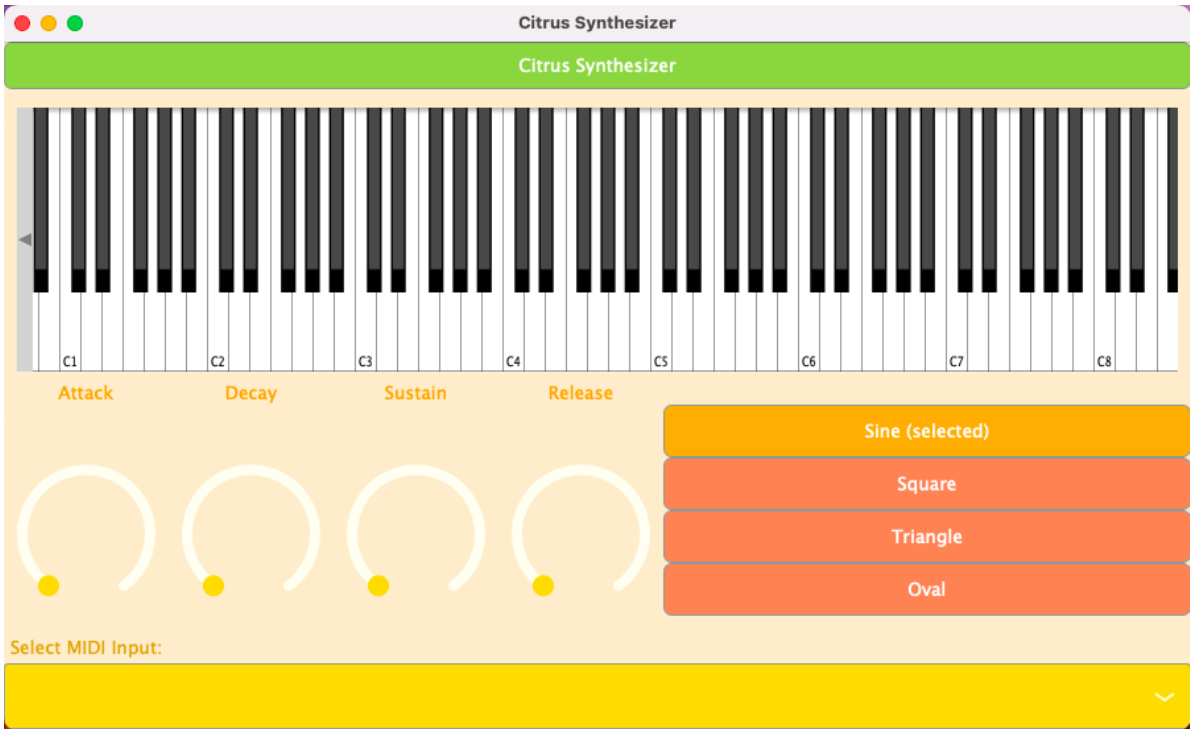
Envelope and Waveform Generation for Digital Synthesiser



Crystal McCay
Master of Science, Computer Science
Maharishi International University
CS 588 Master's Thesis Research
Dr. Catherine Gorini

Contents

1 Overview

1.1 Introduction

The topic of computer programming for music applications produces interesting and challenging problems for both musicians and software developers. At the intersection of these subjects, a surprising abundance of topics is available for technology professionals to address, due to the numerous and widely varying possibilities available to musicians for all stages of music practice and creation. The choices a musician might make about the tools he or she uses are highly personal, and they range from categories like instruments and hardware to audio software products. The tools chosen could be used to assist in a practice routine, simulate a physical instrument, extend instrument functionalities, or deliver a live performance. In many cases, the portability and flexibility of software provide appealing alternatives for musicians to utilize and explore.

1.2 Summary

The purpose of this research project is to explore topics in audio signal processing and produce a music application with a selection of musical parameters to manipulate sound. The name of the application is Citrus Synthesizer and it is designed with JUCE Framework for use on multiple platforms, including Mac, Windows, and Linux.

The application features an on-screen keyboard, four dials to control the shape of the sound produced, four buttons to select the waveform shape, and a dropdown menu from which users can select an external interface to control the on-screen keyboard. This application allows a user to create and shape sounds for use in a musical or entertainment context. If the app is controlled by an external musical device, then it can serve as an extension and expansion of that device’s sonic capabilities.

2 Foundations

Acoustics

The structure and functionality of this project’s application are centrally founded on musical terminologies and phenomena, so this section is dedicated to providing some essential definitions and discussions.

