A Sines+Transients+Noise Audio Representation for Data Compression and Time/Pitch Scale Modifications

Scott N. Levine[*]
scottl@phc.net
http://webhost.phc.net/ph/scottl

Julius O. Smith III
jos@ccrma.stanford.edu
http://www-ccrma.stanford.edu/~jos

Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, CA 94305-8180, USA

Abstract

The purpose of this paper is to demonstrate a low bitrate audio coding algorithm that allows modifications in the compressed domain. The input audio is segregated into three different representations: sinusoids, transients, and noise. Each representation can be individually quantized, and then easily be time-scaled and/or pitch-shifted.

1 Introduction

The goal of this paper is to present a new representation for audio signals that allows for low bitrate coding while still allowing for high quality, compressed domain, time-scaling and pitch-shifting modifications.

In the current MPEG-4 specifications, there are compression algorithms that allow for time and pitch modifications, but only at very low bitrates (2-16 kbps) and relatively low bandwidth (at 8 kHz sampling rate) using sinusoidal modeling or CELP [1]. In this system, we strive for higher quality with higher bitrates (16-48 kbps), while allowing for high bandwidth (44.1 kHz sampling rate) and high quality time and pitch scale modifications.

To achieve the data compression rates and wideband modifications, we first segment the audio (in time and frequency) into three separate signals: a signal which models all sinusoidal content with a sum of time-varying sinusoids [2], a signal which models all attack transients present using transform coding, and a Bark-band noise signal [3] which

[*]Work supported by Bitbop Laboratories.

models all of the high frequency input signal not modeled by the transients. Each of these three signals can be individually quantized using psychoacoustic principles pertaining to each representation.

High-quality time-scale and pitch-scale modifications are now possible because the signal has been split into sines+transients+noise. The sines and noise are stretched/compressed with good results, and the transients can be time-translated while still maintaining their original temporal envelopes. Because of phase-matching algorithms, the system can switch between sines and transients seamlessly. In time-scaled (slowed) polyphonic music with percussion or drums, this results in slowed harmonic instruments and voice, with the drums still having *sharp* attacks.

In this paper, we will first describe the system from a high level point of view, showing how the input audio signal is segmented in time and frequency. We will then spend one section on each of the three signal models: sines, transients, and noise. In each of these sections, we will also describe their separate methods of parameter quantization. Afterwards, another section will be devoted to compressed-domain time-scale modifications.

2 System Overview

The purpose of this system is to be able to perform high-quality modifications, such as time-scale modification and pitch-shifting, on full-bandwidth audio while being able to maintain low bitrates. Before delving into our hybrid system, we will first mention other successful systems, along with their advantages and disadvantages.

2.1 Other Current Systems

The current state-of-the-art transform compression algorithms can achieve very high quality results (perceptually lossless at 64 kbits/sec/channel) but cannot achieve any time or pitch-scale modifications without independent post-processing modification algorithms [4].

The most recent phase vocoders can achieve high quality time and pitch-scale modifications, but currently imposes a data expansion rather than a data compression [5]. The parameters in this class of modeling method are $2\times$ oversampled FFT coefficients. Once expressed in magnitude and phase form, they can be time-scaled and pitch-scaled. Because of the oversampling, there are now twice as many FFT coefficients as original time coefficients (or corresponding MDCT coefficients). In addition, it has not been shown how well these time and pitch-scale modifications will perform if the FFT magnitude and phase coefficients are quantized to very low bitrates.

Sinusoidal+noise modeling has been developed for high quality time and pitch-scale modifications for fullband audio, but is currently limited to monophonic sources and necessitates hand tweaking of the analysis parameters by the user [6]. This user interaction would be unacceptable for a general purpose audio compression system. The system also has difficulties modeling sharp, percussive attacks. These attack signals are not efficiently represented as a sum of sinusoids, and the attack time is too sharp for the frame-based noise modeling used in the system. In addition, the system of [6] typically gives a data expansion rather than a data compression, since its goal was a transformable audio representation and not compression.

Sinusoidal modeling has also been used effectively for very low bitrate speech [7](2-16 kbps/channel) and audio coding [8]. In addition, these systems are able to achieve time and pitch-scale modifications. But these systems were designed for bandlimited (0-4 kHz) monophonic (*i.e.* single source), signals. If the bandwidth is increased, or a polyphonic input signal is used, the results are not of sufficiently high quality.

2.2 Time-Frequency Segmentation

It is evident that none of the individual algorithms described in the previous section can handle both high quality compression and modifications. While sinusoidal modeling works well for steady-state signals, it is not the best representation for attack transients or very high frequencies (above 5 kHz). For this reason, we segment the time-frequency plane into three general regions: sines, transients, and noise. In each time-frequency region, we use a different signal representation, and thus different quantization algorithms.

The first step in the segmentation is to analyze the signal with a transient detector. The details of the transient detector will be discussed in section 4.1. This step segments, in time, the input signal between attack transients, and non-transient signals. Below 5000 Hz, the non-transients are modeled by multiresolution sinusoidal modeling [2], which will be described in Section 3. Above 5000 Hz, the non-transients are modeled using bark-band noise envelopes, similar to those techniques developed in [3], which will be described in Section 5. The transient signals, between 0-16 kHz, are modeled using variants of current transform coding techniques [4], which will be described in section 4. This time-frequency segmentation can be seen in Figure 1. The overlap regions between the sinusoids and the transients are phase-matched, so no discontinuities can be heard. This will also be discussed in Section 3. Incremental improvements to the time-frequency segmentation that allow for lower bitrates and higher fidelity synthesis will be described later in the paper.

2.3 Reasons for the Different Models

Sinusoidal modeling is used only for the non-transient sections of the audio because attack transients cannot be efficiently modeled by a set of linearly ramped sinusoids. It is possible to model transients with a set of sinusoids, but such a system would need hundreds of sinusoidal parameters, consisting of amplitudes, frequencies, and phases. In this system, we attempt to model only the steady-state signals with sinusoids, thus allowing for an efficient representation.

Sinusoidal modeling is only used below 5000 Hz because for most music (but not all), there exists very few isolated, tonal sinusoidal elements above 5000 Hz. This is consistent with results found in the speech world [9]. It is very inefficient to model high frequency noise with sinusoids, and it is also very difficult to track stable, high frequency sinusoids reliably in loud high-frequency background noise. A residual noise model from 0 to 5 kHz is currently being investigated. If one wanted to listen to a pitch pipe or a single glockenspiel, then there certainly are stable high-frequency sinusoids present. But for most music that people listen to, this is not the case. We could have included an additional octave of sinusoids, but this would have added a considerable amount to the total bitrate, and would only benefit a very small percentage of sound examples.

