University of Sydney

DESC9115 Digital Audio Systems (2012)

Instructor: William L. Martens
Student: Tsun Kit Lam
SID: 420088455

Assignment 2: (LPC Cross-Synthesizer with Distortion)
Due Date: 8 May, 2012

Product Name: Bongo Wizard
Product Type: LPC Cross-Synthesizer (with Distortion effect)
Product Usage: used specifically to process recorded bongo samples.
Programmer: Tsun Kit Lam

---

1. Problem Description

Computer music technology is widely adopted by musicians, producers and engineers. At some stages of music production, digital technology is not only used as a supplementary resource, but is also being used as a replacement of certain traditional, physical process of music production. For example, a digital sampler, which is often used as a replacement of physical instruments as well as the recording process.

Software samplers allow users to program music without having to use different instruments and expensive recording equipment. However, as with many inventions, there are disadvantages laying behind. As far as musical concerns, neutrality is a very import element throughout many genres, the use of synthetic instruments often rise the conflict. For example, a recorded bongo hit that is repeatedly played throughout the whole song, will sound very unnatural to the listeners, because a real sounding bongo has various sonic characteristics every time when being hit, depending on how hard the bongo player hits it, how quick that he or she hits, and which spot of the bongo is being hit. These will effect the amplitude, pitch and eventually the overall sound of the bongo.

While most of the samplers provide users the accessibility to adjust the amplitude of the sound samples (with the parameters of attack, decay, sustain and release), the neutrality of music requires more fine adjustment. A real bongo has a different rate of decay at different frequencies, for instance, a high bongo is likely to have a faster decay rate than a low bongo. Another problem with samplers is that the overall spectral envelope of the sound is fixed, which means the resonance of the bongo sample used will be consistence, this makes it more difficult for users to achieve a natural sounding bongo arrangement. With such challenges mentioned, a solution is proposed, the ***Bongo Wizard***. A Linear Predictive Coding Cross-Synthesizer with additional Distortion effect, made specifically for processing bongo samples, allowing users to take advantage of the modern sampler technology, yet maintaining the traditional neutrally of music.

2. Specification

Bongo Wizard is a linear predictive coding cross-synthesizer. The idea is to extract certain characteristic (more details on the following section) of a preset bongo sample then apply it to the user' bongo sample , giving it varieties which can help users achieving musical neutrality. ***Bongo Wizard*** is based on source analysis, source-filter separation and cross-synthesis.Generally, the analysis stage is a series of frames, representing the data of the sound. In most cases, each frame will consist of such parameters:

- the average amplitude of the residual sound and original sound (Gain Comp.)
- the ratio of the two amplitudes
- the estimated pitch
- the frame duration
- the coefficients for the allpole filter, which is basically the resonator structure

