used have the same attributes as described in

$$\exp\left\{-\frac{1}{2}\cdot\left[\left(\frac{\sigma^2_{utt}}{\sigma^2_{speaker}}+\frac{\sigma^2_{speaker}}{\sigma^2_{utt}}-2\right)+(\mu_{utt}-\mu_{speaker})^2\cdot\left(\frac{1}{\sigma^2_{utt}}+\frac{1}{\sigma^2_{speaker}}\right)\right]\right\} \tag{5}$$

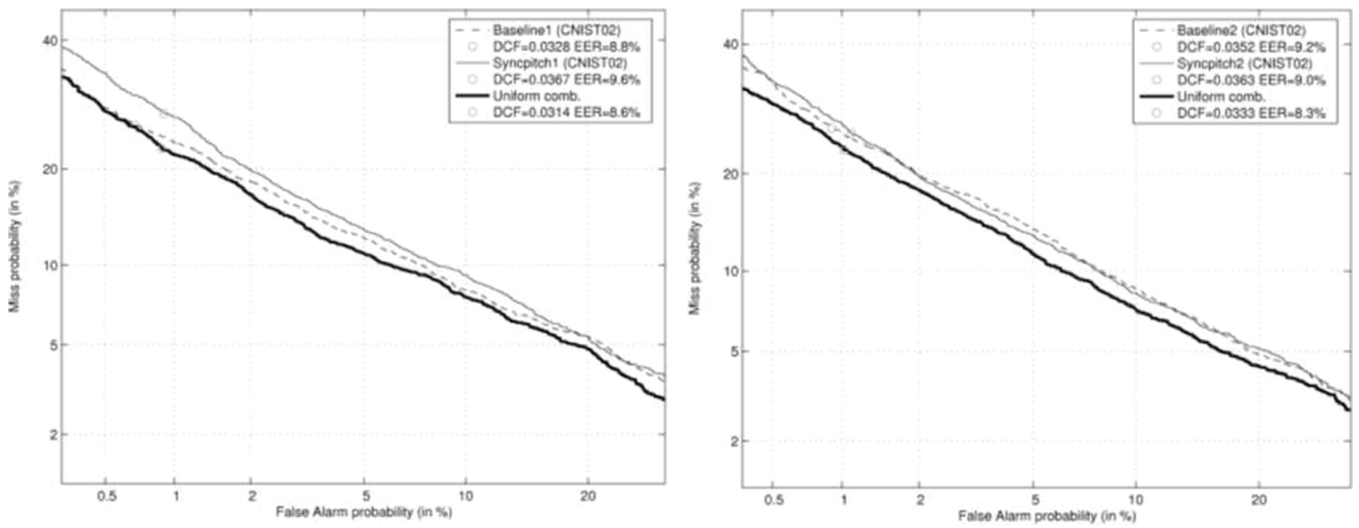


Fig. 11. Performance of syncpitch and baseline systems: (a) system 1 and (b) system 2.

Sections IV-A1 and IV-A2 in terms of GMM and front-end configuration. First, no TNORM score normalization was used. The baseline system with no padpitch processing obtains a DCF of 42.6 and EER of 10.7%. The corresponding padpitch system yields 57.5 DCF and 13.3% EER. Due to this significant relative degradation in performance the padpitch method has not been further tested using TNORM and score fusion, and is assumed not to be useful for speaker verification.

B. Speech Recognition Experiments

We tested the three PPS algorithms on the Aurora-2 noisy digits task [18]. Both the training and test sets use material from the TIDigits corpus [19], downsampled to 8 kHz, filtered to simulate a telecommunications channel, and mixed with realistic noise at a given SNR. We used the multi-condition training set, which consists of 8440 utterances from 110 adult speakers (55 male and 55 female). The 8440 utterances are split into 20 subsets of 422 utterances each, where each subset has one of four noises added to it at one of five possible SNRs (no noise, 20 dB, 15 dB, 10 dB, or 5 dB). For testing, we used test set C, which contains two different noises, one of which is present in the training set and one of which is not, added at seven different SNRs (no noise, 20 dB, 15 dB, 10 dB, 5 dB, 0 dB or −5 dB). Material in test set C also contains a channel mismatch versus the training set.

Our goal in the speech recognition experiments was not to evaluate the suggested algorithms for robust speech recognition, but rather to perform an initial evaluation of the PPS algorithms on a widely used task that is reasonably challenging. As such, the results that were of the most interest to us were the low-noise results. However, we are presenting results in all conditions because of an interesting effect we found in the very high-noise conditions.

