# IMPROVED MINIMUM CONVERTED TRAJECTORY ERROR TRAINING FOR REAL-TIME SPEECH-TO-LIPS CONVERSION

*Wei Han*[†⋆] *Lijuan Wang*[†] *Frank Soong*[†] *Bo Yuan*[⋆]

[†] Microsoft Research Asia, China
[⋆]Department of Computer Science, Shanghai Jiao Tong University, China
{weihan_cs, boyuan}@sjtu.edu.cn
{lijuanw, frankkps}@microsoft.com

## ABSTRACT

Gaussian mixture model (GMM) based speech-to-lips conversion often operates in two alternative ways: batch conversion and sliding window-based conversion for real-time processing. Previously, Minimum Converted Trajectory Error (MCTE) training has been proposed to improve the performance of batch conversion. In this paper, we extend previous work and propose a new training criteria, MCTE for Real-time conversion (R-MCTE), to explicitly optimize the quality of sliding window-based conversion. In R-MCTE, we use the probabilistic descent method to refine model parameters by minimizing the error on real-time converted visual trajectories over training data. Objective evaluations on the LIPS 2008 Visual Speech Synthesis Challenge data set shows that the proposed method achieves both good lip animation performance and low delay in real-time conversion.

***Index Terms*—** speech-to-lips, minimum converted trajectory error, real-time conversion

## 1. INTRODUCTION

Speech-to-lips systems that synthesizes lip animation from given speech offer a wide range of useful applications. For example, generating lip animation from acoustic speech animation is crucial for animating avatar for human computer interaction in video games or other augmented reality scenario. As audio and visual information compensate each other in human communication, an animated avatar is also beneficial in human-to-human interactions, e.g. internet videophones with low network transmission rate.

There are many attempts at modeling the relationship between audio (speech) and visual (usually lips, sometimes also upper face) signals. Most of them are generative probabilistic models that make assumptions about the underlying probability distribution of audio-visual data. Typical model assumptions are Gaussian Mixture model (GMM), Hidden Markov Model (HMM), Dynamical Bayesian Network (DBN) [1] and Switching Linear Dynamical System (SLDS) [2], in order of increasing model complexity. With sufficient training data, all these methods are capable of converting speech to synchronous lips movement.

Besides the quality of converted lips movements, another challenge is low latency required by certain real-time applications, e.g., videophones. Negative delay effects include: 1) The visual animation is rendered behind corresponding audio, giving rise to audio-visual asynchrony to which human perception is sensitive [3]; 2) The audio could be delayed until visual information become available. Instead of asynchrony, in this case the delay is also inconvenient in conversational interactivity. In a closely related area, voice conversion [4], the latency issue has been addressed by sliding window-based conversion [5]. Compared to conventional GMM based batch conversion, the method achieves low latency while keeping comparable performance.

In our previous work [6], we proposed Minimum Converted Trajectory Error training which, unlike maximum likelihood criteria, directly minimizes the converted trajectory error over training data, thereby improves the quality of GMM-based batch conversion. Combining MCTE training with sliding window-based conversion, however, does not lead to optimal performance for real-time conversion, because MCTE specifically optimizes for batch conversion which convert all frames in an utterance at one time. The mismatch between conversion schemes used in training (batch) and testing (sliding window-based) leads to a degradation of conversion performance.

In response to this mismatch, in this paper, we extend the original MCTE to a more general form and propose R-MCTE which accounts for the converted trajectory errors in sliding window-based conversion. In contrast to MCTE training which updates GMM model parameters at every utterance of training data, R-MCTE performs model update at each individual frame in a sliding window manner. By making the conversion method in training and testing consistent, R-MCTE can significantly improve the performance of sliding window-based conversion, demonstrated by objective evaluations on a public data set.

The remainder of this paper is organized as follows. Batch and sliding window-based speech-to-lips conversion are described in Section 2. In Section 3 we firstly review MCTE for batch conversion and then propose R-MCTE for sliding window-based conversion in detail. We present experimental results in Section 4 and conclusions in Section 5.

## 2. GMM BASED CONVERSION

The speech-to-lips conversion consists of two stages: training and conversion, as illustrated in Fig. 1. During training, acoustic and visual features are extracted from parallel training data, forming two feature spaces. A statistical model is automatically trained to characterize the joint feature space. Later at the conversion stage, previously unseen sequences in the acoustic space will be mapped to the visual space. The mapped visual feature sequences are used for rendering animations of lips movements.

