Implementation of an analysis/synthesis system on a DSP56001 for general purpose sound processing

Rupert C. Niebel and Michael Warsch
Skalitzer Str. 62, D-1000 Berlin 18
Technical University of Berlin, EE Department, D-1000 Berlin 18
niebel@opal.cs.tu-berlin.de

Abstract

The following paper discusses the implementation of the digital phase vocoder on a DSP56001 system. The digital phase vocoder represents an analysis/synthesis system providing separation from frequency and time domain information using short-time Fourier transformation (STFT) and a special conversion technique. Due to this technique the system offers several possibilities of parameter modification of the input signal such as pitch transpose and time scaling. If no parameter modification is performed the system acts as an identity system.

1 Theory

1.1 Analysis

The analysis part of the system can be regarded as an implementation of a digital bandpass filterbank using STFT to represent the input signal. Analysis is performed using a weighted overlap-add structure and the time aliasing technique. The input signal is first weighted by an appropriate window function by convolution. The window function is designed by multiplying a Kaiser window1 \( w_{K A I S}(n) \) with a si-function \( b(n) \). The resulting analysis window has the form:

\[
h_{ANA}(n) = b(n) \cdot w_{K A I S}(n).
\] (1)

The si-function is the time domain representation of a lowpass filter in the frequency domain. The lowpass filter is used to separate a single spectral band of the bandpass filterbank. The analysis technique is to modulate each band of the filterbank back into the baseband thus being able to decimate the sample rate. Sample rate decimation is performed by skipping \( M \) samples between successive analysis blocks. For ideal filters the maximum value of \( M \) can be set to \( K \). For the present case of non-ideal filters \( M \) should be set to a somewhat lower value. We found out that \( M = \frac{K}{2} \) is a suitable maximum value for \( M \).

The analysis window is divided into \( I \) groups of \( K \) samples where \( K \) is the number of channels. If \( n_{groups} \) denotes the number of blocks of \( K \) samples in one half of the symmetric window then the window length \( N \) for an odd number of samples is:

\[ N = 2 \cdot n_{groups} \cdot K + 1. \]

The weighted input sequence for a block of \( N \) samples with block index \( m \) is of the form:

\[
y_{in}(n) = \sum_{m=-(N-1)/2}^{(N-1)/2} h_{ANA}(-n) \cdot x(n + mM).
\] (2)

Time aliasing the weighted input sequence means to add all corresponding samples of each group thus creating an appropriate input sequence with length \( K \) for the STFT. Therefore the time aliased sequence has the form:

\[
y_a(r) = \sum_{k=0}^{n_{groups}-1} \sum_{r'=-k}^{K-1} y_{in}(n).
\] (3)

This transformation is performed using a FFT algorithm. The resulting short-time spectrum is:

\[
X_k(m) = \sum_{r=0}^{N-1} y_a(r) \cdot W_{K}^{mr} \quad \text{with} \quad k = 0, 1, \ldots, K - 1 \quad \text{and} \quad W_{K} = e^{j2\pi/K}.
\] (4)

The modulation of the higher frequency bands back into the baseband is automatically realized by multiplying the time aliased input sequence \( y_a(r) \) with the sine and cosine coefficients of the FFT algorithm represented by \( W_{K}^{mr} \). After calculating the first block (\( m = 0 \)) the analysis window is shifted forward \( M \) samples to calculate the next block (\( m + 1 \)).

1For our purposes the Kaiser window has proved to be best, but for other applications any other window function like Hamming, Hanning, etc. can be chosen.
1.2 Frequency computation

The instantaneous frequencies for each channel are computed by converting the real and imaginary outputs of the FFT into polar coordinates. This is done using the well-known formulae:

\[ A(n) = \sqrt{R(n)^2 + \Omega(n)^2} \]  
\[ \Theta(n) = \arctan \left( \frac{\Omega(n)}{R(n)} \right) \]

for calculating the magnitude \( A(n) \) and the phase \( \Theta(n) \). Instantaneous frequency can be calculated using the first order approximation of the derivative of the phase:

\[ \frac{d\Theta(n)}{dt} = \Delta \Theta(n) = \Theta(n) - \Theta(n-1). \]

Therefore the instantaneous frequency components are:

\[ f(n) = \frac{d\Theta(n)}{dt} = \Delta \Theta(n) = \arctan \left( \frac{\Omega(n)}{R(n)} \right) - \arctan \left( \frac{\Omega(n-1)}{R(n-1)} \right) \]

Eq 7 means to subtract the phase values of successive blocks in the corresponding channels.

Before calculating the above parameters, one additional step has to be considered. Due to the analysis process \( M \) samples have been skipped between successive blocks. Regarding the calculated phase values as the differences in phase between successive samples of the input sequence a large error might occur. The calculated differences do not relate to successive input samples but to samples of the spaces apart. For the magnitude values this is usually not very important because they normally change slowly. Phase values might change very rapidly in contrast. So if the calculated phase differences are regarded as "true" values, significant phase changes might be lost, leading to erroneous instantaneous frequencies. Prior to converting to polar coordinates an interpolation to a somewhat higher sample rate has to be performed. If the original sample rate is denoted \( \text{sample rate} \cdot \frac{Q}{M} \). Similarly the number of skipped samples \( M \) can be lowered alternatively.