We compared the performance of four different speech recognition systems trained on the Aurora-2 multi-condition training set and tested on the Aurora-2 test set C. The four recognizers differed only in the way the input signal was processed prior to feature computation. The baseline system performed no processing, while the three test systems used depitch, syncpitch, or padpitch processing to reduce the effects of pitch structure on the recognition features. The features are 13 MFCC features computed using the following steps.

1. The input signal is segmented into 25-ms. windows with an overlap of 10 ms. between windows.
2. Mean removal and linear de-trending are performed on each window separately.
3. The magnitude spectrum for each window is computed by weighting with a Hamming window, zero-padding to 256 samples, and applying an FFT. A small constant is added to each spectral magnitude to avoid problems caused by long stretches of digital zeros in the input.
4. The spectral magnitudes are binned into 24 bins using a Mel filter bank. The filters are equally spaced on the Mel frequency scale, and have a triangular shape when the filter weights are plotted as a function of linear frequency.
5. MFCC coefficients are computed by taking the log of the binned spectral values and then taking the DCT. Only the 13 lowest-order coefficients (including $c0$) are retained.

The recognition features are the 13 MFCC features plus their first and second temporal derivatives. Prior to the computation of the derivatives, the static MFCC features were normalized. Coefficients $c1$-$c12$ were mean-normalized. $c0$ was normalized by finding the maximum value of $c0$ within an utterance and subtracting it from all frames in the utterance. The first temporal derivatives are computed from the static MFCC features by linear regression over a 9-frame window centered on the current frame. Likewise, the second temporal derivatives are computed from the first derivatives by performing linear regression over a 9-frame window centered on the current frame. A more typical window size for the regression is 5 frames, but as will be demonstrated, the longer window yields better performance on the Aurora-2 task.

In the speech recognition system we tested, words are modeled as phone sequences using an alphabet of 22 phones. Each phone is modeled by a three-state, left-to-right HMM. The subphonetic states in the HMMs are context-dependent, with variants identified using decision trees that ask questions about the identity of neighboring phones within an 11-phone window. Cross-word context dependency exists only for words to the left, that is words that precede the current word. The decision trees define a set of 127 context-dependent sub-phonetic states, each of which is modeled by a mixture of up to 128 diagonal-covariance Gaussians. The total number of mixture components in the system is about 9500. All four systems use exactly the same set of context-dependent models (and, therefore, the same decision trees) and the same allocation of mixture components to context-dependent states. The decision trees and mixture allocation come from the Aurora-2 system described in [20]. The acoustic models for all systems are trained using four iterations of EM. In the first iteration, the state posterior probabilities are computed using the MMI-trained models and the features from [20], but the sufficient statistics are accumulated in the new feature space. This process is commonly called single-pass retraining. The three subsequent training iterations used the standard forward-backward algorithm. Recognition is performed using an unpruned Viterbi search on a precompiled decoding graph. Acoustic likelihoods are scaled by 1/15, and a word-insertion penalty of -1 is used. There are no grammatical constraints in the search: any word may follow any other.

Table II summarizes the results of our speech recognition experiments. A comparison of the baseline system and the baseline with a 5-frame window used for the computation of the temporal derivatives shows that the use of the longer 9-frame window leads to consistent performance improvements across conditions, with the exception of the clean condition, where there is no difference in error rates. The difference in performance attributable to the different window sizes increases with decreasing SNR. The choice of window over which to perform linear regression in the temporal derivative computation requires a trade-off between the bias and variance of the derivative estimates. Longer windows reduce the variance of the estimate, but also lead to inaccurate derivative estimates for rapidly changing signals. For the small-vocabulary Aurora-2 task, the more stable estimates provided by a longer window are advantageous. Note that all of the x-pitch systems also used the 9-frame window to compute derivatives.

Comparing the baseline and x-pitch results, it is clear that the lossiness of the pseudo pitch synchronous processing has a strong impact on speech recognition performance. The depitch system, which uses the lossiest algorithm, is worse than the baseline in all conditions. In contrast, the syncpitch system outperforms the baseline in the noisiest condition (-5 dB SNR), but is less accurate than the baseline in the other conditions. The least lossy algorithm, padpitch, outperforms the baseline in the -5 dB and 0 dB SNR conditions, matches the baseline in the 5 dB SNR condition, and is less accurate in the other conditions.

