# Real-Time Signal Estimation From Modified Short-Time Fourier Transform Magnitude Spectra

Xinglei Zhu, Gerald T. Beauregard, *Member, IEEE*, and Lonce L. Wyse

***Abstract*—An algorithm for estimating signals from short-time magnitude spectra is introduced offering a significant improvement in quality and efficiency over current methods. The key issue is how to invert a sequence of overlapping magnitude spectra (a "spectrogram") containing no phase information to generate a real-valued signal free of audible artifacts. Also important is that the algorithm performs in real-time, both structurally and computationally. In the context of spectrogram inversion, structurally real-time means that the audio signal at any given point in time only depends on transform frames at local or prior points in time. Computationally, real-time means that the algorithm is efficient enough to run in less time than the reconstructed audio takes to play on the available hardware. The spectrogram inversion algorithm is parameterized to allow tradeoffs between computational demands and the quality of the signal reconstruction. The algorithm is applied to audio time-scale and pitch modification and compared to classical algorithms for these tasks on a variety of signal types including both monophonic and polyphonic audio signals such as speech and music.**

***Index Terms*—Magnitude-only reconstruction, real-time systems, signal estimation, spectrogram inversion, time-scale modification (TSM).**

## I. Introduction

MAGNITUDE spectra and their time sequences in the form of spectrograms are widely used for time-frequency representations of audio signals such as speech and music. The magnitude spectrum of a discrete-time signal $x(n)$ is typically obtained from the short-time Fourier transform (STFT), which is defined as

$$X(mS,\varpi)=\sum_{n=-\infty}^{\infty}x(n)w(n-mS)e^{-j\varpi n} \qquad (1)$$

where $w$ is the analysis window, $S$ is the analysis step size, and $m$ is the index of the frames of the STFT. The complex-valued STFT is a complete and reversible time-frequency representation. The time-domain signal is uniquely determined by its STFT representation and *vice versa*. Using the STFT, the short-time Fourier transform magnitude (STFTM) spectrum of $x(n)$ is

$$|X(mS,\varpi)|=\left|\sum_{n=-\infty}^{\infty}x(n)w(n-mS)e^{-j\varpi n}\right|. \qquad (2)$$

The terms in (2) are the same as in (1). By combining the real and imaginary part of each spectral frequency component into a single number, the magnitude spectrum provides a valuable visualization tool with a strong correspondence to how audio signals are perceived in terms of frequency content. For example, formants, sibilants, and stop consonants are clearly visible in the STFTM of speech signals, and a skilled viewer can sometimes even identify words.

In many signal processing applications, the STFT phase information is lost or not applicable, so that it is desirable to construct the time-domain signal from a STFTM or a modified STFTM (MSTFTM). An example of the usefulness of time-domain signal reconstruction from STFTM is in time–frequency blind source separation. When the individual sources are separated into their STFT power spectra sequences, the STFT phases of each source are unknown. The original STFT phase of the mixture is not a good choice for the individual sources, since typically the original sources are all still audibly present. The STFTM of each individual source can be considered a modified STFTM. If it were possible to "invent" phases to convert the modified STFTM to a time-domain signal without introducing distracting artifacts, then we could avoid using the mixed-source phase information. A second example is the time-scale modification of audio signals. The STFT phase of the extended signal is different from the original and needs to be determined in some way. In this paper, we will present algorithms for time-scale modification and pitch modification by constructing signals from the modified STFTM.

Generally, the STFTM is not reversible, since the time-domain signal cannot be uniquely determined from its STFTM only. For example, obviously $x(n)$ and $-x(n)$ have the same STFTM. For an arbitrary signal, even when a given magnitude spectrum is calculated from a real signal, there is generally no way to exactly convert the magnitude spectrum back into the original time-domain real signal. Furthermore, in some applications, we have only a modified (or arbitrary) spectrogram, where the series of overlapping magnitude spectra might not be a valid representation of any real-valued audio signal at all. In such cases, we would like to find a real-valued signal whose spectrogram is as close as possible to the modified or target spectrogram.

Manuscript received June 6, 2006; revised February 26, 2007. This work was supported by the Institute for Infocomm Research (IIR), A-STAR, Singapore. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Rudolf Rabenstein.

X. Zhu is with the Media Understanding Department, Institute for Infocomm Research, Singapore 119613 (e-mail: xzhu@i2r.a-star.edu.sg).

G. T. Beauregard is with muvee Technologies, Singapore 188974 (e-mail: g.beauregard@ieee.org).

L. L. Wyse is with the Faculty of Arts and Social Sciences, National University of Singapore, Singapore 119077 (e-mail: lonce.wyse@nus.edu.sg).

Digital Object Identifier 10.1109/TASL.2007.899236

STFT magnitude spectrum inversion has also been referred to as "phase retrieval" and "signal reconstruction from STFTs" in literature spanning the last 20 years or so. The relationship between the STFT magnitude and phase is studied in [4], and the role of the specific choice of analysis window in [9] and [11]. Under certain conditions, such as the minimum and maximum phase assumption discussed in detail in [4], [7] and a set of specific conditions developed in [5], a time-domain signal can be uniquely determined by its STFT magnitude or STFT phase. The exact recovery requires part of the time-domain signal such as half of the samples [3], or some phase information such as a spatial sample [8] or one bit of phase information [6]. In practice, the additional information may not be available. Quite a few iterative algorithms have also been explored [1]–[3], [5]–[8], [10], [12], [23] to reconstruct the time-domain signal by minimizing the mean square error (MSE) between the given STFTM and the STFTM of the reconstruction. The MSE function is defined as

$$D_M\left[x(n),x'(n)\right]=\sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\times\int_{\varpi=-\pi}^{\pi}\left[|X(mS,\varpi)|-|X'(mS,\varpi)|\right]^2 d\varpi \tag{3}$$

where $|X(mS,\varpi)|$ is the STFTM of original signal $x(n)$ and $|X'(mS,\varpi)|$ is the STFTM of the estimation $x'(n)$. Convergence of the iterative process is provable for some of the algorithms [1], [13].

