Presented at the 1996 International Conference on Acoustics, Speech, and Signal Processing, Atlanta, GA, May 7-10, 1996.

AUTOMATIC AUDIO MORPHING

Malcolm Slaney, Michele Covell and Bud Lassiter

Interval Research Corporation
1801 Page Mill Road, Building C
Palo Alto, CA 94304; USA
malcolm@interval.com

ABSTRACT

This paper describes techniques to automatically morph from one sound to another. Audio morphing is accomplished by representing the sound in a multi-dimensional space that is warped or modified to produce a desired result. The multi-dimensional space encodes the spectral shape and pitch on orthogonal axes. After matching components of the sound, a morph smoothly interpolates the amplitudes to describe a new sound in the same perceptual space. Finally, the representation is inverted to produce a sound. This paper describes representations for morphing, techniques for matching, and algorithms for interpolating and morphing each sound component. Spectrographic images of a complete morph are shown at the end.

1. INTRODUCTION

This paper describes techniques to automatically morph from one sound to another. In video, morphing is a process of generating a range of images that smoothly move from one image to another. In a good morph, the in-between images all show one object smoothly changing its shape and texture until it turns into another object. We would like the same thing to happen in an audio morph. A sound that is perceived as one object should change smoothly into another sound, maintaining the shared properties of the starting and ending sounds and smoothly changing the other parameters.

Figure 1 shows a block diagram of our approach. Audio morphing is accomplished by representing the sound in a multi-dimensional space that can be warped or modified to produce a desired result. After matching components of the sound, a morph smoothly interpolates the sound amplitudes to describe a new sound in the same perceptual space. Finally, the representation is inverted to produce a sound. The body of this paper describes representations for morphing, techniques for matching, and algorithms for interpolating and morphing each sound component. Spectrographic images of a complete morph are shown at the end.

2. RELATED WORK

Previous work in audio morphing has used sinusoidal analysis [1]. This paper describes techniques based on magnitude spectrograms. In a sense we have taken sinusoidal analysis to its limit and allowed any sound to be easily and completely represented. Far from complicating the problem, spectrograms make morphing easier because it is no longer necessary to track sinusoids and their phase [2].

Work described elsewhere [3] allows spectrograms, without their phase information, to be inverted to find a sound that has the same magnitude spectrogram. Using magnitude spectrograms to represent the sound allows us to make dramatic changes to the spectrograms and not worry about the phase. The phase will be recovered later as part of the spectrogram inversion process.

This paper describes techniques for automatically morphing one sound into another. We use a rich, multi-dimensional representation to describe sound, so it is no longer easy to see the best matches. Auto-correspondence methods, as described for video [4], provide accurate matches without human intervention. A similar philosophy is used here.

This work is different from that reported on voice transformations [5]. Voice transformations change one speaker's utterances to match the *statistical* properties of another voice. Thus every time an /a/ is spoken, the formant frequencies are changed to match the target speaker's formants. This work, on the other hand, generates new sounds that are in between two exemplars.