While it is difficult to precisely define music, it may be generalized that musical activities tend to produce sound, and to arrange some sort of relationship between sound and silence. Before the emergence of electrical and digital instruments, instrumentalists traditionally created sound through **acoustic** means, such as by blowing air into a wind instrument, striking percussion, or strumming strings. These actions produce **sound waves**, or disturbances that travel through a medium from one location to another location. The topic of acoustics encompasses the way that sound waves are propagated and affected by objects and environments.

Sounds possess a multitude of qualities and attributes that help to distinguish each sound's unique character. Some of the essential attributes in question include **frequency** and **amplitude** (**Figure 1**). Frequency refers to the number of times that a sound wave repeats itself per period of time, and it determines the pitch of a sound. As frequency increases, so does pitch. As for amplitude, this term refers to the maximum extent of a vibration measured from the position of equilibrium. This quality determines the loudness of a sound.

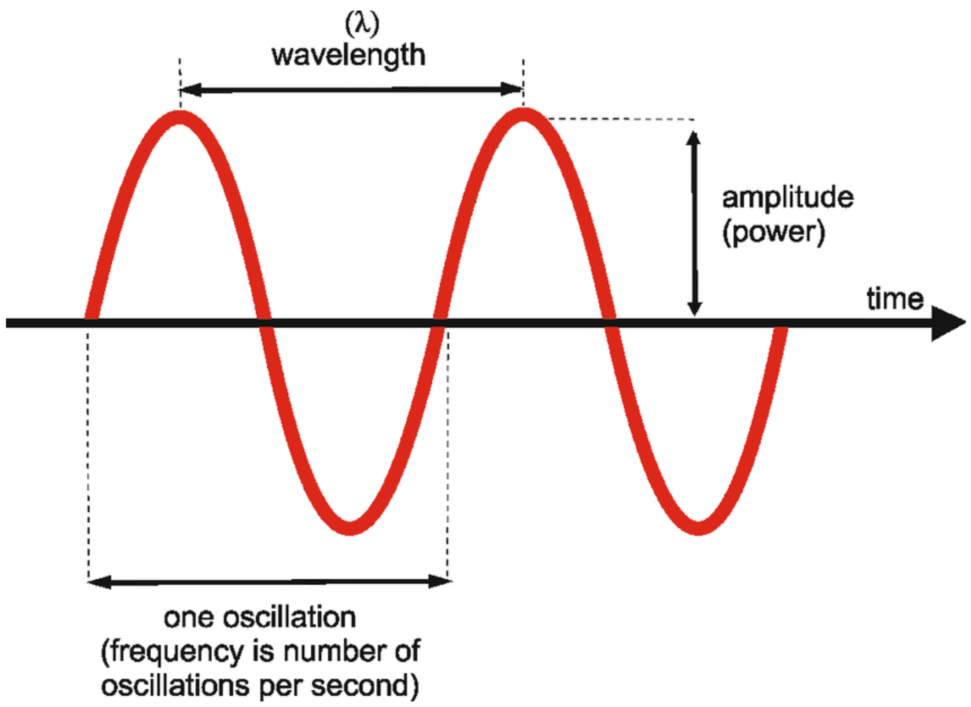


Figure 1

Many other physical phenomena influence the unique characteristics of a sound. The material, size, and playing techniques can all influence **timbre**, or the distinct color of tone that something produces. This is not specific to musical contexts; musical objects purposefully exploit the fundamental physical phenomena that give objects, textures, and voices their distinct sound qualities.

A noteworthy characteristic of acoustic sound waves is that they are **analog** or continuous. This means that for every instantaneous value on the time axis, there exists a corresponding frequency value. The methods of manipulating such waves can be performed mechanically. In the context of playing an instrument, this entails transformation and articulation techniques performed by the musician.

With this method of sound manipulation being an inherently physical phenomenon, limitations on an instrument's range of sonic capabilities are determined by its physical attributes, such as size, shape, and material. However, due to the nature of digital sound, which is explored in more detail following this section, some argue that the integrity and intimacy of acoustic sounds is unmatchable by those of digital instruments.

3 Synthesizer

A **synthesizer** is an electronic musical instrument that generates audio signals. There are a variety of waveform generation methods available to a synthesizer, some of which will be discussed in later sections.

Musical instruments began to converge with electricity in the early-mid 1900s, and this combination began to develop more widely around the 1960s. Many of these early synthesizers were electrical, so they employed analog signal processing techniques. Later, around the 1980s, digital versions of these instruments started to become more popularized (Kuehnl, 2015).