Both the training and conversion stages can be done in several alternative ways. In this section, we introduce two schemes of GMM based conversion: batch conversion and sliding window-based real-

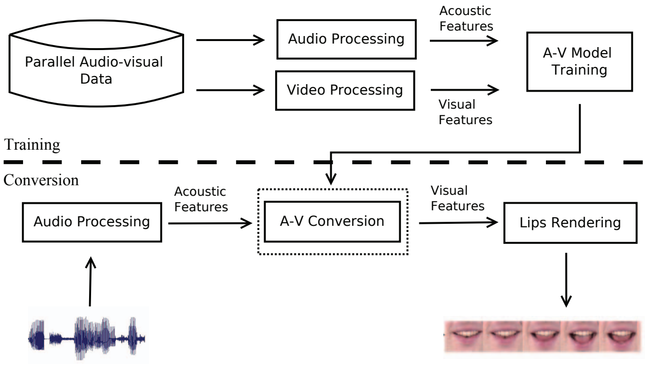

**Fig. 1**. Diagram of speech-to-lips conversion

time conversion. Model training will be discussed in the next section.

### 2.1. Batch conversion

We denote acoustic and visual feature sequences, and their time derivatives as,

$$
\begin{aligned}
x &= [x_1^\top, \ldots, x_T^\top]^\top, & y &= [y_1^\top, \ldots, y_T^\top]^\top \\
\triangle x_i &= \frac{1}{2}(x_{i+1} - x_{i-1}), & \triangle y_i &= \frac{1}{2}(y_{i+1} - y_{i-1}) \\
X_i &= [x_i^\top, \triangle x_i^\top], & Y_i &= [y_i^\top, \triangle y_i^\top] \\
X &= [X_1^\top, \ldots, X_T^\top]^\top, & Y &= [Y_1^\top, \ldots, Y_T^\top]^\top
\end{aligned}
$$

where $x$ and $y$ are acoustic and visual feature sequences of one utterance. We further augment the feature vector with its time derivatives (or dynamic features [7]) $\triangle x_i$ and $\triangle y_i$, so that the sequence $X$ and $Y$ can be represented as linear transformations of static vectors,

$$X = Wx,\ Y = Wy \tag{1}$$

where the transformation $W$ has a same form for all sequences [7].

In the GMM based approach [4], every $X_t$ and $Y_t$ are assumed to be independently drawn from a mixture of Gaussian distributions,

$$P(X_t, Y_t|\lambda) = \sum_{m=1}^{M} w_m \mathcal{N}(X_t, Y_t; \mu_m, \Sigma_m) \tag{2}$$

where $m$ is the index of mixture component, $w_m$, $\mu_m$ and $\Sigma_m$ denote for the mixture weight, mean and covariance of $m^{\text{th}}$ Gaussian. $\lambda = \{w, \mu, \Sigma\}$ denotes for the set of GMM parameters.

With GMM, batch conversion of a sequence is formulated as,

$$
\begin{aligned}
P(Y|X, \lambda) &= \prod_{t=1}^{T} P(Y_t|X_t, \lambda) \\
&= \prod_{t=1}^{T} \sum_{m_t=1}^{M} P(m_t|X_t, \lambda) P(Y_t|X_t, m_t, \lambda) \qquad (3) \\
\hat{y} &= \arg\max P(Y|X) \qquad (4)
\end{aligned}
$$

In practice, we make several approximations to reduce the complexity in solving Eq. 4. First, the summation in Eq. 3 is approximated by the Maximum A Posterior (MAP) mixture component, $\hat{m}_t$,

$$P(Y|X, \lambda) \approx \prod_{t=1}^{T} P(\hat{m}_t|X_t, \lambda) P(Y_t|X_t, \hat{m}_t, \lambda) \tag{5}$$

$$\hat{m}_t = \arg\max P(m|X_t, \lambda) \tag{6}$$

With this approximation, Eq. 4 can be solved in a closed form,

$$\hat{y} = (W^\top D_{\hat{m}}^{(Y)-1} W)^{-1} W^\top D_{\hat{m}}^{(Y)-1} E_{\hat{m}}^{(Y)} \tag{7}$$

where

$$E_{\hat{m}}^{(Y)} = \left[E_{\hat{m}_1}^{(Y)}, \ldots; \ldots; \ldots, E_{\hat{m}_t}^{(Y)}\right] \tag{8}$$

