CUTE: A CONCATENATIVE METHOD FOR VOICE CONVERSION USING EXEMPLAR-BASED UNIT SELECTION

Zeyu Jin[1,2*], *Adam Finkelstein,*[1] *Stephen DiVerdi,*[2] *Jingwan Lu,*[2] *Gautham J. Mysore*[2]

[1]Princeton University
Princeton, NJ 08540, USA

[2]Adobe Research
San Francisco, CA 94103, USA

ABSTRACT

State-of-the art voice conversion methods re-synthesize voice from spectral representations such as MFCCs and STRAIGHT, thereby introducing muffled artifacts. We propose a method that circumvents this concern using concatenative synthesis coupled with exemplar-based unit selection. Given parallel speech from source and target speakers as well as a new query from the source, our method stitches together pieces of the target voice. It optimizes for three goals: matching the query, using long consecutive segments, and smooth transitions between the segments. To achieve these goals, we perform unit selection at the frame level and introduce triphone-based preselection that greatly reduces computation and enforces selection of long, contiguous pieces. Our experiments show that the proposed method has better quality than baseline methods, while preserving high individuality.

Index Terms— Voice conversion, unit selection, concatenative synthesis, exemplar-based

1. INTRODUCTION

The goal of voice conversion (VC) is to modify an audio recording containing the voice of one speaker (the *source*) so that the identity sounds like that of another speaker (the *target*) without altering the speech content. Approaches to VC typically rely on a training set of parallel utterances spoken by both the source and target. State of the art **parametric methods** [1, 2] then explicitly model a conversion function mapping from the source to the target in some feature space such as MFCC [3] or STRAIGHT [4]. A new source utterance (the *query*) may be transformed into the feature space and then mapped through the conversion function to match the target. The output of such parametric methods must be re-synthesized from these features, and artifacts are inevitable since these feature spaces do not perfectly model human voice. Thus, the converted speech usually has a muffled effect [5] as a result of re-synthesis.

Voice conversion can be used in many applications such as generating speech synthesis voices with small samples [6] and bandwidth expansion [7]. In order to avoid artifacts due to re-synthesis, an alternative to the parametric approach relies on **unit selection**. The basic idea is to choose segments of the target speaker's training samples whose corresponding source samples sound like the query, while also seeking smooth transitions between neighboring segments. Modern text-to-speech synthesis systems [8] demonstrate that unit selection can generate high quality speech with high individuality, which is crucial for VC. These systems require very large training sets (many hours up to days) as well as substantial human annotation. Yet, in typical VC applications, we have a limited training set (such as one hour) and usually no manual effort.

Researchers have investigated several approaches for overcoming these limitations in the context of unit selection. For example, the method of Fujii et. al. [9] and Sndermann et. al. [10] relies on automatic speech recognition to generate phoneme level annotations for units. As noted by the former, segmentation errors among the phonemes yield significant artifacts. Another line of research reduces the need for very large datasets by performing frame-level unit selection (thereby increasing the number of units) [11].

Wu et al. further improve frame-level unit selection using exemplars [12], defined as time-frequency speech segments that span multiple consecutive frames [13]. We find that this method greatly reduces distortion from frame-based unit selection, but there remains a significant quality gap between this approach and phoneme-level unit selection where substantial user annotation is involved.

We introduce a hybrid method called CUTE that leverages four major components described in various contexts in the literature:

- Concatenative synthesis
- Unit selection
- Triphone pre-selection
- Exemplar-based features