Many analog era controls set the standard for the parameters included in a typical processing library. According to Wiley (2011), “Many digital implementations of audio effects are in fact emulations of their analog ancestors.” One area that has emerged out of this evolution is digital filter design. The interest to replicate analog designs in the digital domain gives rise to virtual filter design. However, the expansive evolution of the digital domain has also inspired an alternative motivation for the design of effect technologies, which are not bound to the confinements of analog imitation, but which explore new grounds of distinct sound creation.

Digitally produced sounds offer the benefit of being highly versatile and even reprogrammable. One instrument can produce fifty or one hundred different possible sounds. While this may seem promising, many artists and listeners are firmly devoted to acoustic techniques.

4 Audio Processing

4.1 Concepts

Signal processing is a field that focuses on analyzing, modifying and synthesizing signals. It may be useful to consider two main divisions of this area into analog signal processing and digital signal processing. Analog signal processing involves the representation, encoding, or transformation of continuous representations of source signals, while digital signal processing achieves these operations on discrete time signal representations. **Figure 2** illustrates a continuous waveform in gray and a discrete representation of that same waveform in red.

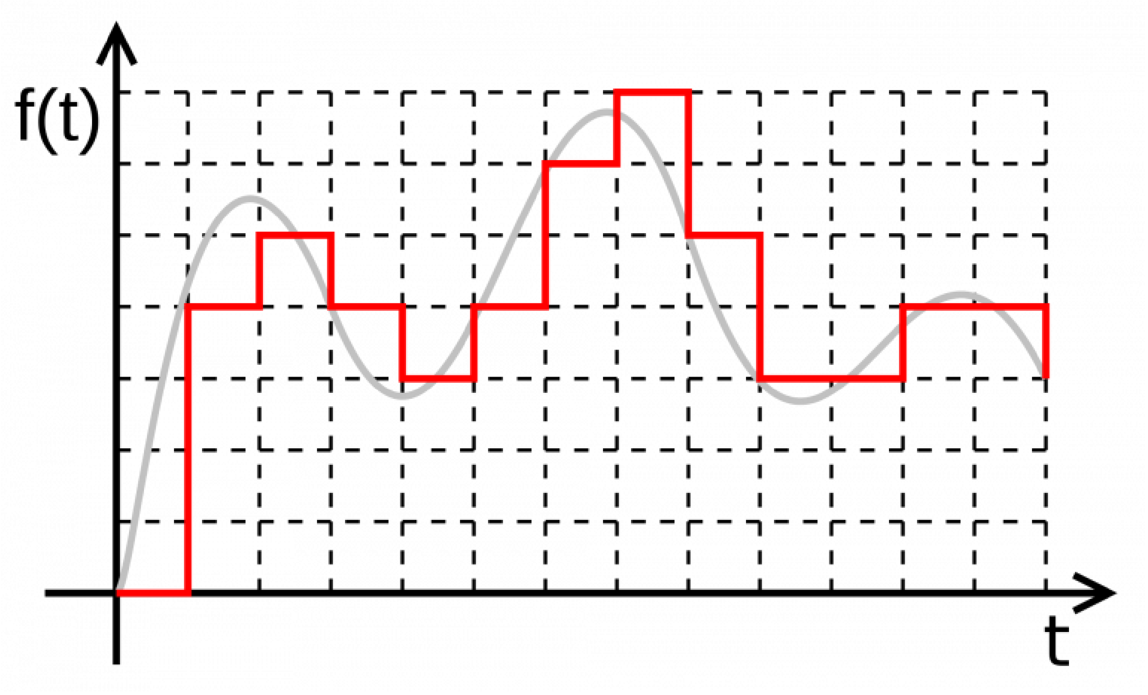


Figure 2

Contrary to acoustic waves, discrete time signals do not have a frequency value for every instantaneous value on the time axis. This is due to the nature of encoding sound information in a digital format, which requires discrete values to be recorded at regular intervals over the duration of the wave's propagation. The process of taking regularly timed instantaneous snapshots of a continuous waveform in order to represent it digitally is called **sampling**. Figure 2 illustrates this phenomenon. Each block on the x-axis represents a consistent interval of time at which the original signal's instantaneous value is recorded. Each instantaneous snapshot, which is simply a numeric value of the wave's y-axis location, is known as a **sample**. The speed at which signal samples are taken per unit of time is known as **sampling rate**.

A chief concern of sound processing is preserving the quality and integrity of the original sound. At the end of the processing chain, encoded samples will need to reconstruct the original wave that they approximated. In this process of **digital to analog conversion** (DAC), an analog wave will be constructed for output from the system. Because the previous stage of processing discarded portions of the original waveform and the output will be continuous, approximations of the discarded data will need to be provided to fill the time in between each of the samples. This process is known as **interpolation**.

