Vocoder-side Voice Morphing for TTS

Product Requirements

Why Voice Morphing?

Recording new voices has huge cost.

Goal Hierarchy

1. No intelligibility penalty
2. High-quality (Naturalness)
3. Speaker similarity

Outline

- Background
- The Matching-Under-Transform problem
- The Matching-Minimization algorithm
- The Optimal-Dynamic-Frequency-Warping-and-Weighting algorithm
- Voice-Morphing algorithm design
- Results

Background

- Voice Conversion:
  - Traditional: Parallel corpora: [Stylianou], [Kain]
  - Modern: Non-parallel corpora: [Mouchtaris], [Erro], [Godoy], [Rosec], [Silen]
  - Adaptation HMM-based TTS: [Tokuda], [Zen], [Yamagishi], etc
    - speaker factorization, eigen-speakers

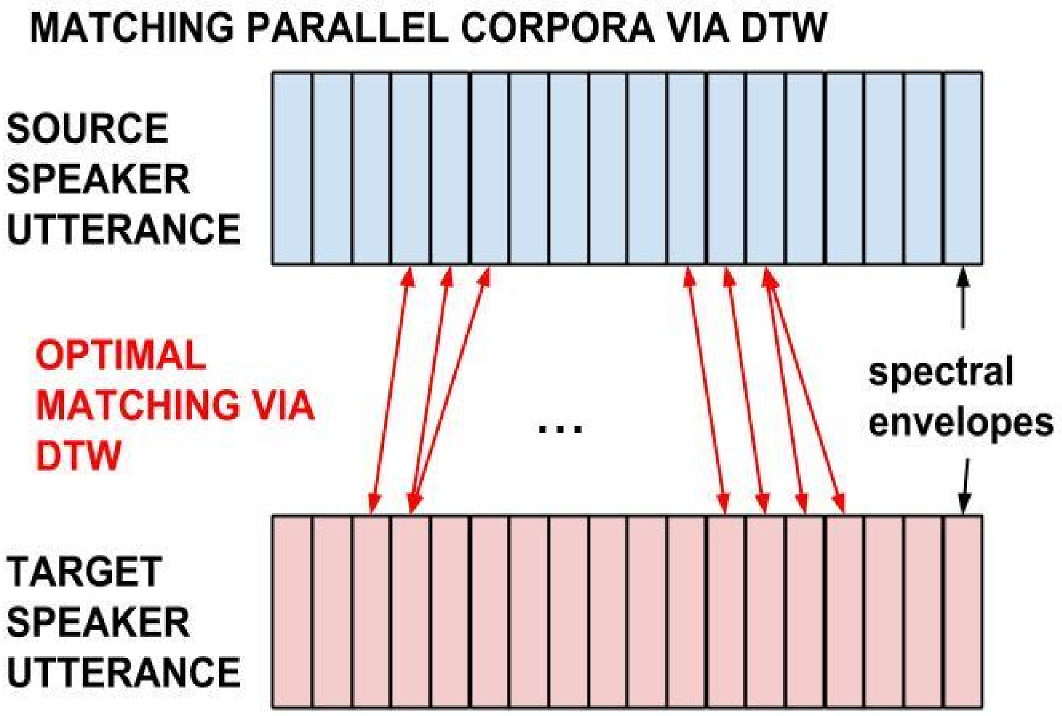
- GMM-based regression function: [Stylianou],[Kain]

$$F(\vec{x}) = \sum_{k=1}^{K} p(k|\vec{x}) \left[\vec{\mu}_{y,k} + \Sigma_{yx}^{k} \Sigma_{xx}^{k,-1} (\vec{x} - \vec{\mu}_{x,k})\right]$$

