Wavetable synthesis

What is wavetable synthesis?

Similar to simple digital sine wave generation/additive synthesis but extended at least two ways.

- Waveform lookup table contains samples for not just a single period of a sine function but for a single period of a more general waveshape.
- Mechanisms exists for dynamically changing the waveshape as the musical note evolves:
  thus generating a quasi-periodic function in time.

Not to be confused with common PCM sample buffer playback: soundcards

Wavetable synthesis: Examples

PPG Wave Series: Implementation of wavetable synthesis employed an array containing 64 pointers to individual single-cycle waves.

Waldorf Microwave: Next generation PPG.

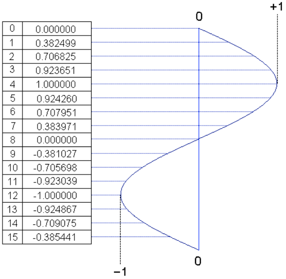
Roland D-50 (and Roland MT-32/variants:) "Linear Arithmetic" synthesizers — combined complex sampled attack phases with less complex sustain/decay phases (basically a wavetable synthesizer with a 2-entry wave sequence table).

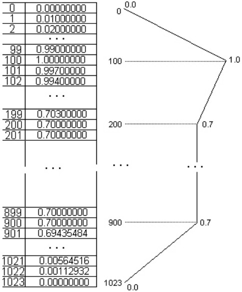
Prophet-VS, (Sequential Circuits)

Korg Wavestation: "Vector synthesis" — move through wavetables and sequences arranged on a 2-dimensional grid.

Wavetable Basics: Making Waves

- The sound of an existing instrument (a single note) is sampled and parsed into a circular sequence of samples or wavetables:
  - each having one period or cycle per wave;
  - A set of wavetables with user specified harmonic content can also be generated mathematically.
- At playback, these wavetables are used to fetch samples (table-lookup)
- However the output waveform is not normally static and evolves slowly in time as one wavetable is mixed with another, creating a changing waveform via ADSR Enveloping.
- Looping maybe used to slow, reverse wavetable evolution





Wavetable Basics: Practicalities

Put more simply, a wavetable synthesiser will store two parts of an instrument's sound.

- A sample of the attack section (e.g. the sound of the hammer hitting a piano string)
- A small segment of the sustain portion of the instrument's sound.

When triggered:

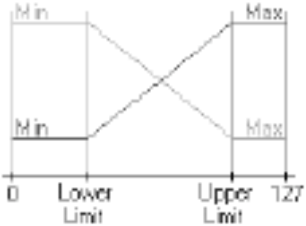
- The attack sample is played once immediately followed by a loop of the sustain segment.
- The endlessly looping segment is then enveloped to create a natural sounding decay (dying away of the sound).

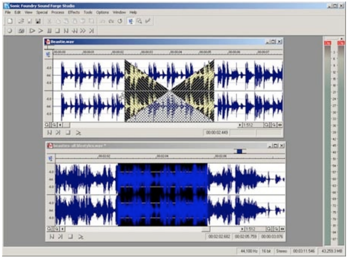
Wavetable v. Sample Playback

- **Differs** from simple sample playback as
  - Output waveform is always generated in real time as the CPU processes the wave sequences
  - Waves in the tables are rarely more than 1 or 2 periods in length.

Simplest idea: Linear crossfading

- Crossfade from one wavetable to the next sequentially.
- Crossfade = apply some envelope to smoothly merge waveforms.





Wavetable Synthesis Example

Simple example — create one sine wave and one saw and then some simple cross-fading between the waves: wavetable_synth.m.

wavetable_synth.m:

```
f1 = 440; f2 = 500; f3 = 620;
Fs = 22050;

%Create a single sine waves
y1 = synth(f1,1/f1,0.9,Fs,'sine');

doit = input('\nPlay/Plot Raw Sine
    y1 looped for 10 ...
   seconds? Y/[N:]\n\n', 's');
if doit == 'y',
figure(1)
plot(y1);
loopsound(y1,Fs,10*Fs/f1);
end
```

```
%Create a single Saw wave
y2 = synth(f2,1/f2,0.9,Fs,'saw');

doit = input('\nPlay/Plot Raw saw
   y2 looped for 10 ...
   seconds? Y/[N:]\n\n', 's');
if doit == 'y',
figure(2)
plot(y2);
loopsound(y2,Fs,10*Fs/f2);
end
```

wavetable_synth.m (Cont.)

Making the crossfades

```
%concatenate wave
ywave = [y1 , y2];

% Create Cross fade half width
% of wave y1 for xfade window
xfadewidth = floor(Fs/(f1*2));
ramp1 = (0:xfadewidth)/xfadewidth;
ramp2 = 1 - ramp1;

doit = input('\nShow Crossfade
            Y/[N:]\n\n', 's');
if doit == 'y',
figure(4)
plot(ramp1);
hold on;
plot(ramp2,'r');
end;
```

```
% Apply crossfade centered over
% the join of y1 and y2
pad = (Fs/f1) + (Fs/f2)
          - 2.5*xfadewidth;
xramp1 = [ones(1,1.5*xfadewidth),
            ramp2, zeros(1,floor(pad))];
xramp2 = [zeros(1,1.5*xfadewidth),
             ramp1, ones(1,floor(pad))];

% Create two period
% waveforms to fade between
ywave2 = [y1 , zeros(1,Fs/f2)];
ytemp = [zeros(1,Fs/f1), y2];

ywave = ywave2;
```

wavetable_synth.m (Cont.)

