WHITE PAPER

Direct Synthesis versus Wavetable Synthesis

by Phil Burk

Mobileer, Inc.
75 Pleasant Lane
San Rafael, CA
94901 USA
1-415-453-4320

www.mobileer.com

Direct Synthesis versus Wavetable Synthesis

by Phil Burk, Mobileer, Inc.

Polyphonic ringtones have become a major source of revenue for carriers and content producers. The standard format for polyphonic ringtones is an SP-MIDI file using General MIDI with 128 possible melodic instruments and 47 percussion instruments. What is the best way to create the sound of these 175 musical instruments in software?

Mobile device platforms vary greatly in memory size and performance. In order to achieve the optimal MIDI solution for a given platform it is necessary to consider the memory, speed and sound quality tradeoffs involved in the various synthesis techniques. This paper will explore two possible solutions.

Direct Synthesis and Wavetable Synthesis

There are two basic approaches to synthesizing musical sound. The oldest technique, which we will call **"Direct Synthesis"** involves creating a sound purely from mathematics. Signals from oscillators, envelopes, and filters are combined in various ways to mimic the timbres and contours of the desired instrument. Envelopes provide a contour and are used to control the amplitude and other slowly changing aspects of a sound. Filters can change the spectrum of a sound and create dramatic "wah-wah" effects or more subtle changes in timbre. Another common technique is frequency modulation, or FM. This involves using one oscillator to rapidly offset the frequency of another. This technique can generate complex spectra that are easily controlled. Direct Synthesis works well for many instruments including bells, organs, flutes, wood blocks, and others.

It is difficult, however, to match the sound of a concert grand piano or a Stradivarius violin with a few oscillators and filters. So another technique called **"Wavetable Synthesis"** was invented. Wavetables are digital audio recordings of the actual instrument that serve as the basis for the synthesized sound. This wavetable recording can then be shaped with envelopes and filters to create the final sound.

This seems like the perfect solution. Simply record the sound of all the notes you need on all the instruments and it will sound great. But an audio recording can consume lots of memory. A low piano note may sustain for 20-30 seconds, or longer. A 20 second, 16 bit recording of a single piano note at 44100 Hz occupies 1,764,000 bytes. If we plan to create the sound of all the notes of all 175 instruments in less than one megabyte then we obviously must use some tricks.

Wavetable Synthesis Techniques

Here are some techniques that are used in wavetable synthesis to reduce the amount of memory required.

Looping: One way to reduce the memory burden is to take a small section of the sound and play it repeatedly. This *loop* can be quite small and provides a steady tone. Often the initial portion of a sound, the *attack*, is played first followed by the loop. The attack is a critical part of a sound and is very important in how people perceive the timbre of an instrument.

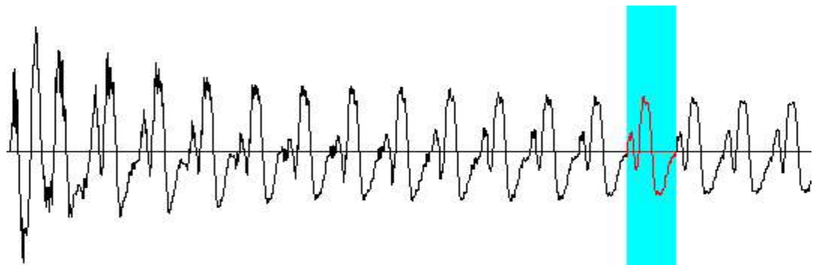

Illustration 1 - Electric Bass showing Attack and Highlighted Loop

Pitch Shifting: Another way to reduce the memory burden is to only store a few selected notes from the instrument. If you have to play a note that is not one of the selected notes then you can shift the pitch of the nearest note up or down to achieve the desired pitch. The farther you shift the pitch then the more unnatural it will sound. Shifting by a few semitones to an octave up or down is usually OK.

Resampling: A recording of a low note on an acoustic bass does not have many high frequency partials. So you can resample the wavetable for that note to a lower sample rate without greatly affecting the timbre.

Enharmonic Partials and Looping

Looping is a powerful technique for saving memory in wavetable synthesis. But there is a limitation that one should be aware of when creating wavetables for certain kinds of instruments. Instruments with complex enharmonic spectra cannot be implemented using short loops.

According to Fourier theory, any sound can be constructed by adding together pure sine waves of various frequencies. Those sine wave components are known as *partials* and define the *spectrum* of a sound. Most instruments have harmonic partials which means that the frequencies of the partials are integer multiples of the fundamental pitch. For example, a piano playing a Concert A at 440 Hz would have harmonic partials at 880 Hz, 1320 Hz, and so on.

If you play a loop of a wavetable that is exactly one period at the fundamental frequency then the harmonic partials will fit within that loop. That is because the period of the harmonic partials are the fundamental period divided by N.