Background

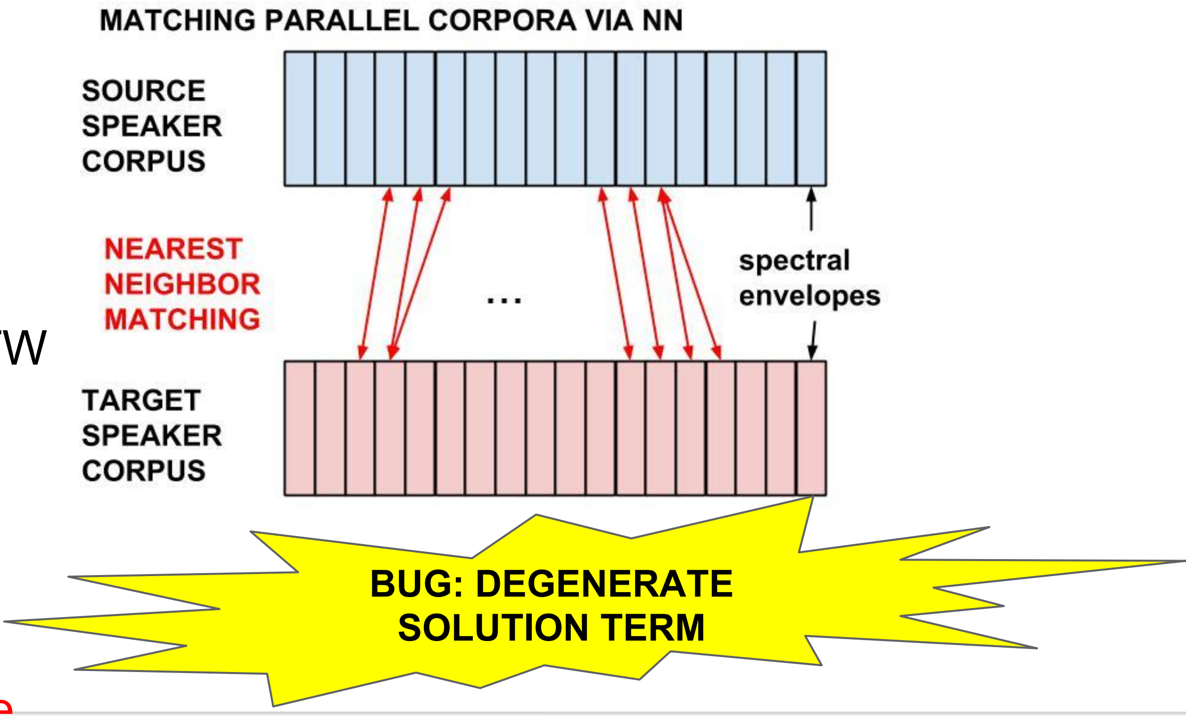
- **Traditional approach** ([Moulines], early 90s)
    a. align spectra via DTW

- **Improved Traditional approach** ([Stylianou], mid 90s)
    a. align spectra via DTW
    b. convert spectra
    c. re-align converted spectra
    d. re-convert spectra
    e. iterate until convergence



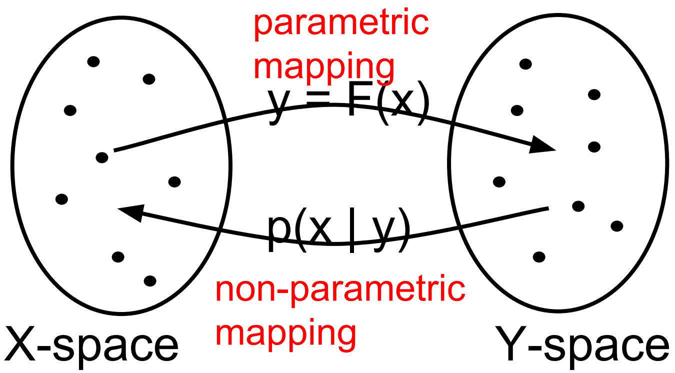
Background

- **Matching Minimization:**
  - We do not need aligned utterances → replace DTW with Nearest Neighbor
  - NN matching error < DTW matching error
- **INCA algorithm:**
  a. match spectra via NN
  b. convert spectra
  c. iterate until convergence



Problem: Matching-Under-Transform

Problem: Match the vectors of a dataset Y ~ P(Y) to the vectors of a dataset X ~ P(X) under a compensating transform Y = F(X).



