A Robust Front-End Algorithm for Distributed Speech Recognition

Yan Ming Cheng¹, Dušan Macho¹, Yuanjun Wei¹, Douglas Ealey², Holly Kelleher², David Pearce², William Kushner¹ and Tenkasi Ramabadran¹

Human Interface Lab, Motorola Labs
¹ Schaumburg, IL, USA; ² Basingstoke, UK
ycheng@labs.mot.com

Abstract

This paper presents the robust front-end algorithm that was submitted by Motorola to the ETSI STQ-Aurora DSR working group as a proposal for the Advanced DSR front-end in January 2001. The algorithm consists of a two-stage mel-warped Wiener filter, a waveform processor, a channel-normalized mel-frequency cepstral calculation and a subsystem of post-cepstral processing according to the reliability of mel-spectral components, etc. The output of this algorithm, a set of Mel-Frequency Cepstral Coefficients (MFCC), is compressed and encoded based on ETSI ES 201-108 standard; and then it is transmitted at 4800 bps. Compared to ETSI standard MFCC front-end, the proposed algorithm delivers an improvement of 42.64% in performance on the Aurora 2 database, which is required by this Eurospeech Special Event. With a very simple frame deletion algorithm based on a Voice-Onset Detection (VOD), the improvements were significantly boosted to 47.58% on the same database. In this paper we also give further insights about the proposal by providing performances and analyses with the Aurora SpeechDat-Car databases.

1. Introduction

As Automatic Speech Recognition (ASR) technology becomes more and more appealing to wireless applications, more emphasis has been placed on applications conducted in automobile environments or hands-free communication. In these situations the noise robustness determines the usability of ASR systems. The performance of current ASR systems radically deteriorates when input speech is interfered by noise (in most cases, background noise or background speech). This fact reduces the success of an ASR system in real-world applications. The first standard for a Distributed Speech Recognition (DSR) front-end based on the Mel-Cepstrum was set by ETSI in Jan 2000. The ETSI STQ-Aurora DSR working group is now developing a standard for an Advanced Front-end, which will deliver significant performance improvements over the Mel-Cepstrum in background noise. In this paper we describe the recent (January 2001) submission by Motorola to the working group as the candidate for the advanced front-end algorithm standard.

In the next section, we give the full description of the proposed algorithm and the theoretical background of each of its major components. In the third section, we evaluate some practical issues, such as the resource consumption of its implementation, and experimental results of its performances. Finally, we give our conclusion and remarks.

2. Advanced front-end algorithm

The overall front-end system of our proposal is sketched in Figure 1.

![Diagram](https://via.placeholder.com/150)

Figure 1: The front-end system diagram

The following list denotes the acronyms used in the diagram.

- **2WF**: two-stage mel-warped Wiener filter,
- **SDs/EstSNR**: silence detectors and SNR estimation, respectively,
- **SWP**: SNR-dependent waveform processing,
- **CNMFCC**: channel-normalized MFCC,
- **AdjCC**: cepstral coefficient adjusting,
- **SM**: silence mixture,
- **CMP/DCMP**: compression/decompression,
- **ENC/DEC**: encoder/decoder, and
- **∆/∆∆**: Delta/acceleration calculation.

2.1. Two-stage mel-warped Wiener filter

This technology was original developed in [1]. Its purpose is to suppress noise, especially strong and colored noise, while overcoming some of the weaknesses of the original Wiener filter. At each stage the filter transfer function can be generically expressed as:

\[
H(m) = \lambda(m) \begin{pmatrix} \hat{R}_y(m) - \rho \hat{R}_n(m) \\ \hat{R}_n(m) \end{pmatrix}^T \phi(m)
\]