Inspired by the speaker recognition experiments, in which the combination of standard and syncpitch models through score fusion yielded improved performance, we tested a simple combination of standard, syncpitch and padpitch systems for speech recognition. The combination was performed with ROVER [21]. For each test utterance, the hypothesized word strings from each of the three systems were aligned against one another, then within a single alignment bin, the most frequent word in the bin was selected as the final hypothesis. The results of this test are also presented in Table II in the row labeled "ROVER." In two conditions (0 dB SNR and clean) the ROVER combination is more accurate than any of the component recognition systems, while in the other conditions one of the component recognizers is better. This suggests that the three recognition systems (baseline, syncpitch and padpitch) tend to have correlated errors, and that system combination is not a promising strategy for using PPS processing for speech recognition.

In summary, we only observe improved recognition accuracy for the syncpitch procedure in the noisiest test condition (-5 dB SNR) and for the padpitch procedure in the two noisiest test conditions (-5 dB and 0 dB SNR). We investigated this performance improvement in high-noise conditions to try to determine its origin. We first checked to see if the padpitch improvements in the -5 dB and 0 dB SNR conditions were gender-dependent. This hypothesis was based on the observation the fusion of the nondep and depitch systems was found to give a more consistent false rejection rate for high-pitch speakers in the speaker recognition tests. A breakdown of the baseline and padpitch systems' results by gender showed that there was no consistent dependence between gender and system performance, however.

We next checked to see if improvement in noisy conditions was simply a result of the padpitch system's producing acoustic likelihoods with a different dynamic range than the baseline system. We considered this possibility after noting that in the -5 dB SNR condition the baseline and padpitch systems had very different distributions of error types. The baseline system had an overall word error rate of 64.6%, comprising 13.7% substitutions, 48.7% deletions and 2.3% insertions, while the padpitch system had an overall word error rate of 60.7%, comprising 17.7% substitutions, 39.4% deletions and 3.6% insertions. These very different error distributions can be caused by the different systems producing acoustic likelihood scores with different dynamic ranges. Such a difference can be compensated by adjusting the word insertion penalty in the decoding. We therefore tested a number of different word insertion penalties with the baseline system to see if we could reproduce the padpitch results (both absolute word error rates and the distribution across error types). If we could reproduce the padpitch results in the baseline system simply by changing the baseline's word insertion penalty, this would strongly suggest that the difference between the systems was simply a matter of different dynamic ranges for the acoustic likelihoods. However, we were not able to reproduce the padpitch results with the baseline system by adjusting the insertion penalty. We therefore conclude that some other effect of the padpitch analysis leads to better performance in the -5 dB SNR condition.

Finally, we examined a number of individual test utterances for which the padpitch processing had led to better recognition performance. We found that in the -5 dB condition, only a very small number of frames had voicing scores high enough to be altered by the padpitch processing. In a number of the test utterances examined, we found that none of the frames altered by the padpitch processing aligned to words corrected by padpitch processing. From this we conclude that the improvements in the -5 dB SNR condition are due to changes to the global scores of different hypotheses rather than simply local changes in the scores of certain frames. Unfortunately, while these tests help to characterize the effects of padpitch processing in noisy conditions, they still do not provide an explanation for the improved performance.

V. Conclusions and Future Work

This paper introduced a family of pitch adjustment algorithms referred to as pseudo pitch synchronous (PPS) speech processing. The proposed algorithms adjust the presence of pitch information in the speech signal either by removing pitch information (depitch), interpolating pitch cycles of the LPC residual signal (syncpitch) or zero padding incomplete pitch cycles (padpitch). PPS algorithms were tested for a telephone text independent speaker verification task and for a speech recognition task in clean and noisy conditions.

It is shown that the presence of pitch information in the MFCC features carries a good deal of speaker-specific information. For speaker recognition, the removal of pitch information from the speech signal using depitch causes a substantial accuracy degradation. However, depitch is useful for pitch-mismatched subsets, where combining scores obtained on depitched speech with baseline system scores provides a significant improvement: from 17.1% EER to 13.2% EER. It is also shown that by combining scores obtained on depitched speech, the distribution of errors across speakers and trials becomes more uniform, with almost no loss in the overall accuracy. This is an important property in field deployments of speaker recognition technology where individual speaker experience is as important as the average user error rate. The tested syncpitch speaker recognition systems performed consistently worse than their corresponding baseline systems. However, a simple score average combination between the syncpitch and baseline systems yielded a performance improvement of up to 10% relative, implying a low correlation between the errors of the two systems. Using a combination of syncpitch with a conventional system on the 2002 NIST evaluation complete dataset, a best DCF of 31.4 and a best EER of 8.3% were obtained (for two separate systems). The padpitch technique does not seem to be useful for speaker recognition. Overall, PPS processing of speech was found to be useful for speaker recognition. This is also supported by a recent study on pitch synchronous processing [22] that has shown accuracy improvements over conventional MFCC processing.