$$D_{\hat{m}}^{(Y)-1} = diag\left[D_{\hat{m}_1}^{(Y)-1}, \ldots; \ldots; \ldots, D_{\hat{m}_t}^{(Y)-1}\right] \tag{9}$$

and

$$E_{\hat{m}_t}^{(Y)} = \mu_{\hat{m}_t}^{(Y)} + \Sigma_{\hat{m}_t}^{(YX)} \Sigma_{\hat{m}_t}^{(XX)-1} (X_t - \mu_{\hat{m}_t}^{(X)}) \tag{10}$$

$$D_{\hat{m}_t}^{(Y)} = \Sigma_{\hat{m}_t}^{(YY)} - \Sigma_{\hat{m}_t}^{(YX)} \Sigma_{\hat{m}_t}^{(XX)-1} \Sigma_{\hat{m}_t}^{(XY)} \tag{11}$$

Second, to have a robust estimation of covariance matrix $\Sigma$, we assume the off-diagonal terms in $\Sigma_m^{(XY)}$ and $\Sigma_m^{(YX)}$ to be all null, and $\Sigma_m^{(XX)}$ and $\Sigma_m^{(YY)}$ to be diagonal. In other words, correlations between different dimensions in the joint audio-visual feature space are ignored. Eventually, Eq. 10 and Eq. 11 are simplified to be,

$$E_{\hat{m}_t}^{(Y)} \approx \mu_{\hat{m}_t}^{(Y)},\ \ D_{\hat{m}_t}^{(Y)} \approx \Sigma_{\hat{m}_t}^{(YY)} \tag{12}$$

Batch conversion comes with a latency time no less than the length of one utterance.

### 2.2. Sliding window-based conversion

To achieve low latency in real-time conversion applications, GMM based batch conversion has been adapted to generate trajectory with sliding windows. In particular, for the estimation of $\hat{y}_t$, a sliding window of length $L = L_f + L_b$ is taken around time $t$. Here $L_f = L_{\text{forward}}$ denotes for the number of look-ahead frames after $t$, whereas $L_b = L_{\text{backward}}$ is the number of frames in the past. At time $t$, the statistics in the sliding window can be denoted as,

$$X_{t,L} = [X_{t-L_b+1}^\top, \ldots, X_t^\top, \ldots, X_{t+L_f}^\top]^\top \tag{13}$$

$$E_{\hat{m},t,L}^{(Y)} = \left[E_{\hat{m}_{t-L_b+1}}{}^{(Y)}, \ldots; \ldots; \ldots, E_{\hat{m}_{t+L_f}}{}^{(Y)}\right] \tag{14}$$

$$D_{\hat{m},t,L}^{(Y)} = diag\left[D_{\hat{m}_{t-L_b+1}}{}^{(Y)}, \ldots; \ldots; \ldots, D_{\hat{m}_{t+L_f}}{}^{(Y)}\right] \tag{15}$$

Following Eq. 7, the solution in this sliding window is,

$$\tilde{y}_{t,L} = (W_L^\top D_{\hat{m},t,L}^{(Y)}{}^{-1} W_L)^{-1} W_L^\top D_{\hat{m},t,L}^{(Y)}{}^{-1} E_{\hat{m},t,L}^{(Y)} \tag{16}$$

All frames in the window are converted together, but we only keep the converted result for current frame $t$ as the estimation of $\hat{y}$, while neglect all others in the window. That is,

$$\hat{y}_t = \beta_L \tilde{y}_{t,L} \tag{17}$$

where $\beta_L = [0, \ldots, 0, 1_{L_b+1}, 0, \ldots, 0]$ is a row vector for picking up the $(L_b + 1)^{\text{th}}$ element of $\tilde{y}_{t,L}$.

In contrast to batch conversion, the sliding window-based approach generate lip movement $\hat{y}_t$ for the $t^{\text{th}}$ frame as soon as the $(t + L_f)^{\text{th}}$ frame $X_{t+L_f}$ becomes available, thus effectively reducing the latency to the number of look-ahead frames. On the other hand, few or no look-ahead frames often results in a quality degradation of converted trajectories [5]. Therefore, look-ahead frames can be treated as a trade-off between latency and conversion performance.