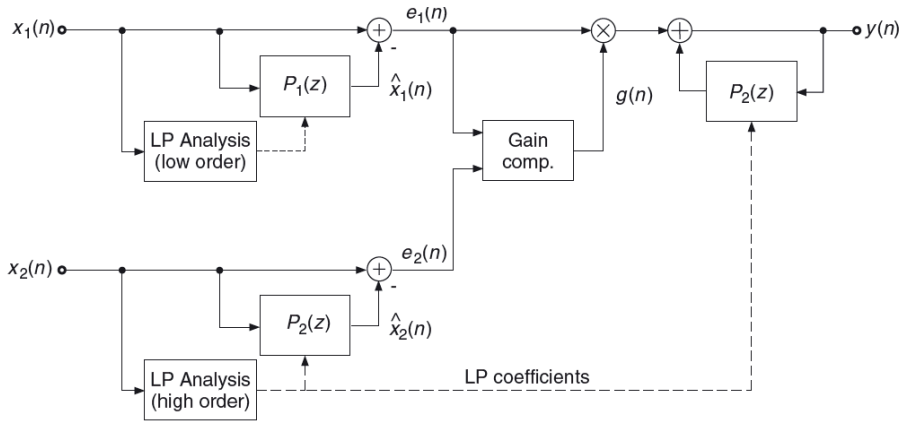


Figure 1: *General LPC Cross Synthesis Structure*

Most plugins are made for dealing with a wide range of sound. However, as mentioned previously, ***Bongo Wizard*** is a processor made specially for processing bongo samples only, therefore some of the general approaches for general Linear Predictive Coding Analysis may not be necessary. First of all, due to the nature of a bongo, which is a very short decaying instrument, the use of frames during the analysis stage will be unnecessary. Instead, the analysis will be performed on the entire sound throughout the duration, and then being averaged. Secondly, ***Bongo Wizard*** is suppose to be used as a supplementary processor for a sampler, in other words, most of the typical LPC parameters mentioned earlier may overlap with the sampler, because the sampler will most likely to provide the parameter of amplitude, and pitch adjustment for the users. This leaves only a few parameters needed for ***Bongo Wizard***, that is the filter coefficients of the sample input by the users, and the coefficients of the chosen bongo sounds (provided by this plugin. To extend that, the plugin also allows user to determine how much resonance (filter coefficient) of the input bongo sample will be extract, and how much resonance of the bongo sound provided by the plugin will be cross-synthesized with user's original bongo sample.

In addition, the plugin provides not only a variety of bongo sound presets, but also a set of resonance of various experimental sound presets as well, with interesting resonating structure for users to cross-synthesize with their bongo sample. The plugin also provides a distortion effect for experimental purpose fro users, which is different than any distortion that the users can get. Generally, users may apply distortion on bongo samples before and after it is processed by an effect processor.The distortion effect of this plugin however, is applied within the signal chain of cross-synthesis. It distorts the excitation (impulse response) of the user's input bongo sample before it get cross-synthesized.

In summary of Specification, ***Bongo Wizard*** removes the resonance of the plugin's bongo samples from the excitation signal (which is the hit of the bongo), then isolate the impulsive signals and the resonance resonances excited by these impulsive signals. Afterwards, it reconstruct the original signals by filtering the obtained impulsive signals with the resonant filter obtained from the bongo sample provided in the plugin.

3. Implementation

The proposed system was implemented using MatLab. The original `filter.m` and `lpc.m` MatLab function are obtained inside Matlab, and the distortion Matlab code is obtained from Udo Zolzer (Helmut Schmidt University). Such mentioned function and

codes were later implemented to construct the algorithm of ***Bongo Wizard***. The implementation process of ***Bongo Wizard*** is break down 6 parts in the following section.

Step 1: Linear Predictive Coding of Bongo Sample (Input by Users)

The `lpc.m` function (obtained from Matlab) is adopted for purpose of linear predictive coding. The function uses the mathematical approach of:

$$\hat{x}(n) = \sum_{k=1}^{p} akx(n-k) \quad \textbf{(1)}$$

From the equation above, the input sample ***x(n)*** is predicted by a combination of past samples of the input signal (in linear nature). ***p*** is the prediction order and a_k are the prediction coefficients[1].

By implementing `lpc.m` in the algorithm of ***Bongo Wizard***, where `a=lpc(x,p)`, it will calculate the coefficients of ***x*** (the input bongo sample). Users are able to determine the amount of ***p,*** which dictates how many poles will be used to model the resonance.

Step 2: Filter of Bongo Sample (to obtain excitation signal minus resonances)

The `filter.m` function (obtained from Matlab) is adopted for the purpose of filtering. The function uses the mathematical approach of

$$e(n) = x(n) - \hat{x}(n) = x(n) - \sum_{k=1}^{p} akx(n-k) \quad \textbf{(2)}$$

The final result ***e(n),*** which is the prediction error, is the difference between the original input signal ***x(n)*** and its prediction ***x̂(n).*** The calculation of ***e(n)*** is done by using a transversal (direct) FIR filter [2]:

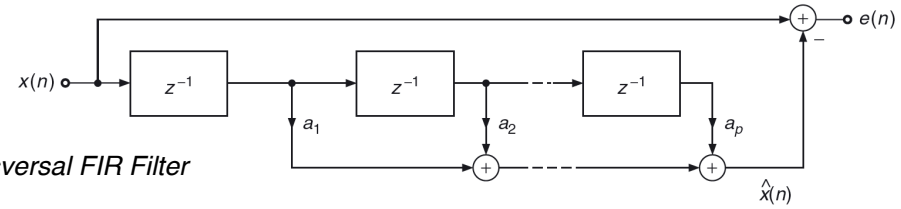


Figure 2: *Transversal FIR Filter*

In Matlab, `bongo_residual=filter(a, 1, x)`, ***a*** is the obtained LPC coefficients of the input bongo sample, and ***x*** is the bongo sample it self. This will result as an excitation signal without the residual (resonances) the user's bongo sample.