Matching-Under-Transform

- Two sets of vectors: X, Y
- Parametric mapping: X →Y
  - transform that compensates speaker differences

$$F(\vec{x}) = \vec{\beta} + A\vec{x}$$

- Non-parametric mapping: Y → X
  - finds correspondences

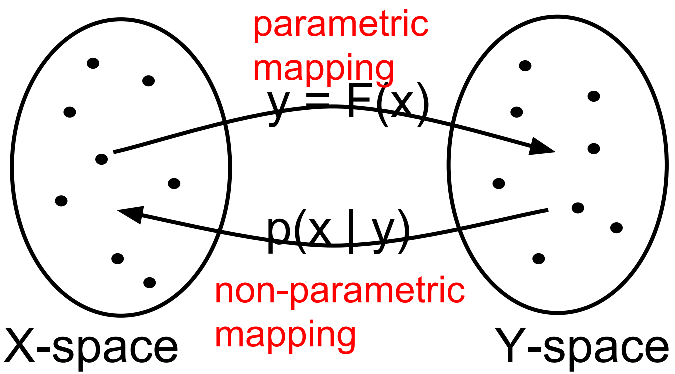
$$p(\vec{x}_n | \vec{y}_q)$$

- Distortion criterion:

$$d(\vec{y},\vec{x}) = (\vec{y} - F(\vec{x}))^T W (\vec{y} - F(\vec{x}))$$

- Global distortion:

$$D = \sum_q p(\vec{y}_q) \sum_n p(\vec{x}_n | \vec{y}_q) d(\vec{y}_q, \vec{x}_n).$$



Matching-Under-Transform

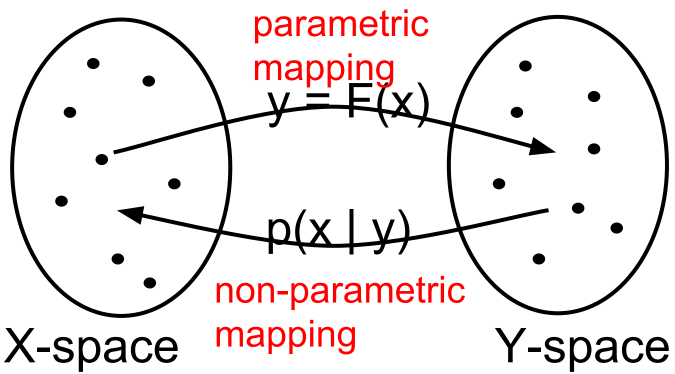
- Deterministic Annealing:

$$D' = D - \lambda H(X|Y)$$

- Optimizing non-parametric mapping (many-to-many):

$$p(\vec{x}_n|\vec{y}_q) = \frac{\exp\{-\frac{1}{\lambda}d(\vec{y}_q, \vec{x}_n)\}}{\sum_i \exp\{-\frac{1}{\lambda}d(\vec{y}_q, \vec{x}_i)\}}.$$

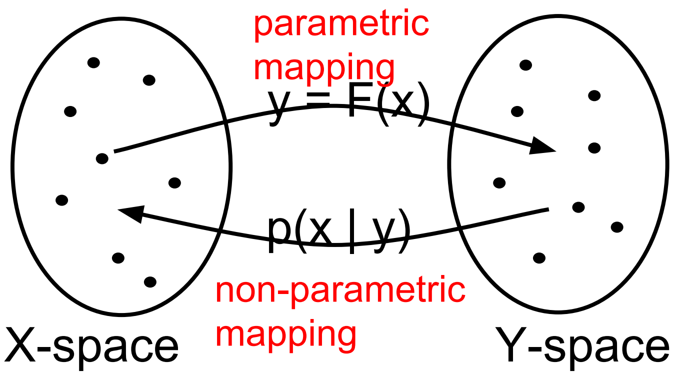
- Optimizing parametric mapping:
  - closed form solution
- When $\lambda \rightarrow 0$ the stochastic many-to-many mapping becomes a deterministic many-to-1 matching



Matching-Minimization

$$p(\vec{x}_n|\vec{y}_q) = \frac{\exp\{-\frac{1}{\lambda}d(\vec{y}_q,\vec{x}_n)\}}{\sum_i \exp\{-\frac{1}{\lambda}d(\vec{y}_q,\vec{x}_i)\}}.$$

$\lambda \to 0$: Nearest Neighbors



Matching-Minimization algorithm: Iteratively optimizes parametric & non-parametric mappings until convergence:

1. optimize non-parametric mapping given transform
2. optimize transform given non-parametric mapping

Matching-Minimization

More expressive transforms:

GMM-based regression function:

$$F(\vec{x}) = \sum_{k=1}^{K} p(k|\vec{x}) \left[\vec{\mu}_k + \Sigma_k \vec{x}\right]$$

