International Journal of Scientific Engineering and Research (IJSER)
www.ijser.in
ISSN (Online): 2347-3878
Volume 3 Issue 2, February 2015

Music Synthesis using Sinusoid Generator, ADSR Envelope Generator and Composer Code

Tony Mathew[1], Bimal M Abraham[2], Robin Scaria[3]

[1]Christ University Faculty of Engineering, Department of Electronics & Communication Engineering, Kumbalagodu P.O., Bangalore, India

[2]Christ University Faculty of Engineering, Department of Electronics & Communication Engineering, Kumbalagodu P.O., Bangalore, India

[3]Christ University Faculty of Engineering, Department of Electronics & Communication Engineering, Kumbalagodu P.O., Bangalore, India

Abstract: ***The ability to synthesize waveforms through digital methods is a popular technique. This method can be found in many applications such as data communications devices (modems), software radios, and DTMF (Touch Tone) generators. One of its most familiar consumer oriented applications is in music synthesis. In this application, the musician often has control over many instruments and sound effects all from a single synthesizer. Waveform synthesis can be taught early in a typical Digital Signal Processing (DSP) course to illustrate some of the applications of sampling and reconstruction theory. In addition hands-on practice with waveform synthesis can be made very interesting in the context of computer music. Two tools used are a tone (sinusoid) generator and an ADSR envelope generator used to shape the amplitude of the tone, i.e. amplitude modulation. The amplitude of the tone can "fit" inside a curve often called the Attack-Decay-Sustain-Release (ADSR) envelope. These two tools form the basis of the project where we can experiment with computer-based music and musical synthesis using MATLAB's built-in sound capabilities and the PC's sound card.***

Keywords: music synthesis, dsp, matlab, ADSR

1. Introduction

The ability to synthesize waveforms through digital methods is a popular technique. This method can be found in many applications such as data communications devices (modems), software radios, and DTMF (Touch Tone) generators. One of its most familiar consumer oriented applications is in music synthesis. In this application, the musician often has control over many instruments and sound effects all from a single synthesizer. Waveform synthesis can be taught early in a typical undergraduate Digital Signal Processing (DSP) course to illustrate some of the applications of sampling and reconstruction theory. In addition hands-on practice with waveform synthesis can be made very interesting in the context of computer music. In this paper we outline a waveform synthesis project in which we code two simple tools in MATLAB. These tools are a tone (sinusoid) generator and an envelope generator used to shape the amplitude of the tone, i.e. amplitude modulation. These two tools form the basis of the project where we can experiment with computer-based music and musical synthesis using MATLAB's built-in sound capabilities and the PC's sound card.

2. Implementation

The implementation of a music synthesizer (AM-based) involves three codes: 1) tone synthesizer or sinusoid generator, 2) ADSR envelope generator, 3) composer/player code. We assume digital synthesis at a rate of fs = 16,000 samples per second. At this rate we are able to reproduce all piano frequencies according to Nyquist theory.

2.1 ADSR Envelope Generation

The sound output of musical instruments does not immediately build up to its full intensity nor does the sound fall to zero intensity instantaneously. It takes a certain amount of time for the sound to build up in intensity and a certain amount of time for the sound to die away. The period of time during which a musical tone is building up to some amplitude (volume) is called the "attack time" and the time required for the tone's intensity to partially die away is called its "decay time." The time for final attenuation is called the "release time." Many instruments allow the user to hold the tone for a period of time which is known as the "sustain time" so that various note durations can be achieved. The amplitude of the tone can "fit" inside a curve often called the Attack-Decay-Sustain-Release (ADSR) envelope.

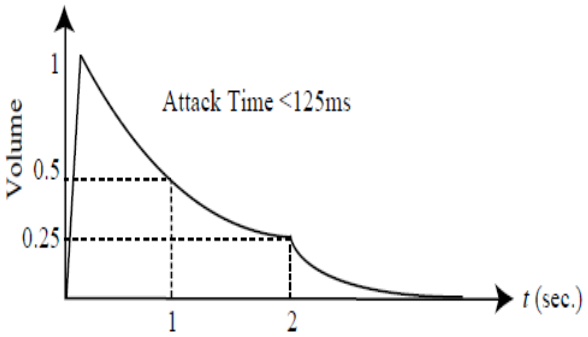


Figure 1: ADSR Envelope for Piano

A synthesizer duplicates the intensity (volume) variation of the tone by multiplying (modulating) the amplitude of the sinusoid with a scale factor dictated by the ADSR envelope, $a(t)$

$$y(t) = a(t) * x(t) \quad (1)$$

International Journal of Scientific Engineering and Research (IJSER)
www.ijser.in
ISSN (Online): 2347-3878
Volume 3 Issue 2, February 2015

The resulting signal, $y(t)$ is referred to as the amplitude-modulated (AM) tone.

