Available online at www.sciencedirect.com
ScienceDirect
Speech Communication 48 (2006) 1262–1272
SPEECH COMMUNICATION
www.elsevier.com/locate/specom

Dynamic programming approach to voice transformation

Özgül Salor *,[1], Mübeccel Demirekler

Department of Electrical and Electronics Engineering, Middle East Technical University, 06531 Ankara, Turkey

Received 7 June 2005; received in revised form 14 April 2006; accepted 19 June 2006

Abstract

This paper presents a voice transformation algorithm which modifies the speech of a source speaker such that it is perceived as if spoken by a target speaker. A novel method which is based on dynamic programming approach is proposed. The designed system obtains speaker-specific codebooks of line spectral frequencies (LSFs) for both source and target speakers. Those codebooks are used to train a mapping histogram matrix, which is used for LSF transformation from one speaker to the other. The baseline system uses the maxima of the histogram matrix for LSF transformation. The shortcomings of this system, which are the limitations of the target LSF space and the spectral discontinuities due to independent mapping of subsequent frames, have been overcome by applying the dynamic programming approach. Dynamic programming approach tries to model the long-term behaviour of LSFs of the target speaker, while it is trying to preserve the relationship between the subsequent frames of the source LSFs, during transformation. Both objective and subjective evaluations have been conducted and it has been shown that dynamic programming approach improves the performance of the system in terms of both the speech quality and speaker similarity.

Keywords: Voice transformation; Speaker transformation; Codebook; Line spectral frequencies; Dynamic programming

1. Introduction

The aim of voice transformation (VT) is to modify the speech of a source speaker such that it is perceived as if spoken by a target speaker. A considerable amount of effort has been dedicated to the problem of voice transformation in the last two decades (Abe et al., 1988; Valbret et al., 1992; Childers, 1995; Mizuno and Abe, 1995; Lee et al., 1995; Stylianou et al., 1998; Arslan, 1999; Kain, 2001). There are various applications of a VT system. Using VT technology, new synthesis voices can be created by transforming the voice of the existing inventory to a new speaker's voice in a text-to-speech system. VT system would require a much smaller inventory than the original text-to-speech inventory, which saves time and disk space. Another application can be developing the voice of a speaking-impaired person, who can provide limited amount of speech data. A VT system could also be used as a preliminary step to speech recognition to reduce speaker variability.

In general, all VT systems have two modes: training and transformation. In the training mode, the

* Corresponding author. Tel.: +90 312 2101310; fax: +90 312 2101315.
E-mail address: ozgul.salor@bilten.metu.edu.tr (Ö. Salor).
[1] Present address: Institute of Space Technologies Research, TÜBİTAK, METU Campus, Ankara, Turkey.

0167-6393/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.specom.2006.06.003

system uses source and target speech inventory to estimate a transformation function that maps the acoustic space of the source speaker to that of the target speaker. Once the training is achieved, the system is ready to transform the source speaker's speech to the target speaker's speech. The acoustic space of the speakers can be represented by various acoustic features. Formant frequencies (Abe et al., 1988; Mizuno and Abe, 1995), LPC cepstrum coefficients (Lee et al., 1995; Stylianou et al., 1998), and line spectral frequencies (Arslan, 1999; Kain, 2001; Salor et al., 2003) have been used. The transformation function can be a continuous function applied to the features (Stylianou, 1999; Kain, 2001; Toda, 2003; Salor, 2005), or it can be a discrete mapping from the feature space of the source speaker to that of the target speaker (Abe et al., 1988; Arslan, 1999; Salor and Demirekler, 2004). The discrete mapping is in general a codebook mapping, in which a one-to-one correspondence between the spectral codebooks of the source speaker and the target speaker is developed. These methods usually face several problems such as degradation of the speech quality because the parameter space of the converted envelope is limited to a discrete set of envelopes. These methods may also result in high distortions between LPC spectrums of the neighboring frames due to independent transformation of the successive frames, which cause audible buzzy sounds or clicks.

In this work, we have aimed to obtain a voice transformation system inside the decoder part of a MELP speech coding algorithm. The idea is that the coded parameters could be used to produce the voice of another person at the end point of the coder. Therefore, we have focused on improving the quality of a codebook based voice transformation system. Here, we propose a dynamic programming approach to codebook based VT methods to overcome the problems of discontinuities and high distortions in speech. Dynamic programming approach considers the spectral distance between successive frames of the source speaker during transformation, while it is giving the chance to one of several target codewords to be selected at every frame instead of using a one-to-one mapping between the source and target speaker codewords. It has been observed that dynamic programming increases speech quality.

2. Algorithm description

This section provides a general description of the voice transformation algorithm. An overview block diagram of the VT system is given in Fig. 1. The speech model is based on the traditional *Linear Prediction Coding* (LPC) parametric model. The spectral characteristics are represented by line spectral frequencies (LSFs) and the spectral transformation from source to target is applied to the source speaker's LSFs. The reason for selecting LSFs is that these parameters are closely related to formant frequencies which carry speaker individualities (Arslan, 1999). LSFs have been used to represent the vocal tract parameters for VT throughout this work.

LPC residual is used as an approximation to the excitation signal during synthesis and average pitch of the excitation signal is modified such that the average pitch of the transformed sentence is the same with that of the target speaker. The algorithm will be described under two main sections: training and transformation.