For block-diagonal matrix (MCEP + Delta + Delta-Delta) with block Σ'_k:

$$\Sigma_k \vec{x} = (\vec{x}^T \otimes I_D) R \Sigma'_k \equiv X'_n,$$

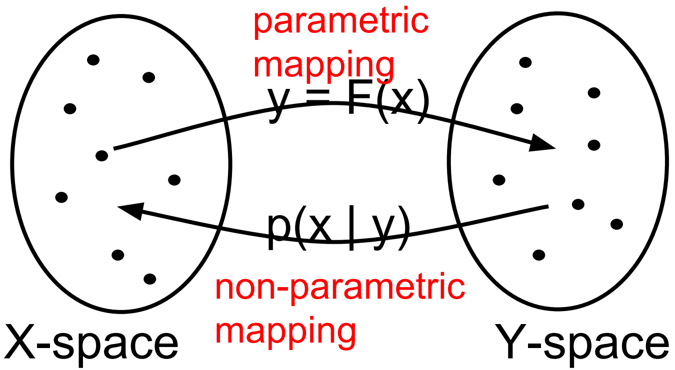
The solution is:

$$\vec{\gamma} = -\left(\sum_q p(\vec{y}_q) \sum_n p(\vec{x}_n|\vec{y}_q) \Gamma_n^T W_q \Gamma_n\right)^{-1} \left(\sum_q p(\vec{y}_q) \sum_n p(\vec{x}_n|\vec{y}_q) \Gamma_n^T W_q \vec{y}_q\right).$$

Matching-Minimization

Matching-Minimization as an adaptation algorithm:

- Recovers both transform & matching (local minima)
- Non-parametric on the data: makes no assumptions regarding the underlying distributions in X, Y
- Parametric only on the transform
- Uses Mean-Squared-Error instead of likelihood



- Other applications:
  - Nearest-neighbor-like non-parametric adaptation for ASR: VTL, recording compensation, data denoising, near/far field, mixed narrowband/wideband training data

Optimal Dynamic Frequency Warping & Weighting

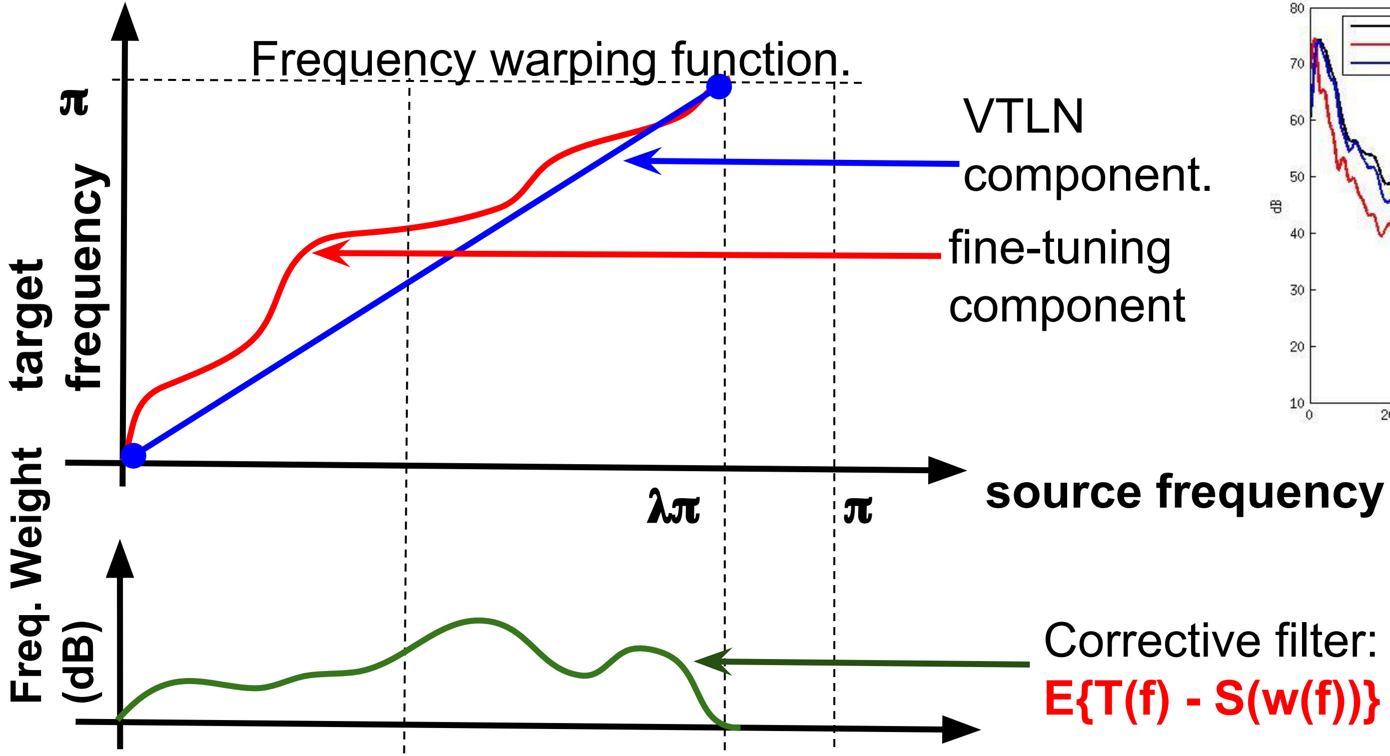
Morphed Source Spectral Envelope: (w(f) the frequency warping function and B(f) the frequency weighting):