In the following drawing we can see a fundamental pitch in blue. The black line is the N=2 partial and the green line is the N=3 partial. They combine to form the actual instrument waveform which is in red. Note that all the partials and their sum begin and end at zero. This makes it easy to form a clean loop.

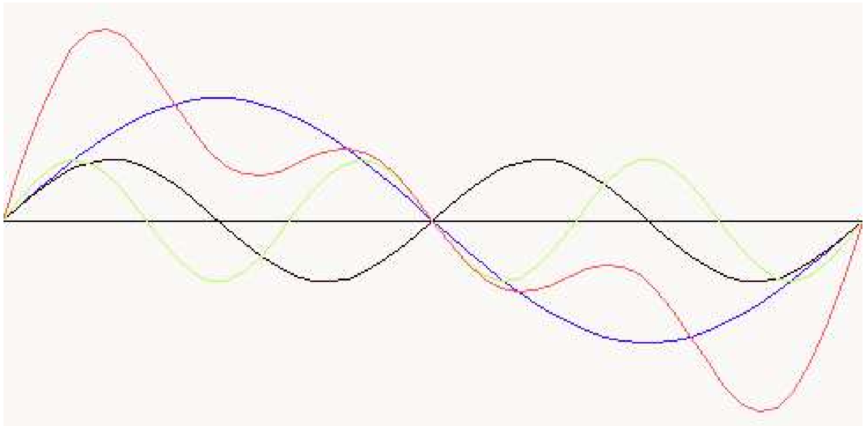

Illustration 2 - Signal composed of Harmonic Partials

In the next drawing we see that the black N=2 partial is replaced by an enharmonic partial that is 2.7 times the fundamental. It does not fit cleanly within the loop. Also notice that their sum (red), does not line up at the end of the loop.

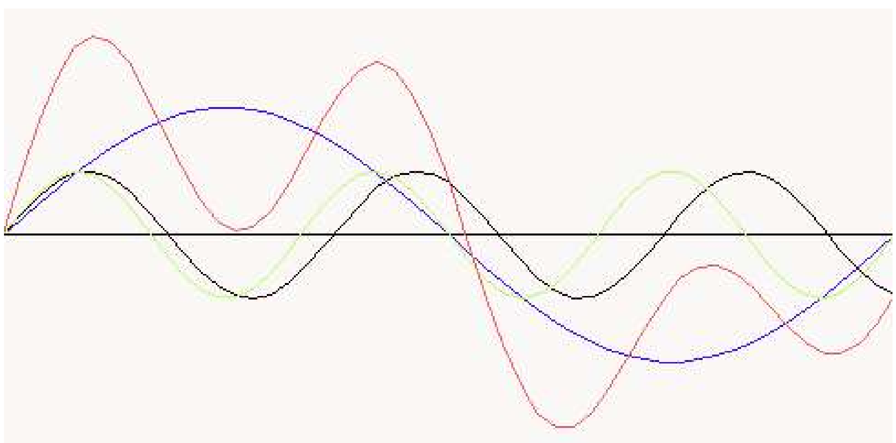

Illustration 3 - Signal with Enharmonic Partials

Why do we care about enharmonic partials? Because many instruments, particularly percussive instruments like Bells, Steel Drums, or Cymbals, have lots of enharmonic partials and cannot be implement easily using looping wavetables. They require a complete wavetable without a loop, which may require lots of memory. So looped wavetables are generally only useful for instruments that have harmonic partials after the attack.

Comparing Direct versus Wavetable Synthesis

The following table compares the two synthesis techniques on a feature by feature basis.

Feature	***Direct Synthesis***	***Wavetable Synthesis***
Memory Footprint	The biggest advantage of direct synthesis is that it consumes very little memory. A complete instrument library can be contained in less than 20 KB.	The quality of a wavetable synthesizer increases with increasing size. A studio quality wavetable library for desktop computers may be several hundred megabytes. But a reasonable instrument library can fit in a few hundred KB.
Realism	Quite good for bells, organs, pads, sound effects, and many percussion instruments. But familiar instruments with complex timbres like piano, violin and trumpet may sound a bit "synthetic".	Very good for instruments with short sounds like a snare drum. Also very good for sounds that rapidly approach a steady timbre like piano and clarinet because they can be looped effectively. Not so good for sounds with enharmonic partials like harsh bells. Also sounds that have a lively timbre may be difficult to reproduce because looped wavetables have a fixed spectrum.
Audio Fidelity	Can be very high fidelity because they are mathematically pure and can be calculated with high precision. But high pitched notes may contain non-band-limited partials that cannot be represented at the chosen sample rate. This can result in some digital aliasing that adds very soft but unexpected frequencies to a tone.	Can be very good if the original recordings are clean. Pitch shifting of the notes requires interpolation which can add some very slight harmonic distortion. If the wavetables are too far apart in pitch then you may hear the jump between one wavetable and the next when playing scales.
Software Complexity	More difficult to implement in software than wavetables because more algorithms are required. Editing the instruments requires a knowledge of various synthesis techniques to achieve an accurate timbre. One must also analyze the desired instrument timbres so that the synthesizer can recreate the spectrum accurately.	Somewhat easier to implement the synthesizer software because they all use the same wavetable oscillator. But one must load and manage the wavetable data efficiently. Instrument editing involves a knowledge of waveform editing but it is easier to achieve an accurate sound because you can "cheat" and use a recording of the desired instrument.