where: \(\hat{R}_y(m), \hat{R}_n(m)\) are the estimated spectra of autocorrelation sequences of noisy signal and noise, respectively, \(m\) is mel-index distributed uniformly on a mel-scale, \(g\) is the root compression constant and is determined...
experimentally, and $\rho$ is a constant between 0 and 1, which controls the aggressiveness of the noise suppression. At the first stage, $\rho$ is set to 0.7, which allows significant noise removal while introducing least distortion, due to the nature of Wiener filter. At the second stage, $\rho$ is varied according to the signal type of the current frame and an estimated global SNR. If the signal type of the current frame is noise, $\rho$ is between 0.6 and 1.0 and is linearly inverse-proportional to the SNR (note that the SNR is the estimated global SNR up to current frame). Otherwise, $\rho$ is set to 0.6. $\phi(m)$ is the parameter that controls the distortion introduced by Wiener filter. At the first stage, it is set to a constant of 0.5. At the second stage, it is a function of $m$, and it is inverse proportional to $m$ (when $m$ is small, it is close to 0.5, and when $m$ is large, it approaches 0). The rationale behind the function is that if a constant is used here at the second stage, it behaves like a traditional Wiener-filter, introducing white spectral noise as the spectral distortion. In other words, the distortion variance is constant across frequency. For a voiced signal, speech energy concentrates at low frequency. Thus, the signal-to-distortion ratio will be much lower at high frequencies than at lower frequencies, making distortion of the speech information more significant at these frequencies.

The function $\phi(m)$ will ensure a constant signal-to-distortion ratio across frequency instead of a constant distortion. From another angle, the real noise in wireless and automobile environments has energy concentrated at low frequency as well. This function also ensures that more noise suppression will occur in lower frequency range. $\lambda(m)$ is the parameter for extreme colored noise suppression. It can be expressed as:

$$
\lambda(m) = \frac{2\theta}{M} \sum_{w} \hat{R}_w(m) \cdot \hat{R}_w(m) > \lambda
$$

otherwise

$$
\lambda(m) = \frac{2\theta}{M} \sum_{w} \hat{R}_w(m) \cdot \hat{R}_w(m) > \lambda
$$

The behavior of this equation is that when an extreme peak occurs in estimated noise spectrum, the $\lambda(m)$ tends to close to zero around the peak’s frequency, or the Wiener filter suppresses any output.

### 2.2. SNR-dependent waveform processing (SWP)

This technology was originally developed in [2]. In SWP, the periodicity of the instant SNR in the voiced speech is exploited for enhancing the energy of the higher SNR waveform portions and reducing the energy of the lower SNR waveform portions. Given a voiced frame, the smoothed instant energy contour (dashed line) is first computed by using the Teager energy operator [3]. A moving-average operator smoothes the contour. Then, by using a simple peak-picking algorithm, the peaks of the instant energy contour are found. Note that the peaks are marked by the actual fundamental frequency. A rectangular window function $\pi(t)$ (dot-dashed line) is constructed by using the positions of peaks. On the right side of each peak, a rectangular window is placed with the width defined as a fixed percentage of the distance between the two actual peaks (this percentage was determined experimentally). The waveform portions selected by the rectangular windows are considered as higher SNR portions and the remaining waveform is considered as lower SNR. The improved SNR waveform is synthesized as:

$$
\hat{s}(t) = \alpha \cdot \pi(t) s(t) + \beta (1 - \pi(t)) s(t).
\alpha \geq 1.0, \quad 0 < \beta \leq 1.0
$$

where $s(t)$ is the output waveform of the two-stage Wiener filter, while $\alpha$ and $\beta$ are enhancing and attenuating constants, respectively, which were determined experimentally. This waveform processing was only applied to the speech frames detected by our speech/noise detectors.

### 2.3. Channel-Normalized Mel-Frequency Cepstral Coefficient (CNMFCC) calculation

In order to compensate for the channel diversity in speech recognition, a simple channel-normalization process is embedded into the conventional MFCC calculation [4]. The normalization process is a simple first-order low-pass filter applied to the output of each mel-filer-bank:

$$
\log \hat{E}(m,n) = \mu \log E(m,n) + (1 - \mu) \log E(m,n)
$$

where $m$ and $n$ are the mel-filer index and frame index, respectively. In order to ensure that this process only removes the long-term spectrum, the only parameter $\mu$ is set to a high value, e.g., 0.98. $E(m,n)$ is the output spectrum of the previous stages processing.