2.1. Training

The output of the training part will be the histogram matrix, denoted by *Hist* and the target speaker's transition probability matrix denoted by T. Both of these matrices are square matrices of size $L \times L$, where L is the codebook size of both speakers. Details of obtaining these matrices are explained in detail in Sections 2.1.2 and 2.1.3 after

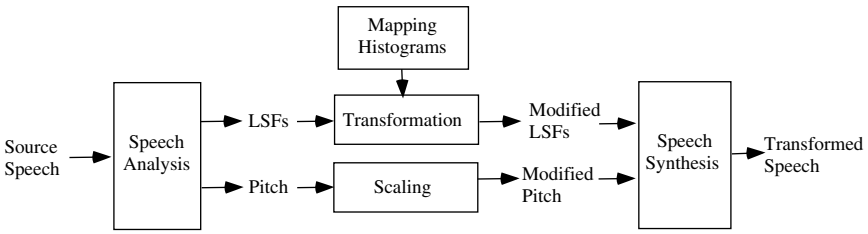

Fig. 1. Overview block diagram for the voice transformation system.

the explanation of the speech data and time alignment procedures.

2.1.1. Speech data and time-alignment

In natural speech, the durations of the speech units vary from speaker to speaker. During training, the system tries to estimate a transformation function which can predict the features of the target speaker from the features of the source speaker. Therefore, the feature streams from the source and the target speakers should be time-aligned to obtain the transformation function that gives the relationship between the source and target features of equal phonetic context. The goal of time-alignment is to modify the source and the target speaker feature streams in such a way that the resulting feature streams describe approximately the same phonetic content. Time-alignment in this work has been achieved by selectively deleting or repeating frames from the target speaker feature stream to match the number of source frames within phonetically equivalent regions, through a *dynamic time warping* (DTW) algorithm (Huang et al., 2001, page 383). The distance criterion used in the DTW algorithm is the Bark-weighted RMS error in dB between the power spectra of the two speakers at those frames (Cohn and Collura, 1997). Bark weighting has been reported to have greater correlation with subjective evaluations because of down-weighting the higher frequencies.

Speech data used for this VT research consists of 235 triphone-balanced sentences collected from two native male speakers of Turkish and the sampling frequency is 8 kHz. During training the orthographic transcription is available along with the training data. All sentences of both speakers in the speech database have been segmented in phoneme level using the phonetic aligner described in Salor (2005). The segmentation algorithm uses Melcepstrum coefficients and delta coefficients within an HMM framework. Silence frames at the beginning and end of the sentences are neglected during both training and transformation throughout this work. After the segmentation, time alignment between the speakers is achieved among the frame groups belonging to the same phoneme, which have been determined during the phonetic segmentation. Once the time alignment is achieved, both speakers have the same number of matched speech frames for each sentence.

Speech model used is based on the MELP coder (MELP, 1997) whose analysis and synthesis modules depend on the traditional LPC parametric model (Markel and Gray, 1976). A 10th order LPC analysis is performed on the input speech using a Hamming window of length 200 samples (for a sampling frequency of 8 kHz) centered on the last sample of the current frame. MELP frames are 180 sample non-overlapping frames, which are the same with the frames used during DTW for time-alignment described above in this section. Then the linear prediction coefficients are converted into LSFs. For each frame, an LSF vector f of length 10, is obtained. f is quantized using a multi-stage vector quantizer (MSVQ). The MSVQ codebook consists of four stages of 128, 64, 64, and 64 levels respectively. The quantized vector is the sum of the vectors selected by the search process, where one vector is selected from each stage.

In this work we have not considered increasing the segment length to increase the transformed speech quality from a length of 180 samples, because this would require a collection of more data from both the source and the target speaker for obtaining reliable mapping histograms. An extreme case for this recommendation of increasing segment size would be using very large units (words, even sentences) of the target speaker as in some corpus-based unit selection TTS systems. In that case, after automatic recognition and annotation of the speech of the source speaker, speech would be synthesized using the units from the corpus of the target speaker. This would increase the speech quality as in TTS systems. Also prosody characteristics of the target speaker would be automatically included into the system. However, this approach would require a huge corpus both for the source and the target speaker.

MSVQ codebook of MELP has been used for quantizing the LSF spaces of the speakers. A reduced set of L LSF vectors out of MELP's 4 stages of MSVQ specific to each speaker have been obtained to represent the LSF spaces of the speakers. Speaker-specific codebooks that contain L vectors are generated by selecting the most frequently used LSF combinations out of the first 3 stages for each speaker. From this analysis, two codebooks, C_X and C_Y, which contain L speaker-specific quantized LSF vectors are obtained for source and target speakers respectively. Note that same codebook size, L, is the same for both speakers in this work.

2.1.2. Obtaining the mapping histograms

LPC analysis has been applied to all frames of the training sentences. LSFs from all frames have

been quantized based on the source and target LSF codebooks, C_X and C_Y. Codeword indices from N training frames are collected in source and target index vectors, X and Y. X and Y are vectors of length N, which is the total number of training frames, and they contain the LSF codeword indices belonging to source and target frames which are time-aligned. From X and Y, an $L \times L$ histogram matrix which includes the occurrence numbers of the corresponding source and target codewords in the N-frame corpus has been obtained. The elements of the histogram matrix, $Hist(i,j)$, show how many times the LSF vector corresponding to the ith codeword of C_X encountered to the LSF vector corresponding to the jth codeword of C_Y in the training sentence set.

2.1.3. Obtaining the transition probabilities of the target speaker

Despite showing the spectral mapping between the source and target speakers, histogram matrices do not contain time information about the spectral changes. This information is modelled in this work as a probability transition matrix T of the target speaker. T is obtained only for the target speaker and it is an $L \times L$ matrix. $T(i,j)$ represents the transition probability from the target codeword y_i in one frame, to another target codeword y_j in the next frame. $T(i,j)$'s are the estimated probabilities based on the empirical probabilities obtained from the target data during training.