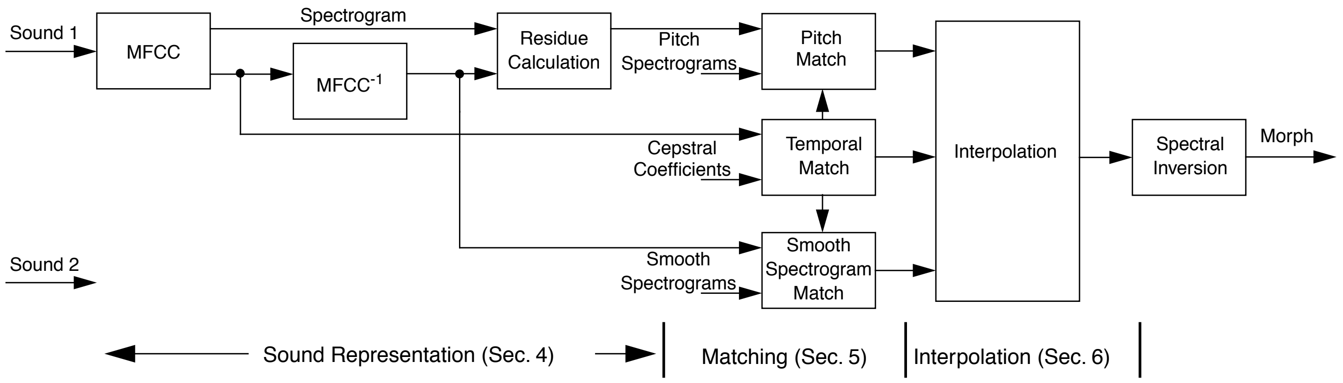

Figure 1. The three stages of audio morphing, representation, matching, and interpolation, are shown. The signal path for representing sound 2 is not shown to simplify the drawing.

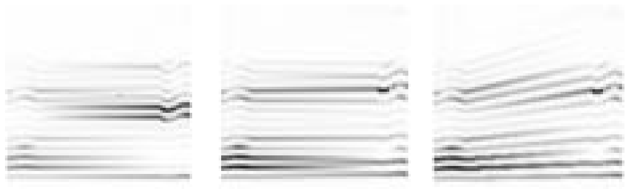

Figure 2. These three magnitude spectrograms show simple spectrogram morphs. The left example shows a cross-fade between /a/ and /i/. The middle example shows a cross-fade between two /a/'s at different pitches. The lines crossing near the middle cause the morph to be perceived as two sounds. The right example shows the same start and end as the middle example, but the harmonics are aligned before cross-fading. The pitch splitting is fixed, but the formant frequencies have moved.

3. TEMPORAL ASPECTS OF AUDIO MORPHING

Time is a special component of sound. Sound does not exist without time. This simplifies audio morphing because sounds that happen at the same time are perceived together. Thus an audio morph should keep simultaneous components of a sound aligned in time throughout the morph. This means that, unlike image morphing, an important dimension of sound, the timing, can be considered independently of the other sound dimensions. The morphs described here consider time separate from the other dimensions of the auditory signal. As will be shown, the separability of the temporal dimension simplifies all aspects of audio morphing.

Time complicates other aspects of audio morphing. Most importantly, there are three kinds of audio morphing. In the simplest case, the two sounds are stationary and we can describe the sounds as points in a high-dimensional space. The dimensions of this space include spectral shape, pitch, rhythm and any other perceptually relevant (and quantifiable) auditory dimensions. We morph between the two sounds by tracing a path between the two points in an appropriately warped space. This is directly analogous to the image morphing case. In the simplest form, a steady vowel morphs into a single note from an oboe.

The second kind of morph is between moving objects. The morph starts with the characteristics of the first sound and slowly changes to have all the characteristics of the second. This is directly analogous to morphing between videos of two different objects.

Finally, there is a unique kind of audio morph which is generated by smoothly changing a repetitive sequence of sounds. The word xxx changes to yyy in a sequence of steps. Each step is small and in the middle of the sequence the word sounds like something in between xxx and yyy. The result is a cyclostationary morph. It is cyclic because we play the sound repetitively to affect the morph. It is stationary since each sound instance is a completely stationary (no change) example of the range of in-between sounds.

4. REPRESENTATIONS

A proper representation of sound is key. In video, a retinotopic image is natural and easy for humans to change. There is no obvious choice for audio. Conventional spectrograms can represent any sound, but cross-fading spectrograms does not produce convincing morphs.

The problems with spectrograms are illustrated in the three examples of Figure 2. In these examples, magnitude spectrograms of two vowels are interpolated. A short section of the original voice with vibrato is included at the beginning and end of the spectrogram to provide context. The middle of the spectrogram shows the morph.