It is critical to achieve the right sampling rate, and this must balance opposing concerns. As the sampling rate increases, the digital encoding more precisely approximates the original waveform. However, much processing will need to occur every second while delivering a stream of continuous audio data, which can be extremely demanding on memory and hardware. As the sampling rate decreases, less memory is required, but the reconstruction of the wave becomes unacceptably inexact, producing unwanted sounds or **noise** in the reconstruction signal. **Figure 3** depicts how a sampling rate that is too low can produce a wildly different output wave compared to an input. This occurrence is known as **aliasing**. The Nyquist Sampling Theorem specifies the constraints for minimum sampling rates in order to accurately represent an input signal.

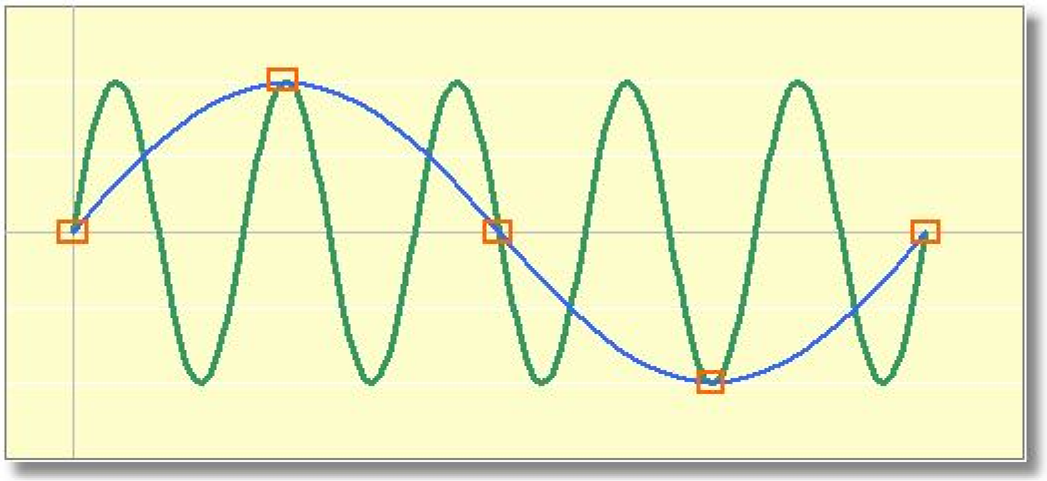


Figure 3

4.2 Uses

Acoustic principles about waveforms can be used in audio processing in order to represent, transmit, and deliver sounds with specific sonic qualities. Because the signals are represented discretely as a collection of individual numeric values, it is common to perform mathematical operations on them to produce desired effects. A simple case of this is amplitude modification, which can be performed through multiplication of all samples by a desired reduction factor, in order to lower the loudness.

Digital audio effects can also be achieved through signal processing algorithms, and this topic holds special significance in the discussion of sound character and musical qualities. When transformations are performed on analog inputs, a chief concern is to preserve the character and richness of the original sound, while also imposing a filter that produces high-quality output.

4.3 Waveform Synthesis

In cases where there is not an external audio source, there are a variety of methods available in order to create sound. Since a sound will not be coming from an outer source, the sound must be internally generated. Two commonly discussed methods of computer **sound synthesis** are frequency modulation synthesis and wavetable synthesis. Frequency modulation synthesis involves the manipulation of frequencies by other frequencies to produce more complex wave shapes (Wilson, 2022). This technique may be used to achieve engaging digital audio effects. Wavetable synthesis utilizes the sounds captured by an audio source as the source of output audio (Future Music, 2021).

The simplest strategy used to generate a pure tone is sine wave synthesis. This involves calculating the discrete time values of a sine wave based on the sine wave formula, and interpolating those values to construct an output sound wave. The desired pitch is achieved by instructing an **oscillator** to produce a sine wave at a specific frequency. An oscillator is simply an element that generates a signal.

Certain attributes of waves allow the output waveform to be calculated from a minimal amount of information. Each waveform repeats itself exactly after every **period** has elapsed. Therefore, continuously new information is not necessary to synthesize the wave for indefinite durations; only the discrete values from a single period need to be known, and the values can simply loop around every period for as long as the note is extended. Some synthesizers utilize this approach instead of continuously recalculating the same values every period.

A synthesizer may utilize any of these possible approaches to create sound sources. This project utilizes single cycle wave calculations in order to generate a sine wave and sine wave variations.

$$y = asin(bx + c) + d$$

To produce a digital representation of a sine wave, the audio source calculates the discrete values of a sine wave at a determined sample rate. The Nyquist sampling theorem specifies that the minimum effective sampling rate of a signal depends on its highest frequency.