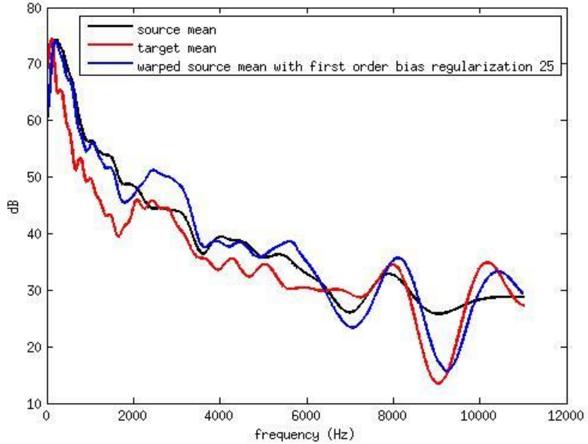
$$\hat{T}_n(f) = S_n(w(f)) + B(f)$$

Error Criterion:

$$\epsilon = \frac{1}{N} \sum_{n=1}^{N} \int_0^{\pi} (T_n(f) - S_n(w(f)) - B(f))^2 \, df$$

Optimal Dynamic Frequency Warping & Weighting

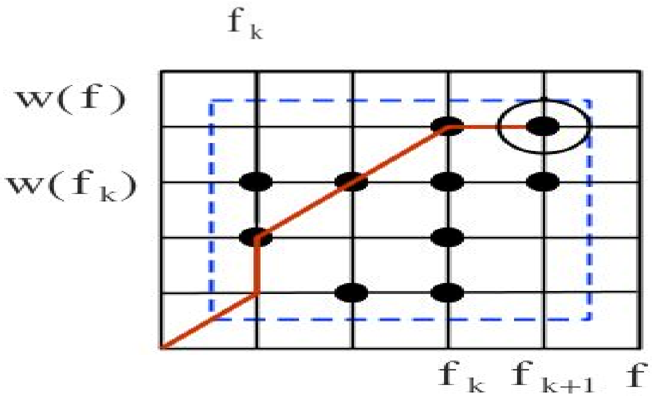




Optimal Dynamic Frequency Warping & Weighting

Optimize jointly the frequency warping function and the frequency weighting in the *continuous* frequency domain using a variant of the Viterbi algorithm:

$$\hat{w}, \hat{B}(f): \underset{w,\ B(f)}{\operatorname{argmin}}\ \epsilon$$



- Frequency domain is discretized in many frequency bins.
- For each frequency bin, a frequency warping and weighting

  are estimated via a viterbi iteration.

- Viterbi is used to find the optimal path.
- Search neighborhood in black dots.