3. Transformation

Transformation of the LPC spectrum is achieved using the histogram matrix. Following sections present the baseline transformation method and the improved VT method based on dynamic programming approach.

3.1. The baseline system

The method for the baseline system is mapping the LSF vector corresponding to the ith index of the source to j_m,

$$j_m = \arg\max_j Hist(i,j) \tag{1}$$

which is the target index, that corresponds to the most frequently occurring index when the source index i occurs in the corpus. During synthesis, the mapped LSF vector is used to estimate the LPC filter of the target speaker. The average pitch value of the source speaker is modified in the MELP synthesis framework by modifying the frequency locations of the quantized FFT peaks of the residual signal. The modified residual and the transformed LPC spectrum are used to resynthesize speech. MELP applies linear interpolation between the LSFs of adjacent frames during synthesis. The main shortcoming of this mapping method is that the parameter space of the converted envelope is limited to a discrete set of envelopes, which reduces the voice quality of the transformed speech. This method may also result in high distortions between the LPC spectrums of the neighboring frames, which causes audible buzzy sounds or clicks. In Fig. 2 spectrograms of the output of the baseline transformation system and the original target speaker's utterance are given. Spectral discontinuities are observed in the converted speech. Also, limitation of the target spectrum has caused unvoiced frames to occur in the middle of the illustrated part of the converted speech.

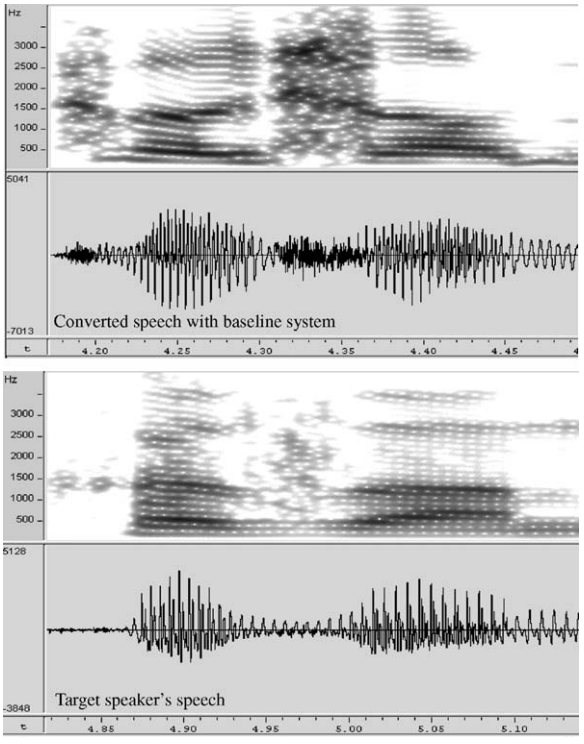

Fig. 2. Comparison of the transformed speech output of the baseline system and the target speaker's speech. Illustrated part is the word *kazandi* in Turkish. Sampling frequency is 8 kHz.

3.2. Dynamic programming for LSF transformation

In order to obtain a higher quality synthetic speech, we have applied a dynamic programming approach to determine the best target index that corresponds to the source index. This approach aims to reduce the distortion between the neighboring frames, while it is giving a chance to every target index, j, to be used depending on its occurrence rate corresponding to the ith index of the source speaker in the histogram matrix.

Dynamic programming approach starts by forming the histogram matrix of the given sentence, H^{sen}. H^{sen} is an $M \times L$ matrix where M is the number of frames in the sentence and L is the size of the LSF codebook as indicated before. It is obtained from the previously mentioned $L \times L$ histogram matrix, *Hist*, of the source and target speakers.

Let the source speaker's quantized LSF indices of one sentence be $X^{\text{sen}}_{1\times M}$, where M is the number of frames in the sentence. The aim here is to obtain the histogram matrix for the given sentence, H^{sen}. Every row of H^{sen} corresponds to a frame of the sentence and every column corresponds to one of the quantization levels of the target speaker. Rows of H^{sen} matrix are copied from the rows of the histogram matrix, *Hist*. Note that rows and columns of *Hist* correspond to quantization indices of the source and target speakers respectively. Let the codebook C_X of L codewords of the source be denoted by $C_X = [x_1, x_2, \ldots, x_L]$, and those of target $C_Y = [y_1, y_2, \ldots, y_L]$, where x_i and y_i represent LSF vectors in the codebook. The set of LSF indices obtained for the sentence of the source speaker are indicated by $x[n]$, whose elements are the indices of the LSFs in the codebook C_X. Here n is the frame number in the sentence, which runs from 1 to M. Using $x[n]$, the sentence dependent histogram matrix, H^{sen}, is obtained as:

$$H^{\text{sen}} = \begin{bmatrix} Hist(x[1],1) & Hist(x[1],2) & \ldots & Hist(x[1],L) \\ Hist(x[2],1) & Hist(x[2],2) & \ldots & Hist(x[2],L) \\ \vdots & \vdots & \ddots & \vdots \\ Hist(x[M],1) & Hist(x[M],2) & \ldots & Hist(x[M],L) \end{bmatrix}_{M\times L} \quad (2)$$

where M is the number of frames in the sentence. As an example, if $x[1] = 3$ and $x[2] = 7$, then the 1st row of H^{sen} is the 3rd row of *Hist* and its 2nd row is the 7th row of *Hist*.