Griffin and Lim [1] proposed an algorithm (hereafter referred to as the G&L algorithm) iteratively applying a forward and inverse Fourier transform to converge toward a time-domain signal with the desired spectrum. The Fourier transform is used to extract the phase from an estimation of the time-domain signal. This phase information is then combined with the target magnitude spectrum and used to compute an inverse Fourier transform to generate the next estimate of the time-domain signal. In this paper, we use the term "magnitude spectrum constrained transform" (or "M-constrained transform") to refer to the calculation of the IFFT using the derived phase and the target magnitude spectrum, and a "transform iteration" or "iteration" to refer to the forward and (M-constrained) inverse transform pair. In the G&L algorithm, each transform iteration is concurrently applied to all frames.

The central contribution of this paper is a real-time method for constructing a high-quality time-domain signal from an overlapping series of MSTFTMs, or spectrogram. This paper is organized as follows. In Section II, requirements for a practical real-time spectrogram inversion method and the details of a family of algorithms that address these requirements are presented. In Section III, experimental results using the real-time magnitude spectrogram inversion algorithm is shown. In Section IV, we apply the magnitude spectrogram inversion method to time-scale modification, and in Section V, to pitch modification. In Section VI, we draw some conclusions.

## II. Real-Time STFTM Spectrogram Inversion

In many applications, the spectrogram inversion algorithm is required to execute in real-time. For example, for real-time communications in the presence of noise, spectral subtraction is one technique used to improve the speech quality and reduce noise. After the noise reduction process, the spectra must be converted back into a time-domain signal. Indeed, any streaming audio signal processing where an output stream is generated while concurrently reading the input stream requires algorithms capable of real-time performance since the magnitude spectrum sequence of the whole signal is clearly not available at the time the processed output needs to be delivered.

The G&L algorithm uses the following function to update the estimate in each iteration:

$$x^{i+1}(n)=\frac{\sum_{m=-\infty}^{\infty}w(n-mS)\frac{1}{2\pi}\int_{\varpi=-\pi}^{\pi}\hat{X}^i(mS,\varpi)e^{-j\varpi n}d\varpi}{\sum_{m=-\infty}^{\infty}w^2(n-mS)} \tag{4}$$

where $\hat{X}^i(mS,\varpi)$ is the STFT of $x^i(n)$ with the magnitude constraint

$$\hat{X}^i(mS,\varpi)=X^i(mS,\varpi)\frac{|X(mS,\varpi)|}{|X^i(mS,\varpi)|}. \tag{5}$$

Execution speed is an issue with the G&L algorithm because it requires the computation of a large number of Fourier transforms for each frame in order to achieve high-quality reconstruction. More fundamentally, the classic G&L algorithm is inappropriate for use in real-time applications because it requires the magnitude spectra from all frames, past and future, to guarantee that the spectral error decreases monotonically.

A practical real-time magnitude spectrogram inversion algorithm is expected to fulfill the following requirements.

1) Structural requirement: The algorithm should reconstruct frames using only temporally local and possibly past information rather than remote or future time information.
2) Computational load requirement: The amount of computation required to reconstruct the audio signal should be low enough to be used in real-time applications across as wide a variety of platform types as possible.
3) Quality requirement: The reconstruction should be accurate if a target signal is known, and free of distracting artifacts. This is a perceptual requirement even when the magnitude spectra have been modified or not created from a time-domain signal that can be used as a "ground truth" from which to derive an objective error measure.
4) Flexibility requirement: The inversion algorithm should reconstruct better quality signals with more computational resources. This property is not necessary, but important for usability. With this property, the same magnitude spectrogram inversion algorithm could be used in situations with a variety of real-time versus quality demands.

In this section, we introduce several real-time spectrogram inversion algorithms which, taken together, fulfill the above four requirements. They are the real-time iterative spectrogram inversion (RTISI) algorithm and the RTISI with look-ahead (RTISI-LA) algorithm with several implementation strategies.

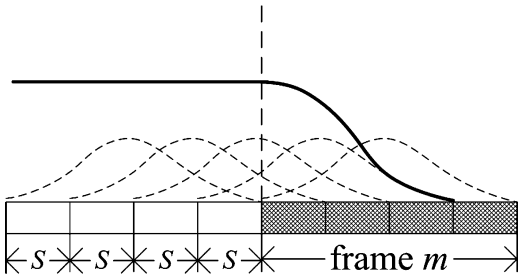



Fig. 1. Illustration of the partially reconstructed frames of signal $y(n)$. Before frame $m$ is estimated, there exists an overlap-added result of the frames $m-1$, $m-2$, $m-3$ in the range of the frame $m$ window. The solid line shows the time-domain contour of the previously constructed signal, and the dotted lines show the overlap-added windows. The shaded part is the range of frame $m$. $S$ is the synthesis step size between two adjacent frames.

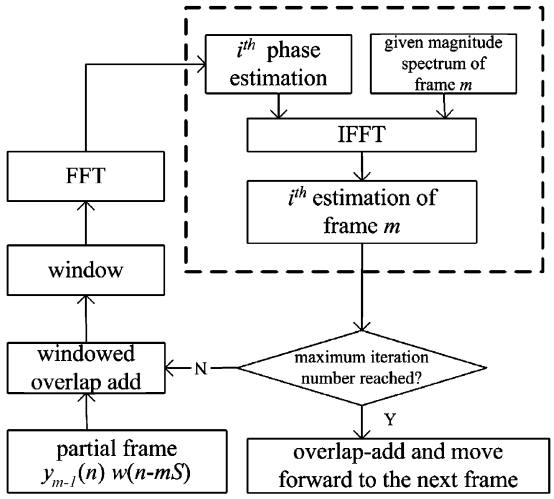



Fig. 2. Frame-by-frame iterative phase estimation process. The dashed square is the magnitude spectrum constrained transform ("M-constrained transform"), based on the phase estimate and the given target magnitude spectrum.

We will show the details of different approaches and discuss the advantages and disadvantages of each of them.

### A. RTISI Algorithm

To fulfill the structural requirement, the signal should be constructed according to the time sequential order (frame-by- frame). In the classic G&L algorithm, all the frames are updated concurrently. To fulfill the computation load requirement, the number of transform iterations must be kept to a minimum. In the classic G&L algorithm, the slow convergence over transform iterations makes it inapplicable to real-time applications. We address both of these real-time issues by employing a G&L iteration strategy on the current frame alone, using information from the audio frames already reconstructed that overlap with the current frame to construct an initial current frame phase estimate. Using a better initial estimate of phases for each frame, the computation process is sped up significantly. We call this algorithm RTISI [14].