Transform coding is used for modeling transients so that the attacks of instruments can be faithfully reproduced without using many bits. Because transform coding is a waveform coder, it can be used to give a high-precision representation over a short time duration (about 66 ms). Whenever an audio signal is to be time-scaled, we simply translate the transform-coded, short-time transients to the correct new places in time. More details will be provided in section 6.

When the signal is not being modeled as a transient, the system splits the bandwidth between 5-16 kHz into six bark-band regions. The high-frequency bandwidth is then modeled as a sum of white-noise bands modulated by separate amplitude envelopes. Again, for most signals, this model is sufficient. More details will be described in Section 5.

3 Multiresolution Sinusoidal Modeling

Sinusoidal modeling has proved to be a good representation for modeling monophonic music [6] and speech [7], but has only recently been used for wideband audio compression [10]. Certain problems arise when switching from monophonic speech/audio to polyphonic audio. A single fundamental frequency can no longer be assumed, and thus no pitch-synchronous analysis can be performed.

The problem to then be solved is choosing a proper analysis window length. One would like to have a long window to guarantee good frequency resolution at low frequencies. On the other hand, one would like to have as short a window as possible to reduce the pre-echo artifacts (see Figure 2). With a pitch-synchronous analysis, one could choose an adaptive window length that is two to three times longer than the current fundamental period.

Because multiple pitches and instruments may be present, we use a multiresolution sinusoidal modeling algorithm [2]. We split the signal into three different octaves, and use different window lengths in each octave. Each octave uses 50% overlap. See the table below for the parameters used in this system:

frequency range	window length	hop size
0-1250 Hz	46 ms	23 ms
1250-2500 Hz	23 ms	11.5 ms
2500-5000 Hz	11.5 ms	5.75 ms

In the time-frequency plane, this segmentation can be visualized as in Figure 3. Each rectangle shows the time-frequency region that sinusoidal $\{amp, freq, phase\}$ parameters can be updated. For example, in the lowest octave, sinusoidal parameters are only updated every 23 ms (the hop size in that octave). But in the highest octave, parameters are updated every 5.75 ms. Usually, there are about 5-20 sinusoids present in each octave at any one time.

3.1 Analysis Filterbank

In order to obtain these multiresolution sinusoidal parameters, we use a $2\times$ oversampled, octave-spaced, filterbank front-end. Each octave output of the filterbank is analyzed separately by a sinusoidal modeling algorithm with different window lengths. The reason we oversample the filterbank by a factor of 2 is to attenuate the aliasing energy between the octaves below audibility. If we used a critically sampled filterbank, such as a discrete-time wavelet transform, each octave output would have aliased energy from the neighboring octaves. This aliased energy would introduce errors in the sinusoidal modeling. For more details on the filterbank design, see [2][11].

3.2 Sinusoidal Parameters

In each l^{th} frame of analyzed audio, in a given octave, the system produces R^l sets of $p_r^l = \{A_r^l, \omega_r^l, \phi_r^l\}$ (amplitude,frequency,phase) parameters based on maximum likelihood techniques developed by Thomson [12] and previously used for sinusoidal modeling by Hamdy, *et al.*[10]. For a given frame, indexed by l, the synthesized sound is:

$$s(m + lS) = \sum_{r=1}^{R^l} A_r^l \cos[m\omega_r^l + \phi_r^l] \qquad m = 0, \ldots, S-1$$

where S is the length of the octave-dependent hop-size, shown in the previous table in Section 3. To be able to synthesize a signal without discontinuities at frame-boundaries, we interpolate the sinusoidal parameters between for each sample m from the observed parameters at $m = 0$ and $m = S$. The amplitudes are simply linearly interpolated from frame to frame. The phase and frequency interpolation will be later be discussed in Section 3.3.

In the next sub-sections, we will show how we first track sinusoids from frame to frame and then compute a psychoacoustic masking threshold for each sinusoid. Based on this information, we then decide which sinusoids to eliminate from the system and how to quantize the remaining sinusoids.

3.2.1 Sinusoidal Tracking

Between frame l and $(l-1)$, the sets of sinusoidal parameters are processed through a simplified peak continuation algorithm. If $|A_i^l - A_j^{l-1}| < \text{Amp}_{\text{thresh}}$ and $|\omega_i^l - \omega_j^{l-1}| < \text{Freq}_{\text{thresh}}$ then the parameter triads p_j^{l-1} and p_i^l are combined into a single sinusoidal trajectory. If a parameter triad p_i^l cannot be joined with another triad in adjacent frames, $\{p_j^{l-1}, j = 1, \ldots, R^{l-1}\}$ and $\{p_k^{l+1}, k = 1, \ldots, R^{l+1}\}$, then this parameter triad becomes a trajectory of length one. With these sets of sinusoidal trajectories, we now begin the process of reducing the bits necessary to represent the perceptually relevant information.

3.2.2 Masking

The first step in reducing the bitrate for the sinusoids is to estimate how high the sinusoidal peaks are above the masking threshold of the synthesized signal. In each octave of sinusoidal modeling, we compute a separate psychoacoustic masking threshold using a window length equal to the analysis window length for that octave. The model used in this system was based on the MPEG psychoacoustic model II. For details on computing the psychoacoustic masking thresholds, see [13].

In each octave, we compute the masking threshold on an approximate third-bark band scale, or the *threshold calculation partition domain* in [13]. From 0 to 5 kHz, there are about 50 non-uniform divisions in frequency that the thresholds are computed within. The i^{th} sinusoidal parameter triad in frame l, p_i^l, then obtains another field, the masking threshold, m_i^l. The masking threshold m_i^l is the difference between the energy of the

i^{th} sinusoid (correctly scaled to match to domain of the psychoacoustic model) and the masking threshold in its third-bark band [in dB].

Not all of the found sinusoids estimated in the initial analysis [12] are stable sinusoids. We only desire to encode sinusoids that are stable sinusoids, and *not* model noisy signals with several closely-spaced sinusoids. We use the psychoacoustic model, which has a tonality measure based on prediction of FFT magnitudes and phases, to double-check the results of the initial sinusoidal estimations.

As can be seen in Figure 4, shorter trajectories have (on average) a lower signal-to-masking threshold. This means that many shorter trajectories will be masked by longer, more stable trajectories. A possible reason for this trend is that the shorter trajectories are attempting to model noise, while the longer trajectories are actually modeling sinusoids. In [13], a stable sinusoid will have a masking threshold at -18 dB in its third-bark band, while a noisy signal will have only a -6 dB masking threshold. Therefore, tonal signals will have a larger distance to the masking threshold than noisy signals. A simple graphical example of the masking thresholds of stable sinusoids can be seen in Figure 5. The signal-to-masking thresholds and trajectory lengths will be important factors in determining which trajectories to eliminate, and how much to quantize the remaining parameters.