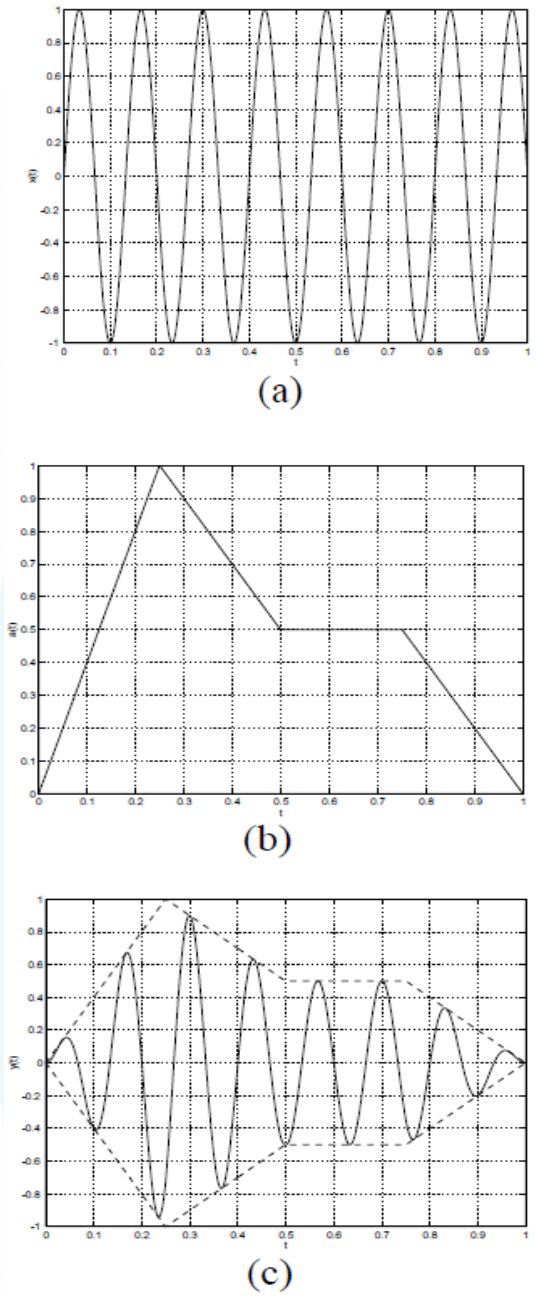


Figure 1: (a) Sinusoid, (b) ADSR envelope, (c) Amplitude modulated (AM) Sinusoid

2.2 Envelope Generator

As described earlier, the envelope will give the sinusoid a volume characteristic which as a first approximation, imitates that of a real instrument. The envelope values are stored as a single vector so that a simple element-by-element product between the sinusoid vector and the envelope vector yields the amplitude modulated sinusoid. The envelope is constructed one segment (A, D, S, and R) at a time. We approximate each segment with a simple exponential which rises or decays asympotically to the target value. This approximation then leads to a simple digital filter implementation (difference equation) which we are familiar with, whose response yields samples of an exponential curve. In addition, we allow for a gain parameter to control the speed at which the exponential reaches the target value. The difference equation is given by a single-pole filter,

a(n)=âg+(1-g)a(n-1)

where $a(n)$ are the envelope values, ˆ a is the target value, and g is the gain parameter.

C. Composer/Player Code

The final code segment generates sinusoids with the proper frequency and an ADSR envelope to amplitude modulate the sinusoid.

3. Result

3.1 MATLAB Code

Function to generate ADSR envelope

```
function[a]=adsr_gen(target,gain,duration)
fs=16000;
a=zeros(fs,1);
duration=round(duration./1000.*fs);
start=2;
stop=duration(1);
%attack phase
for n=(start:stop)
a(n)=target(1)*gain(1)+(1-gain(1))*a(n-1);
end
%Sustain phase
start=stop+1;
stop=start+duration(2);
for n=(start:stop)
a(n)=target(2)*gain(2)+(1-gain(2))*a(n-1);
end
%Release phase
start=stop + 1;
stop=sum(duration);
for n=(start:stop)
a(n)=target(3)*gain(3)+(1-gain(3))*a(n-1);
end
```

