Replicant Orchestra

Creating virtual instruments with software samplers

Aleksi Vuolevi

Bachelor's thesis
May 2018
Degree Programme in Media
Digital Sound and Commercial Music

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Media
Digital Sound and Commercial Music

ALEKSI VUOLEVI:
Replicant Orchestra
Creating virtual instruments with software samplers

Bachelor's thesis 63 pages
May 2018

Sampling is a widely utilized media production technique. It has a multitude of uses in music, sound design, movie and TV production, and in interactive media like games. By sampling source sounds like acoustic instruments, virtual representations of them can be created. Modern modular samplers break apart the sounds of instruments and provide access to the low-level variables that together create the whole. With the possibilities of that degree of control, a lot of new problems are introduced.

Sampling is a broad topic and information about it can sometimes be fragmented and superficial. The purpose of this thesis was to gain a deeper understanding about the process by assembling a comprehensive overview of the topic and examining the commonly used methods and problems they are meant to solve. The benefits and drawbacks of the various methods were researched.

Utilizing various sampling-related literary sources, the thesis analyzed the conventional techniques and workflows that have become prevalent in applications of sampling, and reviewed the associated terminology and other key concepts. Example use cases were introduced, pointing out practical problems that arise from them, and knowledge from closely related fields like audio recording, editing and mixing was applied to identify and analyze problems specific to sampling.

It was observed that the various methods were susceptible to compromises such as increase of memory requirements, complication of the initial sample recording process, and sound quality side effects such as phase cancellation, inconsistencies, unnaturalness and repetitiveness. The benefits and compromises of the methods also vary depending on the properties of the specific source sound.

Knowledge about the different methods and their associated benefits and drawbacks is important to optimize the production process of sample sets. Instrument-specific considerations need to be taken into account when deciding the best methods to emulate their character. Better evaluation of these factors can lead to better planning and pre-production, and can save a lot of effort and resources in the process.

Key words: sampling, sampler, virtual instrument, music technology

CONTENTS

ABBREVIATIONS AND TERMS

AIFF	Audio Interchange File Format. An audio file format developed by Apple Inc. Commonly used for storing uncompressed audio data. Default format for Mac OS
Bitstream	Sequence of bits which is used to represent information in digital form
Byte	a unit of digital information consisting of 8 bits
Chipmunk effect, the	A technique used to create cartoony high-pitched voices. The recording is done at a lower speed, and then sped up to pitch-shift the voice to a higher register
DAW	Digital audio workstation
DFD	Direct-from-disk audio streaming
Flam	Two almost simultaneous notes, which are still heard separately
Forte	Musical term denoting a dynamics level of ‘loud’
Generation loss	Loss of quality caused by subsequent codings or conversions of data
GUI	Graphical user interface
HDD	Hard disk drive
Key scaling	a parameter that is assigned key scaling follows the pitch of the incoming MIDI notes. Sometimes also called pitch tracking or key following
Legato	Articulation where the notes are smoothly connected together
Masking	A psychoacoustic phenomenon where the presence of a sound hinders or prevents the perception of another sound
MIDI	Musical instrument digital interface
One shot	this setting makes the group not react to note-off events, so the samples are always played back in their entirety. Useful for not having to hold notes down to hear the sustain part of drums for example.
Ostinato	Repetitive pattern of notes
Piano	Musical term denoting a dynamics level of ‘soft’
RAM	Random access memory

Spiccato	Articulation with a sudden attack and separation between the notes
SSD	Solid state drive
TDM	Time Division Multiplexing, an audio processing plugin architecture released in 1993. It utilizes external hardware processors. Developed by Digidesign (merged with Avid in 1994)
Transient	A distinct attack part of sounds, which is usually seen as a peak in the overall amplitude
VI	Virtual instrument
WAV	Waveform Audio File Format; an audio file format developed by Microsoft and IBM. Commonly used for storing uncompressed audio data. Default format for Microsoft Windows

1 INTRODUCTION

Sampling is an important technique in various media production subcategories such as music production, sound design, musical composition for film and TV productions, and in interactive media like games. Along with synthesis, it's one of the main means of electronic sound production. By capturing samples of acoustic instruments and playing them back according to a set of rules corresponding to musical information, it can create virtual representations of the original sources. The main advantage of this method is that it can very realistically imitate the many tonal characteristics of the source sounds. (Hosken 2011, 195, 233.) Sampling can be used with any kinds of sounds - not just musical instruments - and that makes it useful in a wide range of applications.

Samplers are either hardware devices or computer programs that - using the recorded samples - translate musical information like note sequences or performance data into sounds according to their programming. Modern software samplers have evolved into complex programs with a lot of advanced features. Their user manuals spanning hundreds of pages describe the various features and how to use them with the provided interface.

However, they don't usually give a good overview of the whole programming process, but instead focus on the specific features of the device/program. Some other sources like books and articles provide a wider perspective on the topic, but due to its broad scope, parts of the information are sometimes omitted, or only specific parts of the whole are covered extensively.

Additionally, sources like sampler manuals rarely explain what exactly are the problems the provided methods are trying to solve and what's causing them in the first place. Other important aspects that are often omitted in the documentation are the benefits and downsides of these methods.

The purpose of this thesis is to provide an overview of the process of programming samplers and review the associated technical concepts. It seeks to clarify the exact nature of the problems that the methods in sampling have been developed for, and what is causing them. The methods are further investigated in order to find out their respective benefits and possible drawbacks and compromises. It also aims to determine the possible use cases for the different methods.

2 FRAME OF REFERENCE

2.1 The importance of sampling

A number of advantages are gained by using virtual instruments. They offer a huge variety of sounds at a minimal investment: there's no need to own the actual acoustic instruments or other sound sources. It's not necessary to know their specific playing techniques either: MIDI controllers and sequencers provide familiar interfaces with which to create musical information which is translated by the sampler to the sounds produced by the instrument's specific playing techniques.

Samplers can provide instant access to a sound palette that is generally beyond the reach of most people: expert performances of whole orchestras, drum kits and all kinds of exotic instruments and sounds, many recorded in world-class acoustic environments and engineered by professionals with top quality recording equipment. (Dunkley & Houghton 2011; Stewart 2017a.)

These are significant considerations for me personally, and in my projects I use a lot of drum, piano and orchestral samples for example. Without samplers I would not have access to these sounds at all. This reliance on the sampler technology was an important factor for the desire to gain a deeper understanding of its details, and therefore I ended up selecting it as the topic for my thesis.

2.2 Terminology and definitions

The term **sampling** has at least three different meanings in the context of audio, according to Hosken (2011, 233):

At the most fundamental level, it refers to the process of **capturing and measuring analog signals in order to create a digital representation of the waveform**. This is also known as analog-to-digital conversion (ADC). (Russ 2009 56-57.) The signal is measured (sampled) at a constant interval, referred to as the sampling rate. The amplitude of an analog AC signal is measured at every sampled point, and the result is quantized to a set of discrete values, the resolution of which is determined by the number of bits that are used to represent them (referred to as the **bit depth**, not to be confused with **bit rate** which in the context of audio denotes information density in file formats). (Russ 2009, 59, 64; Huber 2007, 165; Hosken 2011, 82-83.) This format of representing the continuous analog signal in digital form is called **pulse-code modulation** (PCM) (Russ 2009, 58-59). This data can be then read, manipulated, replicated and reproduced with a computer.

Another meaning for sampling stems from the practice of **recording and re-using audio from existing sources** like songs, movies or any other media, in order to create new compositions or other productions. The sampled content can be anything from a single drum hit that will be reused in a new composition, to longer segments of songs (typically the instrumental breaks) that can be chopped up and rearranged like a collage, to sound effects or spoken word samples for example. (Howell 2005d; Couchman & Dearcangelis 2016.)

Third meaning for sampling denotes the process of **recording sounds from musical instruments or other sources, and using these samples to create a virtual instrument**. These **virtual instruments** can be sequenced, or played in real time with a MIDI controller, and they can mimic the performance and tonal characteristics of the original sound source. They are widely used for music and other media like movies and TV, but sampling can use any kinds of recorded sounds, which is useful in a wide range of applications in media production. (Hosken 2011, 195.)

In contrast to the act of recording, as in capturing complete performances, sampling-related recording usually means capturing audio snippets like single notes, phrases, or other musical gestures or noises. The recorded samples are loaded into a tool called **sampler** that can be either hardware or software. (Hosken 2011, 198.)

A **sampler patch** is the collection of data that represents the virtual instrument. It utilizes one or more samples from the source sound, and defines how the samples are triggered by MIDI notes, and how the resulting sound is modified by assignment of modulation and effects. (Hosken 2011, 241.)

Sampler patches are usually a part of a larger collection, tied together thematically by instrument type or ethnicity, or musical genre for example. These collections are called **sample libraries** (Sweetwater 2014).

2.3 History of sampling

2.3.1 Pioneering devices

The history of sampling extends back to the late 1940s to the conception of a device by an American inventor named Harry Chamberlin: the tape replay keyboard. The invention - which he decided to call simply the ”Chamberlin” - was a pioneering instrument which introduced the idea of using a keyboard manual to play back prerecorded sounds from a variety of conventional instruments in real time. (Howell 2005a.) As Mr. Chamberlin remarked in an interview for the Crawdaddy! magazine in April 1976:

> If I can put my finger down and get a Hammond organ note, why can't I pick a guitar note or trombone note and get that under the keys somehow and be able to play any instrument? As long as I know how to play the keyboard, I could play any instrument! (Epand 1976)

The way tape replay keyboards work is to essentially have a tape machine assigned to each of the keys of the instrument. When the key is pressed, a pad under the key presses a tape into contact with a replay head, and a pinch roller presses another part down against a capstan mechanism, which starts pulling the tape onward. When the key is released, a spring pulls the tape back, resetting its position for the next note. (Russ 2009, 187.)

The recorded sounds would typically be about 8 seconds long, and if the key was held down longer than that, the tape would run out and the note would just stop sounding. Due to the abundance of moving parts in the mechanism, the tape replay keyboards aren't very resilient, and require careful calibration. The keyboard of the earliest versions of the instrument had a limited range of just under three octaves (G2-F5), or 35 semitones. (Howell 2005a; Howell 2005e.)

The novel approach of the instrument garnered considerable interest in musicians and other music industry affiliates when Mr. Chamberlin brought the first model of the instrument to an annual conference of National Association of Music Merchants (NAMM), which was held in Chicago that year in 1956. This led to a successful launch of the product and the beginning of production of commercially available Chamberlins. (Epand 1976.)

A few years later a salesman employed by Chamberlin went missing with a showpiece model of the instrument (Epand 1976). It was later discovered that the salesman had travelled to England and provided the technical details of the tape replay keyboard to an engineering company called Bradmatic Inc, starting a partnership with them and Eric Robinson Organisation. Bradmatic embraced the design and started developing their own version of the instrument. The Eric Robinson's company was renamed to Mellotronics Ltd, and in 1963 it released the first incarnation of the new instrument, dubbed the Mellotron Mark 1. (Reid 2002.)

This was eventually discovered by Harry Chamberlin, when Mellotronics started looking into distribution in the United States, where Chamberlin had patents for the device. Legal proceedings followed, and the end result was a settlement between Chamberlin and Mellotronics in the form of a royalty deal. (Epand 1976.)

In the end though, Mellotron was the instrument that would be remembered, likely because they ended up being used on albums of acclaimed English acts such as the Beatles, Genesis and King Crimson, among others (Howell 2005a).

Eventually, advances in music technology led to more modern instruments like synthesizers becoming increasingly affordable and commonplace, and caused a shift towards the signature sound of the Eighties that we recognize today. This led to a decline in the popularity of the innovative tape replay keyboards, but at the same time, the emerging technology of audio digitalization brought about another innovation: digital sampling.

2.3.2 Shift to digital

The technology that became known as PCM was invented in 1937 by Alec Reeves, who was working at the research laboratory of International Telephone & Telegraph. Subsequent advances in technology - most notably the invention of the transistor in 1947 - made practical applications of PCM possible, and the first commercial digital recorder (Soundstream) was released in 1975. (Reid 2002.) During the seventies the experiments with sampling using the digital technology increased, and while most of the implementations weren’t quite ready for the spotlight in a commercial sense, many individual inventors discovered the power of digital audio.

Some early adaptations of the technology include devices like EMS MUSYS (1969) and Computer Music Melodian (1976), which both utilized Digital Equipment Corporation’s PDP-8 minicomputers (Gardner 2012; The Pennsylvania Gazette 2013). The Melodian - an invention of Harry Mendell - was adopted by Stevie Wonder to be used on the soundtrack ’Journey Through "The Secret Life of Plants"’ which was released in 1979. An example of the uses of the instrument includes utilizing sampled bird sounds to create a melody line. The music technology manufacturers were finally catching up to the rising popularity of sampling. Yamaha acquired a license to the technology behind the Melodian for example, and used it to manufacture computer chips that would be used in a wide range of sampling-related products. (The Pennsylvania Gazette 2013.)