### 2.4. Post-cepstral processing for robustness

#### 2.4.1. Adjusting cepstral coefficient based on reliability of spectral components

![Figure 3: Unreliable spectral locations after noise suppression.](image)
to an inaccurately calculated MFCC. Figure 3 shows an illustrative example where there are the unreliable locations. The idea is to reduce the weighted contribution of the unreliable locations to the MFCC calculation. The approach employed here is to estimate MFCC through a weighted least-mean-square process based on our knowledge about the unreliable spectral locations, which are obtained during the Wiener filter process:

$$\hat{C}_j = \arg \min_{\hat{C}_j} \left\{ \Delta = \sum_{m=0}^{M-1} A(m) \left( \log \hat{E}(m) - \sum_{\text{reliable-locations}} \hat{C}_j \cos \frac{2\pi \cdot r \cdot m}{M} \right)^2 \right\}$$

(5)

$$A(m) = \begin{cases} 1.0 \text{ reliable-locations} \\ <0.9 \text{ colored – noise – peaks} \\ 0.9 \text{ unreliable-locations} \end{cases}$$

There is no closed form solution for the above equation. The approximation we take here is one-step gradient descent algorithm:

$$\hat{C}_j = C_j + \eta \frac{\partial \Delta}{\partial \hat{C}_j}$$

(6)

where \( C_j \) is CNMFCC, as described in previous paragraph, and \( \eta \) is the step size.

2.4.2. Silence mixture

The Aurora Speech-Dat-Car databases contain relatively long periods of noise ahead and after the utterance. These are the cause of excessive insertion errors when one evaluates this front-end algorithm with the database. An effective way of dealing with the long silences is to delete them, presented in the next section, but here we introduce another solution. The silence mixture is to use a weighting average to mix an estimated silence cepstral mean, given a partial utterance, with a silence frame currently detected. Such a mixture increases the uniformity of the cepstral coefficients in the silence segment so as to reduce the chance of insertion errors. The silence cepstral mean is obtained through a running mean silence segment so as to reduce the chance of insertion errors.

2.5. Cepstral coefficient compression-decompression and encoding-decoding

In order to transmit the cepstral coefficients through an erroneous channel, one has to provide both a compression-decompression algorithm and an encoding-decoding algorithm to fulfill the requirements of the ETSI Aurora working group. We use the split-VQ technology in [4] for the compression and decompression, and we use the error resilient technique in [4] to encode and decode the compressed cepstral coefficients. In total, we are able to transmit the cepstral coefficient vector sequence at 4800 bps.

3. Implementation and Experiments

3.1. Implementation and system latency

The 16-bits implementation complexity estimation of the proposed algorithm based on [5] is 7.1 weighted-MOPS. The ROM usage is 4.0 K-words (16 bits a word). The RAM usage is 2.1 K-words. The latency introduced by the proposed algorithm is 125 ms.

3.2. Experimental databases and conditions

The experimental databases and evaluation conditions and metrics are described in [6]. In brief, there are two types of databases: Aurora 2 database and multilingual car database (Italian, Finnish, Spanish, German and Danish). “SpeechDat-Car”. The evaluation conditions for Aurora 2 database are divided into multiple training and clean training; and that for “SpeechDat-Car” into WM (well matched), MM (medium mismatched) and HM (highly mismatched) conditions. A weighting scheme integrates individual performance into an overall performance figure. Our speech recognition experiments were done using HTK for both delta and acceleration calculations (delta span is 3), training and recognition with ETSI provided scripts. The proposed front-end algorithm delivers 12 cepstral coefficients and log energy. After generating 13 delta and 13 acceleration coefficients, there are 39 coefficients in total as input to speech recognizer.