This combination offers a significant improvement over previous approaches such as phoneme-based and exemplar-based unit selection. The basic idea of our method is to concatenate segments from the target speaker's voice so that they match the query and form long consecutive segments, with smooth transitions between them. To have sufficient units and flexibility in choosing the segments, we define unit selection at the frame level. To enforce smooth transition, we compute features using exemplars, concatenated spectral representations of multiple consecutive frames. To better match the query prosody, we exchange the $F0$ contour of source and target voices so that the query $F0$ is transformed and matched to the target $F0$ directly. Finally, we introduce a triphone preselection method to sift the candidate list before unit selection, ensuring large consecutive segments to be preserved in the candidate list. To evaluate our method, we performed two listening tests with human subjects: an XAB preference test and an ABX individuality test. The results show that our method offers significantly higher quality than two other unit selection based methods, and that high individuality is maintained. The contributions in this paper are:

- Adapting exemplar-based unit selection framework for concatenative synthesis.
- Using phonetic information to improve exemplar-based unit selection for voice conversion.
- Experiments showing that the proposed method has better quality than baseline methods, while preserving high individuality.

*This work was partially performed while interning at Adobe Research

2. EXEMPLAR-BASED UNIT SELECTION FOR CONCATENATE SYNTHESIS

2.1. Pairing

Given the source speaker's frame sequence $X = \{x_1, x_2, ..., x_N\}$ in the time domain and the target speaker's frame sequence $Y = \{y_1, y_2, ..., y_M\}$, we first perform dynamic time warping (DTW) to align the source voice to the target in order to obtain frame-wise pairings $\mathcal{A} = \{(x_{n_1}, y_{m_1}), (x_{n_2}, y_{m_2}), ..., (x_{n_K}, y_{m_K})\}$ where for $n, m = 1, 2, ..., k$, $n_k \in [1..N]$ and $m_k \in [1..M]$. We then construct a mapping function $r(j) = Y_j$ where $Y_j = \{y | n_k = j, (x_{n_k}, y) \in \mathcal{A}\}$. This is the mapping we use to translate pairings to actual frames.

2.2. Exemplar feature extraction

Having multiple parallel frames, we define an *exemplar frame* as two parallel sequences of source and target frames with the central frame aligned. With a central frame (x_{n_k}, y_{m_k}), an *exemplar frame* is defined as:

$$\begin{pmatrix} x_{n_k-t} & ... & x_{n_k-1} & x_{n_k} & x_{n_k+1} & ... & x_{n_k+t} \\ y_{m_k-t} & ... & y_{m_k-1} & y_{m_k} & y_{m_k+1} & ... & y_{m_k+t} \end{pmatrix} \quad (1)$$

We define our exemplar as a concatenation of the weighted features of the member frames. Suppose a parallel frame (x_{n_k}, y_{m_k}) has feature vector f_{n_k,m_k}, then an exemplar is:

$$[w_{-t} f_{n_k-t,m_k-t}, ..., w_0 f_{n_k,m_k}, ..., w_t f_{n_k+t,m_k+t}]^T \quad (2)$$

The weights $w_{-t:t}$ are used to attenuate the influence of non-central frames, emphasizing the central frame. A triangule-shaped window function may be used for weights:

$$w_i = ((t - |i| + 1)/(t + 1))^\beta \quad (3)$$

Here $\beta > 0$ is used to control the amount of attenuation - the higher the value, the less emphasis on the non-central frames. An *exemplar* has better temporal resolution than a large frame, which allows us to model transition between phonemes. Using exemplars in unit selection also improves smoothness because similar exemplars share similar contexts; when concatenating these frames using overlap-add, the overlapping region is likely to produce less artifacts.

In this work, we define the weight function as a triangular function ($\beta = 1$) and we use two sets of exemplars: (1) matching exemplars includes the MFCCs of the source frames concatenated with the target frames' F0 (in log space), denoted as $\{A_1, ..., A_K\}$; we use these exemplars to match the spectral envelope and prosody of the query. (2) concatenation exemplars are formed by the MFCCs and the F0 of the target frames, denoted as $\{B_1, ..., B_K\}$. These exemplars are used to enforce transition smoothness between the selected frames.