The phase vocoder provides information about the difference between the center frequency of a given channel and the frequency of the component found in that channel. Therefore only the amount of change in the difference in phase between successive samples in that channel is considered. Frequency values are determined adding the corresponding center frequency \( 2\pi k/K \) of channel \( k \) to the phase difference in that channel. This means that to determine the frequency at a given point in time you have to sum all the preceding phase differences and add it to the center frequency of that channel. In analogy to an oscillator, the amount of change in phase difference equals the change of increment in a wavetable-look-up oscillator. So if the change of phase difference for a given channel is zero for all samples, the frequency in that particular channel is exactly equivalent to the corresponding center frequency.

1.2.1 Refinements to the conversion process

Before calculating the final phase differences some special cases have to be considered. If the magnitude of a component in a given channel equals zero, there is no sense in calculating the frequency at all. Therefore the frequency value is set to zero in that case.

Because of the structure of the analysis process, all phase values are bandlimited to \(-f_{\max}/K \leq \Theta(n) \leq f_{\max}/K\). If the amplitude envelope of the channel signal changes sign, large phase jumps may occur. These jumps are so called "jitters". If a "jitter" occurs the phase difference between adjacent samples will probably have a higher value at under normal circumstances. In fact an additional phase jump of \( \pi \) will take place. To compensate for this change the calculated phase value for the component is limited to the allowed bounds. This process is called "phase wrapping". Whenever the absolute phase difference is greater than \( \pi/2 \), \( \pi \) is added to the calculated phase value until it lies in the allowed bounds. To adjust the envelope the sign of the magnitude of the component is toggled each time \( \pi \) is added or subtracted[3].

Finally the calculated phase differences have to be divided by \( M/Q \) because all values were determined for the decimated sample rate \( \text{sample rate} \cdot \frac{Q}{M} \).
1.3 Synthesis
Prior to resynthesizing, the interpolation of the calculated values to the original sample rate has to be performed. Furthermore the center frequencies $\omega_k$ have to be calculated:

$$\omega_k = \frac{2\pi k}{K}. \tag{8}$$

According to the preceding section a phase value for each sample in each channel has to be determined:

$$\Phi_k(n) = \Phi_k(n - 1) + \mu_k \cdot \left[ \Theta_k(n) + \omega_k \right]. \tag{9}$$

The phase value is recursively calculated by summing up all preceding phase values and adding the center frequency of the channel. To start the calculation, the values for magnitude and phase have to be initialized. Each channel is supposed to start at its center frequency with a magnitude of 1.0. Therefore the initialization is:

$$A(-1) = 1.0 \quad \text{and} \quad \Phi_k(-1) = -\omega_k \cdot \mu_k.$$  

$\Phi_k(-1)$ has to be initialized with negative sign. The center frequency of each channel is added each time $\omega_k$ is calculated. According to the oscillator analogy, to start at the center frequency means to output the first sample in the wavetable. If $\Phi_k(-1)$ were initialized with positive sign the center frequency would be added twice thus starting the output with the second sample. In analogy to the oscillator $\Theta_k(n) + \omega_k$ corresponds to the increment in the wavetable lookup. The factor $\mu_k$ is the pitch transpose factor and is described in the next section. The resynthesized samples are then given by:

$$\hat{x}(n) = \sum_{k=0}^{K/2} |a(k)| \cdot A_k(n) \cos \Phi_k(n). \tag{10}$$

The factor $a(k)$ is used for scaling purposes. Due to the fact of processing only real input signals and the resulting symmetry, half of the FFT output can be skipped. The information in channel 1 is identical to that in channel $(K/2) + 1$ and so on. Only channel 0 and $K/2$ contain exclusive information. Therefore the information in channel 1 up to $(K/2) - 1$ counts twice. So $a(k)$ is set accordingly:

$$a(k) = \begin{cases} 1 & \text{for } k = 0, K/2 \\ 2 & \text{else} \end{cases}$$

1.4 Parameter modification
Modification of the time varying parameters can easily be done. Alteration of the pitch of the input signal is performed by changing $\mu_k$ to an appropriate value. For $\mu_k = 1.0$ no modification of pitch is taking place. Setting $\mu_k = 2.0$ transposes the input signal up by one octave. As can be seen in the preceding section, $\mu_k$ affects the center frequencies and the calculated instantaneous phase values. For $\mu_k = 2.0$ all values are doubled thus unspawning the signal one octave up. Appropriate values for $\mu_k$ will transpose the signal by the desired amount.

Time scaling is performed by interpolating less or more samples prior to the synthesis algorithm thus changing the number of samples to be resynthesized and therefore the length of the output signal respectively.

2 Implementation
For computational efficiency a so-called Real FFT algorithm was used thus calculating two input blocks with one FFT algorithm instead of filling the imaginary inputs with zeros for each block. So the analysis procedure computes two input blocks for the FFT algorithm. After finishing the analysis, interpolation to an intermediate sample rate is performed. The next step is to calculate the magnitudes and phase differences due to the tricky conversion process.

Prior to synthesis, interpolation to the original sample rate is performed. Finally synthesis is processed by summing up all the channel information for each output sample. After resynthesizing two blocks the program can loop back to the analysis process to compute the next two blocks.
2.1 Performance

Fig. 1 shows an example of the program's performance. For simulation an AM signal was generated as shown in Fig. 1(a). The generated signal consists of 304 samples. The parameters of the program were chosen as follows: $g = 3$, $K = 32$, $M = K/8$. Because of the relatively low value of $M$, no interpolation after analyzing the signal needed to be performed. Fig. 1(b) shows the synthesized output signal. The input signal was time-scaled to 72% and pitch transposed one octave up simultaneously. As can be seen, the program performs quite satisfactorily.

![Diagram](image)

Figure 1: Generated input signal (a) and performed output signal (b)

References