Suppose we already have reconstructed the first $m-1$ frames of the synthesis signal, which we denote as $y_{m-1}(n)$. Let us consider the problem of generating frame $m$. The signal frames already generated at this point are illustrated in Fig. 1. All of the algorithms use the scaled Hamming window

$$w(n)=\begin{cases}\frac{2\sqrt{S}}{\sqrt{(4a^2+2b^2)L}}\left(a+b\cos\left(2\pi\frac{n}{L}\right)\right), & \text{if } 1\le n\le L\\ 0, & \text{otherwise}\end{cases} \tag{6}$$

where $S$ is the step size between adjacent frames, $L$ is the window length, $a=0.54$ and $b=-0.46$. In our signal reconstruction system, we typically use $L=4S$, so that the sum of the squares of the overlapping scaled Hamming windows is always 1.

As shown in Fig. 1, before we estimate frame $m$ (for $m>1$), the overlap interval is partially filled by the former frames. Unless otherwise specified, we use a fixed 75% synthesis window overlap (i.e., $S=L/4$) so that the $m$th partial frame comes from the overlap-added results of the estimation of the frames $m-1$, $m-2$, $m-3$ of $y(n)$, while the fourth quarter of frame $m$ is all zero. To distinguish the partially filled frame from the fully constructed frame $m$, we notate the former partially filled frame as $y_{m-1}(n)w(n-mS)$, where $w(n)$ is the window function. Now, we estimate frame $m$ and overlap-add it with the partial frame $y_{m-1}(n)w(n-mS)$ to generate $y_m(n)$.

To generate an initial estimate for the phases of frame $m$, we compute the phase of the partially reconstructed signal using an analysis window positioned at the partially constructed frame $m$. This ensures that even without iterating, the initial phase estimate for frame $m$ will provide good phase continuity with the partially-reconstructed signal. The Fourier transform of this partial frame is calculated with the scaled Hamming window (6). The resulting phase information is combined with the target magnitude spectrum for the M-constrained transform step. The inverse Fourier transform of this new frequency-domain signal produces a new estimate of frame $m$. If the target number of iterations has not been reached, frame $m$ is added to the partial frame $y_{m-1}(n)w(n-mS)$, the window is applied, and then the Fourier transform of the windowed summation is calculated to get a new estimate of the phase. We use the update (4) from the G&L algorithm in the RTISI iterative process, but instead of updating the estimate of the whole signal $x(n)$, in each step, we update the estimate of the current frame only. The frame-by-frame iterative process of the RTISI algorithm is shown in Fig. 2.

For the first frame of the signal, we do not have a partial frame to be added to our estimate. Any initial phase can be used as the initial phase estimation for the first frame. In our experiments, we simply use a zero initial phase estimate with the target magnitude spectrum and follow the above iterative process to generate the first frame of $y(n)$.

When the iterative process ends, frame $m$ is combined with the partial frame $y_{m-1}(n)w(n-mS)$, and the process continues with successive frames until we reach the end of the spectrogram.

RTISI fulfills the structural and the computational load requirements for real-time applications. However, the flexibility requirement is not yet met. The performance in terms of spectral error reaches an asymptote during the iterative process and does not increase with further iterations. The reason is that when constructing frame $m$, only the information from the current and the previous frames are utilized. The RTISI-LA algorithm was developed to address the flexibility requirement and further improve the quality.

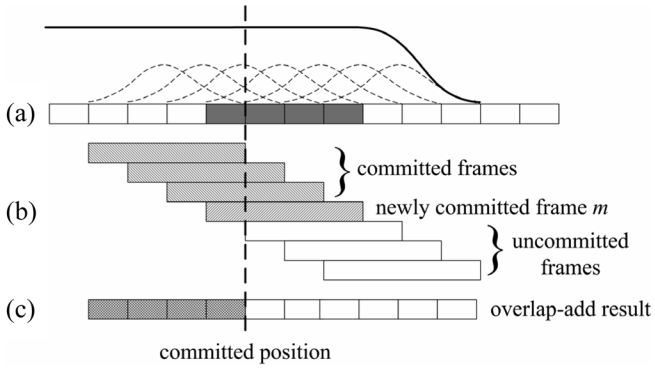



Fig. 3. RTISI with look-ahead after committing frame $m$. (a) The constructed signal with an indication of the contour (solid line) and overlapped windows (dashed line). (b) The frames currently in the frame buffer. There are three kinds of frames in the buffer: frames committed in the previous process; the newly committed frame $m$; and the uncommitted frames. (c) The overlap-add result of the frames in the buffer. The shaded part is the fully committed signal ready for output.

### B. Look-Ahead Strategy

When generating frame $m$ with RTISI, the initial phases are estimated from the partial frame, and then the transform iterations are applied on that frame only. The phases of frame $m$ are then fixed or "committed," and $S$ new samples are ready for output. In this process, the future frames are not considered when choosing the phases for frame $m$. By contrast, in RTISI with look-ahead, $k$ future frames influence the reconstruction of frame $m$. After frame $m$ is generated, it is kept uncommitted until frame $m+k$ is generated.

In Fig. 3, the commitment of frame $m$ is shown with $k = 3$. The shaded portion of Fig. 3(a) shows the position of frame $m$. We use a frame buffer [shown in Fig. 3(b)] to hold earlier committed frames that overlap frame $m$, frame $m$ itself, and $k$ uncommitted "future" frames. We typically use 75% overlap between the adjacent frames in our system so the number of committed frames that overlap with frame $m$ is 3. We will show the performance comparison of different frame overlaps in Section III-A. In Fig. 3, the number of look-ahead frames $k$ is 3, but it can be any nonnegative integer. If the initial phase estimation technique outlined in the next section is not used, RTISI-LA with $k = 0$ (no look-ahead frames) is identical to RTISI.

When frame $m$ is initially generated, we leave it uncommitted in the frame buffer and move forward until we reach frame $m+k$. Then we use the partial frame to estimate the initial phase for frame $m+k$ and apply the transform iteration to each uncommitted frame in the frame buffer (frames $m$ to $m+k$) using the corresponding magnitude spectrum. We overlap-add all the frames in the frame buffer and obtain an overlap-add result, as shown in Fig. 3(c). At this point, frame $m$ is still marked as uncommitted. We read in all the uncommitted frames (frame $m$ to $m+k$) using the scaled Hamming window from the overlap-add result. This is repeated over all uncommitted frames until the desired number of iterations is reached. At this point, frame $m$ is marked as committed and ready for output since the estimate of its phases will not undergo any further modification.