2.3. Pre-selection of candidates

The next several steps involve finding the exemplars that match the query. The first step is to construct exemplars from $\{q_1, .., q_S\}$ and calculate the exemplar features $\{Q_1, ..., Q_S\}$ using the matching feature descriptors we defined in section 2.2. Because the source $F0$s are in a different range of the target $F0$s, we adapted logarithm Gaussian normalized transformation [3] to convert query $F0$ contour to the target's:

$$\log p_{\text{conv}} = \mu_{\text{tgt}} + \frac{\sigma_{\text{tgt}}}{\sigma_{\text{scr}}} (\log p_{\text{scr}} - \mu_{\text{scr}}) \quad (4)$$

where μ_{tgt} and σ^2_{tgt} are the mean and variance of the target voice's log F0s; μ_{scr} and σ^2_{scr} are the mean and variance of the source voice's log F0s. p_{scr} and p_{conv} are the query and the converted F0s.

Then, for each query exemplar, we want to select one exemplar from the dataset that resembles the query and makes a smooth transition to the next exemplar. The Viterbi algorithm can be used for this purpose. However, it does not scale well with the number of states (exemplars) [14]. We pre-select the candidates per query exemplar in order to reduce run-time computation. One way to do this is to compute K nearest neighbors and choose the candidates that are closest to queries [12]. In this work, we found that pre-selecting exemplars that share the same phoneme as the query exemplar generates better result. We will present our approach in Section 3. Denote the candidates (indices of exemplars) for Q_k as $\{c_{k1}, c_{k2}, ..., c_{ku_k}\}$ where u_k is the number of candidates for the k-th query exemplar.

2.4. Optimization

The objective of matching is to select one candidate source frame per query so that the corresponding target frames when concatenated together speak the same word as the query and at the same time sound natural without artifacts, noise or unnatural prosody. To translate that into an optimization scheme, we want to minimize the target cost $\mathcal{T}(q_s, A_{c_{s,j}})$ between the query q_s and a candidate exemplar with index $c_{s,j}$ and concatenation cost $\mathcal{C}_s(B_{c_{s-1,i}}, B_{c_{s,j}})$ between neighboring candidates with subscripts $c_{s-1,i}$ and $c_{s,j}$:

$$\min_{d_1,...,d_S} \mathcal{T}(Q_1, A_{c_{1,d_1}}) + \sum_{s=2}^{S} \mathcal{T}(Q_s, A_{c_{s,d_j}}) + \mathcal{C}_s(B_{c_{s-1,d_{j-1}}}, B_{c_{s,d_j}}) \quad (5)$$

Note that different exemplar features are used for the matching and the concatenation costs. One difference between this scheme and previous unit selection method is that concatenation cost function varies over s. In other words, some statistics about the query q_s is used in calculating the cost function. Here are the equations we use to calculate costs

$$\mathcal{T}(Q, A) = \|Q - A\|_2 \quad (6)$$

$$\mathcal{C}_s(B_i, B_j) = d(B_i, B_j)\mathcal{P}(i, j)\mathcal{R}(E(q_s)) \quad (7)$$

In equation 7, $d(B_i, B_j)$ is defined as the Euclidean distance between the first $2t$ frames in exemplar B_i and the second last $2t$ frames from exemplar B_j, modeling the smoothness at the overlapping region between frames. The second term is defined as break costs, which prefers candidates that are originally neighbors; it also allows skips and repeats, adding more flexibility in the query length.

$$\mathcal{P}(i, j) = \begin{cases} 0, & j - i = 1 \\ c_s, & j - i = 2 \\ c_r, & j - i = 0 \\ c_b, & \text{else} \end{cases} \quad (8)$$

c_s, c_r and c_b are skip, repeat and break penalty multipliers. Through experiment, we found 10, 2, 2 are reasonable choices for c_b, c_r and c_s. The last term $\mathcal{R}(E(q_s))$ is penalty reduction considering the significance of the query. $E(q_s)$ extracts the features for significance and in our experiment we define it as energy times periodicity extracted using YIN [15] algorithm. It means we prefer breaks at silences and noisy parts such as background noise or unvoiced consonants. Mapping $\mathcal{R}$ normalizes the value to be in [0,1].