For speech recognition on the Aurora-2 noisy digits task, the depitch system performed consistently worse compared to the baseline system, and the syncpitch system performed slightly better in the -5 dB condition, but performed worse in other conditions. The padpitch algorithm provided improved accuracy in the -5 dB and 0 dB tasks, same performance in the 5 dB task, and degraded performance in the other conditions. This suggests that the distortions introduced by the x-pitch processing have a greater impact on speech recognition performance than variability in the MFCC features stemming from pitch variation. The speech recognition experiments used 13 MFCC coefficients compared to 19 coefficients used for speaker verification, and therefore are predicted to have less pitch information present initially. However, re-running the recognition experiments with 19 dimensional MFCC produced very similar results, implying that pitch information is still strongly present in the lower dimension MFCC coefficients. A ROVER system combination did not provide any substantial accuracy improvement, unlike for speaker recognition where combining PPS systems with a baseline system provided performance improvement. The reason for the speech recognition performance improvement when using PPS processing in noisy conditions is not clear. However, we observe the following: Depitch is the worst performing technique, followed by syncpitch and padpitch, implying that a milder PPS technique can be more beneficial in improving speech recognition performance. It was also found that in the noisier conditions fewer frames were processed by the padpitch algorithm compared to the cleaner conditions (due to the lower values of voicing scores). This may suggest that the improvement in the noisier conditions is simply related to the PPS processing was less lossy because of the lower frame conversion rate. We therefore conclude that PPS techniques probably should be applied selectively to certain frames rather than to all voiced frames. We also find the fact that the PPS techniques are more useful for speaker recognition rather than speech recognition to reinforce this conjecture. The reason is that in the speech recognition system errors propagate across frames (because of the HMM structure), and therefore degraded likelihoods in the beginning of an utterance will affect likelihoods at the utterance end, whereas the text independent speaker recognition system is more tolerant of individual frame errors. Future work will test performing PPS processing only in a subset of the frames where the expected spectrum estimation error is significant (due to the incorrect alignment of pitch cycles to the frame size).

References

[1] T. F. Quatieri, R. B. Dunn, and D. A. Reynolds, "On the influence of rate, pitch, and spectrum on automatic speaker recognition performance," in *Proc. ICSLP*, 2000.

[2] T. F. Quatieri, R. B. Dunn, D. A. Reynolds, J. P. Campbell, and E. Singer, "Speaker recognition using G.729 speech CODEC parameters," in *Proc. ICASSP*, 2000.

[3] G. R. Doddington, M. A. Przybocki, A. F. Martin, and D. A. Reynolds, "The NIST speaker recognition evaluation – Overview, methodology, systems, results, perspective," *Speech Commun.*, vol. 31, no. 2–3, pp. 225–254, Jun. 2000.

[4] . [Online]. Available: http://www.nist.gov/speech/tests/spk/index.htm

[5] A. Martin and M. Przybocki, "The NIST 1999 speaker recognition evaluation – An overview," *Digital Signal Process.*, vol. 10, no. 1, pp. 1–18, 2000.

[6] G. Saon, M. Padmanabhan, and R. Gopinath, "Eliminating inter-speaker variability prior to discriminant transforms," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, 2001.

[7] W. Hess, *Pitch Determination of Speech Signals*. Berlin, Germany: Springer Verlag, 1983.

[8] R. D. Zilca, J. Navratil, and G. N. Ramaswamy, "Depitch and the role of fundamental frequency in speaker recognition," in *Proc. ICASSP*, 2003.

[9] ——, "Syncpitch: A pseudo pitch synchronous algorithm for speaker recognition," in *Proc. Eurospeech*, 2003.

[10] G. Doddington, W. Liggett, A. Martin, M. Przybocki, and D. Reynolds, "Sheep, goats, lambs and wolves: A statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation," in *Proc. ICSLP*, 1998.

[11] *Proc. 2002 NIST Speaker Recognition Evaluation Workshop*, May 2002.

[12] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET curve in assessment of detection task performance," in *Proc. Eurospeech*, 1997.

[13] J. Navratil and G. N. Ramaswamy, "DETAC: A discriminative criterion for speaker verification," in *Proc. ICSLP*, 2002.

[14] M. J. Carey, E. S. Parris, H. Lloyd-Thomas, and S. Bennett, "Robust prosodic features for speaker identification," in *Proc. ICSLP*, 1996.

[15] J. Navratil and G. N. Ramaswamy, "The awe and mystery of T-norm," in *Proc. Eurospeech*, 2003.