Since the frames are overlap-added to create the output signal, the reconstructed output signal in a given time range is not finalized until all frames in that time range have been committed. For the case where $L = 4S$, four overlapping frames contribute to each block of $S$ output samples. As shown in Fig. 3(c), when frame $m$ is marked as committed, the previous three frames would have already been committed, so a block of $S$ new output samples is ready for output. Then, we remove frame $m-k$, which is committed and ready for output, from the frame buffer and read in the partial frame $m+k+1$ and repeat the above iterative process for frames $m+1$ to $m+k+1$.

### C. Initial Phase Estimation

Spectrogram inversion can be seen as a mapping from a series of overlapping magnitude spectra to a time-domain signal in such a way that minimizes a magnitude spectrum distance function. There may be no zero-error solution, and even when one exists, it may not be unique without *a priori* knowledge or the specification of additional constraints. In many applications, a signal spectrogram that closely matches the target would also be useful. For example, in segregating speech from noise, a "good sounding" reconstruction is more important than a sample-accurate reconstruction of the original. "Good sounding" in this context would mean intelligible speech free of audible artifacts that might reduce the performance of a human or machine speech recognizer. Phase continuity across the frames of a spectrogram is important for avoiding distracting artifacts that can be present even when magnitude spectral error is low.

Iterative strategies typically seek a local error minimum instead of a global minimum. The G&L algorithm [1] uses such a strategy. In G&L the MSE monotonically decreases as the iteration number increases. However, since there is generally no *a priori* knowledge of the original signal, G&L usually starts from an all-zero or random phase. The result of such an uninformed initial phase estimate is that G&L needs a large number of iterations to approach a spectral error minimum and generate a high-quality reconstruction. The RTISI algorithm starts from a much better initial phase estimation based on the partially reconstructed frame, and this greatly reduces the number of iterations required to achieve a given error criteria. The better initial phase estimation means that the iterative process starts from a position much closer to a local error minimum which can then be reached more quickly.

### D. Asymmetric Analysis Window

In the algorithms presented so far, the initial phase estimate for frame $m$ is obtained by overlap-adding the previous frames using the best available phase estimates for those frames to generate an estimated time-domain signal, applying a Hamming window to that time-domain signal, applying a Fourier transform to it, and extracting the phases. This approach guarantees that the initial phase estimate for frame $m$ will have good phase continuity with previous frames.

When we are estimating the initial phases for frame $m$, we already have phase estimates for previous frames, but not for future frames. Our best estimate of the output signal in the time range of frame $m$ therefore consists of the overlap-added result of frames $m-1$, $m-2$, and $m-3$. The envelope of that signal will naturally be asymmetric, as it does not include the contribution of future frames $m+1$, $m+2$, and $m+3$ which

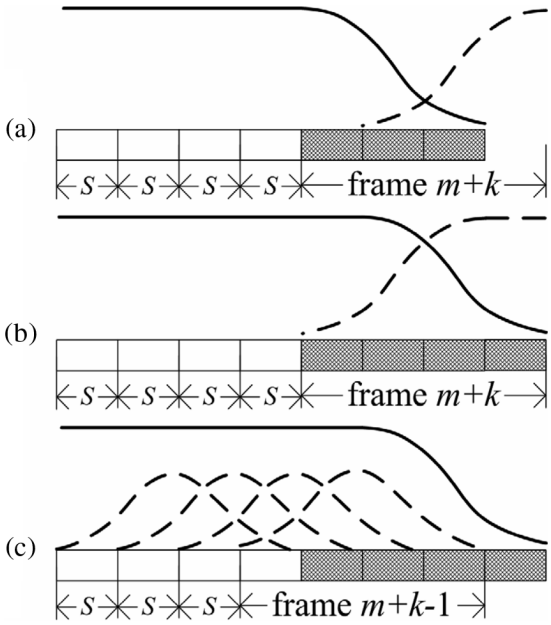



Fig. 4. Analysis windows, shown in dashed line. (a) The asymmetric window for the latest frame $(m+k)$ which is to be initialized. (b) Another asymmetric window for the latest frame $(m+k)$ which has been initialized. (c) The modified Hamming window for other uncommitted frames.

also partially cover the time range of frame $m$. This asymmetry is illustrated in Fig. 1.

Even if the phase estimates for the previous frames $m-1$, $m-2$, and $m-3$ were perfect (i.e., exactly matched the phases in the original signal), the asymmetry of the envelope of those overlap-added frames means that our initial phase estimate for frame $m$ would still be inexact. We can partially compensate for that asymmetry, and hence get a better initial phase estimate, by using an asymmetric analysis window. For the initial phase estimate, a good choice of asymmetric analysis window is a time-reversed version of the aforementioned envelope, as shown with a dashed line in Fig. 4(a).

Applying the dashed window in Fig. 4(a) produces a symmetric signal contour which is used to generate the initial phase estimate. The total computation load is essentially unaffected because there are no new iterative steps introduced. After the first iteration for generating frame $m+k$, we overlap it with the signal constructed so far and get a new partial envelope, shown under the solid line in Fig. 4(b). In the later iterations, we use the reverse of the new envelope (including the new frame), as shown in dashed line in Fig. 4(b), as the analysis window of frame $m+k$. For other frames in the frame buffer, we use the modified Hamming window (6) as the analysis window, which is shown in Fig. 4(c) in the dashed line. We will discuss the performance improvement using such an asymmetric analysis window in Section III.

### *E. Performance of the RTISI-LA Algorithm*

The computational load is determined mainly by the total number of forward and inverse Fourier transforms required per frame. For a given frame, the number of transforms is $2p(1+k)$, or the number of transforms per iteration times $p$, the number of iterations per frame, times the number of frames (the current frame plus $k$, the number of look-ahead frames). The original RTISI algorithm as described in [14] achieves good signal reconstruction quality by ensuring good phase continuity between each new frame and previous frames. RTISI-LA achieves better quality by also taking into account information in a small number of future frames when computing phases for the current frame.