The end of the seventies and the early eighties saw the technology entering the commercial market with pioneering units like the Publison DHM 89 B2 digital sampling delay (1978), or the legendary workstations Fairlight CMI (1979) and the Synclavier II (1982-1983). Their commercial success was limited though, because at this stage the pricing of the most of these devices was still prohibitively expensive (for example the Fairlight cost over $25,000). (Solida 2017.) During the 1980s several new products were introduced which slowly brought the price point down into the reach of an average musician’s budget.

Some of the more notable ones of that era include the first reasonably priced ($1695) sampler Ensoniq Mirage (1985), and the drum samplers SP-12 (1986) and its successor SP-1200 (1987). The SP-1200 became a ubiquitous instrument in the late 80s to early 90s hip hop music. (Solida 2017; Couchman & Dearcangelis 2016.)

Akai was another manufacturer of popular samplers in the 80s and onwards. They began with releasing their S-series samplers, starting off with S612 (1985) and improving upon it with S900 (1986) and S1000 (1988), the latter of which featured CD-quality sampling. Akai also teamed up with the drum machine designer Roger Linn to create the famous MPC60 (1988). (Howell 2005a; Solida 2017.)

In the 90s companies continued to release hardware-based samplers, but their relevance was reducing as personal computers became more prevalent: new powerful ways of working with audio were introduced, with less requirements to maintain external hardware. (Howell 2005a.)

2.3.3 Advent of software

During the 90s audio production workflow gradually integrated the software equivalent for the traditional multitrack tape machines, analog mixing consoles and hardware instruments and sequencers: the digital audio workstation (DAW) (Russ 2009, 400-401).

Another significant development was the release of the VST (Virtual Studio Technology) specification as an open standard by Steinberg in 1996. It allows emulating hardware-based recording tools using digital signal processing techniques powered by the computer's own CPU - as opposed to external signal processors like some earlier plug-in technologies, e.g. TDM - and incorporating them in the DAW environment. This opened audio plugin development to third parties, as encouraged by Steinberg themselves. (Johnson & Poyser 1996; Russ 2009, 385.)

Consequently, software implementations of samplers were realized, and they've made possible numerous benefits compared to the hardware counterparts. The interface for accessing the settings and other functions was no longer restricted to a couple of knobs and buttons and a small LCD screen made to fit into a rackmount piece of gear, instead replaced by the convenient and familiar computer screen and mouse. (Russ 2009, 401.) Also, it's possible to use the DAW's advanced tools do traditionally fiddly tasks like sample editing. Automation for the sampler's parameters can be written in the sequencer. Consolidating the workflow to the DAW can reduce the number of needed analog-to-digital and digital-to-analog conversions, thereby theoretically improving sound quality.

Storing, sharing and selling sample content is facilitated by computer-integrated digital storage and the Internet.

On the other hand, hardware still has some of its advantages; one of them being a familiar interface, with hands-on controls. The layout of the early drum samplers like Linn 9000 and Akai's MPC line has been often imitated in software interfaces, and in turn hardware MIDI controllers have been manufactured which are meant to control that software, retaining the grid layout of pad triggers that was first employed in the hardware drum machines.

Many other best practices ended up being re-implemented in the software versions, but one thing they didn't capture was the characterful sound quality of some of the hardware devices. Many of them had less-than-perfect AD- and DA-converters, and this gritty lo-fi sound was actually preferred by many musicians and producers, and was used as an effect (Couchman & Dearcangelis 2016). In some later software samplers this degradation effect has been added as an optional feature.

Some notable software samplers from the early era include Nemesys Gigasampler (1998), Emagic EXS24 (2000), Native Instruments Kontakt (2002) and Steinberg Halion (2002). Nemesys was bought by Tascam in 2001, and the Gigasampler (later versions known as Gigastudio) line of products was eventually discontinued. Emagic was acquired by Apple in 2002, and their DAW, Logic, became Apple's Logic Pro which integrated the EXS24 (version mkII since 2002) as its default sampler. As of 2018 Kontakt and Halion are also in active development.

Nowadays, the software sampler is an important and often used tool in media production. Some type of sampler plugin comes bundled with the majority of DAWs, and many more are available as independently developed software plugins.

3 SAMPLER ARCHITECTURE AND MODULE HIERARCHY

3.1 Overview of sampler architecture

Software samplers come in many different forms. There are some very simple ones that are designed to load and manipulate only one audio sample at a time (figure 1).

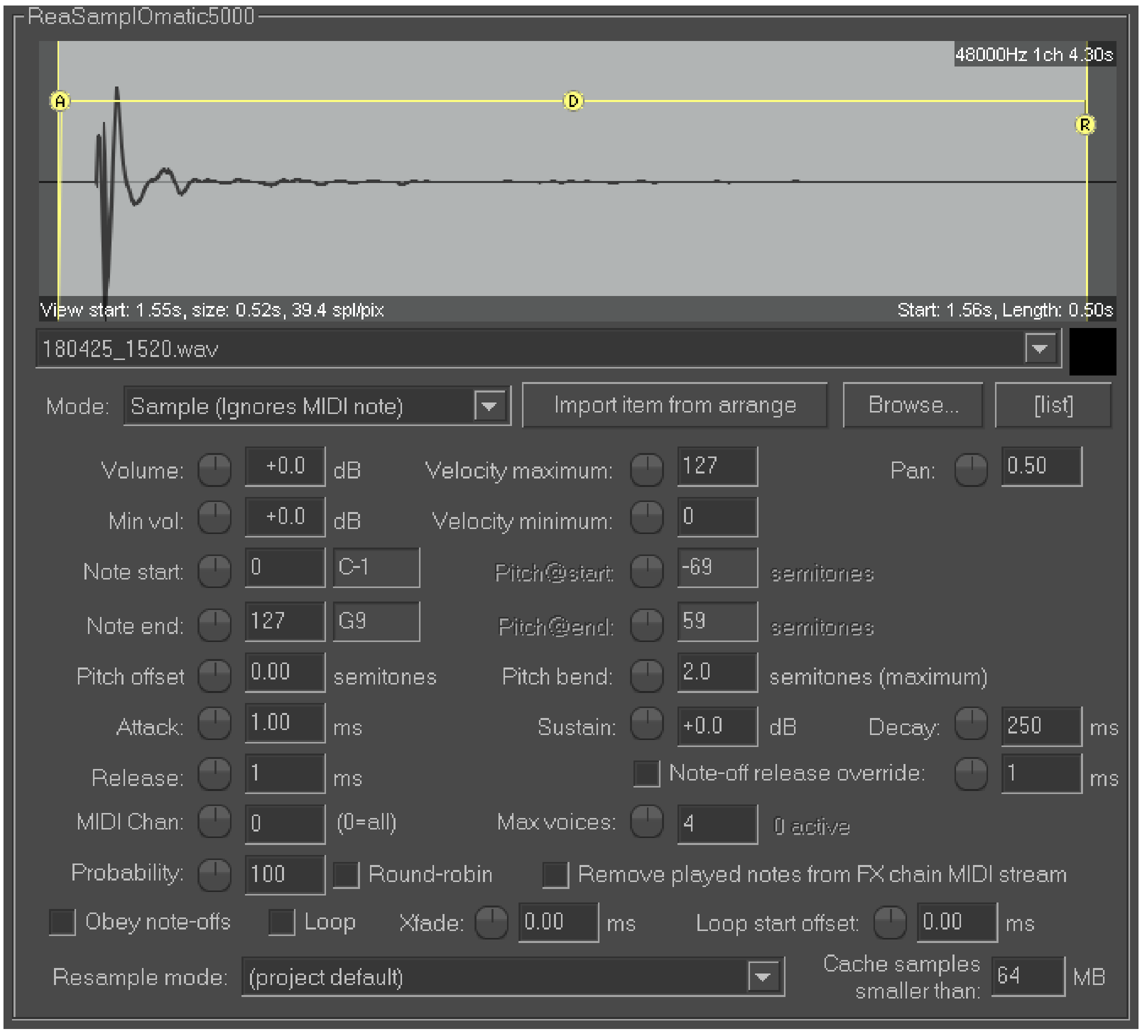

ReaSamplOmatic5000
48000Hz 1ch 4.30s
View start: 1.55s, size: 0.52s, 39.4 spl/pix
Start: 1.56s, Length: 0.50s
180425_1520.wav
Mode: Sample (Ignores MIDI note)
Import item from arrange
Browse...
[list]
Volume: +0.0 dB
Velocity maximum: 127
Pan: 0.50
Min vol: +0.0 dB
Velocity minimum: 0
Note start: 0 C-1
Pitch@start: -69 semitones
Note end: 127 G9
Pitch@end: 59 semitones
Pitch offset 0.00 semitones
Pitch bend: 2.0 semitones (maximum)
Attack: 1.00 ms
Sustain: +0.0 dB
Decay: 250 ms
Release: 1 ms
Note-off release override: 1 ms
MIDI Chan: 0 (0=all)
Max voices: 4
0 active
Probability: 100
Round-robin
Remove played notes from FX chain MIDI stream
Obey note-offs
Loop
Xfade: 0.00 ms
Loop start offset: 0.00 ms
Resample mode: (project default)
Cache samples smaller than: 64 MB

FIGURE 1. Cockos ReaSamplomatic5000 (Vuolevi 2018)

There are some that get their inspiration from the hardware drum samplers, offering a 2-dimensional matrix of cells, where one can load samples of different drum kit pieces for example, and conveniently access them using the familiar interface (figure 2). Many hardware MIDI controllers also have the same kind of layout, so they work very well with these kinds of samplers when real-time performance and MIDI sequence recording is desired.

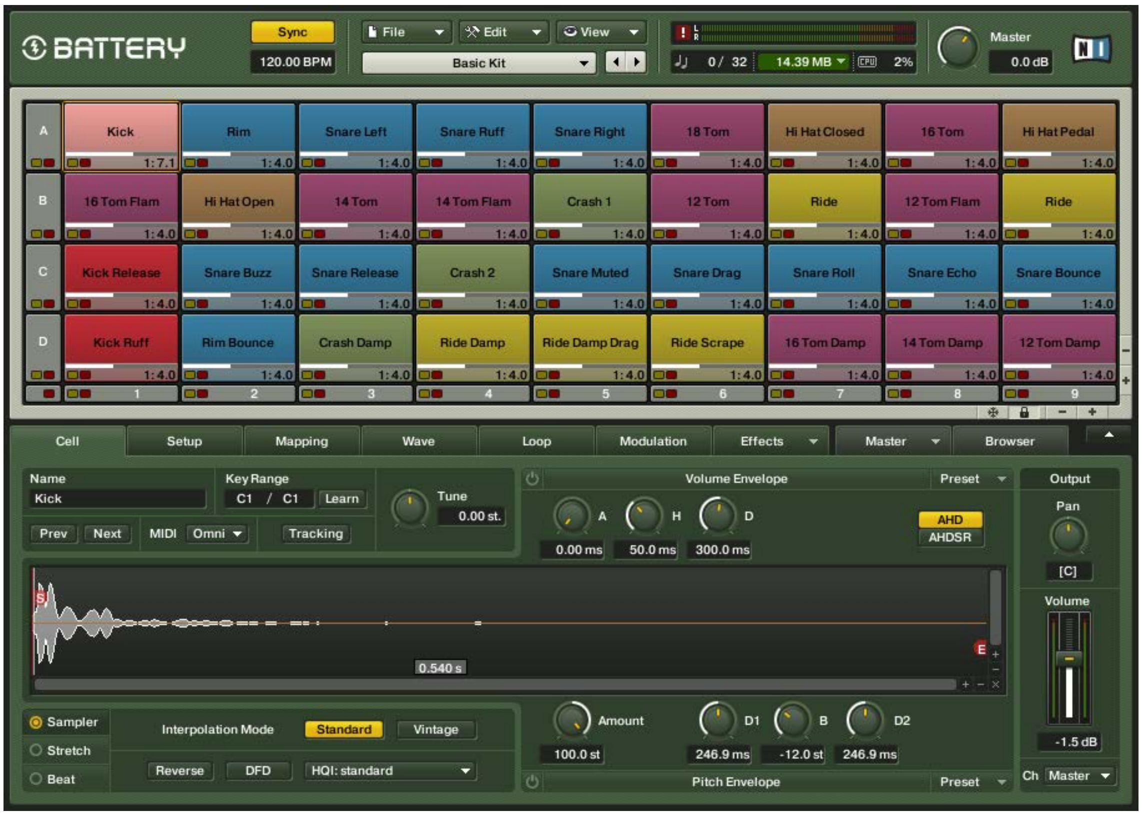


FIGURE 2. Native Instruments Battery 3 (Vuolevi 2018)

A very common task today in music production is using sampled material to reinforce and replace the sound of studio-recorded drums and live drumming (Dunkley & Houghton 2011). Samplers are well suited for this purpose, and some of them are solely dedicated to playing back drum samples (figure 3).