3.2.3 Sinusoidal Trajectory Elimination

Not all sinusoidal trajectories found as described Section 3.2.1 will be encoded. A trajectory that is masked, meaning its energy was below the masking threshold of its third-bark band, will not be encoded. By eliminating the masked trajectories, the sinusoidal bitrate is decreased approximately 30% in typical audio input signals. In informal listening tests, no audible difference was heard after eliminating these trajectories.

3.2.4 Sinusoidal Trajectory Quantization

Once the masked trajectories have been eliminated, the remaining ones are to be quantized. In this section, we will concentrate only on amplitude and frequency quantization. We will discuss phase quantization in Section 3.3. Initially, the amplitudes are quantized with 5 bits, in increments of 1.5 dB, giving a dynamic range of 96 dB. The frequencies are quantized to an approximate just noticeable difference frequency scale (JNDF) using 9 bits.

Because of the slowly varying amplitude and frequency trajectories, we can efficiently quantize the temporal first-order differences across the trajectory. We then Huffman encode these differences. In addition, we can also exploit the inter-trajectory redundancy by Huffman encoding the difference among neighboring trajectories' initial amplitudes and frequencies.

In the previous Section 3.2.3, we eliminated the trajectories that were masked. But, we kept all the other trajectories, even those whose energies were just barely higher than their bark-band masking thresholds. In principle, these lower-energy trajectories should not be allocated as many bits as the more perceptually important trajectories; *i.e.* those having energies much higher than their masking thresholds. A solution that was found to be bitrate efficient and which still sounded good was to downsample these lower-energy sinusoidal trajectories by a factor of two. That is, update the sinusoidal parameters at half of the original rate. On the decoder end, the missing parameters are linearly interpolated.

This effectively reduces the bitrate of these trajectories by 50%, and the total sinusoidal bitrate by an additional 15%.

After testing several kinds of music, we were able to quantize three octaves of multiresolution sinusoids from 0 to 5 kHz at 12-16 kbps. These numbers depend on how much of the signal from 0 to 5 kHz is encoded using transient modeling, as discussed in Section 4. More transients per unit time will lower the sinusoidal bitrate, but the transient modeling bitrate will increase.

3.3 Switched Phase Reconstruction

In sinusoidal modeling, transmitting phase information is usually only necessary for one of two reasons. The first reason for keeping phases is to create a residual error signal between the original and the synthesized signal. This is needed at the encoder, but not at the decoder. Thus, we need not transmit these phases for this purpose.

The second reason for transmitting phase information is for modeling attack transients well. During sharp attacks, the phases of sinusoids can be perceptually important. But in this system, no sharp attacks will be modeled by sinusoids; they will be modeled by a transform coder. Thus, we will not need phase information for this purpose.

A simple example of switching between sines and transients is depicted in Figure 6. At time=40 ms, the sinusoids are cross-faded out and the transients are cross-faded in. Near the end of the transients region at time=90 ms, the sinusoids are cross-faded back in. The trick is to phase-match the sinusoids during the cross-fade in/out times while only transmitting the phase information for the frames at the boundaries of the transient region.

To accomplish this goal, we use cubic polynomial phase interpolation [7] at the boundaries between the sinusoidal and transient regions. We perform phaseless reconstruction sinusoidal synthesis at all other times. Because we only send phase at transient boundaries which happen at most several times a second, the contribution of phase information to the total bitrate is extremely small.

First we will quickly describe the cubic-polynomial phase reconstruction, and then show the differences between it and *phaseless* phase reconstruction. Afterwards, we show how we can switch seamlessly between the two.

3.3.1 Cubic-polynomial Phase Reconstruction

Recall from Section 3.2 that during the l^{th} frame, we estimate the R sets triad of parameters $p_r^l = \{A_r^l, \omega_r^l, \phi_r^l\}$. These parameters must be interpolated from frame to frame to eliminate any discontinuities at the frame boundaries. The amplitude is simply linearly interpolated from frame to frame.

The phase interpolation is more complicated. We first create an instantaneous phase parameter, θ_r^l, which is a function of surrounding frequencies, $\{\omega_r^l, \omega_r^{l-1}\}$ and surrounding phases, $\{\phi_r^l, \phi_r^{l-1}\}$. Because the instantaneous phase is derived from four parameters, we need a cubic polynomial interpolation function. For details of this interpolation function, see [7].

Finally, the reconstruction for frame l becomes

$$s(m + lS) = \sum_{r=1}^{R^l} A_r^l(m) cos[\theta_r^l(m)] \qquad m = 0, \ldots, S-1 \tag{1}$$

3.3.2 Phaseless Reconstruction

Phaseless reconstruction is called *phaseless* because it does not need explicit phase information transmitted in order to synthesize the signal. The resulting signal will not be phase aligned with the original signal, but it will not have any discontinuities at frame boundaries.

Instead of deriving the instantaneous phase from surrounding phases and frequencies, phaseless reconstruction derives the instantaneous phase as the integral of the instantaneous frequency [14]. The instantaneous frequency, $\omega_r^l(m)$, is obtained by linear interpolation:

$$\omega_r^l(m) = \omega_r^{l-l} + \frac{(\omega_r^l - \omega_r^{l-1})}{S} m \qquad m = 0, \ldots, S-1$$

Therefore, the instantaneous phase for the r^{th} trajectory in the l^{th} frame is:

$$\theta_r^l(m) = \theta_r^{l-1} + \omega_r^l(m) \tag{2}$$

The term θ_r^{l-1} refers to the instantaneous phase at the last sample of the previous frame. The signal is then synthesized using Equation (1), but using $\theta_r^l(m)$ from Equation (2) instead of the result of a cubic polynomial interpolation function. For the first frame of phaseless reconstruction, the initial instantaneous phase is randomly picked from $[-\pi, \pi)$.