The cost for the future information is additional delay and a higher computational load. The delay comes in two forms: computational and structural. If the number of iterations per frame, $p$ is held constant, then looking ahead one more frame results in the need to compute $p$ more iterations for every $S$ output samples. Of course, the impact of the computational delay might be negligible on some platforms, but on others, such as on mobile devices, there are strict computational limitations. Furthermore, looking ahead also results in additional structural delay. Frame $m$ is kept uncommitted until frame $m+k$ is initialized. Thus, looking ahead one more frame means that frame $m$ has to be kept uncommitted with an additional delay of $S$ samples, or $S/sr$ seconds, where $sr$ is the sampling rate in hertz. If the sample rate is 44 100 Hz and the frame length $L$ is 1024 samples, then each additional look-ahead frame costs (1024/4)/44 100 Hz $=$ 5.8 ms additional delay. In applications with critical real-time requirements, the delay has to be taken into consideration, and an appropriate number of look-ahead frames should be chosen to achieve the desired balance between performance and delay. Fortunately, most of the improvement in quality results from the first few look-ahead frames in RTISI-LA.

## III. Evaluation

The signal-to-error ratio (SER) function [10] is used in evaluation

$$\text{SER} = 10\log\frac{\sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\int_{\varpi=-\pi}^{\pi}|X(mS,\varpi)|^2 d\varpi}{\sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\int_{\varpi=-\pi}^{\pi}[|X(mS,\varpi)|-|X'(mS,\varpi)|]^2 d\varpi} \tag{7}$$

Since the magnitude spectrogram $|X(mS,\varpi)|$ is given, minimizing the mean square error (3) is equivalent to maximizing the SER function (7).

Our test sample set consists of 32 audio segments, half of them are speech signals (monophonic) and the other half are music signals (polyphonic). Each segment is approximately 15 s long and the sampling rate is 44 100 Hz, except for some downloaded samples whose sampling rate is 22 050 Hz. We use a window length of 23.2 ms in all evaluations, which corresponds to 512 samples at 22 050 Hz.

In Sections III-A–III-C, we evaluate variants of the RTISI-LA algorithm. We then choose a specific variant to compare RTISI-LA with other algorithms in Section III-D.

### *A. Evaluation of Initial Phase Estimation Methods and Frame Overlap Rates in RTISI-LA*

In this section, we compare the SER results using RTISI-LA using various settings on a variety of speech and music signals.

TABLE I
RTISI-LA PERFORMANCE

| Initial phase estimation | step size | Speech SER (dB) | Music SER (dB) |
|---|---|---|---|
| Normal windowing | L/8 | 21.68 | 16.61 |
| | L/4 | 22.72 | 18.22 |
| With asymmetric analysis window | L/8 | 24.96 | 19.72 |
| | L/4 | 24.65 | 19.60 |

In particular, we present results for the two different methods for estimating the initial phases, and for varying amounts of overlap between successive frames.

As discussed in Section III-C, for the initial phase estimation of each new frame, we use either the partial frame's phase or the phase obtained after applying the asymmetric analysis window. Since we are particularly concerned with the performance at low computation loads, we fix the number of look-ahead frames at 3 and the number of iterations per frame at 2 so that the total number of iterations for any individual frame is 8. The SER results are shown in Table I. Using an asymmetric analysis window for RTISI-LA increases performance by approximately 2 dB for both speech signals and music signals, without increasing the number of iterations. Further experiments would be required to determine whether and when this improvement in measured SER results in any subjective audible improvement in quality. In the following, when we refer to RTISI-LA, we mean RTISI-LA with the asymmetric analysis window.

We also compared the performance of RTISI-LA using varying amounts of overlap between adjacent frames, while keeping the number of look-ahead frames fixed at 3. With step sizes $S = L/8$ and $S = L/4$, the corresponding amount of overlap between adjacent frames is 87.5% and 75%, respectively. As shown in Table I, without the asymmetric analysis window, the performance using $S = L/8$ is worse than that of using rate $S = L/4$. This is not surprising, given that the number of look-ahead frames is the same in both cases. For the case where $S = L/4$, using three look-ahead frames means that we use information from all future frames that overlap with the current frame $m$, and the envelope of the sum of the frames in the frame buffer is flat in the time range of frame $m$, as shown in Fig. 3(a). By contrast, if $S = L/8$, there are in fact seven future frames that overlap with the current frame. If the number of look-ahead frames is only 3, this means that when frame $m$ is committed, the envelope of the sum of all frames in the frame buffer is not flat.

When the asymmetric analysis window is applied, using $S = L/8$ gives a slight performance improvement compared to $S = L/4$. Note that using a smaller step size means more frames for a given duration, and an increase in the amount of computation. For example, using an $S = L/8$ requires twice the computation as $S = L/4$, but gives only a slight improvement in SER.

A way to further improve SER performance when using a smaller step size is to look-ahead by more frames to cover all the frames overlapped with the current frame. However, this is computationally expensive. Suppose for example, that we choose the number of look-ahead frames such that all future frames that overlap with the current frame are taken into consideration. When the step size is reduced by half from $S = L/4$ to $S = L/8$, the number of look-ahead frames increases by a factor of 2 as does the total number of frames for a given signal duration. As a result, the computation load increases by a factor of 4.

TABLE II
RTISI-LA WITH DIFFERENT LOOK-AHEAD NUMBERS

| Look-ahead frames | Speech SER (dB) | Music SER (dB) |
|---|---|---|
| 0 | 20.06 | 15.53 |
| 1 | 22.99 | 18.13 |
| 2 | 24.21 | 19.11 |
| 3 | 24.65 | 19.60 |
| 4 | 24.72 | 19.89 |
| 5 | 24.92 | 20.04 |
| 6 | 24.97 | 20.23 |
| 7 | 25.01 | 20.35 |
| 8 | 25.05 | 20.50 |
| 9 | 25.12 | 20.60 |
| 10 | **25.13** | **20.64** |

TABLE III
RTISI-LA PERFORMANCE WITH DIFFERENT NUMBERS OF TRANSFORM ITERATIONS

| Look-ahead frames | Total number of transform iterations per step | Total iterations | Speech SER (dB) | Music SER (dB) |
|---|---|---|---|---|
| 3 | 2 | 8 | 24.65 | 19.60 |
| 3 | 20 | 80 | 28.40 | 23.62 |
| 10 | 20 | 220 | **31.00** | **26.23** |

### B. Comparing Different Amounts of Look-Ahead

In this section, we evaluate the effect of the number of look-ahead frames. The step size is fixed at $S = L/4$, and the number of iterations per step is set to 2. We then use various look-ahead frame numbers from 1 to 10. The SER results are shown in Table II.