Drum samplers can also be used as the sole source of drum sounds for budget productions and demo purposes for example, as the effort and cost of renting a studio to mic up a real drum kit can be a problem. Programming the drums in a sampler can be a workflow benefit too as a quick way to compose, arrange and fine-tune the drum parts. (Messitte 2018.)

Drum samplers can offer deep access to adjusting parameters that are specific to drums as an instrument, like controlling the amount of bleed from the drum kit pieces in the various microphones that were used to record the sample set (FXpansion Audio 2017). This is an especially powerful feature of sample-based drums: with a live-recorded kit, microphone bleed can be an undesirable effect and very hard to eliminate (Accusonus 2017).

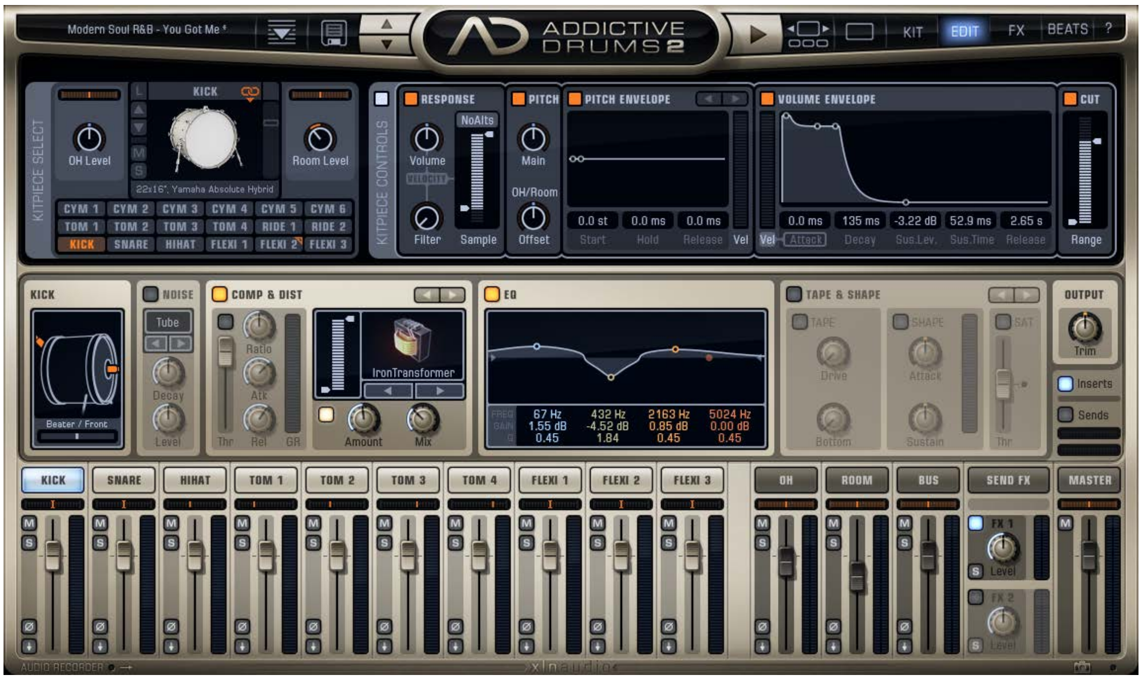
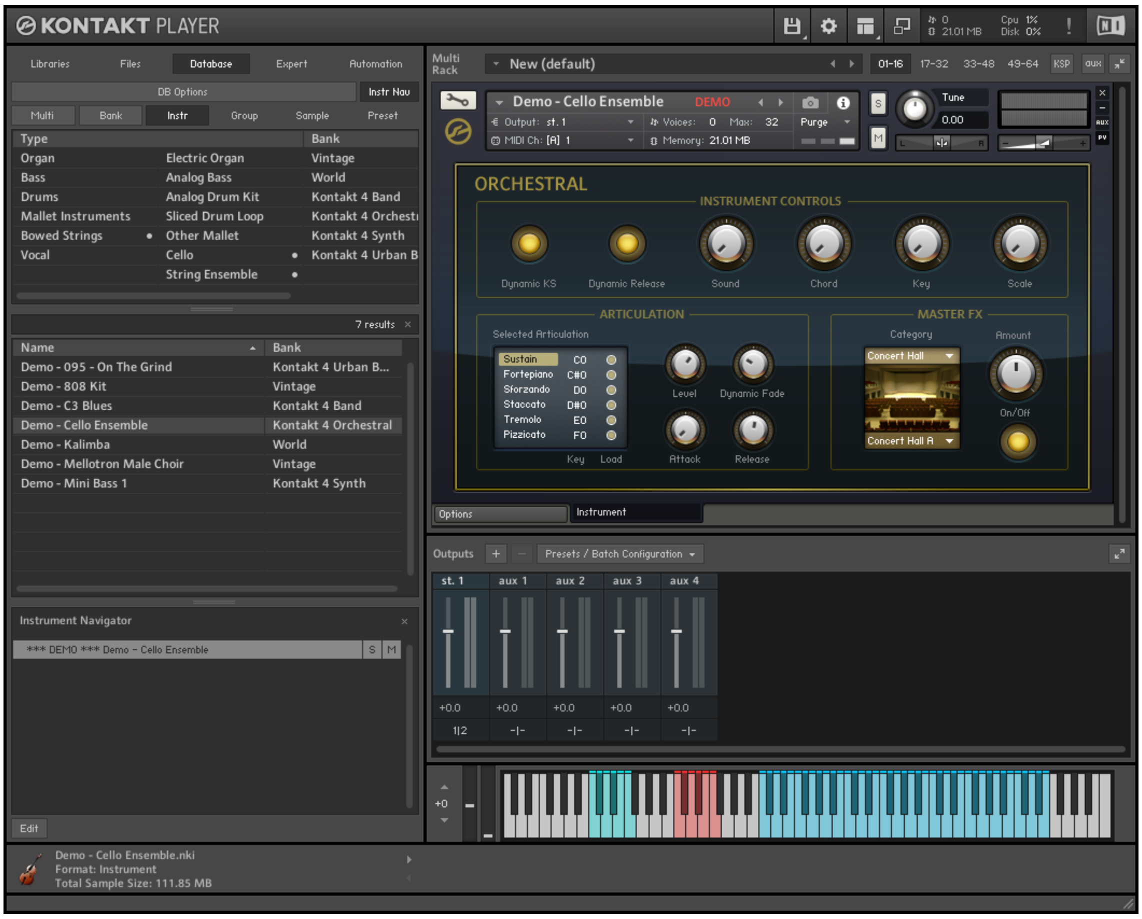


FIGURE 3. XLN Audio Addictive Drums 2 (Vuolevi 2018)

Modular samplers are the most versatile type of sampler software.

Their feature set supports building virtual instruments that can emulate the sound, performance, articulations and other idiosyncrasies of the sampled sound source with convincing accuracy. The modular structure supports creating even very comprehensive patches, like entire virtual orchestras. (Stewart 2017a.)

Modular samplers with scripting capabilities grant further control over the instruments' parameters and interface. The visual appearance of the control section of the instrument can be customized. Scripting can be used to implement high-level controls over complex parameter interactions with a few macro parameters that are exposed to the user. It can be used to create articulations, harmonization and other performance effects through processing incoming MIDI messages, and it can control bundled DSP plugins to further customize the sound of the instruments. (UVI 2018.)



Type		Bank
Organ	Electric Organ	Vintage
Bass	Analog Bass	World
Drums	Analog Drum Kit	Kontakt 4 Band
Mallet Instruments	Sliced Drum Loop	Kontakt 4 Orchestr
Bowed Strings •	Other Mallet	Kontakt 4 Synth
Vocal	Cello •	Kontakt 4 Urban B
	String Ensemble •	

Name	Bank
Demo - 095 - On The Grind	Kontakt 4 Urban B...
Demo - 808 Kit	Vintage
Demo - C3 Blues	Kontakt 4 Band
Demo - Cello Ensemble	Kontakt 4 Orchestral
Demo - Kalimba	World
Demo - Mellotron Male Choir	Vintage
Demo - Mini Bass 1	Kontakt 4 Synth

Selected Articulation	Key
Sustain	C0
Fortepiano	C#0
Sforzando	D0
Staccato	D#0
Tremolo	E0
Pizzicato	F0

Demo - Cello Ensemble.nki
Format: Instrument
Total Sample Size: 111.85 MB

FIGURE 4. Native Instruments Kontakt 5 (Vuolevi 2018)

Modular samplers from various developers differ somewhat in their architecture, but they generally employ a hierarchical structure (figure 5) (Walker 2002; Baer 2018).

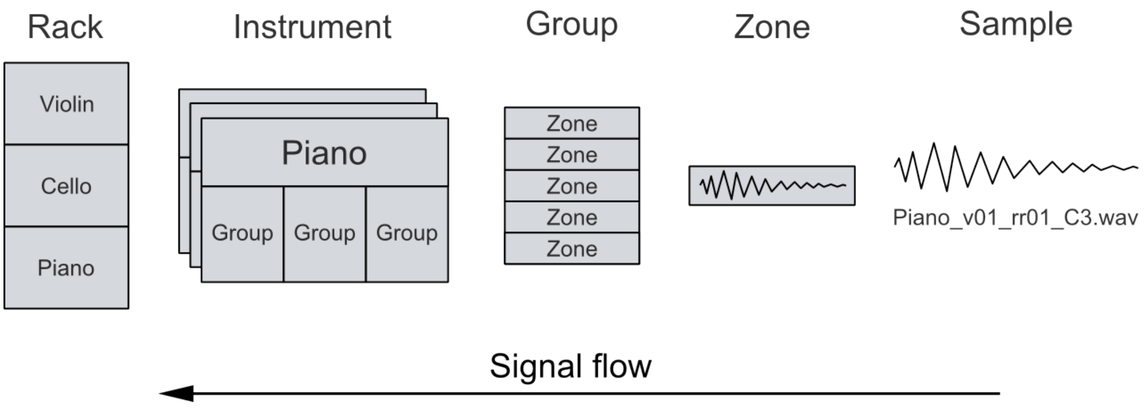


FIGURE 5. Sampler module hierarchy (Vuolevi 2018)

3.2 Audio samples

At the most fundamental level are the recorded audio samples themselves, which contain raw audio data (Native Instruments 2017, 33). The audio files can also contain metadata: for example cue points which can be used to determine **loop point** positions (Brazil 2002).

The files are accessed through the sampler's interface, which usually supports some basic processes like various editing operations and loop point assignment and modification. These alterations can generally be done non-destructively, and they are instead stored in the instrument's patch data. Samplers usually offer a possibility to render them into the original wave files though, which can simplify the programming process further on. (Native Instruments 2017, 180-181.)

3.3 Zone/keygroup

The next element in the hierarchy is the container for an audio sample. The terminology differs here depending on the software's developer, but a common term for it is the **zone**.

They reference the actual audio files on the file system, and are used to map them to MIDI data to define triggering and tuning conditions (Native Instruments 2017, 33). They can contain some basic information like tuning and loop point data, depending on the implementation.

3.4 Group/layer

Next up is the **group**, which encompasses one or more zones.

The group allows configuring the properties of multiple zones together (Steinberg 2017, 217). Grouped zones can share settings like routing and effects, triggering conditions (e.g. playing the sample on a note-off event instead of note-on) and polyphony settings (Native Instruments 2017, 153). An example use would be grouping all the velocity layers of certain MIDI note together, for example to map out a drum kit with kick drum on a C note, a snare sound on D, and so forth, and then defining routing and adding effects to the individual kit pieces.

The groups can have voice/choke settings, which define polyphony count and monophonic behaviors. Monophonic modes can be used to better represent naturally monophonic instruments like woodwinds, preventing two notes from sounding at the same time and facilitating playing techniques like trills.

Monophony features like choking can be used for deadening a playing sound when a new sound sharing the same voice group is triggered. This is used for example to emulate the property of the hi-hat, where closing the pedal cuts off the ringing of its cymbals. Also, lowering the polyphony amount can help with high CPU usage by cutting off decaying sounds with lower volume when new voices are activated. (Native Instruments 2017, 159-160; Steinberg 2017, 89.)