3.3.3 Phase Switching

In this section, we will show how to switch between phase interpolations algorithms seamlessly. As a simple example, let the first transient begin at frame l. All frames $(0, 1, \ldots, l-2)$ will be synthesized using the phaseless reconstruction algorithm outlined in section 3.3.2. During frame $l-1$, we must seamlessly interpolate between the estimated parameters $\{\omega^{l-1}\}$ and $\{\omega^l, \phi^l\}$, using cubic interpolation of Section 3.3.1. Because there were no estimated phases in frame $l-1$, we let $\phi^{l-1} = \theta^{l-1}(S)$, at the last sample of the instantaneous phase of that frame. In frame l, cubic interpolation is performed between $\{\omega^l, \phi^l\}$ and $\{\omega^{l+1}, \phi^{l+1}\}$. But, $\omega^l = \omega^{l+1}$, and ϕ^{l+1} can be derived from $\{\omega^l, \phi^l, S\}$, as was shown in [15]. Therefore, owe need only the phase parameters, ϕ_r^l, for r=$(1, 2, \ldots, R)$ for each transient onset detected.

To graphically describe this scenario, see Figure 7. Each frame is 1024 samples long, and the frames $l-1$ and l are shown. That is, the transient begins at t=1024 samples, or the beginning of frame l. A similar algorithm is performed at the end of the transient region to ensure that the ramped-on sinusoids will be phase matched to the transient being ramped-off.

4 Transform-Coded Transients

Because sinusoidal modeling does not model transients efficiently, we represent transients with a short-time transform coder instead. The length of the transform coded section can be varied, but in the current system it is 66 milliseconds. This assumes that most transients last less than this amount of time. After the initial attack, most signals become somewhat periodic and can be well modeled using sinusoids. First, we will discuss our transient detector, which decides when to switch between sinusoidal modeling and transform coding. Then, we describe the basic transform coder used in the system. In the following subsection, we then discuss methods to further reduce the number of bits needed to encode the transients.

4.1 Transient Detection

The design of the transient detector is very important to the overall performance of the system. The transient detector should only flag a transient during attacks that will not be well modeled using sinusoids. If too many parts of the signal are modeled by transients, then the bitrate will get too high (transform coding has a higher bitrate than multiresolution sinusoidal modeling). In addition, time-scale modification, which will be discussed in Section 6, will not sound as good. If too few transients are tagged, then some attacks will sound dull and have pre-echo problems due to the limitations of sinusoidal modeling.

Two methods are combined in the system's transient detection algorithm. The first method is a conventional frame-based energy measure. It looks for a rising edge in the energy envelope of the original signal over short frames. The second method involves the residual signal, which is the difference between the original signal and the multiresolution sinusoidal modeled signal (with cubic polynomial interpolated phase). The second method measures the ratio of short-time energies of the residual and the original signal. If the residual energy is very small relative to the original energy, then that portion of the signal is most likely tonal and is modeled well by sinusoidal modeling. On the other hand, if the ratio is high, it concludes the energy in the original signal was not modeled well by the sinusoids, and an attack transient might be present.

The final transient detector uses both methods; *i.e.*, it looks at both rising edges in the short-time energies of the original signal and also the ratio of residual to original short-time energies. The system declares a region to be a transient region when both of these methods agree that a transient is occurring.

4.2 A Simplified Transform Coder

The transform coder used in this system is a simplified version of the MPEG-AAC (Advanced Audio Coding) system [4]. It has been simplified to reduce the system's overall complexity. The emphasis in this paper is not to improve the current state of the art in transform coding, but rather to use it as a tool to encode transient signals. In the future, we plan to further optimize this simplified coder to reduce the bitrate of the transients and to introduce a shared bit reservoir pool between the sines, the transients, and the noise modeling algorithms. In this system, the transient is defined as the residual over the detected transient duration after subtracting out the off-ramping and on-ramping sinusoids. A graphical example of a transient can be seen in the second plot in Figure 6.

First, the transient is windowed into a series of short (256 point) segments, using a raised sine window. At 44.1 kHz, the current system encodes each transient with 24 short overlapping 256-point windows, for a total length of 66 ms. There is no window length switching as in AAC since the system has already identified the transient as such. Each segment is run through an MDCT [16] to convert from the time domain to a critically sampled frequency domain. A psychoacoustic model [13] is performed in parallel on the short segments in order to create the masking thresholds necessary for perceptually lossless subband quantization.

The MDCT coefficients are then quantized using scale factors and a global gain as in the AAC system. However, there are no iterated rate-distortion loops. We perform a single binary search to quantize each scale factor band of MDCT coefficients to have a mean-squared error just less than the psychoacoustic threshold allows. The resulting quantization noise should now be completely masked. We then use a simplified version of the AAC noiseless coding to Huffman encode the MDCT coefficients, along with the differentially encoded scalefactors.

4.3 Time-Frequency Pruning

In principle, a time duration of a transient is frequency dependent. We do not have a rigorous definition of transient time duration, other than to generally say it is the time during which a signal is not somewhat periodic. At lower frequencies, this time duration is usually longer than it is at higher frequencies.

We mentioned earlier in this section that transients are encoded in this system for 66 milliseconds. But because a single transient does not have the same length in time at all frequencies, we do *not* need to encode all 66 milliseconds of the transient in every frequency range. In particular, we construct a tighter time-frequency range of transform coding around the attack of the transient. For example, as shown in Figure 8, we transform-encode the signal from 0 to 5 kHz for a total of 66 milliseconds, but we only transform encode the signal from 5-16 kHz for a total of 29 milliseconds. The remaining time-frequency region above 5 kHz not encoded by transform coding is represented by bark-band noise modeling, which will be discussed in the following section.

This pruning of the time-frequency plane greatly reduces the number of bits necessary to encode transients. As will be shown, bark-band noise modeling is a much lower bitrate representation than transform coding. After informal listening tests on many different kinds of music, no differences were detected between using transform coding over all frequency ranges for the full duration of the transient versus just a tighter fit region of the time-frequency plane.

As shown in Figure 8, there are only two frequency regions that have different time-widths of transform-encoded transients. This could easily be generalized to more bands, octave-spaced bands, or even a bark-band scale. By using transform coding only around the time-frequency regions that need it, the bitrates can be lowered further. The remaining regions of time-frequency are modeled using multiresolution sinusoidal modeling and bark-band modeling, both of which have lower bitrate requirements.

5 Noise Modeling

In order to reduce the total system bitrate, we stated previously that we will not model any energy above 5 kHz as tonal (with sinusoids). Above 5 kHz, the signal will either be modeled as a transform-coded transient or as bark-band filtered noise, depending on the state of the transient detector. Bark-band noise modeling bandpass filters the original signal from 5-16 kHz into six bark-spaced bands [17]. This is similar to [3], which modeled the sinusoidal modeling residual from 0-22 kHz with bark-spaced noise modeling. If a signal is assumed to be noisy, the ear is sensitive only to the total amount of short-time energy in a bark band, and not the specific distribution of energy within the bark band. Therefore, every 128 samples (3 milliseconds @ 44.1 kHz), an RMS-level energy envelope measurement is taken from each of the six bark bandpass filters. To synthesize the noise, white noise is filtered through the same bark-spaced filters and then amplitude modulated using the individual energy envelopes.