Adding the crossfade

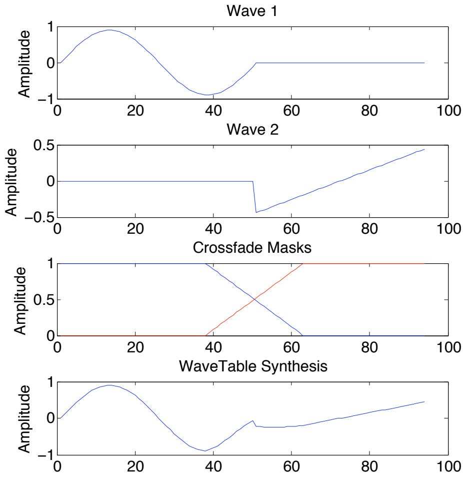
```
% do xfade

ywave2 = xramp1.*ywave2
              + xramp2.*ytemp;

doit = input('\nPlay/Plot Additive
     Sines together? Y/[N:]\n\n', 's');
if doit == 'y',
 figure(5)
 subplot(4,1,1);
 plot(ywave);

 hold off
 set(gca,'fontsize',18);
 ylabel('Amplitude');
 title('Wave 1');
 set(gca,'fontsize',18);
 subplot(4,1,2);
 plot(ytemp);
 set(gca,'fontsize',18);
 ylabel('Amplitude');
 title('Wave 2');
 set(gca,'fontsize',18);
 subplot(4,1,3);
 plot(xramp1);
 hold on
 plot(xramp2,'r')
 hold off
 set(gca,'fontsize',18);
 ylabel('Amplitude');
 title('Crossfade Masks');
 set(gca,'fontsize',18);
 subplot(4,1,4);
 plot(ywave2);
 set(gca,'fontsize',18);
 ylabel('Amplitude');
 title('WaveTable Synthesis');
 set(gca,'fontsize',18);
 loopsound(ywave2,Fs,
          10*Fs/(f1 + f2));
end
```

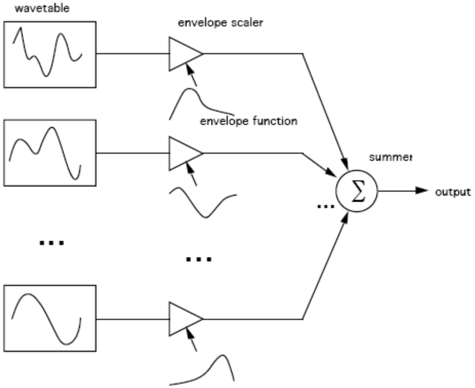
MATLAB Example: Linear Crossfading (Cont.)



Note: This sort of technique is useful to create an ADSR envelope in MATLAB

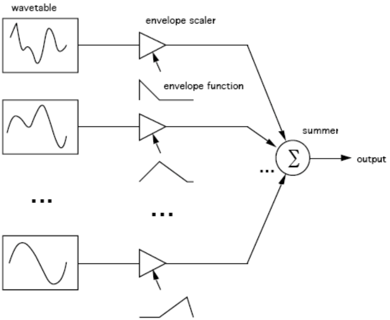
More sophisticated method: Sequential Enveloping

- Example below: two wavetables are being mixed at any one instance of time by moving envelope scale



Linear Crossfading as Sequential Enveloping?

- The simple linear crossfading method can be thought of as a subclass of the more general basis mixing method where the envelopes are overlapping triangular pulse functions.



Wavetable synthesis: Advantages

- Well suited for synthesising quasi-periodic musical tones because wavetable synthesis can be as compact in storage requirements
  - Amount of data being stored and used for this synthesis method is far less than just the PCM sample of same sound.
  - As general as additive synthesis but requires much less real-time computation.
- Wavetable synthesis takes advantage of the quasiperiodic nature of the waveform to remove redundancies and to reduce the data.

Wavetable synthesis: Advantages (cont.)

Enabling Faster Playback

- Precomputes the inverse Discrete Fourier Transform (DFT) of the waveform spectrum before playback
- Rather than computing the inverse DFT in real-time as additive synthesis does.
- Precomputed, real-time synthesis is reasonably simple to implement.

Built in MATLAB Wavetable examples

MATLAB has a basic wavetable synthesiser built-in to its Audio Toolbox:

```
doc wavetableSynthesizer
```

See this MATLAB page for some examples

- Generate Variable-Frequency Staircase Wave:
- Manipulate Audio Samples Using Wavetable Synthesizer
- Modify Wavetable While Stream Processing
- Tune Wavetable Synthesizer Parameters