2.5. Translation and Concatenation

The last step is to convert exemplars to actual frames. Suppose we have selected candidates with indices $\{c_1, ..., c_s\}$, we first remove the repeats and skips and then translate them into target frames using the function $r(j) = Y_j$ defined in section 2.1. Then we concatenate elements in sequences $(Y_{c_1}, Y_{c_2}, ..., Y_{c_s})$ to obtain a series of frames. We obtain the final result using overlap-add [8].

3. TRIPHONE PRESELECTION

We have observed two limitations in the above mentioned method: first, when the query contains unfamiliar prosody (strong emphasis, high or low $F0$s), the query exemplars might lack true nearest neighbors in the dataset and therefore are likely to be matched to target exemplars with incompatible phonemes. Secondly, if a different utterance of the same query word exists in the dataset, the above mentioned method fails to extract the whole word most of the time. We think these problems have to do with the fact that spectral envelope is not completely independent of prosody, especially at extreme $F0$s and emphasis. As a result, different utterances of the same word might have high gross spectral distances with each other, making several frames of the word not appearing in the K nearest neighbors. To encourage concatenating long segments and reduce the negative effects of extreme prosody, we propose using phonetic information to preselect the candidates.

3.1. Pre-selection using phonemes

To obtain phoneme segmentation, we first translate the transcripts into phoneme sequences and then apply forced alignment to align phonemes to the target speaker's voice [16]. For each pair of source and target frames, we annotate it with a phoneme label, such as "AH" or "P". Then for each exemplar, we label it with its central frame's phoneme. In the pre-selection step, for each query exemplar, we include only the candidates that share the same phoneme.

Using phoneme pre-selection, we guarantee that if other utterances of the same query word exist in the dataset, their frames are included in the candidate lists. With a proper break cost, we can extract a full word from the target speech to match the query. This overcomes the issue faced by KNN pre-selection. Unit selection at the frame level with phoneme pre-selection also outperforms phoneme-level unit selection because it allows breaks in the middle of a phoneme, making it possible to form longer or shorter phones. This approach is also robust to phoneme segmentation errors since an incorrectly segmented frame is often dissimilar from the query and thus excluded in the optimization step. The drawback of this method however is that the candidate table gets very large for frequent phonemes such as vowels; because the complexity of Viterbi algorithm scales with the squared size of candidates, this method seems impractical for large datasets. Even if we select the K nearest candidates after phoneme preselection, there is still chance of omitting frames that potentially make up long sequences. Our solution to this drawback is to use Triphones to pre-select the candidates.

3.2. Decision graph for triphone pre-selection

A triphone is a three-phoneme sequence, composed of a center phoneme and its two immediate neighbors. In tasks such as speech recognition and text-to-speech synthesis, triphones are superior to phonemes as they capture contextual information [17]. To preselect exemplar candidates, we propose labeling frames with triphones instead of individual phones. Although the triphone vocabulary is large, the number of instances per triphone is small. Therefore, we adopt a simple scheme to deal with triphones nonexistent in the dataset: first, we look for candidates with exact triphones; if we cannot find enough candidates (e.g. less than 10), we look for candidates with similar diphones; and if we again fail to find sufficient candidates, we apply monophone-based preselection and only include frames that are closest to the query. One may also adopt decision trees used in speech recognition [18] to inform triphone selection procedure.

This preselection scheme has several merits: (1) The frames in the target dataset having exactly the same phoneme sequence with the query will be present in the candidate table. (2) Moreover, we can get away with only a few candidates per query frame - following the decision table, if we find a correct triphone instance, we include all candidate frames in that triphone; if not we look for diphones and include all of them if they exist. If no matching diphone is found, we can include only a small number of frames that has correct phoneme because there will be a break at this point anyway. In this way, the number of candidates can be kept small resulting in faster Viterbi computation. (3) Using the decision graph, the synthesis quality improves with increasing amounts of data: the larger the dataset, the more likely we have triphones in the candidate list.