Voice Morphing Algorithm Design

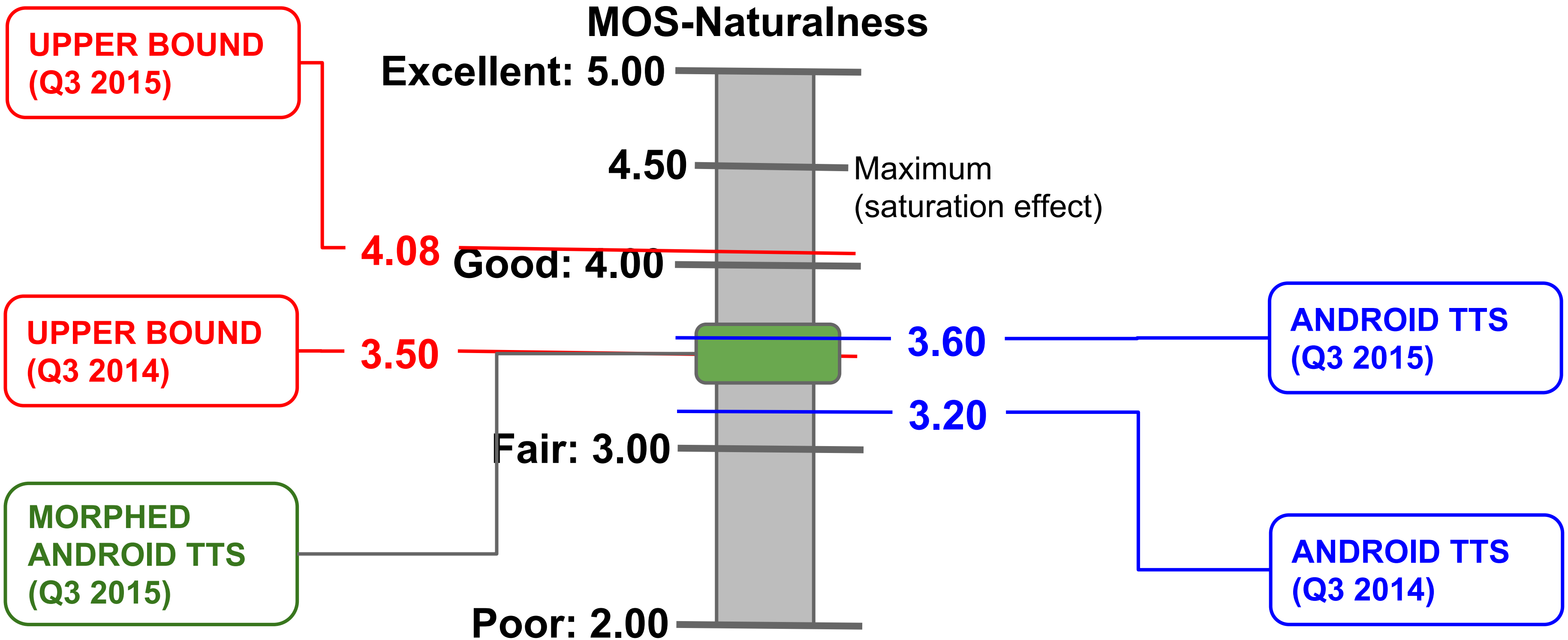
- **Source speaker:** TTS corpus
- **Target speaker:** 10-150 utterances

1. **Alignment step:** match source/target speaker spectra

2. **Training step:** find optimal transform from source->target speaker

3. **Run-Time Synthesis step:** transform TTS output audio.

Results: MOS of Morphed Speech



Results (EN-US, DEV Vocaine-LSTM)

Algorithm	MOS
22KHz PRODUCTION TTS: greco_barracuda_sample_rate_22050_en_us_sfg	**3.798** ± 0.132
MORPHED VOCAINE-LSTM TTS: morph_lstm_en_sfg_dev_vctk_20150603_sfg_to_p362	**3.794** ± 0.097
PRODUCTION TTS: greco_barracuda_sample_rate_16000_en_us_sfg	**3.776** ± 0.117
MORPHED VOCAINE-LSTM TTS: morph_lstm_en_sfg_dev_vctk_20150603_sfg_to_p269	**3.757** ± 0.099
VOCAINE-LSTM TTS: lstm_en_sfg_dev_20150603	**3.737** ± 0.091
MORPHED VOCAINE-LSTM TTS: morph_lstm_en_sfg_dev_vctk_20150603_sfg_to_p330	**3.723** ± 0.115
MORPHED VOCAINE-LSTM TTS: morph_lstm_en_sfg_dev_vctk_20150603_sfg_to_p244	**3.693** ± 0.088
MORPHED VOCAINE-LSTM TTS: morph_lstm_en_sfg_dev_vctk_20150603_sfg_to_p351	**3.677** ± 0.094

Results (EN-US, DEV Vocaine-LSTM)