H^{sen} is transformed to a probability matrix by normalizing its rows. To obtain the probability matrix, P, for the M-frame sentence, we normalize the histogram matrix, H^{sen}, in Eq. (2) such that elements of each row add up to unity. P is again of size $M \times L$. Every column of this matrix corresponds to one of L LSF vectors of the target speaker as illustrated in Fig. 3. Dynamic programming is achieved on the elements of the P matrix as well as on the T matrix, which has been explained before, to determine the best path from frame number 1 to frame number M. An example P matrix for the case when $L = 128$ and $M = 105$ (i.e. the number of frames in the source speaker's sentence is 105) is given in Fig. 4.

To start the dynamic programming constraints should be determined.

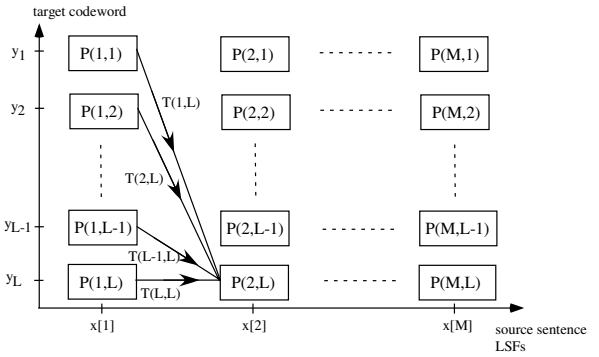

Fig. 3. Dynamic programming along the frames of one sentence for LSF transformation.

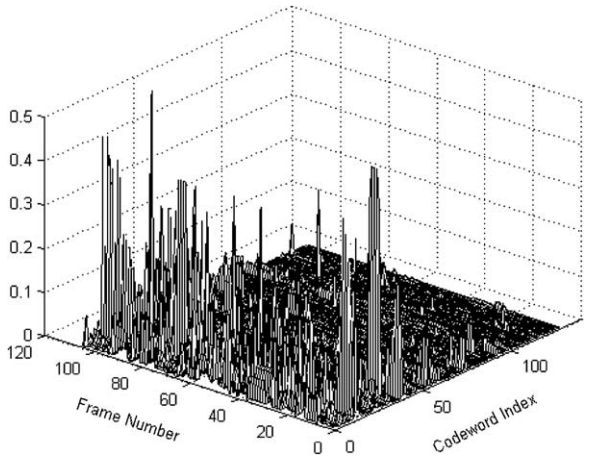

Fig. 4. P matrix.

3.2.1. Constraints

It is well known that discontinuities in the spectrogram of speech cause degradation of the speech quality. Therefore applying a continuity constraint seems to be reasonable during dynamic programming. The spectral distance between the subsequent frames of the source speaker is taken into account during constraint determination. The second aim for applying constraints is to avoid any possible mismatch between the residual signal and the transformed filter. If a sequence of frames belongs to a voiced sound, filter responses of the subsequent frames will be close to each other, while the opposite is true for frames of the plosive sounds or the fricatives. Therefore, only certain paths which satisfy the inequality given in (3) are allowed to be considered during dynamic programming:

$$\begin{aligned}&\mathrm{SD}(\mathrm{LSF}x[n-1]),\mathrm{LSF}(x[n])-D\\&\quad\leqslant \mathrm{SD}(\mathrm{LSF}(y_i),\mathrm{LSF}(y_j))\\&\quad\leqslant \mathrm{SD}(\mathrm{LSF}(x[n-1]),\mathrm{LSF}(x[n]))+D\end{aligned}\tag{3}$$

where, $\mathrm{SD}(\mathrm{LSF}(x[n-1]),\mathrm{LSF}(x[n]))$ is the spectral distance between the source LSFs corresponding to the indices $x[n]$ and $x[n-1]$ at frames n and $n-1$ respectively. Similarly, $\mathrm{SD}(\mathrm{LSF}(y_i),\mathrm{LSF}(y_j))$ is the spectral distance between the target LSFs represented by indices y_i and y_j. D is the allowed distance interval. Dynamic programming is applied to the paths which satisfy this constraint only.

3.2.2. Algorithm details

The idea is to determine the best sequence of target codewords from frame-1 to frame-M, while allowing only certain transitions determined by constraints among the possible target codewords, y_i. The optimization problem can be formulated as follows:

$$\begin{aligned}&\max_{\{i_k\},k=1\ldots M} P(1,i_1)T(i_1,i_2)P(2,i_2)T(i_2,i_3)\ldots\\&\qquad P(M-1,i_{M-1})T(i_{M-1},i_M)P(M,i_M)\\&=\max_{\{i_k\},k=1\ldots M}\left(\prod_{k=1}^{M-1}P(k,i_k)T(i_k,i_{k+1})\right)P(M,i_M)\end{aligned}\tag{4}$$

for all i_k (paths) satisfying the constraint given in (3). The method is explained in Fig. 3.