4. EVALUATION AND DISCUSSION

This section describes subjective and objective evaluations of our method, in comparison with two other baseline methods in the literature. We use CMU arctic corpus for these experiments [19]. We examined four conversion schemes based on four voices: DBL-male to RMS-male (m2m), DBL-male to SLT-female (m2f), SLT-female to RMS-male (f2m), and SLT-female to CLB-female (f2f). The training data is composed of 800 utterances, while 20 utterances are used for evaluation. Transcripts are aligned to utterances via forced alignment implemented in `p2fa-vislab`[1]. No manual segmentation or adjustment is performed in these experiments, so segmentation errors are prevalent.

We obtain frames using a 25ms analysis window and a 12.5ms hop size. To extract exemplar features, we use 24-coefficient MFCCs, excluding the 0th energy coefficient, and the YIN algorithm [15] for F0 detection. We compare our method (CUTE) with two other unit selection voice conversion methods:

1. Phoneme-level unit selection (PUS) [9]: We used the same phoneme segmentation to extract the units. Unlike the experiments in the original paper, we did not perform manual adjustment to phoneme segmentation. This is a baseline used to show that using frame-level concatenation and phonetic preselection can overcome segmentation problems.
2. Exemplar-based unit selection (EUS) [12]: This method is similar to our method without triphone-based preselection and penalty reduction terms in the Viterbi algorithms. In our experiments, we notice that noisy discontinuities are substantially reduced via triphone-based preselection.

4.1. Subjective Evaluation

To compare the quality of the proposed method to the two baseline methods, we conducted a subjective evaluation using an XAB test, as follows. We shuffle the order of the 20 utterances, of which 16 are used to compare methods, and the remaining four are used as a validation described below. For each utterance, we randomly

[1] https://github.com/ucbvislab/p2fa-vislab

select one out of four voice conversion settings (m2m, m2f, f2m, and f2f) and one of the two baseline methods. The subject is shown three tracks: the reference track (X) followed by two other tracks (A&B) – CUTE track and the baseline method, in random order. A subject is asked to choose which sample (A or B) sounds more like the reference track X. In the four validation tests, the A&B tracks (in random order) are our method and a duplicate of the reference track X, which should easily be selected as the preferred track. We assume that when a subject fails any validation tests they are not paying careful attention, and we omit the data for all 20 utterances from that experiment.

We recruited subjects on the Amazon Mechanical Turk (MTurk), a crowdsourcing platform that has become a common tool for perceptual and other human subjects experiments [20]. In each task (called a HIT) the subject provided the 20 XAB answers described above. Of 100 HITs we retain 95 valid results with 1520 (95×16) individual responses. As seen in Figure 1, our method (CUTE) is almost always preferred when comparing to phoneme-level unit selection (PUS), with p-values ≈ 0 in all four cases. The lower quality of PUS is possibly due to segmentation errors and insufficient units. CUTE is also strongly preferred over the exemplar-based unit selection method (EUS) for same-gender voice conversion. For cross-gender tests, the margin is smaller, but significant.

In order to corroborate these outcomes obtained on MTurk, we also ran a duplicate experiment with only a dozen subjects recruited in the research lab at Adobe. Of these, only 9 passed the validation test, insufficient data to break out the four separate gender settings. Nevertheless, the aggregate results (labeled 'lab' in the figure) were similar to the aggregate from MTurk (labeled 'all').