Table 1: Performance improvements compared to the baseline

<table>
<thead>
<tr>
<th>Languages</th>
<th>Italian</th>
<th>Finnish</th>
<th>Spanish</th>
<th>German</th>
<th>Danish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train Mode</td>
<td>Set A</td>
<td>Set B</td>
<td>Set C</td>
<td>Overall</td>
<td></td>
</tr>
<tr>
<td>Multi-condition</td>
<td>27.5</td>
<td>25.66</td>
<td>33.08</td>
<td>28.11</td>
<td></td>
</tr>
<tr>
<td>Clean Only</td>
<td>57.74</td>
<td>60.08</td>
<td>48.28</td>
<td>57.17</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>42.66</td>
<td>42.87</td>
<td>40.68</td>
<td>42.64</td>
<td></td>
</tr>
</tbody>
</table>

Train Mode  SpeechDat-Car

Languages   | Italian | Finnish | Spanish | German | Danish |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Well-Match</td>
<td>24.5</td>
<td>37.9</td>
<td>46.3</td>
<td>12.2</td>
<td>27.6</td>
</tr>
<tr>
<td>Medium Mismatch</td>
<td>38.2</td>
<td>38.6</td>
<td>30.9</td>
<td>24.5</td>
<td>13.1</td>
</tr>
<tr>
<td>High Mismatch</td>
<td>61.8</td>
<td>60.8</td>
<td>64.9</td>
<td>41.7</td>
<td>4.3</td>
</tr>
<tr>
<td>Average</td>
<td>33.72</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.3. Experiments without and with Voice-Onset-Detection (VOD) based frame deletion

As an alternative approach to combat the very long preceding/trailing silences in databases, we have put together a back-end (sever-end) voice-onset-detector (VOD). This VOD is based on the log energy and me-scale spectra at each frame, converted from the output of our algorithm, i.e. the mel-frequency cepstral coefficients. Then, we delete the speech frames prior to the detected voice onset at each utterance, except the first 10 frames. Table 1 shows the performance improvements without deleting initial silences. Based on the ETSI Aurora formula, the overall performance improvement (combining Aurora 2 and SpeechDat-Car) is 37.29%. Table 2 shows the performance with the VOD, which gives a performance of 48.09%. Comparing Table 1 and Table 2, the overall performance improvements are changed from 37.29% to 48.09% due to the VOD. It also can be observed that the radical changes (or improvements) come from the multilingual database car database especially from Italian, Spanish and Danish databases (from 33.72% to 48.42%).
that the Aurora 2 database is converted from well-known TI-digits database that has relatively short silence periods before and after each utterance.

**Table 2:** Performance improvements with VOD-based frame deletion algorithm

<table>
<thead>
<tr>
<th>Training Mode</th>
<th>Set A</th>
<th>Set B</th>
<th>Set C</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicondition</td>
<td>45.94</td>
<td>50.23</td>
<td>43.82</td>
<td>47.56</td>
</tr>
<tr>
<td>Clean Only</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>45.94</td>
<td>50.23</td>
<td>43.82</td>
<td>47.56</td>
</tr>
</tbody>
</table>

4. Conclusions

In this paper, we have described the Motorola proposal for the ETSI advanced DSR front-end speech recognition. The proposal comprises four innovative blocks of processing: an enhanced two-stage mel-warped Wiener filter, SNR-dependent waveform processing, channel-normalized mel-frequency cepstral coefficient computation and post cepstral coefficient modification. We have listed the algorithms used in the proposal and explained the motivation behind each algorithm.

As our performances indicated, this proposal does enhance the robustness of front-end against interfering noise and acoustic channel variation. The proposal is also robust to the wireless channel distortion for the data transmission between client (terminal) and servers in the sense of distributed speech recognition. As a particular case, we have studied a critical component of speech recognition, voice activity detection (VAD), in the context of front-end. A partial realization of VAD, voice-onset-detection (VOD), has significantly improved the performances. The performance improvements are raised from 42.64% without VOD to a 47.58% with a VOD on the Aurora 2 database. When full ETSI databases are considered greater performance gains were observed. For the overall metric the improvements were raised from 37.29% without VOD to 48.09% with VOD.

In conclusion, this proposal delivers a definite robustness improvement of front-end compared to the standard MFCC algorithm.

5. References