$T(i,j)$ in Fig. 3 represents the transition probabilities from the target codeword y_i in one frame, to another target codeword y_j in the next frame. The T matrix obtained from the target speaker's corpus is presented in Fig. 5. As an example, to determine the path towards the node $(2,L)$ the spectral distortion between the source LSF vectors, $\mathrm{SD}(\mathrm{LSF}(x[1]),$

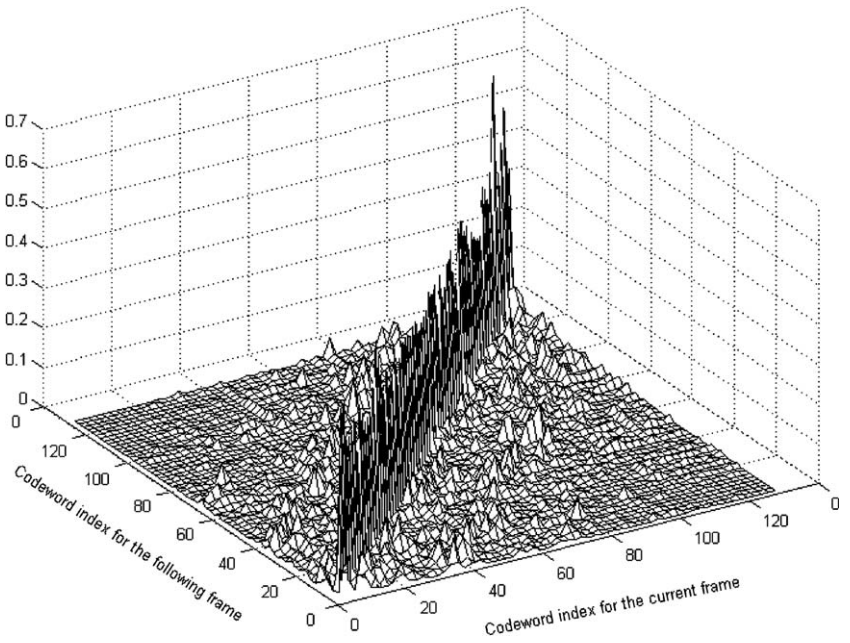

Fig. 5. Transition matrix for the target speaker.

LSF(x[2])), is computed first. Paths which satisfy the inequality given in (3) are selected as *allowable paths* to node $(2, L)$. This is a search on possible y_j's $(j = 1, \ldots, L)$ with an allowed distance interval, D, such that the inequality given in (3) is satisfied with $n = 2$, $y_i = y_j$, and $y_j = y_L$.

Best path to the node $P(2, L)$ is then selected among the allowable paths as the path with the maximum accumulated probability. The maximum possible accumulated probability at the node (i,j) is denoted by PathProb(i,j) and it is the accumulated probability of the best path that ends at the node (i,j). Let the set of j values satisfying the inequality (3) be denoted by ψ. Possible path probabilities are computed by multiplying the accumulated probability on the path towards the node $(2, L)$ with the transition probability $T(j, L)$ and $P(2, L)$ for all $j \in \psi$. Then the maximum probability is selected as PathProb$(2, L)$, which is given as:

$$\begin{aligned}\text{PathProb}(2,L) &= \max_j\{\text{PathProb}(1,j)\times T(j,L)\times P(2,L)\},\quad j\in\psi\\ &= \max_j\{P(1,j)\times T(j,L)\times P(2,L)\},\quad j\in\psi. \end{aligned} \tag{5}$$

Equality of $P(1,j)$ to PathProb$(1,j)$ in Eq. (5) is a special case, due to the initialization of the path-probability matrix, PathProb$_{M\times L}$. Once this algorithm is applied to all frames and PathProb$_{M\times L}$ matrix is obtained with the corresponding path track matrix, Path$_{M\times L}$, the best path (with the highest path probability) is determined. The final row of the PathProb matrix shows the accumulated probabilities along the paths and the highest element of the final row shows the last target LSF vector index of the best path. Path matrix is constructed along with PathProb matrix during dynamic programming. Every node of the Path matrix gives the previous target codeword index of the path with the highest probability to that node. It keeps track of the possible best paths along the sentence. When the final frame is reached, best path is obtained using the final row of the PathProb matrix and the Path matrix. A simple flow chart of the dynamic programming procedure is given in Fig. 6.

The complete pseudo code is given below:

Initialize

$$\text{Path} = \begin{bmatrix} 1 & 2 & \ldots & L \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix} \quad \text{PathProb} = \begin{bmatrix} P(1,1) & P(1,2) & \ldots & P(1,L) \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix}$$

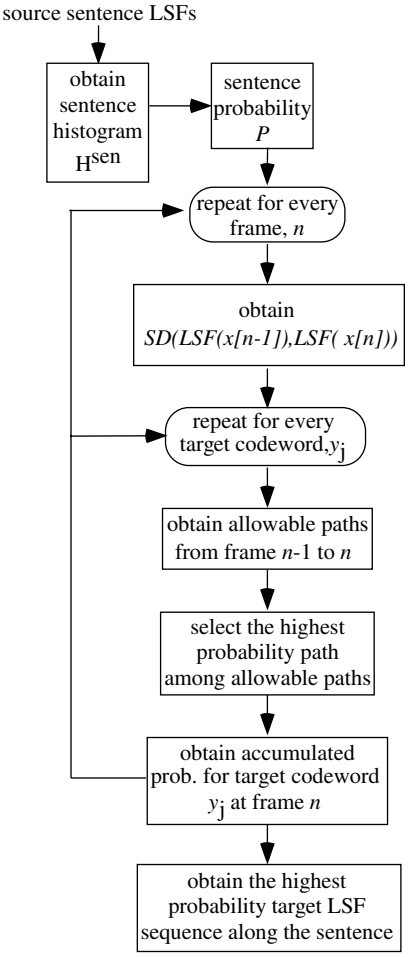

Fig. 6. Flow chart of the dynamic programming procedure.