Certain parameters must be recorded as the calculations for each sample is performed. A period of a wave may be represented by a circle, with one complete cycle around the circle corresponding to one wave period. The **(instantaneous) phase angle** of the wave refers to a specific location or point along the wave period. The **angle delta** refers to the amount by which the phase angle needs to increment in between each point that is recorded as a sample, and it depends on the wave frequency.

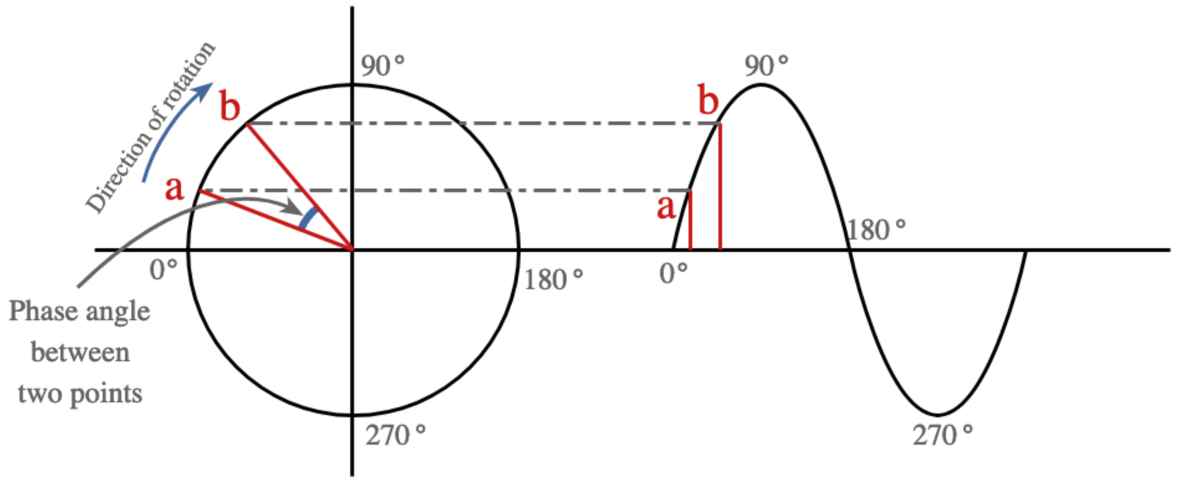


Figure 4

The angle delta is determined based on the current frequency. Because the sampling rate is expressed as the number of **samples recorded** per second, and the frequency is expressed as the number of **periods completed** per second, the two units need to be reconciled to express the portion of the wave period that needs to be traversed in between each sample. This process is called **frequency normalization**, and the unit conversion is represented by the following figure:

$$\frac{frequency}{sampling\ rate} = \frac{cycles\ per\ second}{samples\ per\ second} = \frac{cycle}{sample}$$

When this value is multiplied by the length of a complete wave cycle or 2π, we determine the new angle delta that is appropriate for this specific frequency.

Between each sample calculation, the instantaneous phase will be stored, and then incremented by the angle delta. This calculation may be performed repetitiously over the duration of a note's key press.

4.4 Envelope

Special challenges are introduced when synthesizing sound. In some cases, the aspiration may be to mimic an acoustic instrument. In such a case the digital source is tasked with recreating the depth and tone of a physical entity through digital means, as well as imitating physical phenomena like air resistance and initial force. **Figure 5** illustrates the character differences between a pure sine wave and the sound waves of physical instruments with distinct timbres.