In the first example, the singer's /a/ is cross-faded to her /i/. The morph is convincing because the pitch is similar across the morph. In the second example, the morphed sound has two separate pitches, causing the sound to be perceived as two different auditory objects, and destroying the illusion of a continuous morph. Finally, by scaling the frequencies of the spectrogram, much like sinusoidal analysis would do, interpolation can be done across different pitches to produce a proper morph. However, this simple scaling does not work if there are drastic pitch changes because formants move with the harmonics.

We would like a multi-dimensional representation of sound where each dimension is independent and salient. Then we could morph the sound by simple interpolation in this ideal space. Instead, this work approximates the ideal by decomposing the sound into a smooth spectrogram that represents the broad spectral shape, and a second "pitch" spectrogram that encodes the pitch and voicing of the sound.

We use mel-frequency cepstral coefficients (MFCC) to model part of the sound [6]. Cepstral coefficients are a type of homomorphic processing which allows us to separate the broad spectral characteristics of the sound from the pitch and voicing information. The MFCC coefficients are used in the initial temporal matching and to compute the smooth spectrogram.

MFCC is computed by resampling a conventional magnitude spectrogram to match critical bands as measured by auditory perception experiments. After computing logarithms of the filter-bank outputs a low-dimensional cosine transform is computed.

The MFCC representation is inverted to generate a smooth spectrogram for the sound. After applying the cosine transform again and undoing the logarithm we have a smooth estimate of the filter-bank output. The filter-bank output is then reinterpolated to get a spectrogram. The logarithmic transform and low quefrency cosine transform serve to filter out the pitch information in the spectrogram. MFCC is good at modeling the overall spectral shape, but it doesn't include pitch. When we invert MFCC we get a rough approximation of the spectrogram, but without the pitch information.

It would be nice if we could summarize all the information about pitch with a small number of scalars and then smoothly vary these numbers to get intermediate excitations. For example, we might use one number for the pitch and one to indicate the amount of voicing. Unfortunately, this type of summarization is not sufficient as is seen in speech compression systems. Simple LPC systems suffer from objectionable inaccuracies in the excitation. To provide acceptable reconstructions, a large codebook is needed to summarize the possible residues.

In audio morphing we use a spectrogram of the residue to code the pitch and voicing in the acoustic signal. A conventional short-time spectrogram $S(\omega, t)$ encodes all the information in the signal and the smooth spectrogram $S_s(\omega, t)$ describes the overall spectral shape. Dividing the short-time spectrogram, S, by the smooth spectrogram, S_s, gives us a "pitch" or residual spectrogram, $S_p(\omega, t)$, which describes the pitch and voicing information in the sound. The smooth and pitch spectrograms form the basis of our morphing techniques and are illustrated in Figure 3. We recover the original spectrogram by multiplying the pitch and smooth spectrograms together.

5. MATCHING

Matching is necessary so that we know which features of the first sound correspond to any particular feature of the second. Often a feature has moved and to affect a morph we need to slowly move the feature from where it is in the first sound to its position in the second. There are many ways to perform the matching. Dynamic time warping and harmonic alignment are used to match features in audio morphing.

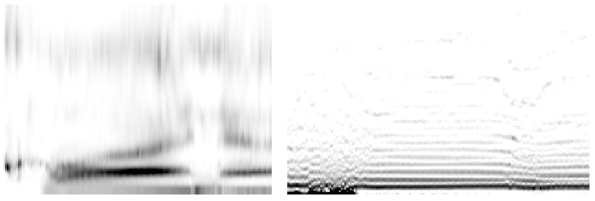

Figure 3. A smooth spectrogram for the word “corner” is shown on the left. Its pitch spectrogram is shown on the right. The smooth spectrogram encodes the broad spectral shapes and the pitch and voicing is encoded in the pitch spectrogram.

Dynamic Time Warping (DTW) is used to find the best temporal match between the two sounds. Over the course of the morph, we want features that are common to both sounds to remain relatively fixed in time. MFCC is often used in modern speech recognition systems as a distance metric and is used here for the same purpose. Using DTW allows us to calculate the best match between the two sounds so that later spectral stages have less work.