Begin

```
for k = 2:M;
   SDs = SD(LSF(x[k]),LSF(x[k-l]));
   for i = 1: L;
      for j = 1: L;
         Path_allow = all j s.t.
            SDs-D <= SD(LSF(y[i]),LSF(y[j]))<=
            SDs + D;
       endfor
       maxSD = 0;
       for n = l : length(Path_allow);
          temp =
         PathProb(k-1, Path_allow(n))P(k,i)T(i,
         Path_allow(n));
          if temp > maxSD;
             maxSD = temp;
             Path(k,i) = Path_allow(n);
            PathProb(k,i) = temp;
         end if
      endfor
   endfor
endfor
```

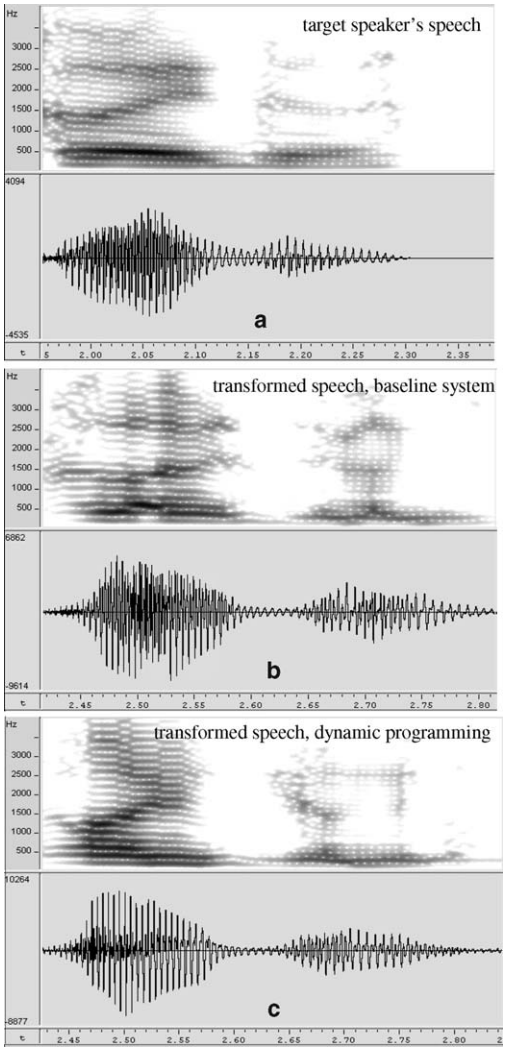

Fig. 7. Comparison of the waveforms and spectrograms of the baseline system and the improved system with dynamic programming: (a) is target speaker's speech, (b) is conversion with baseline system with $L = 128$ and (c) is conversion with dynamic programming with $L = 128$ and $D = 0.16$. Displayed is the Turkish word "aydı".

Once the path is determined, corresponding target codewords, y_i, are used to determine estimated target codeword sequence, $y_{\text{est}}[n]$, for the sentence. The 4th stage source frequencies which are neglected during the transformation are added to y_i's (which are results of the dynamic programming operation) directly. This has the effect of moving the target LSF vector from the codeword vector, y_i, in the same direction with the vector ($\text{LSF}_x[n] - \text{LSF}(x[n])$) where $\text{LSF}_x[n]$ is the original source LSF vector obtained from addition of all 4 stages of MSVQ at frame n.

A comparison of the baseline system output and the improved system output with dynamic programming is given Fig. 7. Dynamic programming reduces the discontinuities at the frame boundaries as observed in the figure.

Synthesis of the transformed speech is achieved by applying the modified LSFs in the MELP speech synthesis framework presented. Residual signal of the source is not modified except for a pitch period modification. The average pitch value is changed in MELP coder inside the decoder block before synthesis. Residual reconstruction is made with the new pitch value at every frame.

Pitch period modification that is used in this work is a simple one. The maximum and the minimum pitch period samples in the VT corpus have been obtained for both of the speakers and a linear relationship has been determined between the two pitch ranges. The first speaker's pitch period ranges from 38 to 83 samples, while the second speaker's pitch range is from 30 to 67 samples with a sampling frequency of 8 kHz. The relationship can be given as:

$$p_2[n] = 0.82p_1[n] - 1.24, \tag{6}$$

where $p_1[n]$ and $p_2[n]$ are pitch values at frame number n of the first and the second speakers respectively.

4. Results and discussion

Speech data for our VT research consists of 235 sentences collected from two male speakers of Turkish. Sentences are selected randomly from the phonetically-balanced 2462-sentence text corpus given in Salor et al. (2002). Speech has been collected in a quiet office environment with a *Sehnheiser ME-64* microphone and sampled at 16 kHz. Then it has been resampled to 8 kHz. This is approximately 15 min of speech data and more than 30,000 non-overlapping 180-sample analysis frames for each speaker, after silence frames are removed. Both speakers have read the same sentence set. Two hundred and fourteen sentences have been used for training and the remaining 21 sentences for the test.

The performance of the baseline VT system discussed in Section 3 has been compared to the method proposed in Section 3.2 with different parameter values using objective evaluations. Subjective listening tests have also been conducted and results are reported.

4.1. Objective evaluations

4.1.1. Distances and the performance index

Two kinds of distances are of interest here to evaluate the VT system: the *transformation* distance $E(t,y)$ and the *inter-speaker* distance $E(x,y)$, where y represents the target speaker's LSF indices, x represents the source speaker's LSF indices, and t represents LSF indices of the transformed speech. The inter-speaker distance describes the degree of difference between the source and the target speakers. These two distances are conceptual and cannot be measured directly, but can be approximated using objective and subjective evaluations (Kain, 2001).