Step 3: Distortion

An additional distortion is implemented to the algorithm of ***Bongo Wizard*** for creative purpose. As mentioned early in the proposal, the distortion is not applied on the output of the system, but within it. It distorts the excitation (impulse response) of the user's input bongo sample. From Step 2, the ***Bongo Wizard*** obtains ***a***, which is the excitation (minus resonance) of the input bongo sample, ***a*** is then route to a the distortion.

Step 4: "To-Be Synthesized" Bongo Presets in *Bongo Wizard*

In order to perform Cross-Synthesis, two sounds are needed. One is input by the users, the other one is provided inside the plugin. The approach is having 50 bongo audio samples and 11 high-resonating sounding audio samples imported to Matlab, and then convert to a set of data to be loaded.

There are 61 cases provided in the Matlab code (e.g. case 1, case 2...) which are created for users to choose from. These cases are a range of data, which are those set of data converted from the 61 bongo audio samples. The cases work as a preset library, for users to determine which type of bongo the user desire to cross-synthesize with.

Step 5: Linear Predictive Coding of "To-Be Synthesized" Bongo Sample

This step is the same as Step 1, except it does LPC analysis/calculation on the bongo sample presets instead of the user input bongo sample. In other words, this steps obtains LPC coefficients of the chosen case (preset) in ***Bongo Wizard.***

e.g. `preset_coeff = lpc(case, p2)`

where `case` *is the case(preset) chosen*

`p2` *is the amount of poles that will be used to model the resonance*

Step 6: Cross-Synthesis

This last step reconstructs the original signals by filtering the obtained impulsive signals with the resonant filter obtained from the bongo sample provided in the plugin.

e.g. `cross_synth_signal = filter(1, preset_coeff,bongo_residual)`

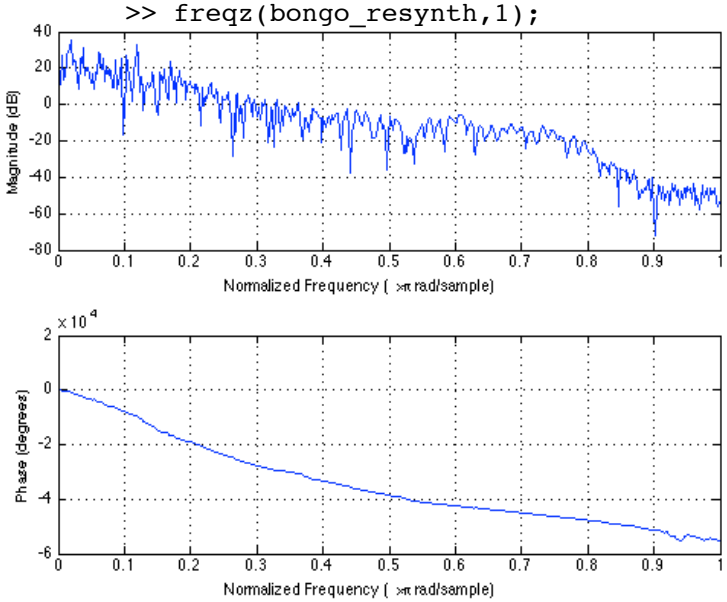
4. Evaluation

After implementing the system on MatLab, it was shown possible to perform accurate cross-synthesis between 2 signal. This allows users to use only one bongo sample from sampler, and still main the neutrality of music, by applying different resonance structure extracted from other bongo presets to it.

To demonstrate, the use of re-synthesis is a good examination tool to determine whether a impulse response and the resonance of a sound can be extract and to be reconstructed accurately.

In Matlab:

```
>> fs = 44100;
>> bongo = wavread('bongo.wav');
>> bongo_coff = lpc(bongo, 24);
>> bongo_residual = filter(bongo_coff, 1, bongo);
>> bongo_resynth = filter(1, bongo_coff, bongo_residual);
>> freqz(bongo,1);
>> freqz(bongo_resynth,1);
```