From Table II, we can see that the performance increases with the number of look-ahead frames when the number of iterations per step is held constant. However, the computational load also increases with the number of look-ahead frames. Looking ahead more than three frames means that we use information from future frames that do not overlap at all with the current frame. The additional indirect information does not provide much SER improvement.

### C. Flexibility of RTISI-LA

RTISI-LA has several parameters that can be used to choose the appropriate balance between computation time and reconstruction quality for an application. The primary factors determining the computation time are the number of iterations and the number of look-ahead frames. The performance of RTISI-LA with different computational loads is shown in Table III.

### D. Comparing RTISI-LA, G&L , and RTISI

In this section, we compare the SER performance of RTISI-LA to that of the G&L and RTISI algorithms. For the

TABLE IV
SER EVALUATION OF RTISI-LA3, G&L, AND RTISI

| Algorithm | Number of transform iterations per frame | Speech SER (dB) | Music SER (dB) |
|---|---|---|---|
| G&L | 8 | 11.77 | 10.61 |
| | 80 | 19.99 | 17.08 |
| RTISI | 8 | 19.12 | 14.72 |
| | 80 | 19.32 | 14.90 |
| RTISI-LA | 8 | 24.65 | 19.60 |
| | 80 | **28.40** | **23.62** |

purposes of this comparison, we use RTISI-LA with three look-ahead frames with the aforementioned asymmetric analysis window. We evaluate these algorithms at eight transform iterations per frame (which easily runs in real-time on a 3-GHz PC), and at 80 iterations per frame representing the near highest-quality reconstruction performance for these algorithms. The SER of the reconstruction result is shown in Table IV.

For both cases, the SER performance of RTISI-LA is substantially better than either G&L or RTISI, whether for speech or music. Even when using only eight transform iterations per frame, RTISI-LA gives better SER than G&L or RTISI using 80 transform iterations per frame. In other words, the measured SER is better despite a reduction in computation load by a factor of 10.

Table IV also shows that the original RTISI algorithm performs better than G&L at eight iterations. However, as the number of iterations is increased, RTISI's SER quickly reaches a plateau, whereas G&L's SER continues to increase, so G&L eventually outperforms RTISI given a large enough number of iterations. With RTISI-LA, by contrast, the SER generally continues to improve as the number of iterations is increased. Even when the number of iterations is extremely large (e.g., 1000, as shown in Fig. 5), RTISI-LA still gives better SER than G&L. A possible explanation for this is that the strictly monotonic error reduction with iterations in G&L coupled with poor initial phase estimates might make better error minima unavailable to the algorithm. The better initial phase estimates permit RTISI-LA to achieve better reconstruction quality than G&L and greatly reduces the computation time; however, the SER does not increase in a strictly monotonic fashion. Fig. 5 shows the SER of the construction result of an orchestral music signal using G&L, RTISI, and RTISI-LA with look-ahead number $k = 3$.

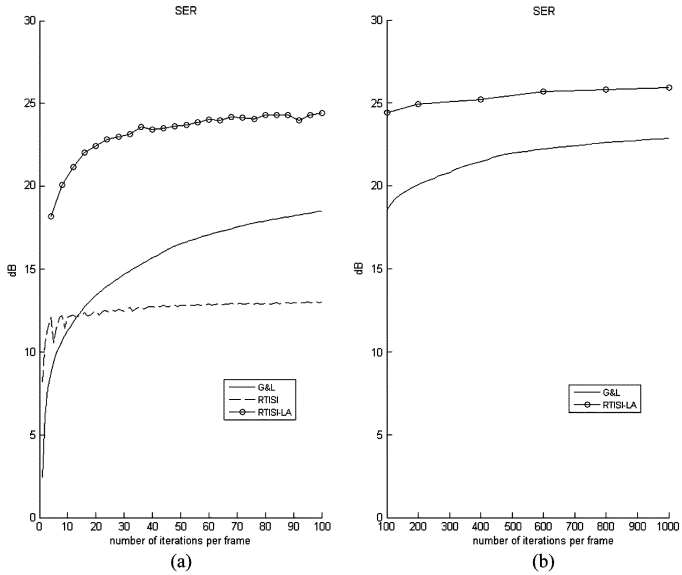



Fig. 5. SER of the construction result of an orchestral music signal using G&L, RTISI, and RTISI-LA (with look-ahead number of 3), respectively. (a) SER of G&L, RTISI, and RTISI-LA for up to 100 iterations. (b) SER of G&L and RTISI-LA for 100 to 1000 iterations.

## IV. TIME-SCALE MODIFICATION (TSM)

TSM of signals has long been a subject of interest in the audio and speech processing domains. A key challenge in TSM is to change the audio rate, while preserving other characteristics such as pitch and timbre. Several approaches have been reported for modifying the time-scale of an audio signal. Such approaches include the G&L method [1], the synchronized overlap and add algorithm (SOLA) [15], and its various modifications such as WSOLA [16], PAOLA [17], PSOLA [18], the phase vocoder algorithm [19], and some methods for building specific models of speech processes such as the vocal tract model [20] and a probabilistic inference model [21]. To achieve time-scale modification for polyphonic signals, the phase vocoder methods [19], [22] are common choices.

For monophonic signals, a time-domain process of overlap and add (OLA) is often used as follows. The original signal is first windowed at length $L$ with an analysis step size $S_a$. Then for each windowed frame a reconstruction signal of the same length is generated and all the regenerated signal segments are overlap-added with appropriate weights for the synthesis step size $S_s$. However, a simple time-domain OLA method does not generally work well because the signal segments being overlap-added may not be consistent with the audio modification rate $S_s/S_a$. Different OLA variants modify the basic process to improve quality. For example, in the SOLA method, the reconstructed frame varies within a small range to maximize a correlation function to improve the consistency of the scaled result. Pitch synchronous methods can also help when the signal is monophonic.

Because traditional magnitude-spectra-only reconstruction methods in [1] and [2] require a large number of iterations of the analysis-synthesis cycle to achieve good performance, the time-domain methods have been considered more economical in computation and have been applied in many commercial implementations. The time-domain TSM methods work well as long as the modification factor is close to 1 and when the signal source is monophonic. The RTISI-LA method works well for both monophonic samples and polyphonic signals.