B.Function to generate sinusoid

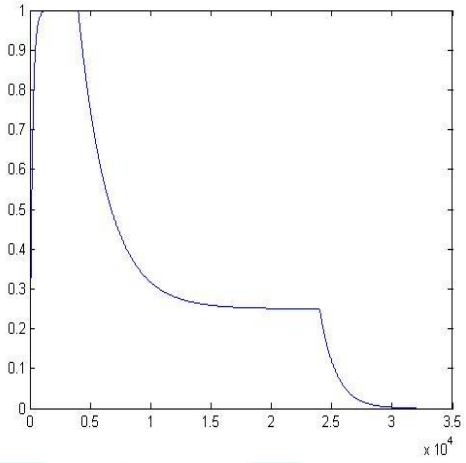
```
function[x]=singen(f,fs,N)
n=(0:N-1);
x=sin(2*pi*f/fs*n);
```

C. Composer/Player code

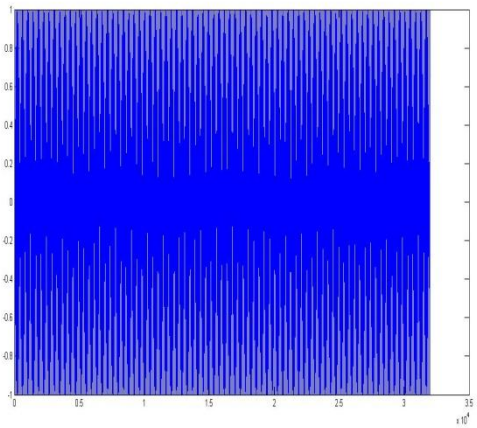
```
function [] = sound_play(f)
target=[0.99999;0.25;0];
gain=[0.005;0.0004;0.00075];
duration=[250;1250;500];
fs=16000;
tot_dur=floor(sum(duration)/fs);
[adsr]=adsr_gen(target,gain,duration);
figure(1)
plot(adsr);
x=singen(f,fs,length(adsr));
figure(2)
plot(x);
b=adsr.';
```

```
y=b.*x; % Modulate
wavplay(y,fs);
figure(3)
plot(y);
```

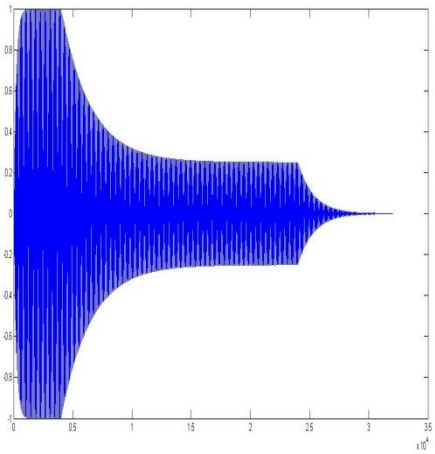
D. Output waveforms
(i.)ADSR envelope



(ii)Sinusoid wave



(iii)AM Modulated signal



4. Conclusion

In this paper we have developed an exercise in computer music. The exercise consists of three MATLAB codes which synthesizes a tone (sinusoid), generates an ADSR envelope used to amplitude modulate the tone, and built a song from the modulated tones.

5. Future Scope

Since the target application is computer music, the ideas can be extended to more general ideas in waveform synthesis.

References

[1] J. Chowning, "The synthesis of complex audio spectra by means of frequency modulation," *Journal of the Audio Engineering Society*, Sept. 1973, vol. 21, no. 7, pp.526–534.
[2] Matlab tutorials [online]. Available: http://in.mathworks.com/academia/student_center/tutorials/

Author Profile

Tony Mathew received the B.E degree in Electronics and Communication Engineering from Anna University, Chennai, India in 2012. He is pursuing his Masters Degree in Communication Systems at Christ University, Bangalore, India. He now with Bharat Electronics Ltd as Project Trainee.

Bimal M Abraham received the B.Tech degree in Electronics and Communication Engineering from M G University, Kottayam, India in 2012. He is pursuing his Masters Degree in Communication Systems at Christ University, Bangalore, India. He now with Bharat Electronics Ltd as Project Trainee.

Robin Scaria received the B.E degree in Electronics and Communication Engineering from Visveswaraiah Technological University, Bangalore, India in 2013. He is pursuing his Masters Degree in Communication Systems at Christ University, Bangalore, India. He now with Bharat Electronics Ltd as Project Trainee.