The groups can also be used to define a pool of samples which are used for sequential and randomized triggering features. The aim is to reduce repetitiveness in the case where successive notes in the incoming MIDI sequence would otherwise trigger the exact same sample multiple times in a row. (Steinberg 2017, 90; Native Instruments 2017 161-163.)

3.5 Instrument/program

A container module for all the groups and zones belonging to an instance of a virtual instrument is called an **instrument**, or sometimes, a **program**.

It has basic parameters like volume and pan so it can be mixed inside the sampler engine with other simultaneously loaded instruments (Steinberg 2017, 82). Other possible options include tuning, transposition of incoming MIDI and limiting the instrument's key and velocity ranges, to split the keyboard range among multiple instruments for example (Native Instruments 2017, 67, 77). Modular samplers often provide insert/send effects and modulation for the instruments (MOTU 2013, 62).

Instruments have a setting for the MIDI input channel and for audio output channels which can be used for internal routing, mixing and effects assignment (Native Instruments

2017, 37). Besides complete sample sets of real instruments, the instrument elements can represent the sample sets of their various articulations (Steinberg 2017, 47).

Some sampler software can display the instruments in a virtual mixer, reminiscent of DAW interfaces, where one can do further balance adjustments and add effects and processing (MOTU 2013, 97-99). This can speed up workflow, as the routing configuration and processing for complex patches can be setup and saved/loaded inside the sampler, making further setup in the DAW unnecessary (Russ 2009, 400).

3.6 Rack/multi

A sampler can host several instruments in a container called a **rack**, or a **multi** for example. The container allows loading the whole state of the module to quickly browse and audition instrument groups like drum kits, ensembles and orchestras.

It can represent an instrument with multiple articulations, by consolidating them into individual instrument elements, which can be assigned to their own MIDI channels (Steinberg 2017, 47). It can also store multi-timbral combinations of instruments, each with their own input, output, and effects assignments and other settings (MOTU 2013, 62). Samplers which support scripting can feature macro controls for the nested instruments' parameters (Native Instruments 2017, 65).

4 VARIANCE OF PITCH

4.1 Keymapping

Keymapping is the process of assigning the recorded samples to MIDI notes in the sampler.

This is usually done in a **mapping editor** that displays the imported samples as rectangular zones in a two-dimensional table diagram, where the horizontal axis represents the MIDI notes and the vertical axis corresponds to note velocity (figure 6) (Steinberg 2017, 70-71). The editor is sometimes also referred to as the mapping matrix.

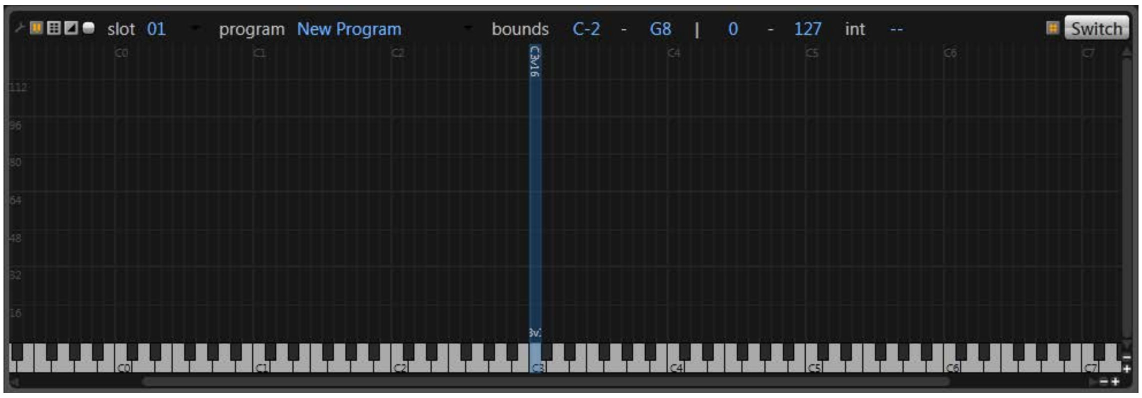


FIGURE 6. A piano sample assigned to the MIDI note 60 in the mapping editor of CWITEC's TX16Wx sampler (Vuolevi 2018)

Once the sample is dropped into this area, a zone is created. It's assigned a **root key**, usually equal to the note that corresponds to the pitch of the sample. If desired, the key zone can be extended to cover any range of notes. If the zone is then triggered by a note that is different from the root key, the sample will be transposed by a number of semitones equal to the number of the triggering note minus the number of the root key.

Extending the zone symmetrically so that the root stays in the middle of the zone ensures that the distance of the triggering note vs. the root key will be as short as possible in case transposition is needed. As Howell (2005b) notes though, sometimes transposing the notes down instead of up might sound better, as up-shifting sounds can lead to timbres that are associated with comical circumstances (*the Chipmunk effect*).

Mapped zones or groups can have settings for key scaling. It can be used for example to restrict a percussion sound to a useful pitch range, or to disable transposition completely in fixed mode (Howell 2005f). Percussion instruments are a particular case where the instruments' natural tone is sensitive to the side-effects of transposition (Russ 2009, 327).

4.2 Multisampling

A melodic instrument that has been sampled only at a single pitch doesn't yet provide a realistic representation of the real instrument's sound. Because of transposition, the samples can sound artificial if a performance necessitates a range of notes to be triggered. (Howell 2005b.)

Transposing a sample one semitone or even two, depending on the instrument's timbre, can be reasonably inaudible, but the further away the key zone is triggered from its root note, the more it begins to sound unnatural (Hosken 2011, 235). The reason for this is that the transposition requires a process called **pitch-shifting**, which results in undesirable qualities or artifacts in the sound when pushed too far.

A basic form of pitch-shifting is achieved simply by slowing down or speeding up the sample, i.e. resampling it at a different rate (Huber 2007, 190). This causes some adverse effects due to the change in length of the sample. The attack (transient) of the sound and the rest of its amplitude contour is warped. It also changes the speed of cyclic articulations like vibrato and tremolo if present in the sound. (Howell 2005b.)

Pitch-shifting can also degrade sound quality by changing the frequency of the formants, which are natural resonances in the spectrum of the sound of vibrating bodies. Normally, formant resonances don't follow the frequency of the vibration, but speeding up or slowing down the playback rate also shifts the formants along with the pitch. This can appear to make the sound sources feel 'bigger' or 'smaller'. (Langford 2014, 246-248.)

Advanced pitch-shifting algorithms compensate for the aforementioned drawbacks: after initially shifting the pitch by resampling, they correct the frequency of the formants and use time-stretching to preserve the original length (Langford 2014, 325). Time-stretching is a process that can alter the length of a waveform without affecting pitch, but it requires interpolation to optimize the amplitude values for the affected digital samples, and this can introduce undesirable artifacts in the sound (Russ 2009, 325).

In order to avoid these negative effects, **multisampling** is used. The instrument is sampled at various different pitches, each of these samples covering their own note, or a range of notes in the mapping area. In practice the range of the key zones will depend on the intended purpose of the sample set (artificial sound might be appropriate for some music genres for example), and the desired realism versus memory footprint.

Generally, a ‘dense’ set of samples more accurately captures the natural timbre variations between notes of different pitches, but instruments vary in their tolerance of transposition, and e.g. bowed strings might be more forgiving than percussion (Russ 2009, 326).

High quality commercial libraries of tonal instruments typically have a zone for each note in the instrument's tonal range. On the other hand some lightweight libraries might utilize a more economic approach and record samples only at minor third intervals for example, each sample thus covering their root note plus the notes directly above and below (figure 7). (Hosken 2011, 235.)

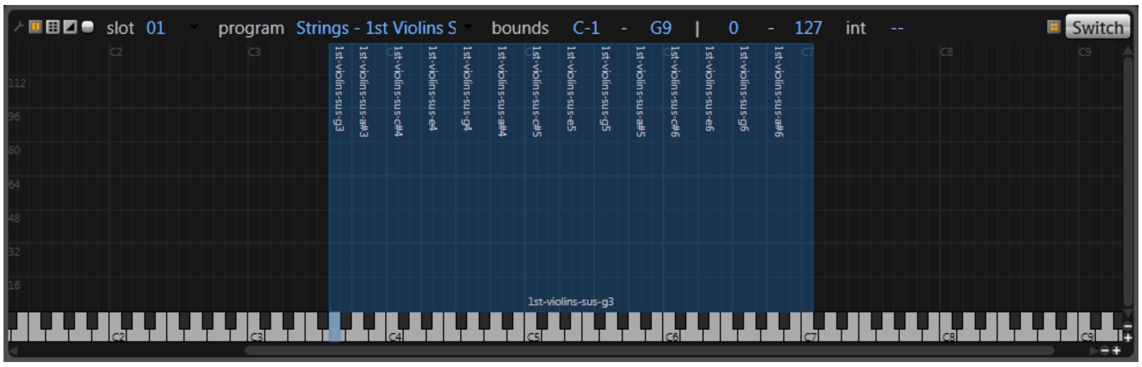


FIGURE 7. Multisampled violin patch (Vuolevi 2018)

4.3 Key zone crossfading

When playing adjacent notes on a sparsely sampled instrument, the notes can sometimes sound jarringly inconsistent. This happens when the consecutive notes land on different key zones which are stretched over too wide a range. The timbres of the notes can be very discrepant, especially after transposition is applied. Some samplers have a feature which can help smoothing the transitions: **key zone crossfading**. This technique overlaps the zones horizontally, and for the notes in the overlapped area, the samples are crossfaded from one to the other, according to the position of the triggering note in the crossfade zone. (Native Instruments 2017, 177.)

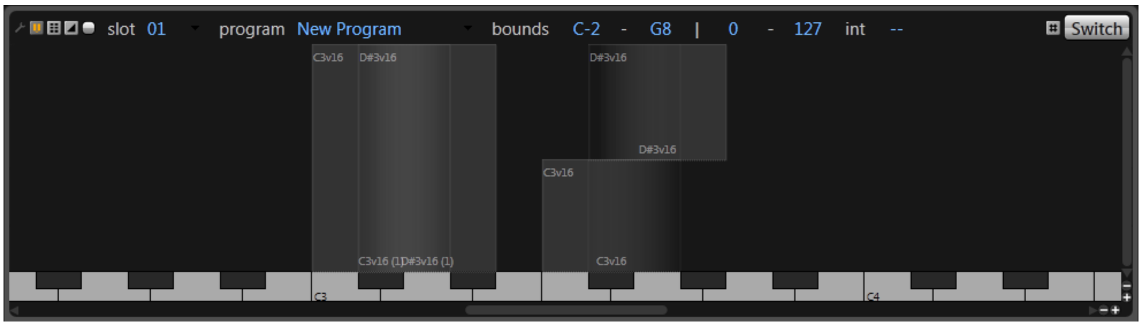

slot 01 program New Program bounds C-2 - G8 | 0 - 127 int --

FIGURE 8. Two crossfaded key zones. Separated duplicates on the right side demonstrate the crossfade gradient (Vuolevi 2018)

This method can cause audible side effects though. If the two notes are too far apart, their timbres and onsets might simply not blend well together, and the two different notes are discernible from the resulting sound. If the zones are close together, the waveforms of the samples tend to be correlated, and - depending on the alignment of the waveforms - adding them together can cause audible phase cancellation.

Mixing waveforms that have a similar frequency but are out of phase can result in a volume discrepancy for the notes in the crossfaded zone, depending on the crossfade type and degree of correlation. It can also manifest as comb filtering in the combined sound, which can be an even more distinct disruption than a volume change. (Langford 2014, 55-56.)

There might also be a slight difference in the tuning of the overlapping notes, especially in samples of instruments where a good playing technique is important for correct intonation (e.g. string instruments with fingerboards).

Overlaying notes with slightly different pitches can lead to chorusing and it can sound like two distinct notes playing on top of each other (Huber 2007, 295; Langford 2014, 57). This can be less distracting on ensemble-type sounds, which already feature chorusing, but can be more of a problem on solo instruments (Magnus 2016). Heavy transposition can also misalign the transients of the notes, which can lead to pronounced **flamming**.

5 VARIANCE OF DYNAMICS

5.1 Velocity switching

Varying the playing dynamics on an acoustic instrument is an effective way to introduce expressiveness to the musical performance.

The whole range of dynamics should ideally be able to be represented by the virtual counterpart that is aiming for realism; it is one of the key components in many acoustic instruments' sound and feel (Howell 2005b).

Samplers offer an easy way to mimic the natural loudness variation by modulating the sample's volume with the incoming MIDI note velocity, but usually this is not enough to achieve a realistic dynamic response (Hosken 2011, 237). This is because changing the playing dynamics on an acoustic instrument also changes the timbre of the notes: the more forceful the playing, the brighter the resulting timbre (Hosken 2011, 205).