5.1 Bark-Band Quantization

After some informal listening tests, quantizing each bark band energy sample to 1.5 dB seemed the largest possible quantization range possible without hearing artifacts. An example of such an envelope can be seen in the top plot of Figure 9. If we Huffman encode this information, the total data rate would be in the neighborhood of 10 kbps. However, it does not seem perceptually important to sample the energy envelope every 128 samples (345 frames/sec). It seems more important perceptually to preserve the rising and falling edges of the energy envelopes. Small deviations in the bark-band energy envelope could be smoothed without audible consequence. The goal is to transmit only a small subset of the energy envelope points, and linearly interpolate the missing points at the decoder.

5.2 Line Segment Approximation

We call the samples of the energy envelopes that are transmitted, *breakpoints*, since they are points at which the straight lines "break" to change slope. We implemented a greedy algorithm [18] that iteratively decides where a new breakpoint in the envelope would best minimize the error between the original and approximated envelope. The number of breakpoints is set to 20% of the length of the envelope itself. Using fewer breakpoints would lower the bitrate, but would introduce audible artifacts in the synthesized noise. An example of an energy envelope reduced by line segment approximation can be seen in the lower plot of Figure 9.

There are now two sets of data to quantize: the timing and amplitude of the breakpoints. We Huffman encode the timing differences, along with the amplitude differences. In addition, there is another Huffman table to encode the first amplitude of each envelope. The initial timing of each envelope can be inferred from timing information of the preceding transform-coded transient signal. If there is a possibility of losing some data in transmission, the time-differential methods will obviously need to be changed. Overall, quantization of the six bands for most signals results in a bitrate of approximately 3 kbps.

5.3 High Frequency Transform Coding

There are certain transients, which we will call microtransients, that are not broadband or loud enough to be triggered in by the algorithm stated in section 4.1. For example, small drum taps like a closing hi-hat sometimes appears as a microtransients. If these microtransients are modeled by bark-band noise modeling, the result will not sound crisp, but rather distorted and spread. The solution is to use transform coding centered around these attacks, but only from 5 to 16 kHz. Because these high frequency transients are very sudden and short, only three transform coding frames of 128 samples each are necessary.

Before and after the sudden transient, bark-band noise modeling is used. See Figure 10 for an example and further discussion.

6 Modifications

Time-scale and pitch-scale modifications are relatively simple to perform on the compressed data because the input audio has been segregated into three separate parametric representations, all of which are well behaved under time/frequency compression/expansion. In this section we will concentrate on time-scale modification. For more details on pitch shifting capabilities, see [19]. Because the transients have been separated from the rest of the signal, they can be treated differently than the sines or the noise. In order to time-scale the audio, the sines and noise components will be *stretched* in time, while transients will be *translated* in time. In the next three subsections, we will discuss in detail how each of the three models are time-scale modified. See Figures 11 and 12 for graphical examples and further explanation.

6.1 Sinusoidal Time-Scale Modification

Since the earliest sinusoidal modeling systems for speech and audio, it has been shown how to time-scale the representation. The synthesis equation (1) for the l^{th} frame is slightly altered by scaling the hop size S by the time stretch factor α:

$$s(m + lS\alpha) = \sum_{r=1}^{R^l} A_r^l(m) cos[\theta_r^l(m)] \qquad m = 0, \ldots, \alpha(S-1) \tag{3}$$

When $\alpha = 1$, no time-stretching is applied. When $\alpha > 1$, the playback speed is slowed but the pitch remains the same. Similarly, when $\alpha < 1$, the playback speed is faster with the same pitch. The amplitude parameters are still linearly interpolated, but over a different frame length. In addition, the instantaneous phase parameter is now interpolated using the phase switching algorithm described in Section 3.3.3 over a different frame length. Even though the cross-fade regions between the sinusoids and the transients now appear at different regions in time, phase-locking is still guaranteed when the sinusoids overlap with the transient signal.

6.2 Transient Time-scale Modification

To keep the sharp attacks inherent in the transients, the transform-coded transients are translated in time rather than stretched in time. Therefore, the MDCT frames are simply moved to their new place in time and played at the original playback speed. Because these signals are so short in time (66 milliseconds), the attack sounds natural and blends well with the time-stretched sinusoids and noise. Thus, attacks are still sharp, no matter how much the music has been slowed down.

6.3 Noise Time-scale Modification

Because the noise has been parametrized by envelopes, it is very simple to time-scale the noise. The breakpoints in the bark band envelopes are stretched according to the time factor, α. Using linear interpolation between the breakpoints, new stretched envelopes are formed. Six channels of bark bandpassed noise are then modulated by these new stretched envelopes and summed to form the final stretched noise. Similarly, efficient inverse FFT methods could be used [3].

7 Acknowledgment

The first author would like to thank Tony Verma for his sinusoidal modeling software core, and for many hours of discussions about parametric coders and compression.

8 Conclusions

We described a system that allows both aggressive data compression and high-quality compressed-domain modifications. By parametrizing sines, transients, and noise separately, we get the coding gain of perceptually based quantization schemes and the ability to perform compressed-domain processing. In addition, we can preserve the sharp attacks of transients, even with large time-scale modification factors. To hear demonstrations of the data compression and modifications described in this paper, see [20].

References

[1] B. Edler, "Current status of the MPEG-4 audio verification model development", *Audio Engineering Society Convention*, 1996.

[2] S. Levine, T. Verma, and J.O. Smith, "Multiresolution sinusoidal modeling for wide-band audio with modifications", *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Seattle*, 1998.

[3] M. Goodwin, "Residual modeling in music analysis-synthesis", *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Atlanta*, pp. 1005–1008, 1996.

[4] M. Bosi, K. Brandenburg, S. Quackenbush, L.Fielder, K. Akagiri, H.Fuchs, M.Dietz, J.Herre, G.Davidson, and Y.Oikawa, "ISO-IEC MPEG-2 Advanced Audio Coding", *Audio Engineering Society Convention*, 1996.

[5] J. Laroche and M. Dolson, "Phase-vocoder: About this phasiness business", *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, NY*, 1997.

[6] Xavier Serra and Julius O. Smith III, "Spectral modeling synthesis: A sound analysis / synthesis system based upon a deterministic plus stochastic decomposition", *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, winter 1990.