Audio morphs with different properties are created with different matching functions. In morphing between two versions of the same song, the melody is important. The temporal matching is done with a distance metric based on the dominant pitch. For other music (i.e. rap) we will want to consider the underlying rhythm.

In this work we have represented pitch and voicing information as a spectrogram. In a pitch spectrogram the pitch information in the sound is visible as a series of peaks. The spacing of the peaks is proportional to the pitch. When the sound is unvoiced the peaks disappear and the “pitch spectrum” is flat.

To smoothly morph the pitch spectrogram we need to match the pitch, if present, and then cross-fade the amplitude at each frequency. Unfortunately, the pitch might be absent or difficult to find at each point. We also have to deal with times when one sound has a pitch and the other doesn’t. When there is a pitch, we want to match it in the two sounds, otherwise we want to cross-fade the noise.

To solve this problem we estimate a pitch for the entire utterance. We use a combination of a conventional pitch scheme and dynamic programming to find a “pitch” everywhere. The basic pitch algorithm (autocorrelation of the peak enhanced waveform) produces many possible pitch peaks. It is difficult to know, without more information, which is the best pitch.

Secrest and Doddington [7] propose using dynamic programming to estimate a pitch that fits the available data (the peaks in the pitch spectrogram) and smoothly changes over time. We use this to calculate a “pitch” estimate for the entire sound, whether it is actually voiced or not.

We use the complete pitch estimate from both sounds to perform the match. It is most important to match the pitch between the two sounds, and less important to match the inharmonic residual. Thus we want to stretch and compress the frequency axis of the pitch spectrograms to make sure the pitch peaks agree before we cross-fade the two spectrograms. Depending on the change in pitch, the unvoiced components of the sound will move in frequency. This is less important than not splitting the harmonics that cause pitch.

Matching the features of the smooth-spectrogram is less critical. Researchers have investigated the proper domain to do interpolation for voice coding [8]. They argue for cross-fading the spectral shapes (without pre-warping) or for interpolating the spectral peak locations by cross-fading line spectral pairs (LSP). Section 7 discusses the results of both approaches.

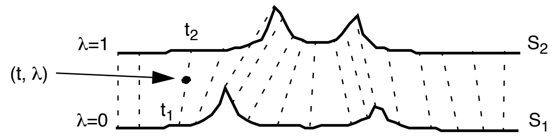

Figure 4. One-dimensional morphing (either in frequency or time) proceeds by warping along the dashed matching lines, $p(t_1) = t_2$, and cross-fading the signals.

6. ONE DIMENSIONAL MORPHING

A morph includes some type of interpolation step. Scalar quantities are easiest to morph because it reduces to a simple cross-fade. If one component of a sound description is loudness, then the loudness of the morph should change smoothly from the loudness of the first sound to the loudness of the second.

Unfortunately, acoustic information is not always scalar. Temporal alignment and spectral warping share the same problem. Given a dense match between two one-dimensional curves, how do we smoothly morph between these curves?

The data we are trying to morph is described as $s_1(t)$ and $s_2(t)$. We want to find a new curve $s(\lambda, t)$ such that the s function is between the s_1 and s_2 curves. Since the match functions are monotonic, we know that matching lines do not cross and for each point (λ, t) there is only line establishing the correspondence. Our problem simplifies to finding the times t_1 and t_2 that should be interpolated to generate the data at (λ, t).

We do this by calculating the path location for all (sampled) values of t_1 and then picking the t_1 whose path is closest to the desired sample point, (λ, t). Given lines ending at t_1 and t_2 as shown in Figure 4, the intersection with the λ morphing line is at

$$\frac{t - t_1}{t_2 - t_1} = \lambda \quad \Rightarrow \quad t = \lambda(t_2 - t_1) + t_1$$

Given the proper values for t_1 and t_2, we generate the new data at (λ, t) by cross-fading the warped signals

$$s(\lambda, t) = (1 - \lambda)s_1(t_1) + \lambda s_2(t_2)$$

This results in a smooth cross-fade between s_1 when $\lambda = 0$ and s_2 when $\lambda = 1$.