An approach that provides a more realistic result in a sampler is called **velocity switching**, and it involves recording multiple samples of the same note with different amounts of force applied in the playing. The samples are stacked up as velocity layers which are represented in the mapping matrix by vertically separated rectangles (figure 9). The vertical axis corresponds to MIDI note velocity and is used to determine which of the layers is triggered by a given note.

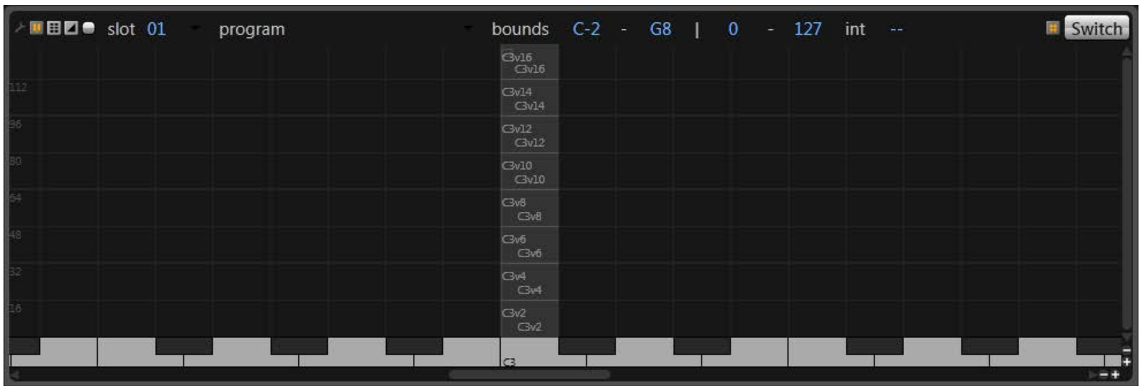


FIGURE 9. Multiple velocity zones for a single note (Vuolevi 2018)

When deciding the amount of velocity layers to be recorded, an obvious consideration would be the instrument’s (or the specific articulation’s) playing technique and its dependency on dynamic variation. Generally, instruments that mostly articulate via dynamics are a good candidate for benefitting from velocity layering.

Piano is an example of an instrument where a wider sampled dynamic range can be helpful (Russ 2009, 327). Other examples include percussion and drums, (electro)acoustic keyboard instruments in general, and rhythmic/percussive articulations of melodic instruments (*spiccato* strings for example).

Increasing the number of velocity layers can also be a source of problems: during the sample recording, the player is required to have a very precise control of the dynamics of their playing to produce a gradual and consistent increase in velocity in all notes across the tonal range of the instrument. Using a large amount of velocity variations will also increase the storage space and RAM requirements. (Hosken 2011, 236.)

Some instruments and articulations that usually don't depend so much on fast dynamic variation could get away with less dense layering (e.g. a string section playing sustained notes).

This approach can split the available velocity range more coarsely, for example based on thinking in terms of musical dynamics terminology: *piano*, *mezzo-forte*, *forte* and so on (Hosken 2011, 237). This has the advantage of being simple to communicate to the performer during the sample recording process, and easier for them to achieve a consistent performance.

5.2 Velocity zone crossfading

One downside to sparse layering is that there can be an abrupt change in timbre when crossing the zone boundaries with successive notes. Some samplers offer a feature that can be helpful for creating a more gradual transition: **velocity zone crossfading**. This feature allows overlapping the zones vertically, applying a crossfade from one to the other (similarly to the previously mentioned method for blending together key zones). When a note is received with a velocity that lands on this crossfaded zone, the two zones' samples are mixed together based on the position of the note’s velocity in the crossfaded zone. (Native Instruments 2017, 177.)

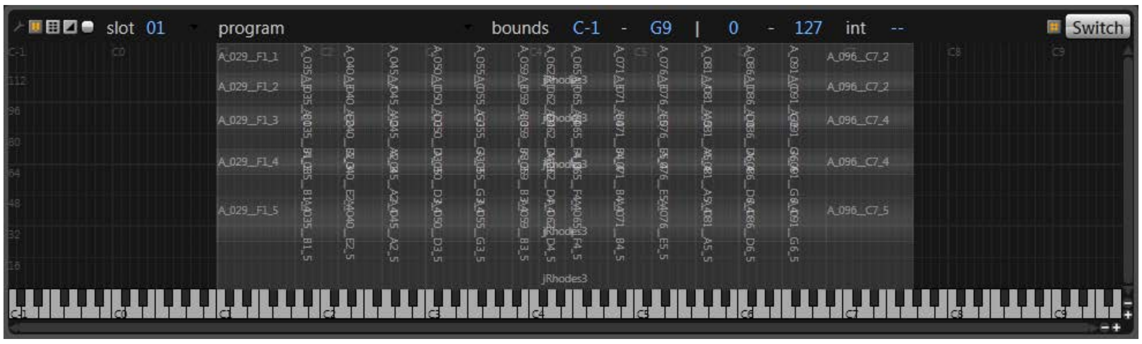


FIGURE 10. A Rhodes patch with crossfaded velocity zones (Vuolevi 2018)

Like key zone crossfading, this method is also prone to the sound quality issues associated with mixing two similar samples together. The degree of correlation between velocity layers is usually very high, and any discrepancies in their phase and onsets might become audible.

For sounds with sharp onsets, a delay of about 10ms or more can begin to introduce flamming to the combined sound (Russ 2009, 43), and short delays can cause audible comb filtering (Hosken 2011, 103). These negatively affect the character of the sound and can degrade the impact of the transient.

5.3 Dynamic crossfading

Sometimes simple velocity switching will not be able to replicate the exact dynamic characteristics of an instrument. This is the case with sustaining instruments - strings, brass or woodwinds for example - where the performer can adjust the force of their playing during the sustain phase (which is integral for performing articulations like *crescendo* and *diminuendo*).

With a real instrument, this transition again changes not only the loudness, but also the timbre of the note. Therefore simply modulating the volume of the playing note in a sampler is not enough to provide a realistic sound. Instead a method is needed to crossfade between the different velocity layers during the sustain part. This is referred to as **dynamic crossfading** (and sometimes simply as velocity crossfading, not to be confused with the aforementioned method of static velocity zone crossfading).

A patch that has this feature typically utilizes the modulation wheel (CC01) to shift the crossfade position between the velocity zones. The note velocity, which usually switches the velocity zone, should be left unassigned. (Stewart 2017a.) This is done so that a difference between the note velocity and modulation wheel position will not cause a jump in the crossfade position when the wheel is moved after a note-on event.

Another modulation source (typically CC11 is used) can be assigned to adjust the volume parameter separately. By separating the control of volume from the timbre his way, interesting sound combinations like soft-but-loud, or intense-but-distant can be performed.

Dynamic crossfading again introduces the risk of comb filtering and flamming with overlapping notes that are not aligned well together. Perfectly matching the sustain parts' phases is not usually feasible with acoustic sound sources though because of slight variations in intonation and possibly different loop lengths for the velocity zones for example.

6 VARIANCE OF CONSECUTIVE NOTES

Many instruments utilize playing techniques that incorporate a quick succession of notes at a constant pitch and approximately the same velocity (e.g. snare roll or ghost notes, or strings playing an *ostinato* pattern).

This can lead to the sampler triggering the same exact sample multiple times in a row, and the result will likely sound obviously artificial. A commonly used method for providing a realistic sound in this case is called **round robin**. (Hosken 2011, 238.)

This function utilizes round robin sample groups for each velocity and note combination in the mapping matrix. The groups contain multiple, separately recorded samples of the same note at the same velocity. In the case of successive triggering, a different sample is picked from the group each time. The samples can be cycled through either sequentially, or in a randomized order (random robin). (Steinberg 2017, 90.)

The sequential method can still end up sounding artificial, for example if the amount of notes in a repeating musical phrase equals the amount of samples in the round robin pool, creating a repeating pattern of sounds.

Conversely, the samples might end up being triggered in a different order upon successive playbacks, for example if the number of round robin layers does not equal (or is not a subdivision of) the number of notes in the MIDI pattern, or if the playback is stopped and restarted in arbitrary positions. This can be a problem if it causes the part to sound slightly different each time it’s played back. Some sample libraries employ a special key switch note to reset the round robin sequence in order to provide the user a way to make the sequences deterministic. (Stewart 2018c.)

Randomized order overcomes the first mentioned problem of sequential triggering, but some sampler designs have a drawback in their implementation: the same sample can end up being picked from the pool multiple times in a row. Others handle this by offering an option to exclude the previously played sample from the selection pool when picking the next one.

The aforementioned problems of repetitiveness/indeterminism stem from an underlying cause: the round robin samples are not matched closely enough in volume, timbre, or the timing and tone of the transients. Including samples that have some distinct character (tonality, intonation, noise, pops or clicks, ringing) compared to the other ones in the pool can lead to some of the notes sticking out.

Depending on the type of instrument, the dimensions of its resonating body can introduce complexities in the vibrations it produces. Another factor is the mechanism of excitation, like the hammer action and bounce of the piano for example. (Russ 2009, 90, 287.) A third component that adds to the complexity is the natural variance in the playing technique of the musician. Due to this chaotic tendency of acoustic systems there will always be slight variation present in separately recorded samples - even if they sound near identical to the ear - and this helps to break up the monotony of successive retriggers (figure 11).

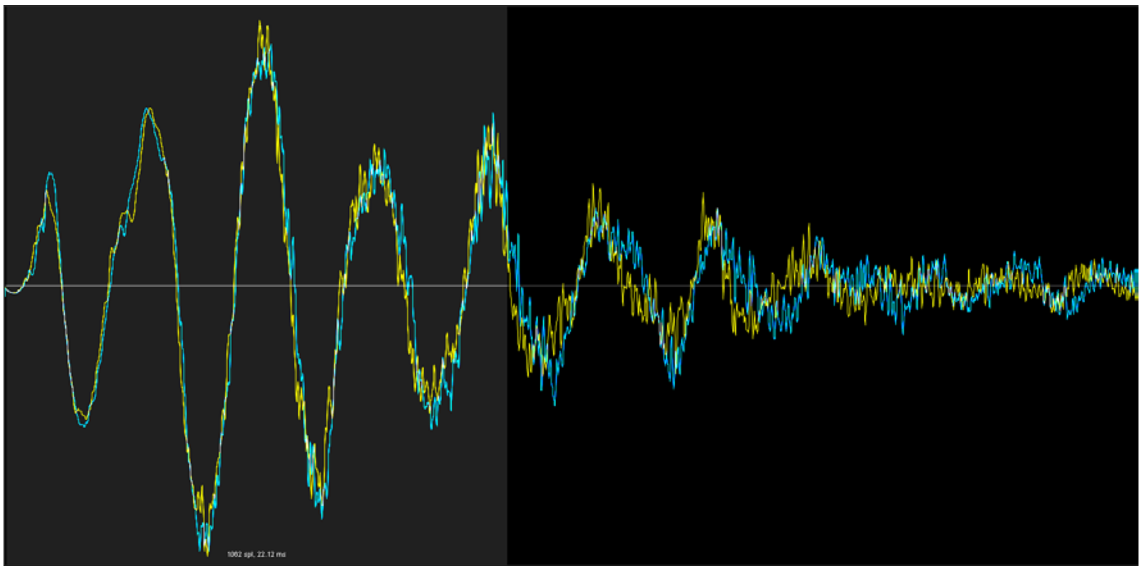


FIGURE 11. Two overlaid waveforms of snare round robin samples recorded from Addictive Drums 2. The waves are highly correlated at the onset, and only start to diverge after about 20ms, when most of the transient has passed (Vuolevi 2018)

The number of necessary round robin layers depends on how fast the sound is likely to be retriggered. Different instrument types (e.g. military snare vs. kick drum) and for example musical genres (e.g. dub vs. thrash metal) have different requirements. When deciding the round robin amount it’s good to keep in mind - especially if the instrument already requires dense layering in the key and velocity axes - that adding more layers multiplies the total number of samples that need to be recorded.

A dedicated drum sampler might use 6 round robin layers per velocity by default, although some provide a lot more (Toontrack 2017). For percussion instruments in general, some minimal round robin implementations offer two alternate samples (for left and right hand strokes).

For already recorded (multisampled) sample sets that don’t feature round robin layers, adjacent key zones can be repurposed as round robin layers at the cost of increased transposition. For a chromatically sampled instrument as an example, three separate layers can be created by grouping key zones that are a minor third apart and extending the range of the zones (figure 12).