To determine the transformation performance objectively, we have used the distance measure used in both LSF quantization and dynamic programming. Mean of this metric is obtained over all test set frames. It is given as:

$$E(x,y) = \frac{1}{M}\sum_{n=1}^{M} \mathrm{SD}(\mathrm{LSF}(x[n]), \mathrm{LSF}(y[n])), \tag{7}$$

where M is the number of frames in the test set, $\mathrm{LSF}(x[n])$ and $\mathrm{LSF}(y[n])$ are the LSF vectors corresponding to the LSF indices, $x[n]$ and $y[n]$, belonging to the source and the target speakers at frame n respectively. Then the LSF transformation performance index is defined as:

$$P_{\mathrm{LSF}} = 1 - \frac{E(t,y)}{E(x,y)}. \tag{8}$$

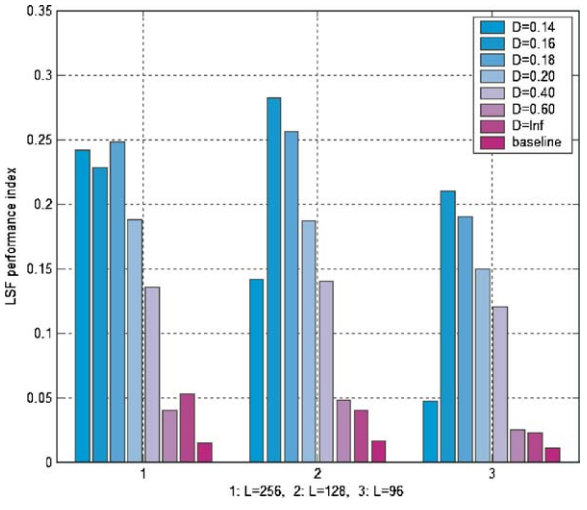

Fig. 8. Performance indices of the system using dynamic programming.

This index has been used by Kain (2001) for objective evaluation of a VT system and it applies a normalization to the transformation distance $E(t,y)$. P_{LSF} is zero when the transformation error equals the inter-speaker distance $E(x,y)$. It approaches to 1 as the transformation error approaches to zero. In an effective VT system transformation error is expected to be below the inter-speaker error, which means $P_{\mathrm{LSF}} > 0$.

4.1.2. Results

There exist two training parameters in our system: the number of reduced codewords, L, and the distance interval, D. We have evaluated the performance indices of the system for $L = 256$, 128, 96, 64, and $D = 0.14$, 0.16, 0.18, 0.20, 0.4, 0.6, ∞. $D = \infty$ means no constraints are applied and all paths are allowed in the dynamic programming procedure. Transformations from Speaker-1 to Speaker-2 and vice versa are tested. Fig. 8 presents the results obtained. The maximum value used for D is approximately 2.3 for each L value, which is the maximum value of the spectral distance between any LSF codewords of the source. Distance interval values of $D > 0.6$ give results which are very close to applying no constraints at all and values of $D < 0.14$ give no solutions, since most of the frame boundaries end up with no allowable paths when constraints are too tight.

LSF performance indices of the system with dynamic programming have been compared to the performance indices of the baseline system as given in Fig. 8. The performance index of the baseline system has been found to be 0.010, 0.011, and 0.011 for $L = 256$, 128 and 64, respectively. Dynamic programming improves the transformation performance as presented in the figure.

The results for $L = 64$ are not given in the figure, because this case gives performance indices below zero, which means that VT system is not successful. The reason for the decrease in the performance when L is reduced to 64 is thought to be the selection procedure of the reduced codewords for the speakers. When the number of codewords are reduced to a very small value, the reduced codebook includes codewords which are very close to each other. The reduction criterion is selecting the most frequently used codewords which is not the ideal method to cover the LSF space of a speaker, when a small-sized codebook is of concern.

It has been observed that, best performance index is obtained for the codebook size, $L = 128$,

and form the allowable distance interval in the dynamic programming, $D = 0.16$. $L = 128$ is probably a trade-off between obtaining correct probabilities from the histogram matrix, and obtaining a reduced codebook which represents the LSF space of the speakers efficiently. When L is high, the probabilities obtained from the histogram matrix are less reliable, but the LSF space representation of the reduced codebook is more efficient.

4.2. Subjective evaluations

The subjective evaluations consist of a speaker-similarity test and a mean opinion score (MOS) test. The speaker-similarity test is an **ABX** test. In the **ABX** text, **A** and **B** represent the original speakers, and **X** is the transformed speech either from **A** to **B** or from **B** to **A**. Subjects are asked to determine whether **X** is more similar to **A** or **B**. Only one original sentence from each speaker is provided to the subjects instead of providing the original forms of all the transformed sentences. The aim is to prevent the speaker-specific long-term behavior of the intonation along the sentences from effecting the decision of the subjects. The test includes 9 transformed **X** sentences (3 transformations for 3 different L values: 256,128, and 96). D is 0.18, 0.16, and 0.16 for L values 256, 128, and 96, respectively, which give the highest performance indices as observed in Fig. 8. Twenty subjects have taken the test. One hundred and seventy-four converted sentences have been detected as the target speaker out of 180 sentences in total. Four of the incorrect decisions were for $L = 128$ and 2 of them were for $L = 256$.