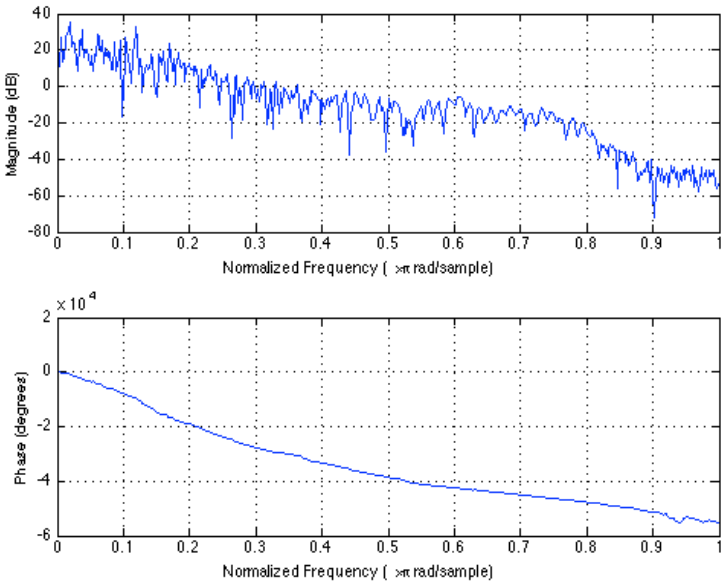


Figure 3 (Left) ***:*** *frequency response of filter of original input bongo signal*

Figure 4 (RIght) : *frequency response of filter of re synthesized bongo signal*

Judging from the sound and the graph of two signal, they appear to be very identical. The accuracy of the re-synthesis construction demonstrated can help indicating that the cross-synthesis will also be quite accurate. By modeling the resonator, ***Bongo Wizard*** can also simulate the decay of the chosen sound (see figure 5).

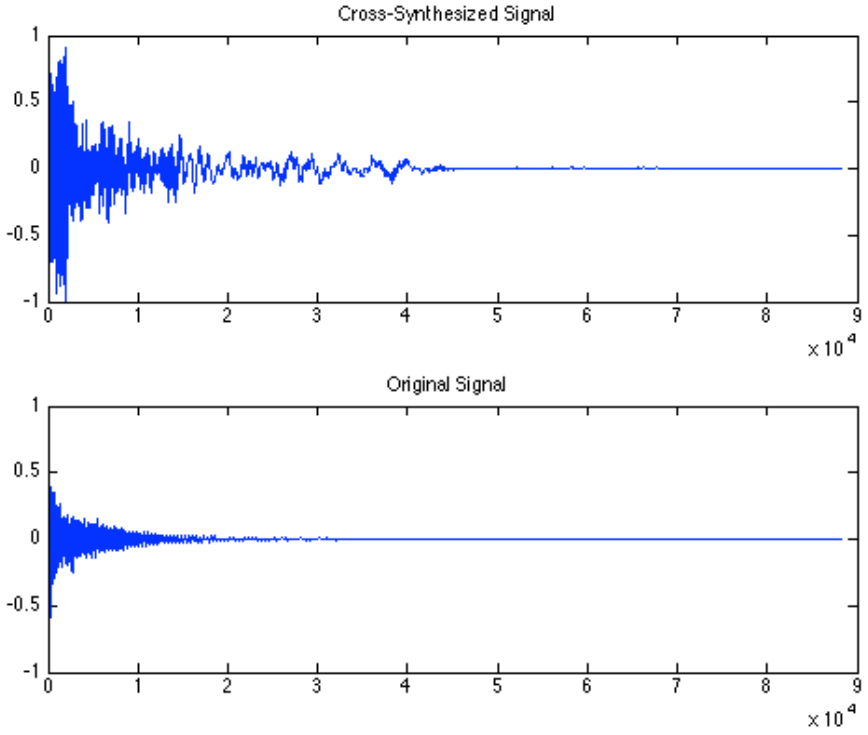


Figure 5: *Time vs Frequency Response of original input bongo and Cross Synthesized bongo*

Bongo Wizard, aimed to help musicians, producers, engineers, to program bongo arrangements without the fear of sounding robotic due to the repeated use of the same bongo sample. The plugin modulate the resonant structure of other bongo samples and allow users to apply those characteristics to their bongo sample.

Reference

[1] J.O. Smith, *Physical Audio Signal Processing: for Virtual Musical Instruments and Digital Audio Effects*, W3K *Publishing, web based, December.2010, pp.186*

[2] U. Zolzer, *DAFX: Digital Audio Effects*, 2nd, John Wiley & Sons Ltd, Hamburg, Germany, May. 2011, pp.283-284