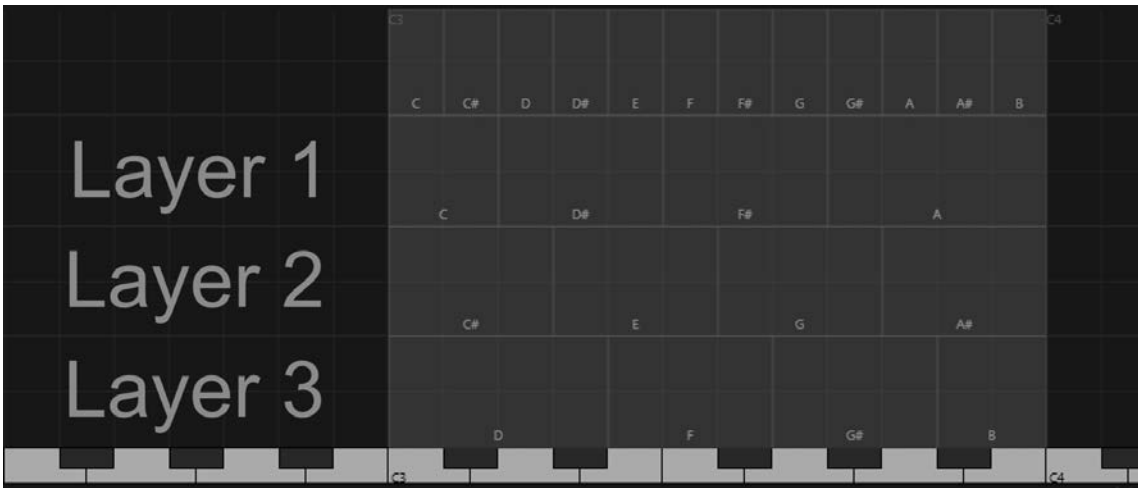


FIGURE 12. Key zones converted to round robin layers (Vuolevi 2018)

Samplers which support scripting can achieve this more easily and flexibly, for example with an algorithm that picks a random sample from a specified range of nearby key zones and pitches it to the playing note (waveforms.fairlyconfusing.net 2014).

Another technique to reduce the number of retriggers - useful in samplers that have no or limited round robin support - is to concentrate the available velocity zones in a range that is likely to be triggered most often on the particular instrument: for example the high end of velocity on percussion. Some samplers which don't have advanced round robin features utilize the technique (figure 13).

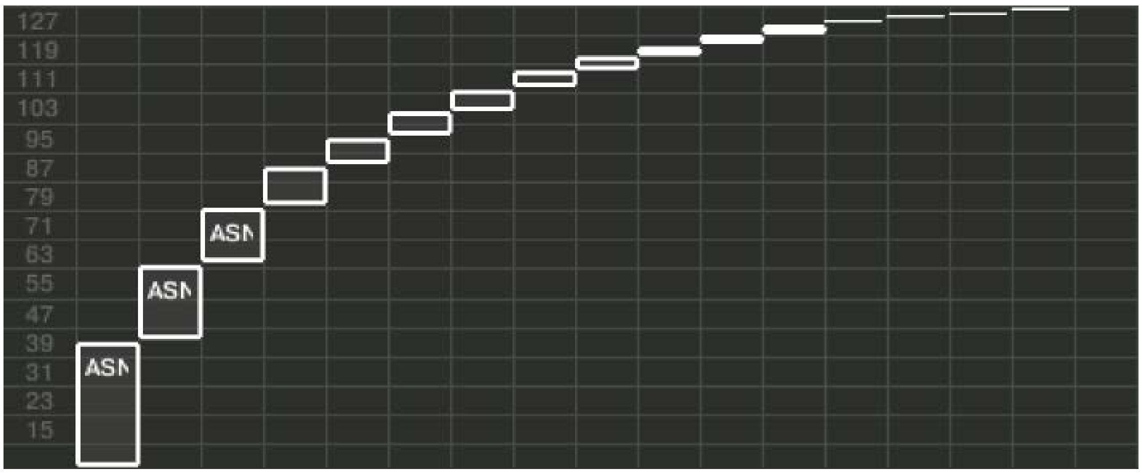


FIGURE 13. Velocity zones of a taiko drum patch in Native Instruments Battery 3 (Vuolevi 2018)

7 LOOPING

7.1 Overview of looping

When sampling infinitely sustaining instruments, for example organs or bowed strings, a question about the ideal sample length arises. If a sample has a length of, say, 3 seconds, but later on a composer using the virtual instrument would like to create a drone note that goes on much longer, the patch will not be able to accommodate that by simply playing back the sample. It also makes no sense to record extremely long sustain samples for every note, because the disk space requirement and loading times will become a problem. To deal with this issue, samplers offer a function called **looping**.

A loop in a sample is a second set of start and end points, usually separate from the actual sample start and end, positioned so that the sustaining part of the sampled note is in between them. When the sampler is playing back the sample, it will repeat the looped area of the waveform - in a manner that is specified by the loop mode setting - until it receives the corresponding note off message. The most common type of loop makes the sample play position jump from the loop end marker to the loop start marker. This is referred to as the ‘forward’ mode. (Native Instruments 2017, 196-199.)

Another option that’s usually offered is called ‘bidirectional’. It causes the playback to change direction when reaching the loop end marker, and plays the sample back in reverse until changing direction again at the loop start point. This mode has limited practical use, because it tends to sound unnatural. (Russ 2009, 322; Howell 2005d.) Backwards playback might introduce weird effects in the sound, and the loop can sound obviously cyclic compared to the forward mode because of the alternating playback direction.

7.2 Loop point matching

When using a one-directional loop mode, a common problem is a ‘popping’ sound, which happens when the playback loops around. This is caused by a discontinuity in the waveform (figure 14), where the playback jumps from the sample at the end of the loop to the sample at the beginning, as these will not likely be closely matched in amplitude. (Hosken 2011, 238.)

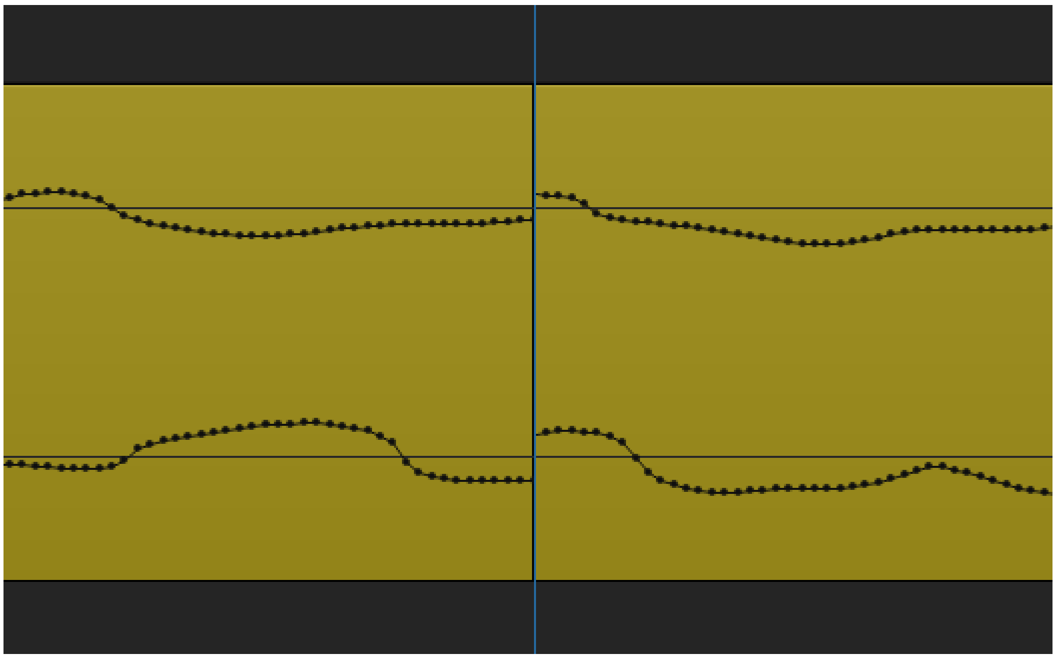


FIGURE 14. Discontinuity in the waveform at loop end and start points (Vuolevi 2018)

One method for making the transition sound more natural is to have the loop points located in **zero crossings** (the points in the waveform where the digital sample is at the vertical center point, signifying an amplitude of zero). This will ensure that the waveform will be continuous across the loop point. (Langford 2014, 28.)

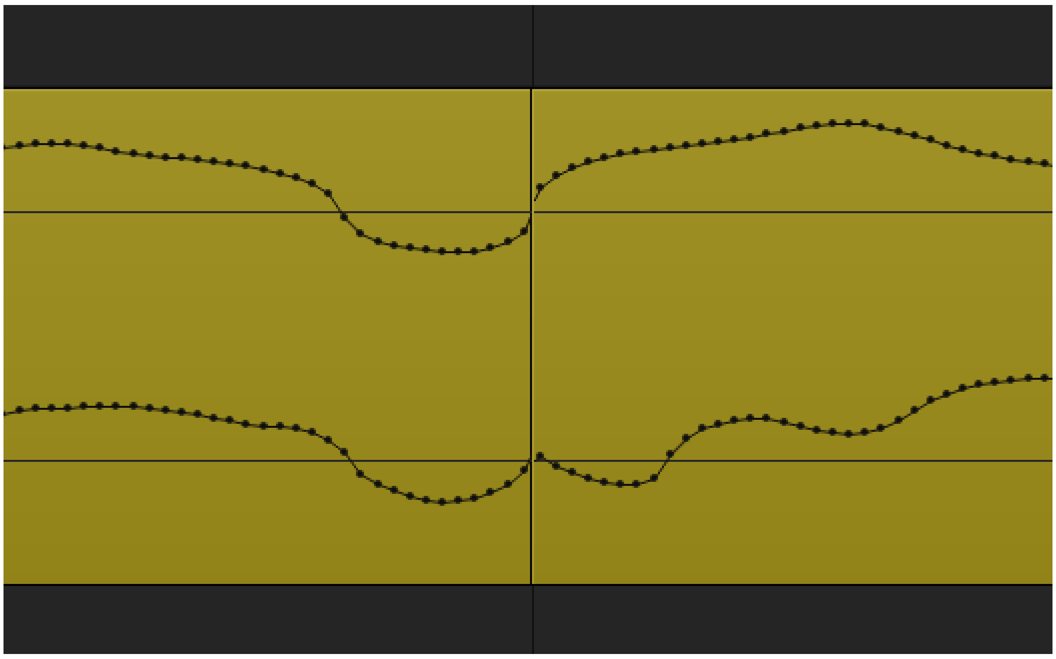


FIGURE 15. Loop points positioned at zero crossings (Vuolevi 2018)

The method improves the sound of the loop transition, but is usually not enough in itself to completely eliminate the offending artifact. Even if the samples at the loop point match in amplitude, the overall waveform shape might present a problem (figure 15). There are two more factors that need to be taken into account: the slope of the waveform and the rate of change of that slope. (Russ 2009, 322.)

7.3 Loop point crossfading

Another method for smoothing the transition is **loop point crossfading**. This function specifies a length at which the sampler will start fading in the beginning part of the sample before reaching the loop end marker, simultaneously fading out the end part. (Howell 2005d.) The fade-in section is taken from the part of the sample that's located before the loop start marker, so crossfading requires leaving a sufficient amount of the sustain sound on the left side of the loop start point.

Using the crossfade function requires the consideration of a couple of factors. The first is that because the sample is crossfaded with another part of itself, there is likely a high correlation between the overlapping waveforms, and phase cancellation can again become a problem. Some samplers offer functionality that helps with visually aligning the waveforms by overlaying the start and end marker positions with a transparent representation of the waveform's phase at the other position (figure 16).

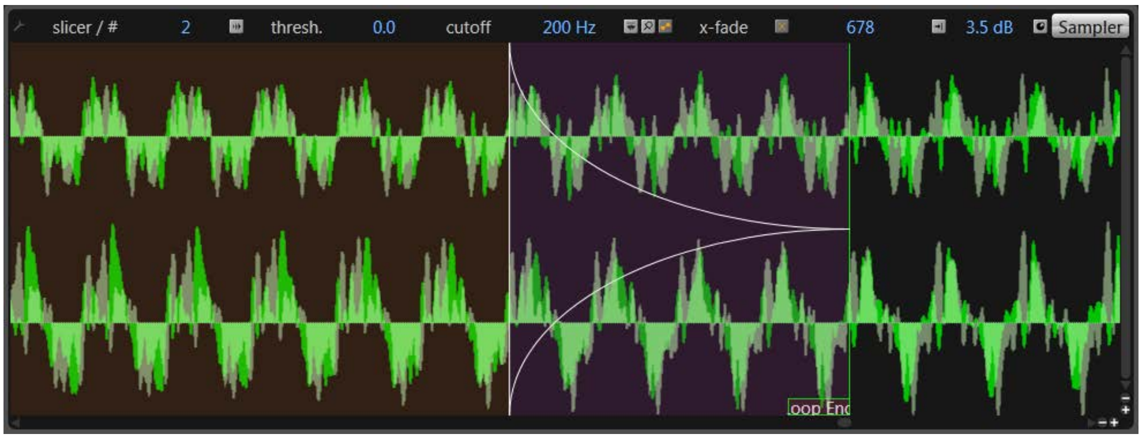


FIGURE 16. Loop point waveform overlay (Vuolevi 2018)

The other factor is the length of the crossfaded area. Longer fades create smoother transitions at the expense of possibly audible phase cancellation. Shorter fades can mask the comb filtering, but the transition can sound unpleasantly sharp if the timbre at the loop points is dissimilar. (Langford 2014, 56-57.)