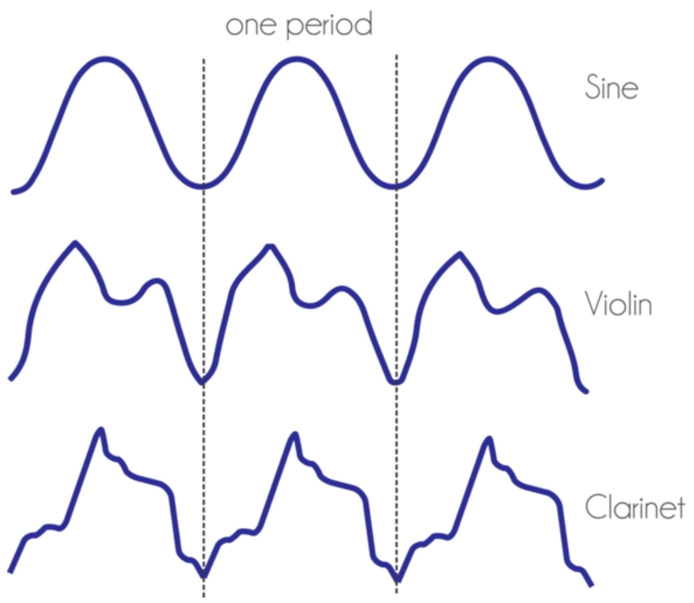


Figure 5

If the sound is being synthesized mathematically as it is in this project, the user may wish for some enrichment and extension to the simple tones produced by the generator. When a physical piano key is first struck, there is an initial spike in the sound produced, known as a **transient**. After this, the initial energy dissipates in two stages: a swift drop in amplitude relative to the transient, followed by a more prolonged reduction in volume as the sound dissipates. The player may also extend the middle portion of the note by engaging the foot pedals.

This sequence of changes from the sound's start to its end describes the **envelope** of a sound. It is made up of four stages: **attack**, or the initial strike of the note; **decay**, the descent from the attack; **sustain**, or the main expanse of the note as it resounds; and **release**, the descent from the sustain period into silence. Together these attributes may be referred to as ADSR. Not only does adding these characteristics produce more musical capacities to the instrument, but it also allows a spectrum of customization scenarios for the user to utilize for specific musical effects. **Figure 6** illustrates an example of envelope shape.

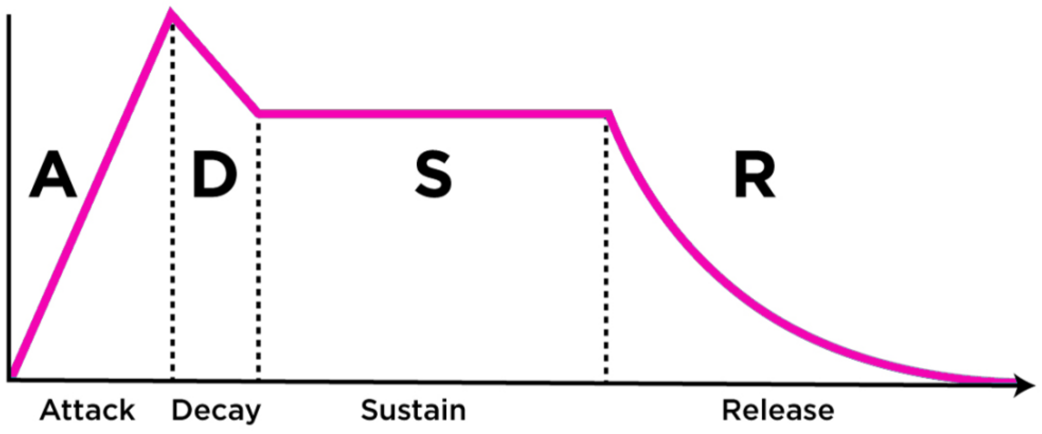


Figure 6

Envelope shaping is essentially varying the amplitude during specific phases of the original signal. It changes the duration that it takes for each stage of the envelope cycle to unfold and therefore modifies the intensity of the sound at different points along the time axis.

5 Design Summary

The principles, concerns, and attributes discussed in the previous sections are brought together in the following sections, and considered from the point of view of the application.

5.1 JUCE Framework

This application was developed with JUCE Framework, an open-source cross-platform C++ application framework for desktop and mobile applications. It is an especially potent tool for developing music and audio applications, offering extensive libraries for advanced audio functionality.

5.2 Application Summary

As mentioned in the introduction, the Citrus Synthesizer application features an on-screen playable keyboard which can be used to trigger sound generation events, as well as four waveform options, a slider for each ADSR property, and a selection menu for external keyboard controller inputs via USB connection.

The user can control the envelope duration and intensities by setting the ADSR parameters and alter the texture of the sound played by selecting the waveform shape that will be used by the synthesizer. Once the desired settings are applied, the user can play the keyboard with the chosen audio effects shaping the output sound.

In order to implement these functions, a chain of communication occurs between the events triggered by the keyboard and the app's audio components. The main application frame inherits the functionality of a JUCE Audio App Component, which must perform operations necessary to handle audio streams, including preparing the audio source, providing playback data for the hardware, and setting up audio channels. This component also maintains a reference to the audio source, which coordinates activity between a synthesizer and the keyboard in the main component. The relationship between these components and the flow of events from the on-screen keyboard to the sound generation step are explored in greater detail in the following section.