[7] T. Quatieri R. McAulay, "Speech analysis/synthesis based on a sinusoidal representation", *IEEE Transactions on Acoustics, Speech, Signal Processing*, August 1986.

[8] B.Edler, H.Purnhagen, and C. Ferekidis, "ASAC - analysis/synthesis codec for very low bit rates", *Audio Engineering Society Convention*, , no. 4179, 1996.

[9] E. Moulines J. Laroche, Y. Styliano, "HNM: A simple, efficient harmonic + noise model for speech", *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, NY*, 1993.

[10] A. Hamdy, K. Ali and Tewfik H., "Low bit rate high quality audio coding with combined harmonic and wavelet representations", *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Atlanta*, 1996.

[11] U. Zolzer N.J. Fliege, "Multi-complementary filter bank", *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Minneapolis*, 1993.

[12] D. J. Thomson, "Spectrum estimation and harmonic analysis", *Proceedings of the IEEE*, vol. 70, no. 9, pp. 1055–1096, September 1982.

[13] ISE/IEC JTC 1/SC 29/WG 11, "ISO/IEC 11172-3: Information technology - coding of moving pictures and associated audio for digital storage media at up to about 1.5 mbit/s - part 3: Audio", 1993.

[14] X. Serra, *A System for Sound Analysis/Transformation/Synthsis based on a Determistic plus Stochastic Decomposition*, PhD thesis, Stanford University, 1989.

[15] T. Quatieri R. McAulay, "Speech transformations based on a sinusoidal representation", *IEEE Transactions on Acoustics, Speech, Signal Processing*, vol. 34, December 1986.

[16] A. Bradley J. Princen, A. Johnson, "Subband/transform coding using filter bank designs based on time domain aliasing cancellation", pp. 2161–2164, 1987.

[17] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models*, Springer-Verlag, 1990.

[18] J. Beauchamp A. Horner, N. Cheung, "Genetic algorithm optimization of additive synthsis envelope breakpoints and group synthesis parameters", *Proceedings of the 1995 International Computer Music Conference, Banff*, pp. 215–222, 1995.

[19] S. Levine, *Parametric Audio Representations for Data Compression and Compressed-Domain Processing*, PhD thesis, Stanford University, expected December 1998, working title, available online at http://www-ccrma.stanford.edu/˜scottl.

[20] S. Levine, "Sound demonstrations for the 1998 San Francisco AES conference", http://webhost.phc.net/ph/scottl/aes98.html.

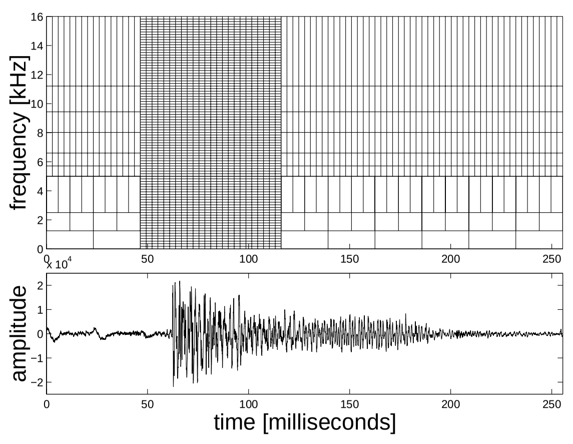

Figure 1: The lower plot shows 250 milliseconds of a drum attack in a piece of pop music. The upper plot shows the time-frequency segmentation of this signal. During the attack portion of the signal, transform coding is used over all frequencies and for about 66 milliseconds. During the non-transient regions, multiresolution sinusoidal modeling is used below 5 kHz and bark-band noise modeling is used from 5-16 kHz.

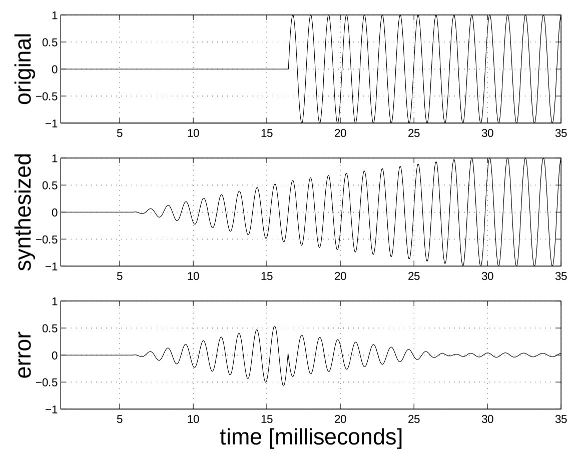

Figure 2: This figure shows the pre-echo error resulting from sinusoidal modeling. Because the sinusoidal amplitude is linearly ramped from frame to frame, the synthesized onset time is limited by the length of the analysis window.

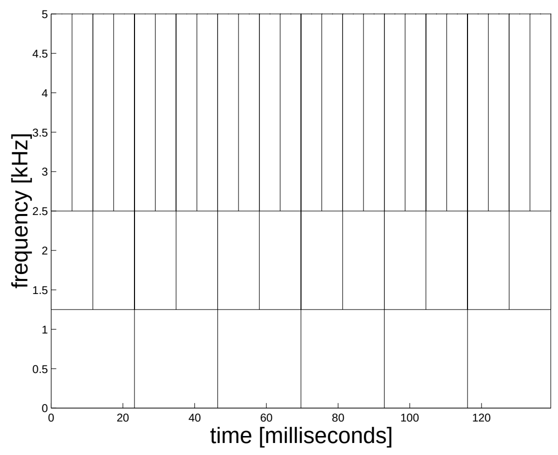

Figure 3: The time-frequency segmentation of multiresolution sinusoidal modeling. Each rectangle shows the update rate of sinusoidal parameters at different frequencies. In the top octave, parameters are updated every 5.75 ms, while at the lowest octave the update rate is only 23 ms. Usually, there are 5-20 sets of sinusoidal parameters present in any one rectangle.

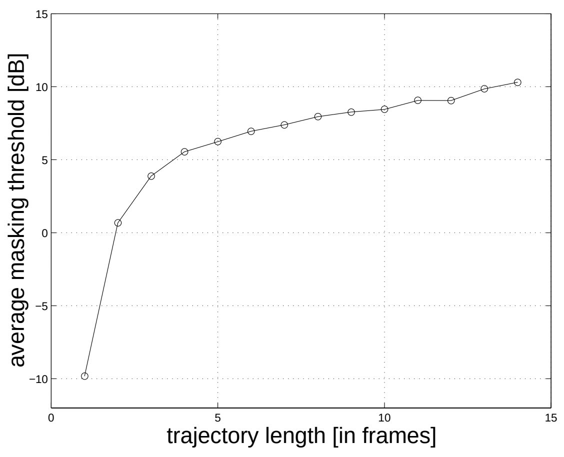

Figure 4: This figure shows how longer sinusoidal trajectories have a higher average maximum signal-to-masking threshold than shorter trajectories. Or, the longer a trajectory lasts, the higher its signal-to-masking threshold. This data was derived from the top octave of 8 seconds of pop music, where each frame length is approximately 6 milliseconds in length.