Waveforms of acoustic instruments exhibit periodic patterns at several time scales: one represents the fundamental frequency of the note, but there can also be recurring phases at various longer intervals. These can be used as visual cues to find crossfade points that will likely match better. (Langford 2014, 29-30, 56.) In addition to matching the pattern contours at longer scale, and the phase of the fundamental during the crossfades, these patterns can be used to mask the crossfade in the character of the sound in two ways:

1. A sound such as a bowed string instrument played in tremolo articulation has a lot of quick, intermittent attacks in it, which can mask a short crossfade in between them (figure 17). If the loop points are also set so that the overall timbre of the loop start and end are similar, the transition can be quite inaudible.

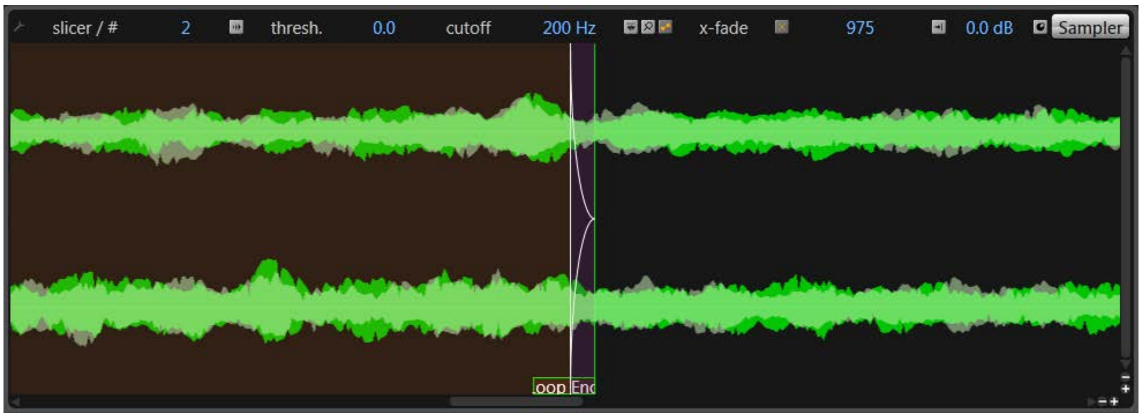


FIGURE 17. Short crossfade (Vuolevi 2018)

2. If the sound has a more steady timbre, such as sustain strings for example, the fade can be positioned so that it follows the slowly evolving contour of the timbre (figure 18). This can disguise the comb filtering as the natural timbre variation pattern of the waveform.

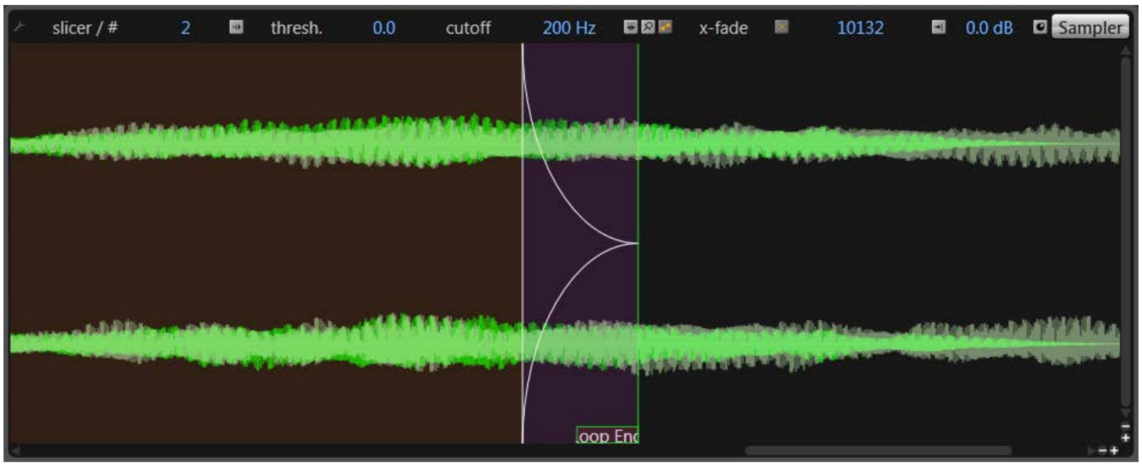


FIGURE 18. Longer crossfade (Vuolevi 2018)

7.4 Other considerations

Beyond these techniques used for smoothing the loop transition, the sample itself needs to possess some qualities for it to be looped properly. First of all, the sustaining part that is looped should be long enough, preferably several seconds. If the sustain part - and therefore the loop - is too short, the timbral pattern it creates will sound apparently repetitive. (Howell 2005d.) On the other hand, in cases where there is little to no timbral variation over time, very short sections can be transparently looped, e.g. a single cycle of an oscillator waveform.

Second, it would be ideal to get the amplitude and timbre of the sustain to match at the loop points. This makes it more difficult to loop sounds that exhibit a decay in amplitude: there will be a volume jump when the playback loops around. (Hosken 2011, 238.) Also, timbres tend to get darker along with the loss of amplitude, because higher harmonics decay faster than lower ones (Hosken 2011, 205).

Two other important, looping-related concepts are **one shot** and **release samples**. One shot samples do not obey note-off events, but are instead always played in their entirety. One shot mode is useful for drum samples for example, where having short notes inadvertently cutting off the sounds is undesirable. Release samples are triggered by note-off events and are used to emulate sounds like mechanical noise from dampers in keyboard instruments. (Mantione 2018; Native Instruments 2017, 199, 222.)

8 MODULATION

8.1 Overview of modulation features of samplers

Modulation is the process of altering target signal's (carrier) properties with another signal (modulator).

In the context of audio the term has roots in the world of modular analog synthesizers, which create sounds by utilizing separate devices - modules - like oscillators, filters and control voltage generators. These parts are connected together by a network of patch cables, and then controlled - or modulated - by the different kinds of voltage generators in order to achieve dynamic variation for the parameters of the sound sources and processors. (Howell 2005f.)

Many modern software samplers offer the flexibility of the modular synthesizers' modulation assignments. With samplers, the source sounds are usually more complex and interesting than oscillator waveforms, and this combined with the comprehensive modulation features presents numerous sound design possibilities. (Howell 2005f.)

In samplers, modulation uses a source signal (dynamic or a static scalar value), which is then mapped to a destination parameter. The modulation mapping can support scaling, inverting and otherwise shaping the effect of the source on the destination parameter (figure 19). (Steinberg 2017, 137-138.)

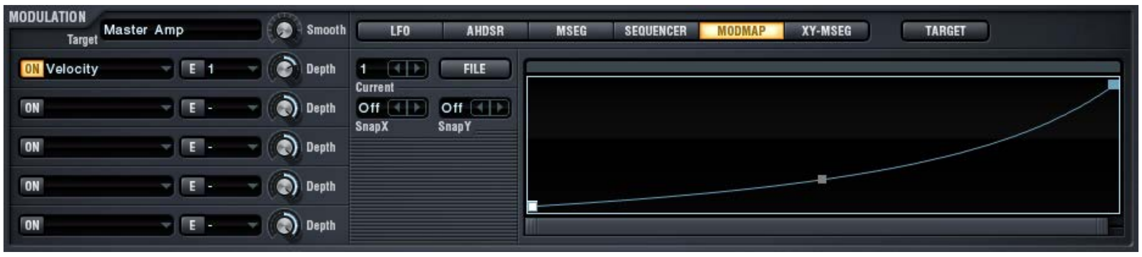


FIGURE 19. Velocity value mapped to master volume through a transfer curve in Camel Audio's Alchemy (Vuolevi 2018)

The modulation process can either be continuous, sequenced, or it can be triggered on certain conditions, such as when the sampler receives a note-on MIDI message (Native

Instruments 2017, 320). Modern samplers can support complex interactions with an arbitrary number of modulation assignments (Howell 2005f). The sources and targets are usually displayed as a list in the interface, but it’s sometimes referred to as the modulation matrix (Steinberg 2017, 137-138), as per the interface that is sometimes used to relay control voltages in modular synthesizers.

8.2 Incoming MIDI data

The MIDI digital communications protocol is used to transmit control and performance data between musical devices. Control data is sent by **System messages**, which relay things like time code, song position and for example raw data for specific devices. Performance data is created by the user manipulating the controller, and this data is transmitted as **Channel Voice messages**. They contain several types of information that can be used as modulation sources. (Huber 2007, 13, 22, 42-45.)

In the MIDI standard, a set of 128 consecutive integer values are represented by seven bits in a MIDI message **byte**, and the leftover bit defines the byte as either a ‘status’ or a ‘data’ message. A serially transmitted cluster of bytes forms the MIDI message. The status byte defines the type of the message and the channel that needs to be used to receive it. The following data bytes define the value of this event. The Channel Voice message types are **Note-On**, **Note-Off**, **Polyphonic Key Pressure**, **Channel Pressure**, **Program Change**, **Pitch Bend Change**, and **Control Change**. (Huber 2007, 16-17, 22.)

The Note-On message for example defines the following parameters: the message type as Note-On and transmitting channel (1-16), the note value in a range from 0 to 127 (with note #60 representing the middle C), and the attack velocity ranging from 0 to 127. A velocity of 0 is usually interpreted as a Note-Off message (Huber 2007, 22-23).

Control Change (CC) messages can be continuous controllers, each of which supports values from 0 to 127. These are usually transmitted as a stream of messages, generated by manipulating controls like the modulation wheel or sliders and knobs. They also transmit switch controller messages, which have a discrete ‘on’ and ‘off’ states, generated by e.g. a hold pedal or other foot switches. (Huber 2007, 28-29.)

MIDI also supports a special pitch bend control message. It differs from regular CCs in that it has a much more accurate resolution, being able to send 16384 different values (2 data bytes are used instead of 1 like with the CC messages). The increased accuracy is necessary because of the ear’s pitch sensitivity. With a lower resolution the steps in pitch could become audible. This message is usually generated by pitch-bend wheels which is found in most MIDI keyboards and other instrument controllers. (Huber 2007, 28.)

Aside from determining which key zones are triggered, the MIDI note-on and note-off events can be used for various modulation purposes, which is sometimes called key following (or keyscaling). For example, assigning note pitch to filter cutoff can be used to create a filter that follows the played note and thus keeps the timbre more consistent over different notes. (Russ 2009, 118.) Another example would be linking note pitch to the amplitude envelope to mimic a property of natural resonators where higher-pitched notes decay faster (AKAI 2000).

Note velocity is another commonly used modulation source. Typically it’s used to create variation in the amplitude in relation to playing dynamics, but it has also other uses than simply linking together velocity and volume.

If the source sound features velocity layers with natural dynamics, velocity can be inversely linked to volume to increase the loudness of softer notes, creating a compression effect. This can also be achieved by normalizing the recorded samples by loudness, and regularly assigning velocity to modulate volume the desired amount (although this has the disadvantage of losing the original loudness-to-timbre relation in the samples). Artificially defining the velocity response this way gives the user control over the instrument’s dynamics, and can reduce the need to apply processing like compression further up the signal chain.

The instrument’s response to playing dynamics can also then be fine-tuned by adjusting the transfer curve of the modulation, which can allow the instrument to adapt more easily to different playing styles (Computer Music 2016).

Another way to alter playing dynamics with velocity is to inversely link it with the sample start parameter, so that lower velocity values move forward the point at which the samples start playing, effectively skipping a part of the transient of the sound. This has the effect of softening the sound at lower velocities, and can be a useful addition to the dynamic response of the sampler patch, especially if the recorded sample set is lacking in velocity layers. (Howell 2005c.)

Velocity modulation can also help to simulate the timbre variation of acoustic instruments in relation to playing force, if that property is not adequately represented by the sample set (e.g. not enough velocity layers). Mapping velocity to the cutoff of a low-pass filter can provide the relationship between dynamics and timbre, although it might not sound as natural as a few good velocity layers. (Hosken 2011, 238.) Playing force also tends to affect the length of the decay of notes, which can be simulated by linking velocity with the amplitude envelope's decay parameter.