## 3. MINIMUM CONVERTED TRAJECTORY ERROR TRAINING

### 3.1. Conventional ML training

As a generative model, a GMM is usually trained by Maximum Likelihood (ML) estimation and the EM algorithm. The log likelihood function for a joint audio-visual GMM is,

$$\begin{aligned} L(\lambda) &= \log(\mathcal{N}([X,Y];\mu_m,\Sigma_m)) \\ &= -\log((2\pi)^D|\Sigma_m^{(XX)^{\alpha_X}}\Sigma_m^{(YY)^{\alpha_Y}}|^{\frac{1}{2}})) \\ &\quad -\frac{1}{2}\alpha_X(X-\mu_m^X)^\top\Sigma_m^{(XX)^{-1}}(X-\mu_m^X) \\ &\quad -\frac{1}{2}\alpha_Y(X-\mu_m^Y)^\top\Sigma_m^{(YY)^{-1}}(Y-\mu_m^Y) \end{aligned} \tag{18}$$

where $\alpha_X$ and $\alpha_Y$ are weighting factors for likelihood in acoustic and visual subspaces, respectively. In our previous work [6], we empirically choose weighting exclusively on the audio subspace ($\alpha_X = 1,\ \alpha_Y = 0$) which results in significant performance improvement than equal weights.

Maximum Likelihood training is an effective way to train the GMM. However, an audio-visual GMM with maximum likelihood on the training data does not necessarily result in the best converted visual trajectories. Alternatively, a discriminative criterion which directly minimizes the task specific error often outperforms ML. For audio-visual conversion, we have proposed a discriminative criterion called Minimum Converted Trajectory Error (MCTE) training which achieves better performance than conventional ML training.

### 3.2. MCTE for batch conversion

MCTE aims to minimize the converted trajectory error, i.e., Euclidean distance between converted trajectory and ground truth over all the training data,

$$D(y,\hat{y}) = ||y-\hat{y}||_2^2 = \sum_{t=1}^{T}||y_t-\hat{y}_t||_2^2 \tag{19}$$

$$L(\lambda) = \frac{1}{N}\sum_{i=1}^{N}D(y^i,\hat{y}^i) \tag{20}$$

Note that in GMM conversion with MAP approximation Eq. 5, the conversion is actually accomplished in two steps. First, a sequence of Gaussian mixtures is estimated from observation $X$ by MAP: $\hat{m} = \arg\max P(m|X,\lambda)$. Then, visual trajectory $\hat{y}$ is generated from mixture sequence $\hat{m}$ by maximizing $P(Wy|\hat{m},\lambda)$, leading to the closed form solution in Eq. 7. Thus, the MCTE loss function Eq. 20 becomes a function of $\lambda^{(Y)}$ for given mixture sequence $\hat{m}$. We minimize it using the probabilistic descend (PD) algorithm.

The probabilistic descend (PD) algorithm updates the model parameters at each training utterance [8]. For the $n^{\text{th}}$ utterance,

$$\begin{aligned} \lambda_{n+1}^{(Y)} &= \lambda_n^{(Y)} - \varepsilon_n\frac{\partial D(y^n,\hat{y}^n)}{\partial\lambda^Y}|_{\lambda^{(Y)}=\lambda_n^{(Y)}} \\ &= \lambda_n^{(Y)} - 2\varepsilon_n(\hat{y}^n-y^n)^\top\frac{\partial\hat{y}^n}{\partial\lambda^{(Y)}} \end{aligned} \tag{21}$$

By Eq. 7,

$$\frac{\partial\hat{y}^n}{\partial E_{\hat{m}_t,d}^{(Y)}} = (W^\top D_{\hat{m}}^{(Y)-1}W)^{-1}W^\top D_{\hat{m}}^{(Y)-1}Z_E \tag{22}$$

where $E_{\hat{m}_t,d}^{(Y)}$ is the $d^{\text{th}}$ dimension of the mean vector of the $t^{\text{th}}$ mixture in the MAP mixture sequence, $Z_E = [0,\ldots,0,1_{1\times D_Y+d},0,0,\ldots,0]^\top$ and $D_Y$ is the dimension of $Y$.

For convenience we denote $v_{t,d} = 1/\sigma_{t,d}^2,\ Z_v = Z_E Z_E^\top$. $\sigma_{t,d}^2 = D_{\hat{m}_t,d}^{(Y)}$ is the variance corresponding of $E_{\hat{m}_t,d}^{(Y)}$. The updating rule for covariance is,

$$\frac{\partial\hat{y}^n}{\partial v_{t,d}} = (W^\top D_{\hat{m}}^{(Y)-1}W)^{-1}W^\top Z_v(E_{\hat{m}}^{(Y)} - W\hat{y}^n). \tag{23}$$

### 3.3. R-MCTE for sliding window-based conversion

As the sliding window-based conversion generates visual trajectories in a different way, the original MCTE does not necessarily lead to reduced conversion errors. To resolve the inconsistency between MCTE training and sliding window-based conversion, in this section we propose R-MCTE which minimizes the generation error for sliding window-based conversion.