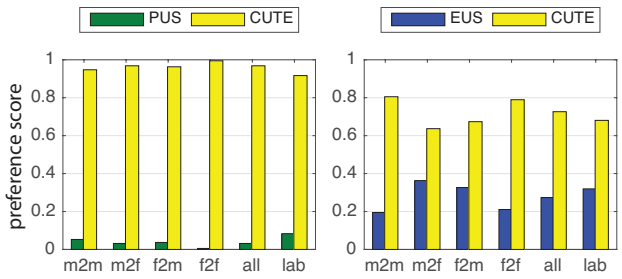

Fig. 1: XAB test results. Our method (CUTE) has better quality than two baseline methods (PUS and EUS – similar to those found in the literature). Label 'all' refers to aggregating the four preceding gender settings, while 'lab' refers to a smaller experiment performed in the lab. Note that p-values ≈ 0 in all cases except for EUS-m2f, EUS-f2m, and EUS-lab with p-values $< 10^{-4}$.

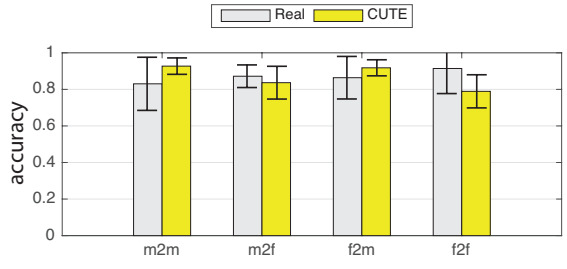

Fig. 2: ABX individuality test results. The performance of subjects is compared between identifying voices for real samples and synthetic samples generated using the proposed method.

We hypothesize two reasons for the significant improvement in CUTE: first, CUTE overcomes phoneme segmentation errors by selecting new concatenation points: if the query has two phonemes and in the candidate table there is no consecutive paths between them, CUTE finds the best concatenation points within both phonemes that produces the least cost. The new cut points are not necessarily at the recognized boundaries that may deviate from the true boundary. Secondly, with triphone preselection, we can reduce noise of concatenating small segments by enforcing less breaks with a large break penalty. Triphone preselection ensures that these larger segments are phonetically consistent with the source. Note that having a large break penalty is not effective in pure frame-level unit selection as it over-smooths the distance and thus may select chunks that are not phonetically consistent.

We also conducted 100 ABX tests on MTurk, in order to test if our method preserves individuality claimed in the phoneme-level unit selection method [9]. In these tests, a subject is presented with Person A's and Person B's voice samples (same gender) and then an unknown sample X. The task is to choose which voice (A or B) is that of the sample X. In these tests, X can be either an actual voice sample from A or B, or generated using our proposed method (CUTE). As shown in Figure 2, the subjects have similar performance on generated samples and on real samples, meaning individuality is well preserved in CUTE.

The audio clips and results of the experiments described in this section may be found at `http://voxogram.com/CUTE/`.

4.2. Objective Evaluation

As an objective evaluation, we computed Mel-cepstral distortion (MCD) [21] to measure the distortion between the target voice sample and the converted samples. The results contradict the subjective evaluation results – MCDs of samples generated by our method are about 1dB higher than those generated by EUS even though the synthesized samples have better perceptual quality verified in the subjective listening tests. We hypothesize two explanations: (1) we use query F0s to control the output and therefore the prosody may vary from the target sample; as noted before, the spectral envelope may vary with prosody – a natural sounding sample can have high distance to other versions of it. (2) MCD is an over-smoothed measure of the spectral envelope, artifacts and noises are evened out. We found that having a low break cost would result in much lower MCD, but because of many breaks, the output sounds noisy.