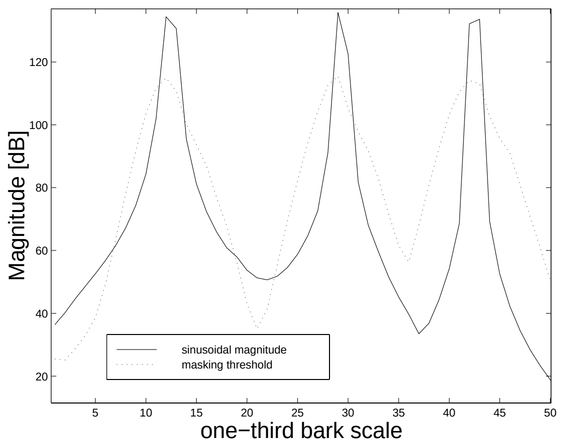

Figure 5: The original spectral energy versus the masking threshold of three pure sinusoids at frequencies 500, 1500, 3200 Hz. Notice that the masking threshold is approximately 18 dB below their respective sinusoidal peaks.

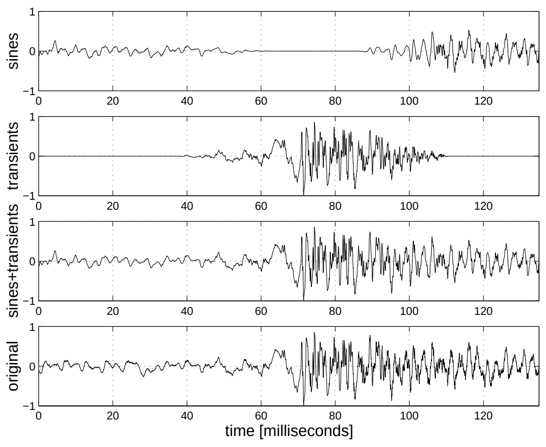

Figure 6: This figure shows how sines and transients are combined. The top plot shows the multiresolution sinusoidal modeling component of the original signal. The sinusoids are faded-out during the transient region. The second plot shows a transform-coded transient. The third plot shows the sum of the sines plus the transient. For comparison, the bottom plot is the original signal. The original signal has a sung vowel through the entire section, with a snare drum hit occurring at t=60 ms. Notice that between 0 and 30 ms, the sines are *not* phase-matched with the original signal, but they do become phase-matched between 30-60 ms, when the transient signal is cross-faded in.

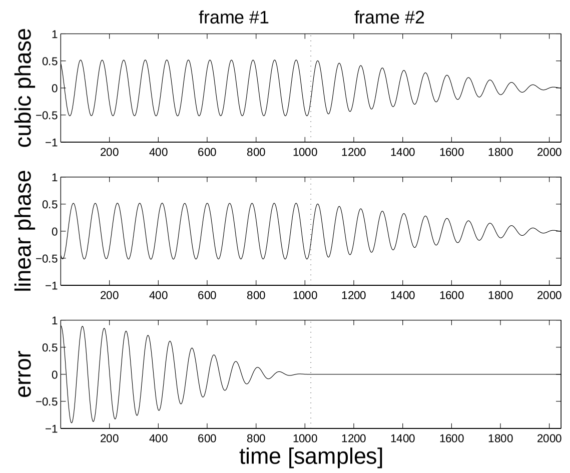

Figure 7: The top signal shows a signal synthesized with phase parameters, and the phase is interpolated between frame boundaries using a cubic polynomial interpolation function [7]. The middle signal is synthesized using no explicit phase information except at the transient boundary, which is at time = 1024 samples. The initial phase is random, and is otherwise interpolated using the switched method of Section 3.3. Over the shown time scale is two frames, each 1024 samples long. Frame #1 shows the middle signal slowly becoming phase locked to the signal above. By the beginning of frame #2, the top two signals are phase locked. The bottom plot is the difference between the top two signals.

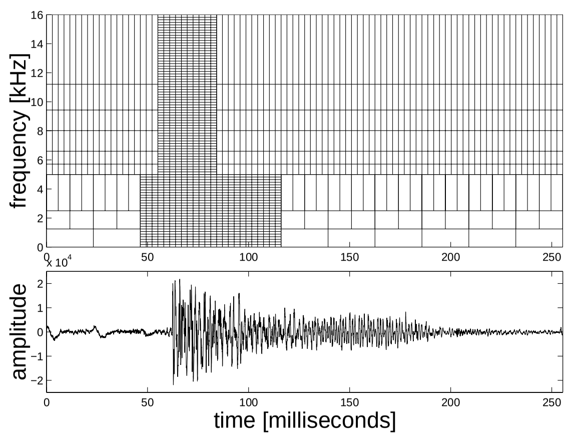

Figure 8: This figure shows how to prune the time-frequency plane for transform coding of a transient. Like Figure 1, the lower plot shows 250 milliseconds of a drum attack in a piece of pop music. The upper plot shows the time-frequency segmentation of this signal. During the attack portion of the signal, transform coding is used for about 66 milliseconds between 0 to 5 kHz, but for only 29 milliseconds between 5-16 kHz. By reducing the time-frequency region of transform coding, the bitrate is reduced as well. During the non-transient regions, multiresolution sinusoidal modeling is used below 5 kHz and bark-band noise modeling is used from 5-16 kHz.

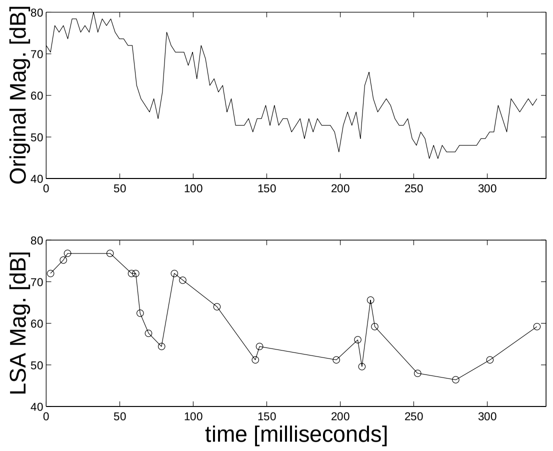

Figure 9: The top plot shows a bark band (8000-9200 Hz) RMS-level energy envelope for about 300 milliseconds. The bottom plot shows the line segment approximated RMS-level energy envelope. The circled points are the transmitted envelope points, and the remaining points are linearly interpolated using the transmitted points.