Recall the definition in Section 2.2, Eq. 17 indicates sliding window based-conversion operates independently on each frame. Consequently, the generation error should be defined as a summation over all frames over training data:

$$L(\lambda) = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T_i}||\beta_L\tilde{y}_{t,L}^i - y_t^i||_2^2 \tag{24}$$

Instead of updating model utterance-by-utterance as in original MCTE, the proposed R-MCTE performs frame-by-frame model updating. Specifically, for the minimization of Eq. 24, the probabilistic descent algorithm updates current model parameter $\lambda_k^Y$ at every $i^{\text{th}}$ frame of $n^{\text{th}}$ utterance in training data,

$$\begin{aligned} \lambda_{k+1}^{(Y)} &= \lambda_k^{(Y)} - \varepsilon_k\frac{\partial||\beta_L\tilde{y}_{t,L}^i - y_i^n||_2^2}{\partial\lambda_k^Y}|_{\lambda^{(Y)}=\lambda_k^{(Y)}} \\ &= \lambda_k^{(Y)} - 2\varepsilon_k(\hat{y}_i^n - y_i^n)\frac{\partial\beta_L\tilde{y}_{t,L}^i}{\partial\lambda^{(Y)}} \end{aligned} \tag{25}$$

In particular, by taking derivative on the sliding window version of generation function, we have,

$$\frac{\partial\beta_L\tilde{y}_{t,L}^i}{\partial E_{\hat{m}_t,d,i,L}^{(Y)}} = \beta_L(W_L^\top D_{\hat{m},i,L}^{(Y)}{}^{-1}W_L)^{-1}W_L^\top D_{\hat{m},i,L}^{(Y)}{}^{-1}Z_E \tag{26}$$

and for variance,

$$\frac{\partial\beta_L\tilde{y}_{t,L}^i}{\partial v_{t,d,i,L}} = \beta_L(W_L^\top D_{\hat{m},i,L}^{(Y)}{}^{-1}W_L)^{-1}W_L^\top Z_v(E_{\hat{m},i,L}^{(Y)} - W\hat{y}_{i,L}^n) \tag{27}$$

where $E_{\hat{m}_t,d,i,L}^{(Y)}$ is the $d^{\text{th}}$ dimension of the mean vector of $t^{\text{th}}$ mixture in the MAP mixture sequence of the sliding window at $i$. Similarly, $v_{t,d,i,L}$ is the definition for elements in covariance matrix.

The updating rule differs from original MCTE by the term $\beta_L$, which could be interpreted as a weighting vector that decides how errors of each converted frame in the sliding window contribute to the updating factor. In theory, $\beta_L$ should be set in the same form as in the conversion. In practice for robust model updating, $\beta_L$ can also be in other forms, e.g. a Gaussian window centers at $(L_b+1)^{\text{th}}$ frame in a sliding window.

|  | Batch |  | Real-time |  |
|---|---|---|---|---|
|  | open | closed | open | closed |
| ML | 9.36 | 8.943 | 9.106 | 8.697 |
| MCTE | 7.624 | 5.206 | 9.103 | 7.118 |
| R-MCTE | 7.588 | 5.695 | **7.967** | **6.252** |

**Table 1**. MSE ($\times 10^5$) of different training criterion on batch and sliding window-based real-time conversion. ($L_b = 190,\ L_f = 10$)

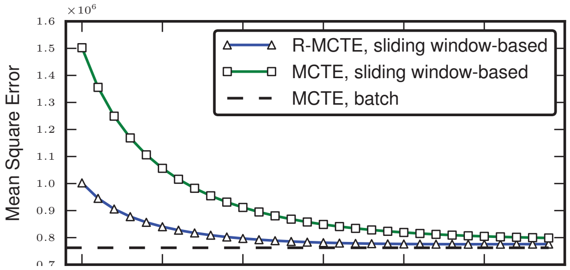

**Fig. 2**. Performance under different number of look-ahead frames. ($L_f = 0 \sim 30,\ L_b = 190$)

## 4. EVALUATION

### 4.1. Experimental Setup

We employ the LIPS 2008 Visual Speech Synthesis Challenge [9] data to evaluate the performance of different training criterion. The dataset consists of 278 video clips, each is an English sentence spoken neutrally by a female British native speaker. Videos are recorded at a 50 Hz frame rate. For acoustic feature extraction, we compute Mel-frequency Cepstral Coefficient (MFCC) from the speech with a 20ms time window shifted every 5ms. For the visual features we perform Principle Component Analysis (PCA) on the automatically detected and aligned mouth image, keeping the first 20 principle dimensions. Visual feature vectors are interpolated to the same frame rate as audio speech MFCCs.