The method of applying RTISI-LA to time-scale modification follows the traditional frequency-domain method. For a modification rate $\alpha$, we use an analysis step size $S_a$ to obtain the STFTM and use a synthesis step size $S_s$ such that $S_a = S_s/\alpha$. The frame lengths in the analysis and synthesis process are both $L$. Here, we use a fixed synthesis step size $S_s = L/4$, which

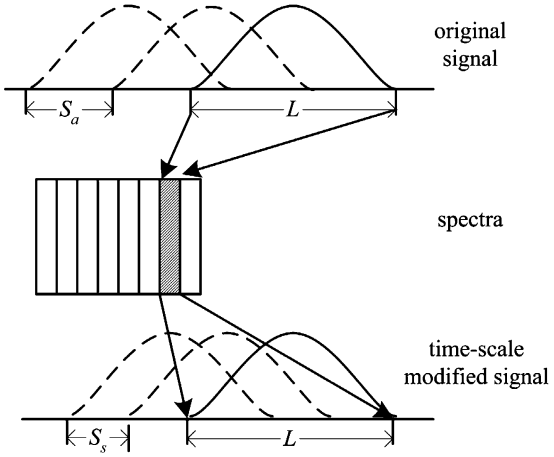



Fig. 6. Time-scale modification in RTISI-LA.

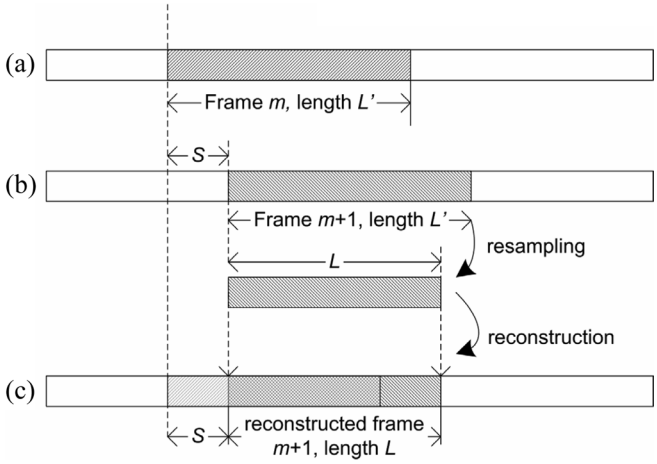



Fig. 7. Real-time pitch modification process. (a) Frame $m$ in the original signal. (b) Frame $m + 1$ in the original signal. (c) Frame $m + 1$ in the reconstructed signal, which is obtained by resampling the original frame $m + 1$ and synthesizing.

keeps computational requirements consistent for various modification rates. The process is shown in Fig. 6.

The analysis step size $S_a$ can be any arbitrary value. If the analysis step size is positive and less than the synthesis step size $S_s$, the result is time-stretching, i.e., the reconstructed audio plays back more slowly than the original. If the analysis step size is larger than the synthesis step size, the result is time-compression (the reconstructed audio plays back faster than the original). The analysis step size can also be negative for "reverse" playback, or even zero.

Note that if the analysis step size $S_a$ exceeds the frame length $L$, there is no overlap between adjacent analysis frames, and the quality of TSM may suffer as a result. This can be avoided by choosing a smaller synthesis step size $S_s$, so that for the maximum desired $\alpha$ (i.e., greatest degree of speedup), $S_a$ never exceeds $L$.

In subjective listening tests, the perceptual quality of reconstructions using RTISI-LA compare favorably with other methods such as SOLA, even for monophonic signals with small modification rates where time-domain methods are at their best. In subjective tests using monophonic signals, RTSI-LA sounds better, especially as the modification rate approaches a factor of 2. This is despite (or perhaps because of) the fact that the first step in the RTISI algorithm is to throw away the phase information. Readers who would like to do their own listening tests can find time modification examples using RTISI-LA at http://www.zwhome.org/~lonce/Publications/RTISI_LA.html.

## V. Pitch Modification (PM)

PM of audio signals is useful in a number of applications such as multimedia audio signal processing, speech synthesis, vocal identity transformation, and creating special sound effects for applications such as karaoke. A simple extension to the original RTISI-LA algorithm allows it to be used for pitch modification, while also providing simultaneous independent control of time-scale modification.

The key is to generate a "stretched" or "squashed" spectrum for each frame for upwards or downwards pitch shifts, respectively. Modifying the spectrum directly is problematic, however, as the bin spacing for typical frame lengths is quite coarse. This makes precise control of pitch shifts nearly impossible.

To alleviate this problem and allow very precise pitch shifts, the pitch is shifted by resampling each frame in the time-domain, prior to computing the STFTM. If the normal analysis frame length is $L$, and the pitch is to be shifted upwards by a factor of $q$, then for each frame we use a block of $L' = qL$ samples, as shown in Fig. 7(b). Resampling is done in the time-domain to generate a frame of length $L$ as shown in Fig. 7(c). The target STFTM is then computed on the resulting frame. Then the RTISI-LA is run exactly as for the case without pitch modification. Note that in the reconstruction process the step size $S$ and frame length $L$ remain constant regardless of the degree of time-scale or pitch modification. In this way, the computational load remains essentially constant for any pitch or time-scale modification factor.

In our current experiments, we use simple linear interpolation, which is computationally inexpensive and provides reasonable sound quality in most cases. Better interpolation algorithms will almost certainly provide some improvement in sound quality. In particular, when resampling to shift the pitch upwards, simple linear interpolation will result in aliasing if the original signal has high-frequency content, and is not low-pass filtered prior to resampling.

In our experiments, we have found that pitch shifting using RTISI-LA works well for pitch shift factors from 1/2 to 2, i.e., pitch shifts downwards or upwards by as much as one octave. Examples of pitch modification using RTISI-LA can be found at http://www.zwhome.org/~lonce/Publications/RTISI_LA.html.

## VI. Conclusion

The RTISI-LA algorithm for constructing real-valued time-domain signals from a sequence of magnitude spectra was presented. The method is based on the classic G&L [1] algorithm, and modified to be structurally and computationally suitable for real-time applications. The initial phase estimates for each frame were shown to be the key for fast convergence. Variations

of the algorithm were developed that allow for a parameterized tradeoff between computational and performance goals.