Algorithm A	Algorithm B	Mean Score	Significant	preference
Morphed Vocaine-LSTM (p362)	Vocaine-LSTM	0.107 ± 0.101	A is significantly better than B	A (38.4%), B (30.5%)
Morphed Vocaine-LSTM (p362)	16kHz Barracuda	-0.191 ± 0.164	A is significantly worse than B	A (37.3%), B (45.9%)
Vocaine-LSTM	16 kHz Barracuda TTS	-0.351 ± 0.170	A is significantly worse than B	A (30.8%), B (50.4%)
Morphed Vocaine-LSTM (p269)	Vocaine-LSTM	0.096 ± 0.060	A is significantly better than B	A (39.1%), B (33.3%)
Morphed Vocaine-LSTM (p362)	22 kHz Barracuda	-0.346 ± 0.173	A is significantly worse than B	A (35.1%), B (51.9%)
Vocaine-LSTM	22kHz Barracuda TTS	-0.611 ± 0.170	A is significantly worse than B	A (26.8%), B (58.8%)

Results - Real-Time Ratio

Conditions

- en-US
- Nexus 7 2013 (deb)

Device	Real-time ratio	
	Without Morphing	With Morphing
deb, Normal speed (1x)	46%	50%
deb, Rapid speed (2x)	67%	76%
deb, Fastest speed (4x)	118%	136%

Examples - Changing voice character per word.

The Bear and the Dragon.

The Bear said "But I am just a bear. How can I fight a dragon?" with a deep voice.

The Wolf and the Dog (fable).

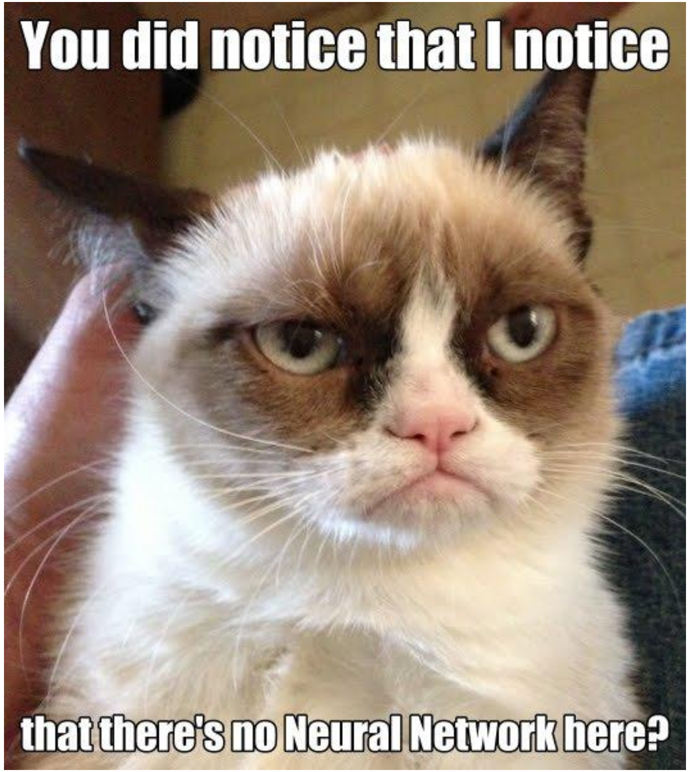
A gaunt Wolf was almost dead with hunger when he happened to meet a House-dog who was passing by.
- "Ah, Cousin",
said the Dog.
- "I knew how it would be; your irregular life will soon be the ruin of you. Why do you not work steadily as I do, and get your food regularly given to you?."
- "I would have no objection",
said the Wolf,
- "if I could only get a place.".

Results

- Quality of morphed voices rivals quality of their source voices.

- Some morphed voices are **significantly better** than their source voices (Post-recording speaker correction?)

- Quality of best morphed voice gets pretty close to the quality of 16 kHz Barracuda TTS (current production TTS).



References

- Yannis Agiomyrgiannakis, "The Matching-Minimization algorithm, the INCA algorithm and a mathematical framework for Voice Conversion with unaligned corpora", ICASSP 2016
- Yannis Agiomyrgiannakis, Zoe Roupakia, "Voice Morphing that improves TTS quality using an Optimal Dynamic Frequency Warping-and-Weighting transform", ICASSP 2016.

“The theory is simple but it is hard to implement.”

The only way to know whether you know the theory is to implement it.