### 4.2. Objective Evaluation

The conversion performance is objectively evaluated by Mean Square Error (MSE) defined as

$$MSE = \frac{1}{T} \sum_{t=1}^{T} ||y_t - \hat{y}_t||_2^2 \tag{28}$$

We conduct "open" and "closed" tests for objective evaluation. In the "closed" test, all 278 sentence in the data set are used for both training and conversion test. In the "open" test, we use a 14-fold cross-validation and the errors are averaged over all folds. In sliding window-based conversion, we set $L_b = 190$ and $L_f = 10$.

The results are shown in Table 1. In batch conversion, R-MCTE retains comparable performance as MCTE, while both are better than conventional ML. In sliding window-based conversion, the proposed R-MCTE shows a significant improvement compared with MCTE and is much closer to the best performance achieved by batch conversion. In other words, R-MCTE narrows the gap between performance of batch and sliding window-based conversion.

Fig. 2 shows the influence of look-ahead frames on conversion performance. Note that we train individual R-MCTE models for each look-ahead number $L_f$. R-MCTE (blue curve) consistently outperforms MCTE (green curve) especially on small look-ahead numbers. Also R-MCTE converges more quickly than MCTE as the number of look-ahead frames increases.

The choice of $L_f$ is critical since it is a tradeoff between performance and latency. In our application for rendering an interactive speech driven avatar, the avatar's mouth needs to move instantaneously in synch with the human user's voice. We choose $L_f = 10$ that results in an audio-visual asynchrony $\approx 50ms$ due to the latency. In speech, audio-visual signal with this amount of asynchrony remains intelligible to human perception [3].

## 5. CONCLUSION AND DISCUSSION

In this work, we propose R-MCTE training to directly refine the model towards minimum error on trajectories generated by sliding window based-conversion. R-MCTE training uses the probabilistic descent algorithm which updates model parameters on every frame in the training data. On the LIPS 2008 data set, the proposed method outperforms original MCTE in sliding window-based conversion. By R-MCTE, sliding window-based conversion achieves good performance comparable to batch conversion, while maintaining a much lower latency time which is suitable for real-time applications. This work has demonstrated the flexibility of MCTE for different conversion methods.

## 6. ACKNOWLEDGMENTS

We thank Dr. Yi-Jian Wu for helpful discussions about this work.

## 7. REFERENCES

[1] L. Xie and Z. Liu, "A coupled HMM approach to video-realistic speech animation," *Pattern Recognition*, vol. 40, no. 8, pp. 2325–2340, 2007.

[2] G. Englebienne, T. F. Cootes, and M. Rattray, "A probabilistic model for generating realistic lip movements from speech," in *NIPS*, 2007.

[3] K. Grant and S. Greenberg, "Speech intelligibility derived from asynchronous processing of auditory-visual information," in *AVSP 2001-International Conference on Auditory-Visual Speech Processing*, 2001, pp. 132–137.

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

[5] T. Muramatsu, Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, "Low-delay voice conversion based on maximum likelihood estimation of spectral parameter trajectory," in *INTERSPEECH*, 2008, pp. 1076–1079.

[6] X. Zhuang, L. Wang, F. K. Soong, and M. Hasegawa-Johnson, "A minimum converted trajectory error (MCTE) approach to high quality speech-to-lips conversion," in *INTERSPEECH*, 2010, pp. 1736–1739.

[7] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *ICASSP*, 2000, pp. 1315–1318.

[8] Y.-J. Wu and R.-H. Wang, "Minimum generation error training for hmm-based speech synthesis," in *ICASSP*, 2006, pp. 89–92.

[9] B.-J. Theobald, S. Fagel, G. Bailly, and F. Elisei, "LIPS2008: visual speech synthesis challenge," in *INTERSPEECH*, 2008, pp. 2310–2313.