Experiments show that with a specific asymmetric analysis window, RTISI-LA achieves its best reconstruction quality. Although increasing the number of iterations generally tends to improve the reconstruction quality, the SER does not increase monotonically, though in practice this does not limit effectiveness.

The RTISI-LA method at eight transform iterations per frame generally approaches or exceeds the performance of the classic G&L algorithm with 80 iterations per frame in terms of the SER. The reduction in computation and consistently high quality across a wide variety of signal types makes RTISI-LA an appropriate choice for many signal processing applications.

## REFERENCES


[1] D. W. Griffin and J. S. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-32, no. 2, pp. 236–243, Apr. 1984.

[2] S. H. Nawab, T. F. Quatieri, and J. S. Lim, "Signal reconstruction from short-time fourier transform magnitude," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-31, no. 4, pp. 986–998, Aug. 1983.

[3] S. H. Nawab, T. F. Quatieri, and J. S. Lim, "Algorithms for signal reconstruction from short-time fourier transform magnitude," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Boston, MA, 1983, pp. 800–803.

[4] B. Yegnanarayana, D. K. Saikia, and T. R. Krishnan, "Significance of group delay functions in signal reconstruction from spectral magnitude or phase," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-32, no. 3, Jun. 1984.

[5] M. H. Hayes, J. S. Lim, and A. V. Oppenheim, "Signal reconstruction from phase or magnitude," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-28, no. 6, pp. 610–623, Dec. 1980.

[6] P. L. Vanhove, M. H. Hayes, J. S. Lim, and A. V. Oppenheim, "Signal reconstruction from signed fourier transform magnitude," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-31, no. 5, pp. 1286–1293, Oct. 1983.

[7] T. F. Quatieri and A. V. Oppenheim, "Iterative techniques for minimum phase signal reconstruction from phase or magnitude," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-29, no. 6, pp. 1187–1193, Dec. 1981.

[8] G. Michael and M. Porat, "On signal reconstruction from fourier magnitude," in *Proc. 8th IEEE Int. Conf. Electron., Circuits, Syst.*, Sep. 2001, vol. 3, pp. 1403–1406.

[9] M. R. Portnoff, "Magnitude-phase relationships for short-time Fourier transforms based on Gaussian analysis windows," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1979, vol. 4, pp. 186–189.

[10] D. W. Griffin and J. S. Lim, "Speech synthesis from short-time fourier transform magnitude and its application to speech processing," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 1984, vol. 9, pp. 61–64.

[11] W. Kim and M. H. Hayes, "Phase retrieval using a window function," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 41, no. 3, pp. 1409–1412, Mar. 1993.

[12] Y. Shapiro and M. Porat, "Optimal signal reconstruction from spectral amplitude," in *Proc. 13th Int. Conf. Digital Signal Process.*, Jul. 1997, vol. 2, pp. 773–776.

[13] V. T. Tom, T. F. Quatieri, M. H. Hayes, and J. H. McClellan, "Convergence of iterative nonexpansive signal reconstruction algorithms," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-29, no. 5, pp. 1052–1058, Oct. 1981.

[14] G. T. Beauregard, X. Zhu, and L. Wyse, "An efficient algorithm for real-time spectrogram inversion," in *Proc. 8th Int. Conf. Digital Audio Effects (DAFX-05)*, Sep. 2005, pp. 116–121.

[15] S. Roucos and A. M. Wilgus, "High quality time-scale modification for speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1985, vol. 10, pp. 493–496.

[16] W. Verhelst and M. Roelands, "An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1993, vol. 2, pp. 27–30.

[17] D. Dorran, R. Lawlor, and E. Coyle, "High quality time-scale modification of speech using a peak alignment overlap-add algorithm (PAOLA)," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2003, vol. 1, pp. 6–10.

[18] E. Moulines and F. Charpentier, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones," *Speech Commun.*, vol. 9, pp. 453–467, 1990.

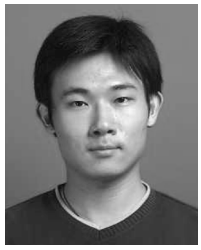
[19] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *IEEE Trans. Speech Audio Process.*, vol. 7, no. 3, pp. 323–332, May 1999.

[20] T. F. Quatieri and R. J. McAulay, "Shape invariant time-scale and pitch modification of speech," *IEEE Trans. Signal Process.*, vol. 4, no. 3, pp. 497–510, Mar. 1992.

[21] K. Achan, S. T. Roweis, and B. J. Frey, "Probabilistic inference of speech signals from phaseless spectrograms," *Neural Inf. Process. Syst.*, vol. 16, pp. 1393–1400, 2003.

[22] M. Dolson, "The phase vocoder: A tutorial," *Comput. Music J.*, vol. 10, no. 4, p. 14, 1986.

[23] M. Slaney, D. Naar, and R. F. Lyon, "Auditory model inversion for sound separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1994, vol. 2, pp. 77–80.

**Xinglei Zhu** received the B.S. degree in computer science from the University of Science and Technology of China (USTC), Hefei, Anhui, China, in 2002 and the M.S. degree in computer science from the National University of Singapore (NUS), Singapore, in 2004.

He is currently a Research Engineer in the Media Understanding Department, Institute for Infocomm Research, Singapore. His research interests include digital signal processing, streaming authentication, and watermarking.

**Gerald T. Beauregard** (M'00) received the B.S. degree in electrical engineering from Queen's University, Kingston, ON, Canada, in 1986, and the M.A. degree in electroacoustic music from Dartmouth College, Hanover, NH, in 1991.

He is currently VP Media Technology at muvee Technologies, Singapore. His research interests include media analysis and processing.

**Lonce L. Wyse** received the Ph.D. degree in cognitive and neural system from Boston University, Boston, MA, in 1994.

He then spent a year as a Fulbright Scholar in Taiwan developing a computational model of pitch perception. He then joined the Institute of Infocomm Research, Singapore, where he headed the Mixed Media Modeling Laboratory, developing real-time interactive sound modeling and synthesis software systems. He recently took a position as an Associate Professor in the Communications and New Media Program, National University of Singapore. He is currently teaching, pursuing research, and doing creative work in sonic arts and interactive media.