Mappings between s_1 and s_2 are described as paths. Path p warps s_2 to look like s_1. Thus p is the mapping that produces the smallest different between $s_2(p(t))$ and $s_1(t)$. Using this, we can simplify the above equation so that the intermediate t is given by

$$t = \lambda(p(t_1) - t_1) + t_1$$

We could use the path map to calculate the appropriate t_2, but we get better results if we repeat the procedure in the other direction. Because of quantization, more than one point along the t_1 axis might map into the same t_2. When λ is equal to one, we will not get an exact copy of s_2, but some points will be slightly out of place. It is better to repeat the procedure used to find the best t_1 to find the best t_2. This is used to calculate the second half of the equation for s above.

7. RESULTS

Figure 5 shows a complete morph. This morph was generated by splitting two sounds (“morning” and “corner”) into smooth and pitch spectrograms. Dynamic programming is used on the peaks in the pitch spectrograms to summarize the pitch information in the two sounds. The MFCC vectors are used in dynamic time warping to time align the sounds. The pitch spectrograms are scaled in frequency to align pitch contours and cross-faded.

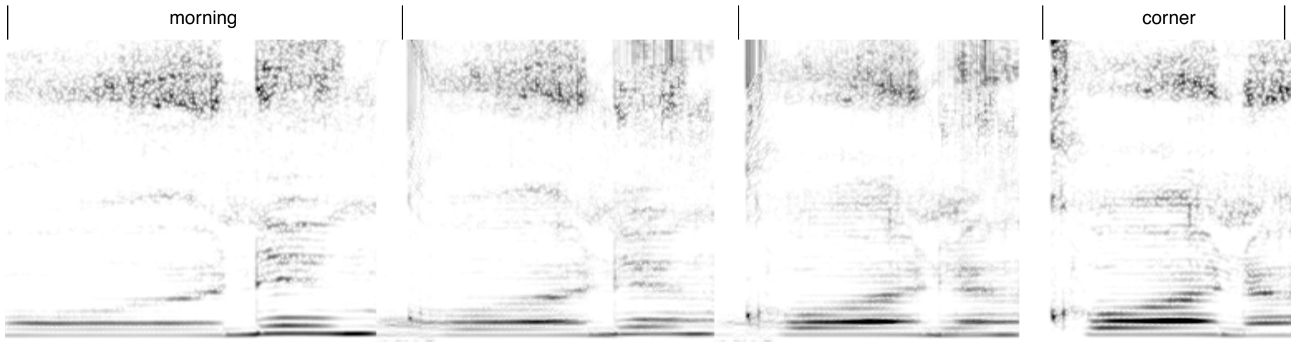

Figure 5. The words “morning” and “corner” are shown in a cyclo-stationary morph along with two of the intermediate words. The time scale of the word is changed during the morph. At each corresponding frame, the pitch, voicing, and spectral shapes are independently morphed. (The original sound samples are from the first verse of the recording of “Tom’s Diner” by Suzanne Vega.)

In the results shown in Figure 5, the smooth spectrograms are cross-faded. The interpolated spectrograms are combined and inverted to recover the morphed sounds. This approach results in high-quality morphs. Part of the reason that this simple approach to smooth-spectrogram interpolation works so well may be because MFCC was used to do the temporal pre-alignment. The pre-alignment helps to insure that the two MFCC vectors are similar. The use of MFCC in many speech recognition systems would imply that a smooth spectrogram as calculated by inverting MFCC is well behaved. Sounds that are close in the smooth-spectrogram representation should sound similar.