6 Interfaces

6.1 MIDI Interface

MIDI, or musical instrument digital interface, is a widely used protocol for encoding musical information in a digital context. As opposed to audio-based sound representations, MIDI messages only include instructions about note events, such as note on, note off, frequency, and intensity. It has many benefits, especially since it requires drastically less storage space than digitized audio alternatives (Heckroth, 1995).

For this project, the main application component holds a MIDI keyboard component. This component listens to screen mouse clicks as well as key presses on the computer keyboard. The state of the MIDI keyboard is monitored by a state object which accepts buffers of MIDI data collected from events on the MIDI keyboard.

6.2 Audio Source

The main component maintains a composition of the audio source for this application, which is a custom class inheriting the functionality of a JUCE Audio Source class. The Synth Audio Source is an intermediary between the main component and the actual sound generation. The Audio Source implementation accepts buffers of MIDI instructions and uses this information to control the sound source, which is a Synthesizer. The Audio Source maintains reference to a synthesizer, and upon construction it oversees the population of the synthesizer with its necessary components.

6.3 Synthesizer

The Synthesizer object requires two subclasses which function together to produce the audio output. These are the Synthesizer Voice and the Synthesizer Sound. The descriptions and data for a sound available to a synthesizer are kept in Synthesizer Sound. The rendering of a sound is performed by the Synthesizer Voice, and it specifies the occurrences that should take place at each phase of the sound's trigger, from start to release. The implementation of the signal generation calculations reside inside of the voice classes.

6.4 Synthesizer Voice

The Synthesizer Voice class is responsible for performing the wave generation calculations discussed in Section 4.

Preparatory calculations about the wave state are performed in the start note function. In this function, the start notification is also given to the ADSR phases, which provides the multiplication factor that is included in the wave function during the rendering step. This factor is multiplied by the instantaneous phase over a period of time in seconds, specified by the dial level placements in the user interface. This will reflect the envelope parameters onto the continuous stream of samples, sculpting the output levels as the note is pressed, held, and released.

Within the rendering implementation of the voice, the discrete values of the sine function are calculated and passed to each channel of the output buffer. This is also where the instantaneous phase is repeatedly incremented by the angle delta amount.

As Wiley discusses, “Signal processing algorithms usually process signals by either block processing or sample-by-sample processing… For block processing, samples are first collected in a memory buffer and then processed each time the buffer is completely filled with new data” (Wiley, 2011). The processing chain in this application utilizes versions of both approaches. On the input side, the MIDI buffer structure collects block sequences of MIDI messages, which are used in the synthesizer rendering, while the output buffer in the synthesizer voice is populated on a sample-by-sample basis according to the retrieved MIDI information.

To generate the user-specified waveform, the sine formula is used as a basis to calculate the wave shapes. The wave shapes available in this application version are sine, square, triangle, and oval. **Figure 7** shows some of the waveforms created by these calculations, excluding the final one. The square wave is a function of the discrete polarity of the sine wave. Any instant that a sine wave is positive, the square wave has an exact value of 1, and when negative, a value of -1. This produces the flat-lined crests and troughs. It is given by the formula $sgn\ (sin(x)\ )$.