[16] G. N. Ramaswamy, J. Navratil, U. V. Chaudhari, and R. D. Zilca, "The IBM system for the NIST-2002 cellular speaker verification evaluation," in *Proc. ICASSP*, 2003.

[17] R. D. Zilca, U. V. Chaudhari, and G. N. Ramaswamy, "The sphericity measure for cellular speaker verification," in *Proc. ICASSP*, 2002.

[18] H.-G. Hirsch and D. Pearce, "The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *Proc. ISCA ITRW ASR2000*, 2000.

[19] R. G. Leonard, "A database for speaker independent digit recognition," in *Proc. ICASSP*, 1984.

[20] G. Saon and J. M. Huerta, "Improvements to the IBM Aurora 2 multicondition system," in *Proc. ICSLP*, 2002.

[21] J. G. Fiscus, "A post-processing system to yield reduced word eror rates: Recognizer output voting error reduction (ROVER)," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, 1997.

[22] H. Nakasone, M. Mimikopoulos, S. D. Beck, and S. Mathur, "Pitch synchronized speech processing (PSSP) for speaker recognition," in *Proc. Odyssey Speaker and Language Recognition Workshop*, Jun. 2004.

[23] R. A. Gopinath, "Maximum likelihood modeling with Gaussian distributions for classification," in *Proc. ICASSP*, 1998.

Ran D. Zilca (M'92–SM'02) received the B.S. degree (honors) in electrical engineering from Ben Gurion University, Israel, in 1993, and the M.S. degree in electrical engineering from Tel Aviv University, Israel, in 1997.

From 1993 to 1999, he was with the Israel Defense Forces (IDF), where he managed different research and development groups working on interdisciplinary projects involving signal processing, pattern recognition, and data mining. From 1999 to 2001, he was a Research Scientist with the R&D Division, Amdocs, Israel, where he conducted research in text independent speaker recognition for landline and cellular environments, focusing on computationally efficient large population open-set speaker identification. During this time, he also managed a group of developers working on a robust voice portal for cellular providers. In 2001, he joined the Human Language Technology Department, IBM T. J. Watson Center, Yorktown Heights NY, where he is conducting research in robust and efficient speaker recognition and interaction models for secure remote user authentication. His current research interests include statistical pattern recognition, statistical modeling, and robust speaker verification and identification.

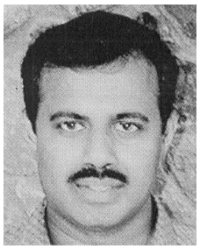
Brian Kingsbury (M'97) received the B.S. degree in electrical engineering from Michigan State University, Ann Arbor, in 1989 and the Ph.D. degree in computer science from the University of California, Berkeley, in 1998.

He is currently a Research Staff Member in the Human Language Technologies Department, IBM T. J. Watson Research Center, Yorktown Heights, NY. His research interests include acoustic modeling and robustness in large vocabulary conversational speech recognition systems.

Jiří Navrátil was born in 1971 in Hodonín, Czech Republic. He received M.Sc. and Ph.D degrees (summa cum laude) from the Ilmenau Technical University, Germany, in 1994 and 1999, respectively.

From 1996 and 1998, he was Assistant Professor at the Institute of Communication and Measurement Technology, Ilmenau Technical University, performing research on speech recognition and language identification. Between 1999–2000, he was a Visiting Scientist, and, since 2001, Research Staff Member in the Human Language Technologies Department, IBM Thomas J. Watson Research Center, Yorktown Heights, NY. He authored over 30 publications on language and speaker recognition and his current interests include methods for voice-based authentication, particularly conversational biometrics.

Dr. Navrátil is the recipient of the 1999 Johann-Philipp-Reis Prize awarded by the Informationstechniscne Gesellschaft im VDE (ITG), Deutsche Telekom, and the cities of Friedrichsdorf and Gelnhausen, Germany.



Ganesh Ramaswamy (M'88) received the S.B., S.M., and Ph.D. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, in 1992, 1993, and 1995, respectively.

In 1995, he joined the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, where he is now the Manager of the Conversational Biometrics and MultiMedia Mining Group, Human Language Technologies Department. His research interests include pattern recognition, statistical signal processing, machine learning, system identification, voice biometrics, multibiometrics fusion, user modeling and personalization, speech coding, speech analytics, multimedia mining, data compression, wireless communication, speech recognition, language modeling, natural language understanding and dialog management. He has filed more than 30 patent applications, and has published more than 40 conference and journal papers. He has received numerous invention achievement awards and technical group awards from IBM.

Dr. Ramaswamy is a member of Eta Kappa Nu, Tau Beta Pi, Sigma Xi, and Malaysian Mensa.