An alternative approach to matching the smooth spectrograms was also considered. This second approach is analogous to Yong's approach of blending LSPs. Instead of using LSP, we applied dynamic warping, this time on the smooth spectra as a function of frequency, to match peaks in the two sounds.[1] The results in our limited testing do not sound as good as cross-fading the smooth spectrograms.

The entire process is not expensive. The cost of the homomorphic processing is dominated by the cost of the initial spectrogram calculation. Depending on the breadth of the search, dynamic time warping can be expensive, as much as $O(N^2)$ where N is the number of spectrogram frames. Spectrogram inversion techniques described elsewhere [3] allow the iterative procedure to quickly converge, often at a cost only four to five times as expensive as the original spectrogram calculation.

Perhaps the biggest obstacle is that spectrogram inversion techniques need overlapping windows in the time domain. Estimating the phase of a spectrogram can be done, but each point in the waveform must be included in two different spectral slices. A four way overlap is even better. Thus using 256 point windows, as in this work, means that a new spectral slice is calculated every 64 points.

8. CONCLUSIONS

Previous work in audio morphing has been shaped by the constrained representations that are used in speech and music synthesis. This paper describes a new approach based on separate spectrograms to encode the pitch and broad spectral shapes of the sound. These spectrograms are independently modified to create pleasing morphs between many sounds.

An important contribution of this work is the realization that, unlike image morphing, audio morphing can effectively be separated into multiple, independent dimensions. This paper has used as its dimensions: time, smoothed spectral shape and high-pass or “pitch” residual. Finally, this work has investigated techniques for matching these independent dimensions.

Future work on audio morphing should revolve around better representations, better matching techniques and more natural sounding interpolation schemes. Spectrograms are a good representation of sound, but better representations will allow the details of the pitch and voicing information to be separated. Automatic correspondence simplifies the morphing procedure, but different matching functions will be necessary for different tasks. Finally more work is needed to find perceptually optimal interpolation functions.

1. Our preference for the smooth spectrogram is because LSP is not a universal representation of sound. It is optimized for voice. The smooth spectrogram contains much the same information as LSP but is more general.

9. ACKNOWLEDGMENTS

We would like to thank Meg Withgott, John Tukey, Golan Levin, David Burgess, John Lewis, Glenn Edens, Steve Saunders, Jim LaBarre, Bill Lynch and Frank Crow for their support, encouragement, ideas and feedback.

10. REFERENCES

[1] E. Tellman, L. Haken, B. Holloway, “Timbre morphing of sounds with unequal numbers of features,” *Journal of the AES*, 43(9), pp. 678-689, 1995.

[2] P. Depalle, G. Garcia, X. Rodet, “Tracking of partials for additive sound synthesis using hidden Markov models,” *Proceedings of 1993 ICASSP,* Minneapolis, MN, vol. 1, pp. 225-8, 1993.

[3] Malcolm Slaney, R. Lyon, D. Naar, “Auditory model inversion for sound separation,” *Proceedings of 1994 ICASSP*, Adelaide, Australia, vol. II, pp. 77-80, 1994.

[4] M. Covell, M. Withgott, “Spanning the gap between motion estimation and morphing,” *Proceedings of the 1994 IEEE ICASSP*, Adelaide, Australia, vol. V, pp. 213-216, 1994.

[5] E. Moulines, Y. Sagisak (editors), “Voice conversion: State of the art and perspective,” Special issue of *Speech Communications*, 16, pp.125-216, 1995.

[6] Malcolm Slaney, “Auditory Toolbox: A MATLAB toolbox for auditory modeling work,” Apple Technical Report #45, 1994 (available from ftp://ftp.apple.com/pub/malcolm/AuditoryToolbox.tar).

[7] B. Secrest, G. Doddington, “An integrated pitch tracking algorithm for speech systems,” *Proceedings of 1983 ICASSP*, Boston, MA, vol. 3, pp. 1352-1355, 1983.

[8] M. Yong, “A new interpolation technique for CELP coders”, *IEEE Trans. on Communications*, 42 (1), pp 34-38, 1994.