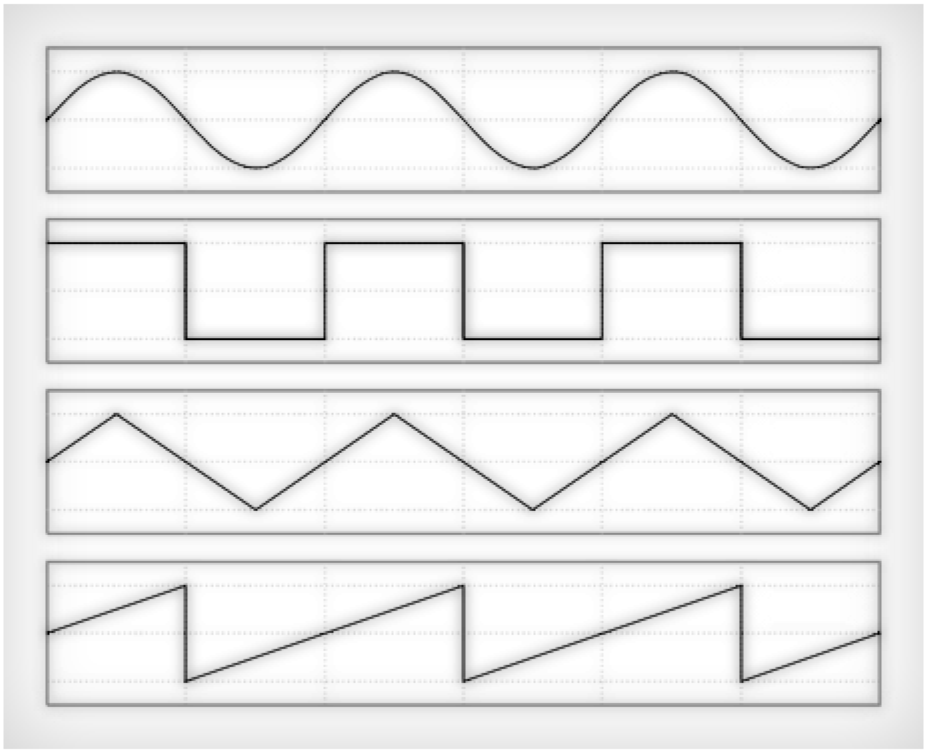


Figure 7

The triangle wave is produced by calculating the arcsine of the sine of x, and is given by the formula $\frac{2}{\pi} sin^{-1}(sin(x))$. Lastly, the oval wave was simply an experiment out of curiosity, and it was customly made to have the shape of an elongated semicircle. It was made with the formula $sin(x)^{3/7}$.

7 Consciousness Connection

As mentioned in Section 2, many audio processing principles are guided and defined by sound behavior in the physical world. The topics of acoustics, as well as psychoacoustics, are the bases from which digital processing concepts are extended. Audio processing for musical applications also inherits many concerns from the music domain.

At the root of all of these concerns is the study of waves in relation to human perception and environments. The fundamental action is the transmission of information and the fundamental concern is how it propagates from the source to the listener. When pondered from this layer, the seemingly diverse subjects that inform this interdisciplinary area all appear to be grasping at the same problem from different angles.

Whether performing music for an audience, speaking to a companion, or sending an audio message, the concern frequently is how clear, intelligible, and meaningful the transmitted information is.

This strikingly demonstrates the principle that infinite possibilities extend from the foundation of knowledge. All sound transmission techniques that were encountered during the course of this research were essentially reinterpretations of acoustic transmission. With the knowledge that a wave is a disturbance that travels through a medium, the type of wave could be converted, the medium could be changed, but the same underlying functionality could still be achieved.

With the digital generation of music, this accomplishment is especially impressive. It entails the internalization of the wave principles into a binary medium that functions as the source of information. The physical intermediaries have been surpassed, and the infinite possibilities of musical creation join with those of digital simulation. Simple mathematical calculations can be expanded into total musical expressions of emotions and moods. This derivation of a more complex product from simpler components is a clear statement of the whole exceeding the sum of its parts.

This entire discussion can be summarized as conscious self-referral. The grand basis of interpersonal sound transmission lies in psychoacoustics, or the way in which minds perceive and interpret sound. Many of the constraints and concerns of signal processing are determined by the outer boundaries of human perception. Designers, musicians, and signal processing engineers create sound products for human perception through human perception. At the community level, this is the concept of infinite self-referral in action.

8 Conclusion

The topic of computer programming for music applications continues to produce interesting and challenging problems for both musicians and software developers. In the completion of this project, many challenges and opportunities for expansion were encountered, and it is worthwhile to imagine the ways to extend what was learned.

One such opportunity exists with the waveform calculations used. As mentioned in Section 4, the cyclic nature of the sine wave offers an opportunity to dramatically reduce the number of sample calculations performed. A better approach is employed by wavetable synthesis, or the storage of the sample values for a single wave period in a data structure or sample buffer. This enables the processor to loop through these output samples instead of redundantly recalculating them.

There are also innumerable possibilities for wave generation beyond simple waveform functions. One such example is frequency modulation synthesis. A technique that will be intriguing to explore is the use of a low frequency oscillator (LFO) to produce effects such as tremolo. This type of effect can unfold over the body of the note instead of just at key phase markers as in the case of ADSR. The foundation built creating this project provides a starting point to more deeply explore signal processing algorithms and digital audio effects generation.

9 References

How Sound Cards Work
Tracy V. Wilson, 2022
How Sound Cards Work

What is wavetable synthesis?
Future Music, 2021
What is wavetable synthesis?

MIDI and Music Synthesis
Jim Heckroth, 1995
MIDI and Music Synthesis

DAFX: Digital Audio Effects
John Wiley and Sons, 2011
DAFX: Digital Audio Effects

History of Computer Music
Eric Kuehnl, 2015
History of Computer Music

Image Sources

Figure 1	ResearchGate
Figure 2	Electronics | Projects | Focus
Figure 3	Peranso
Figure 4	Electronics Notes
Figure 5	The Music Telegraph
Figure 6	Stikeys
Figure 7	AK Waveforms