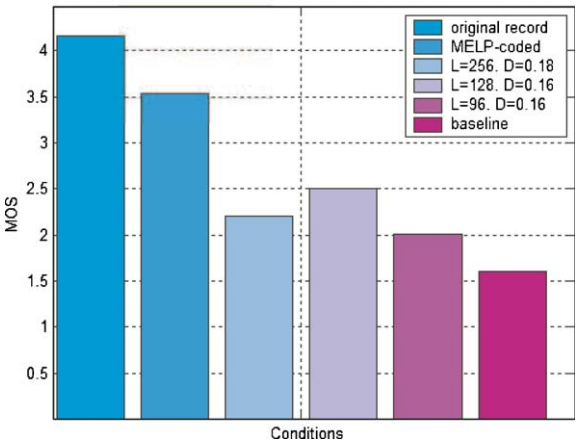

Fig. 9. MOS Test Results.

In MOS test, an opinion score was set to a 5-point scale (5: excellent (perfect speech signal recorded in a quiet booth), 4: good (intelligent and natural like long distance telephone quality), 3: fair (communication quality, but requires some hearing effort), 2: poor (low quality and hard to understand the speech), 1: bad (unclear speech)). Twenty subjects listened 24 sentences each. Four sentences from each group (original record, MELP coded speech, conversion with $L = 256$, conversion with $L = 128$, conversion with $L = 96$, and conversion with baseline system) have been used. Fig. 9 shows the MOS test results. The proposed algorithm has improved the speech quality as observed. The test has been conducted using the D values with the highest objective scores for each codebook size. Scores for MELP-coded and original record cases are also provided for comparison. It is observed that best MOS score is obtained as 2.50 when $L = 128$ and $D = 0.16$, while the score is 3.51 and 4.10 for original recording and MELP-coded speech respectively.

5. Conclusion

A new approach to the concept of voice transformation has been developed in this study. The algorithm is based on the idea of codebook mapping of the spectral features of the source and the target speakers. Some shortcomings of the codebook-mapping based systems, which are the limitation of the target spectral feature space and spectral discontinuities due to the independent mapping of subsequent frames, are overcome by applying a dynamic programming approach. This approach considers all target codewords corresponding to one source codeword using the probabilities obtained from the histogram matrix. Dynamic programming also considers the feature distances between the subsequent frames of the source speaker which reduces residual-filter mismatches in the transformed speech when an LPC-based speech model is used. The performance of the system was tested by objective and subjective listening tests. The objective evaluations verified that the target speaker characteristics are obtained to a large extend when the dynamic programming approach is applied to a baseline codebook mapping based voice transformation system. The subjective evaluations verified that the proposed approach results in convincing voice transformation in terms of speaker identity.

Acknowledgement

This work was supported by TÜBİTAK, Scientific and Technical Research Council of Turkey, through a combined doctoral scholarship.

References

Abe, M., Nakamura, S., Shikano, K., Kuwabara, H., 1988. Voice Conversion through Vector Quantization. In: Proceedings IEEE ICASSP, pp. 655–658.

Arslan, L.M., 1999. Speaker transformation algorithm using segmental codebooks (STASC). Speech Communication 28 (3), 211–226.

Childers, D.G., 1995. Glottal source modeling for voice conversion. Speech Communication 16 (2), 127–138.

Cohn, R.P., Collura, J.S., 1997. Incorporating Perception into LSF Quantization – Some Experiments. In: Proceedings IEEE ICASSP.

Huang, X., Acero, A., Hon, H.W., 2001. Spoken language processing, a guide to theory, algorithm, and system development. Prentice Hall PTR.

Kain, A., 2001. High resolution voice conversion. Ph.D. Thesis, OGI School of Science and Engineering at Oregon Health and Science University, Portland, Oregon.

Lee, S.K., Youn, D.H., Cha, I.W., 1995. Voice Personality Transformation Using an Orthogonal Vector Space Conversion. In: Proceedings of the Fourth European Conference on Speech Communication and Technology (EUROSPEECH 95).

Markel, J.D., Gray, A.H., 1976. Linear Prediction of Speech. Springer-Verlag.

MELP, 1997. Specifications for the Analog to Digital Conversion of Voice by 2400 Bit/Second Mixed Excitation Linear Prediction. Federal Information Processing Standards Publication.

Mizuno, H., Abe, M., 1995. Voice conversion algorithm based on piecewise linear conversion rules of formant frequency and spectrum tilt. Speech Communication 16 (2), 153–164.

Salor, Ö., 2005. Voice Transformation and Development of Related Speech Analysis Tools for Turkish, Ph.D.Thesis, Middle East Technical University, Turkey.

Salor, Ö., Demirekler, M., 2004. Spectral Modification for Context-free Voice Conversion Using MELP Speech Coding Framework. In: Proceedings of the International Symposium on Intelligent Multimedia, Video and Speech Processing (ISIMP 04).

Salor, Ö., Pellom, B.L., Çiloğlu, T., Demirekler, M., 2002. On developing new text and audio corpora and speech recognition tools for the turkish language. In: Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 02).

Salor, Ö., Demirekler, M., Pellom, B.L., 2003. A system for voice conversion based on adaptive filtering and LSF distance optimization for text-to-speech synthesis. In: Proceedings of the 8th European Conference on Speech Communication and Technology (EUROSPEECH 03).

Stylianou, Y., 1999. Assessment and correction of voice quality variabilities in large speech databases for concatenative speech synthesis. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 99).

Stylianou, Y., Cappé, O., Moulines, E., 1998. Continuous probabilistic transform for voice conversion. IEEE Transactions on Speech and Audio Processing 6 (2), 451–454.

Toda, T., 2003. High-Quality and Flexible Speech Synthesis with Segment Selection and Voice Conversion, Ph.D.Thesis, Nara Institute of technology, Japan.

Valbret, H., Moulines, E., Tubach, J.P., 1992. Voice transformation using PSOLA technique. Speech Communication 11 (2), 175–187.