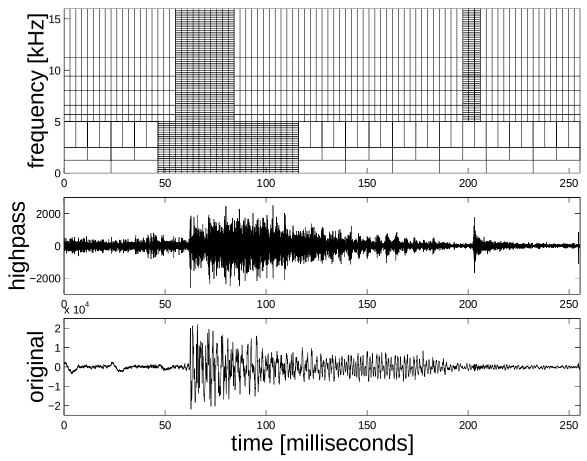

Figure 10: This figure shows how transform coding can preserve sharp, high-frequency attacks. The bottom plot shows the original signal, as shown in Figures 1 and 8. The plot directly above it shows the same signal highpass-filtered, with a cutoff at 5 kHz. Notice that at 200 milliseconds, a transient is observed in the highpassed signal, but not in the lower wideband signal. Accordingly, we segment the time-frequency plane around t=200 milliseconds and between 5 and 16 kHz, and encode that region using transform coding techniques. This preserves the high-frequency transient onset. Bark-band noise modeling is used for surrounding times.

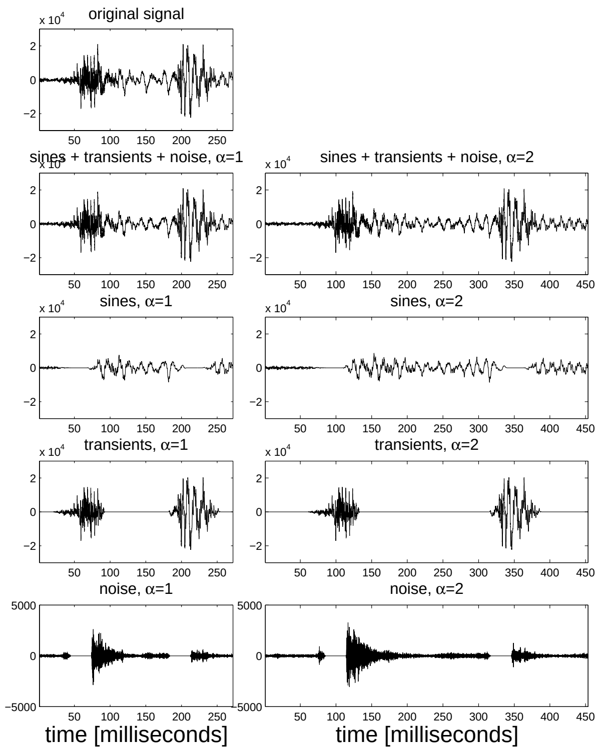

Figure 11: This set of plots shows how time-scale modification is performed. The original signal, shown at top left, shows two transients: first a hi-hat cymbal hit, and then a bass drum hit. There are also vocals present throughout the sample. The left-side plots show the full synthesized signal at top, and then the sines, transients, and noise independently. They were all synthesized with no time-scale modification, at α=1. The right-side plots show the same synthesized signals, but time-scale modified with α=2, or twice as slow with the same pitch. Notice how the sines and noise are stretched, but the transients are translated. Also, the vertical amplitude scale on the bottom noise plots are amplified 15 dB for better viewing.

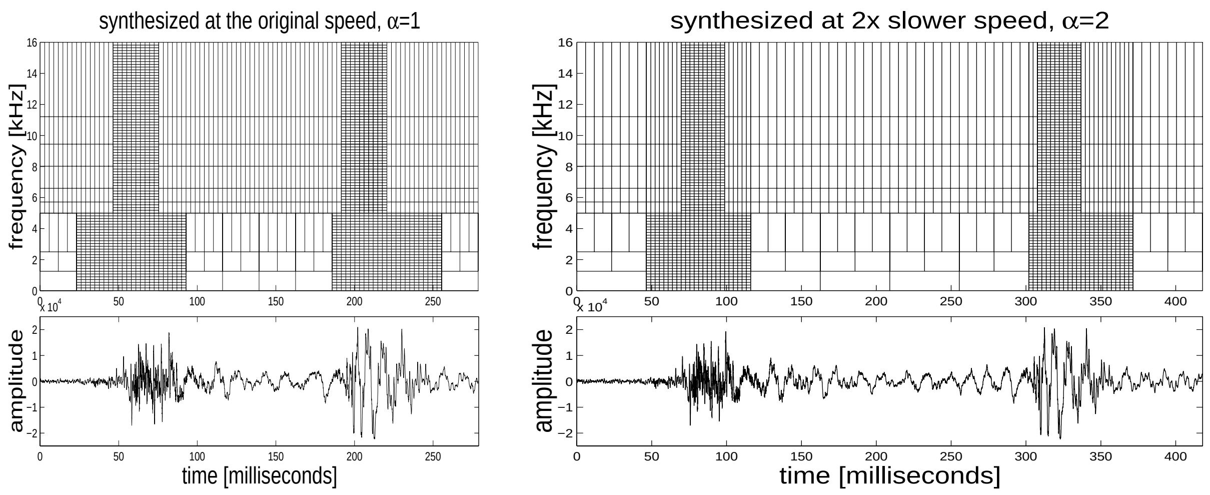

Figure 12: These figures show the time-frequency plane segmentations of Figure 11. The figure on the left is synthesized with no time-scaling, α=1. The figure on the right is slowed down by a factor of two, *i.e.* α=2. Notice how the grid spacing of the transform coded regions are not stretched, but rather shifted in time. However, the time-frequency regions of the multiresolution sinusoids and the bark-band noise have been stretched in time in the right plot. Each of the rectangles in those regions are now twice as wide in time. The exception to this rule is the bark-band noise modeled within the time span of the low-frequency transform-coded samples. These bark-band noise parameters are shifted (not stretched), such that they remain synchronized with the rest of the transient. There are no sinusoids during a transform-coded segment.