Feature	*Direct Synthesis*	*Wavetable Synthesis*
CPU Load	Generally uses more CPU than wavetables because the sound must be created mathematically. Also the code size is slightly bigger because more mathematical functions need to be supported. So it has more impact on the code cache.	Generally uses less CPU because fewer tricks are required to get an accurate sound. But reading large amounts of memory can impact the data cache. This can be reduced by carefully aligning loops on cache boundaries.
Sonic Flexibility	Very good for dynamic sounds that have lots of variation, like sound effects. This is because synthesizers have more parameters that can be tweaked.	Very good for sounds that are the same every time because you can just record any sound you need to reproduce, within the constraints of your memory footprint.

Implementation of Common Instruments

The physics of instruments can vary greatly. Each instrument has its own optimal solution.

Piano: – Pianos are very common in jazz, rock, classical and other forms of music so it is important to have a good piano sound. But it is hard to create a good piano sound without spending lots of memory. Piano is generally the largest instrument in the sound library. Luckily the partials for plucked strings are mostly harmonic so they can fit in a loop. A common way to synthesize piano with wavetables is to play a loop that is slowly faded out using an envelope. When the note is released the envelope drops suddenly.

Steel Drum: This instrument has a very complex spectrum with enharmonic partials. So it cannot be looped easily if implemented using a wavetable. FM synthesis with a complex ratio between the carrier and the modulating oscillators can generate a complex enharmonic spectrum.

Synth Pads: The General MIDI standard specifies many keyboard instruments that are essentially synthesizers. They have complex sounds with lots of modulation. They are, therefore, more accurately modeled using Direct Synthesis.

Trumpet: The trumpet has a very complex attack as the sound builds up within the tube. This is difficult to model accurately with direct synthesis. The trumpet has a steady sustain tone which makes it easy to loop. So it is a good candidate for wavetable synthesis.

Helicopter: The General MIDI Standard specifies several sound effects including a helicopter. The swooshing sound of the helocopter rotors can be generated quite well using a white noise generator and a resonant filter swept by an LFO. A wavetable is not required.

Cymbals: Cymbals are extremely complex and have high frequency partials that increase over time. You can synthesize something noisy that serves the role of a cymbal in a composition. But it doesn't really sound like a cymbal. So for short cymbal sounds like ClosedHiHat a wavetable is a good choice. You can use a wavetable for long cymbal sounds like the Crash Cymbal but be prepared to use up some memory.

A Hybrid Solution

As you can see, some instruments are best implemented using direct synthesis, and some are best implemented using wavetable synthesis. If you have a severe memory constraint then you may need to create all of the instruments using direct synthesis. That is the most memory efficient solution. An example of a MIDI synthesizer product that is based entirely on direct synthesis is the Mobileer ME1000.

If you have a few hundred kilobytes or more to spend on wavetables then you can use a combination of direct and wavetable synthesis. By combining the techniques, you can spend your memory on just the instruments that benefit the most from wavetable synthesis. The remaining instruments can then be implemented using direct synthesis. This will allow you to optimize the sound quality for a given memory footprint. An example of a MIDI synthesizer product that combines direct synthesis and wave synthesis is the Mobileer ME2000. Examples recordings of both the ME1000 and the ME2000 can be heard at "http://www.mobileer.com/ringtones/".

About Mobileer

Mobileer provides manufacturers with innovative polyphonic ringtone software and audio solutions for embedded devices like mobile phones, PDAs, toys, and other handheld devices. Mobileer is led by a seasoned management team comprised of technology executives and music system designers. Mobileer's ringtone and audio solutions are highly portable, integrate quickly and produce modern, high-fidelity sound. Mobileer's R&D is aimed squarely at pioneering solutions for handheld device manufacturers to address the fast-growing consumer market's demand for polyphonic audio capabilities. Mobileer is a privately-held company based in San Rafael, California. Information about Mobileer can be found at www.mobileer.com.

About the Author

Phil Burk is a programmer, researcher, author and musician with extensive experience in music synthesis, audio digital signal processing and embedded systems design. He led a team at 3DO that invented the first DSP based audio system for video game consoles. He has developed numerous computer music software systems including HMSL, JSyn, TransJam, and JavaSonics. His publications include research papers for ICMC, articles for Game Gems and Audio Anecdotes, and he recently co-authored a text book on "Music and Computers" for Key College Press.