5. CONCLUSION

We have proposed a concatenative voice conversion method based on exemplars and phonetic pre-selection of units. We defined two types of exemplars, target exemplar and concatenation exemplar, to allow controlling the prosody with source examples and enforce concatenation smoothness in the target samples. Using phoneme information to pre-select the phonemes, we ensure the longest possible phonetically correct segments to be used in concatenative synthesis. Experiments demonstrate that our CUTE method has better quality than previous exemplar-based voice conversion methods and high individuality comparable to real samples. One limitation is that it still suffers from the lack-of-data problem faced by phoneme-based unit selection – sometimes the result is phonetically correct but unsatisfactory in prosody due to the lack of samples with appropriate F0s. It also relies on speech recognition for phoneme segmentation. Future work includes refining pre-selection to balance phonetic correctness and prosodic continuity.

6. REFERENCES

[1] S. Takamichi, T. Toda, A.W. Black, and S. Nakamura, "Modulation spectrum-constrained trajectory training algorithm for gmm-based voice conversion," in *ICASSP 2015*, 2015.

[2] R. Aihara, T. Nakashika, T. Takiguchi, and Y. Ariki, "Voice conversion based on non-negative matrix factorization using phoneme-categorized dictionary," in *ICASSP 2014*, 2014.

[3] S. Desai, E.V. Raghavendra, B. Yegnanarayana, A.W. Black, and K. Prahallad, "Voice conversion using artificial neural networks," in *ICASSP 2009*, 2009.

[4] H. Kawahara, I. Masuda-Katsuse, and A. de Cheveigné, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, pp. 187–207, 1999.

[5] Y. Stylianou, "Voice transformation: A survey," in *ICASSP*, April 2009, pp. 3585–3588.

[6] A. Kain and M.W. Macon, "Spectral voice conversion for text-to-speech synthesis," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, 1998, vol. 1, pp. 285–288 vol.1.

[7] K. Y. Park and H. S. Kim, "Narrowband to wideband conversion of speech using gmm based transformation," in *ICASSP*, 2000, vol. 3, pp. 1843–1846 vol.3.

[8] P. Taylor, *Text-to-Speech Synthesis*, Cambridge University Press, 2009.

[9] K. Fujii, J. Okawa, and K. Suigetsu, "High-individuality voice conversion based on concatenative speech synthesis," *International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, vol. 1, no. 11, pp. 1617 – 1622, 2007.

[10] D. Sündermann, H. Höge, A. Bonafonte, H. Ney, A. Black, and S. Narayanan, "Text-independent voice conversion based on unit selection," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. IEEE, 2006, vol. 1.

[11] T. Dutoit, A. Holzapfel, M. Jottrand, A. Moinet, J. Perez, and Y. Stylianou, "Towards a voice conversion system based on frame selection," in *ICASSP 2007*, 2007.

[12] Z. Wu, T. Virtanen, T. Kinnunen, E. Chng, and H. Li, "Exemplar-based unit selection for voice conversion utilizing temporal information," in *INTERSPEECH*. 2013, pp. 3057–3061, ISCA.

[13] B. Raj, T. Virtanen, S. Chaudhuri, and R. Singh, "Non-negative matrix factorization based compensation of music for automatic speech recognition.," in *Interspeech*. Citeseer, 2010, pp. 717–720.

[14] G.D. Forney Jr., "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[15] A. Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[16] K. Sjölander, "An hmm-based system for automatic segmentation and alignment of speech," in *Proceedings of Fonetik*, 2003, vol. 2003, pp. 93–96.

[17] L. Rabiner and R. Schafer, *Theory and Applications of Digital Speech Processing*, Prentice Hall Press, 2010.

[18] H. W. Hon and K. F. Lee, "Cmu robust vocabulary-independent speech recognition system," in *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*. IEEE, 1991, pp. 889–892.

[19] J. Kominek and A. W. Black, "The CMU Arctic speech databases," in *Fifth ISCA Workshop on Speech Synthesis*, 2004.

[20] W. Mason and S. Suri, "Conducting behavioral research on Amazons Mechanical Turk," *Behavior research methods*, vol. 44, no. 1, pp. 1–23, 2012.

[21] T. Toda, A.W. Black, and K. Tokuda, "Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 8, pp. 2222–2235, 2007.