8.3 Envelope generator

Envelopes are time-variant signals that are triggered and terminated by MIDI note-on and note-off messages respectively (Hosken 2011, 204). They can approximately represent many dynamic variations that are present in natural sounds and are useful for emulating these features. Perhaps the most common type of envelope generator - usually referred to as the **ADSR envelope** consists of four parts: **attack**, **decay**, **sustain** and **release**. (Howell 2005f.) Each of these represents a distinct part in the signal's time axis (figure 20).

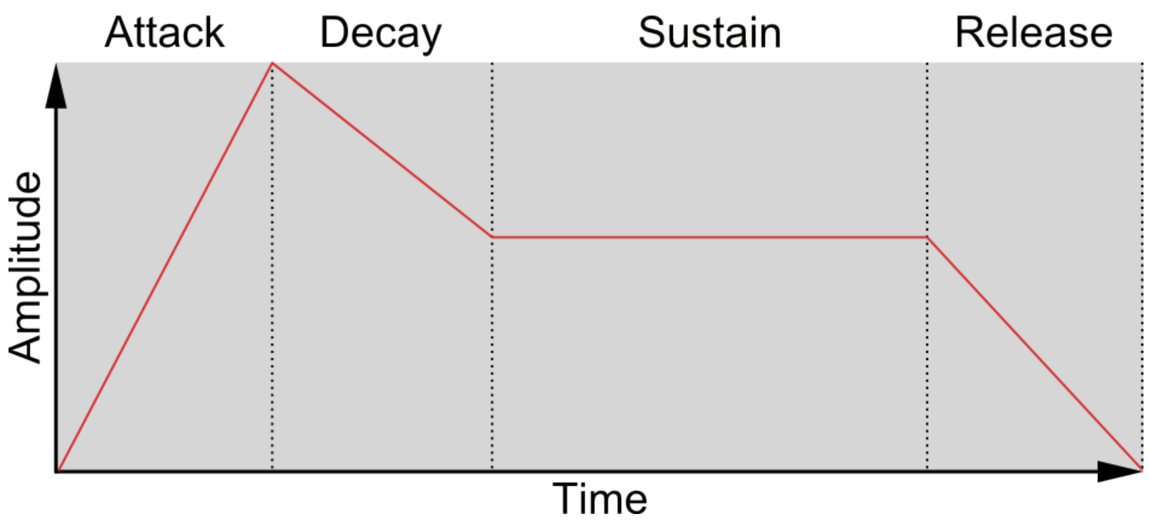


FIGURE 20. The phases of an ADSR envelope (Vuolevi 2018)

1. Upon a note on event, the attack phase of the envelope begins, the length of which determines how long it takes for the signal to reach peak amplitude. The curvature of the travel can usually be transformed.
2. The decay phase follows, during which the signal travels from peak to the sustain amplitude. Together with the attack phase, this forms the 'transient' of the envelope, reminiscent of the initial peak in the amplitude of many naturally occurring sounds.
3. The next phase is sustain, and here the envelope's amplitude stays constant until either a note off event is received, or the sustain part's maximum length value (if available) is reached. The level of the sustain is equal to or less than the peak amplitude. In the latter case, the transient created by the attack and decay phases is emphasized. This effect is more prominent the lower the sustain amplitude is set in relation to the peak.
4. The final phase is release, and it determines the time it takes the signal to reach zero after the sustain phase ends.

The sustain is the distinguishing part of this type of envelope. A player keeps applying force to the instrument to keep the note ringing; this is analogous to the envelope's sustain phase. For example, bowed string instruments work like this, as do blowed ones like brass and woodwinds. (Hosken 2011, 28.)

For instruments with dampers, e.g. the piano, it's the equivalent of holding down the played note. Although the note keeps decaying due to the percussive nature of the instrument - meaning there's no way to keep applying force to the resonating body - the sustain is still distinct from the release part, which happens when the damper is brought into contact with the resonator, making the sound decay more quickly.

A variation on this envelope type can emulate the decay at the sustain phase: **ADBDR**, which stands for **attack**, **decay 1**, **break**, **decay 2**, **release** (figure 21) (Russ 2009, 128).

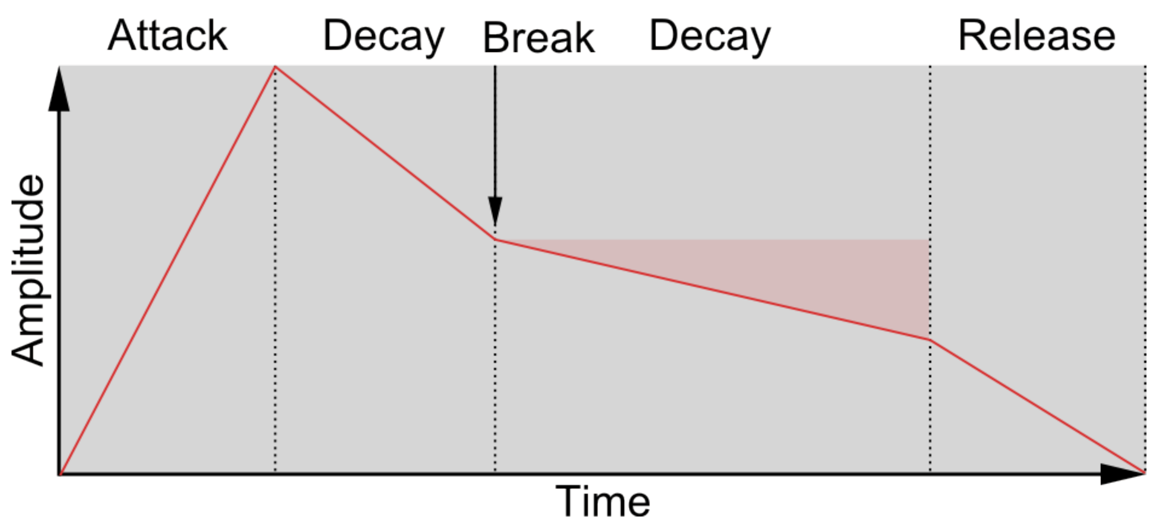


FIGURE 21. ADBDR Envelope (Vuolevi 2018)

Another category of instruments that have their own distinct amplitude contour are percussive instruments like drums and guitar for example. These lack the sustain part, because all the force is applied at once, and the sound begins to decay to zero right after. This too can be presented by the ADSR envelope by setting the sustain amplitude (or duration) to zero, but some samplers offer a specialized envelope generator for this purpose: **AR** (or **AD**), which stands for **attack**, **release** (or **attack**, **decay**). (Hosken 2011, 28.)

There's also a special type of envelope called **DBD** (Decay-break-decay). This envelope is bipolar and can create a shape where the signal crosses over to a negative value (figure 22). It features three controls: **decay 1**, **break** and **decay 2**. Decay 1 sets the time during which signal travels to the position defined by break, and decay 2 determines the decay time from break to zero. This type of envelope can be useful for simulating pitch envelopes of percussive sounds or other fast effects that happen in the attack phase. (Native Instruments 2006, 31-32; 2017, 327.)

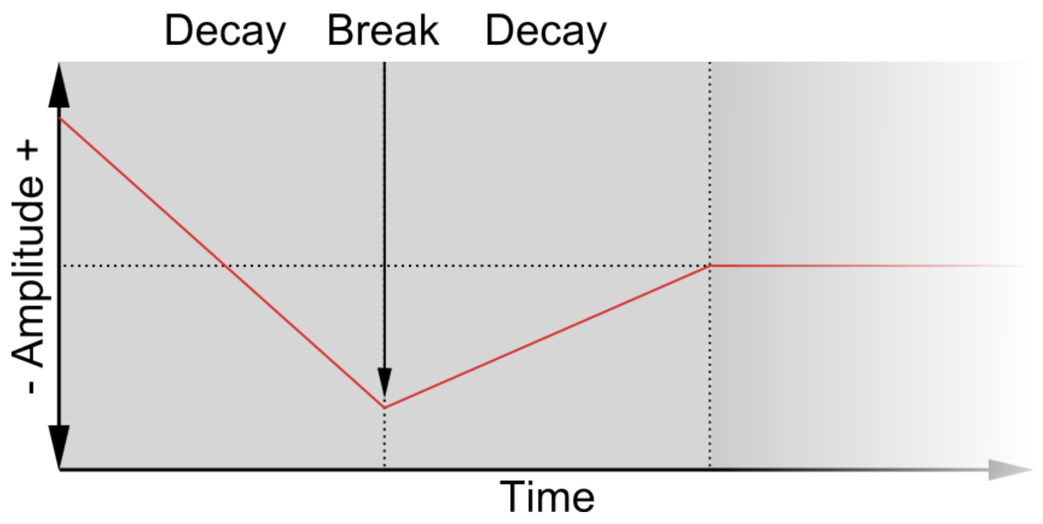


FIGURE 22. DBD Envelope (Vuolevi 2018)

8.4 LFO (low frequency oscillator)

LFOs produce different kinds of waveforms, the frequency of which is usually below the audible range (< 20 Hz). They provide a range of common periodic waveform shapes, such as sine, triangle, square/pulse and saw, and also sometimes a random pattern, which can either jump from one value to the next (hold), or transition smoothly (glide). (Native Instruments 2017, 332; Steinberg 2017, 133.)

A very useful feature is the ability to sync the LFO rate to the host tempo (the tempo that is reported by the DAW to the sampler if it's used as a plugin), as this can provide musically paced modulation signals that work well with the rest of the arrangement. (Howell 2005f.)

LFOs can be used to create movement in sounds, and they're suited for emulating some performance techniques like tremolo and vibrato, by assigning modulation to sample volume and pitch respectively. Despite not necessarily sounding as natural as the genuine articulations, using LFO for creating the effects has a couple major benefits. It's easier to find good loop points in the samples without the periodic articulation patterns. (Howell 2005f.) Also, there is no need to record a separate set of samples where the technique is applied by the performer, reducing the size of the sample set. Another important benefit is that the intensity and speed of the tremolo/vibrato can be fully customized by the user.

Some LFO implementations have a fade-in parameter that can be used to gradually ramp up the strength of the modulation to the specified full value (Steinberg 2017, 132). This can provide a more natural sound for articulations, because in real vibrato for example, the intensity tends to start at zero and ramp up over time (Elsea 2013, 228).

8.5 Other modulation sources

Some samplers offer even more choice regarding modulation sources. There are step sequencers that can create rhythmic patterns, and multi-stage envelopes for creating complex curves. Envelope followers gradually outline the amplitude changes of input signals. (Native Instruments 2017, 329, 334-335.) Some expose ‘dummy parameters’ to the host DAW, so they can be controlled there using automation, and linked to one or more parameters in the sampler (Steinberg 2017, 42).

9 ARTICULATION SWITCHING

9.1 Overview of articulation switching

Acoustic instruments feature many different ways that sound can be produced from them. They can be struck, plucked, scraped, blown, etc. with various materials to produce a huge variety of different timbres. Due to this variety it’s very difficult to capture all of their characteristics in a sample-based virtual instrument. (Hosken 2011, 25; Jensen 2016; Russ 2009, 91.) In practice though, many instruments have a set of established articulations that are most commonly utilized (Stewart 2017c). A comprehensive virtual instrument should strive to provide them in the software form.

Some instruments are simpler in this regard - for example the piano - as their typical articulations can be controlled with playing dynamics and note lengths (e.g. *legato*, *staccato*, accents). On the other hand some instruments’ articulations change their timbres so much that additional sample sets need to be recorded in order to properly emulate them (e.g. *tremolo* and *pizzicato* on a string instrument).

Orchestral instruments for example tend to have many distinct articulations, and orchestral scores make use of the flexibility of the instruments and feature switches between the different articulations as musical effects (Stewart 2012).

Some articulations, e.g. tremolo and vibrato, can be emulated to an extent without additional samples, but usually most playing techniques need to be recorded separately, along with their corresponding key, velocity and round robin layers. A method is then needed for the user of the virtual instrument to control which sample set is used at any particular time.

The individual articulations are loaded into the sampler as groups or instruments as per the sampler’s architecture. These elements provide access to overall adjustments to the contained sample sets, and a means to switch between them with sequenced or played MIDI events. (Native Instruments 2017, 96, 161.)

MIDI provides three different ways to convey the switch message to the sampler:

9.2 Keyswitching

Sampler patches that feature **keyswitching** utilize MIDI notes outside of the keymapped range of the instrument to select between the different articulations (Stewart 2012). When the sampler receives one of the notes assigned to the key switches, it changes the active articulation sample set in the program/instrument bank.

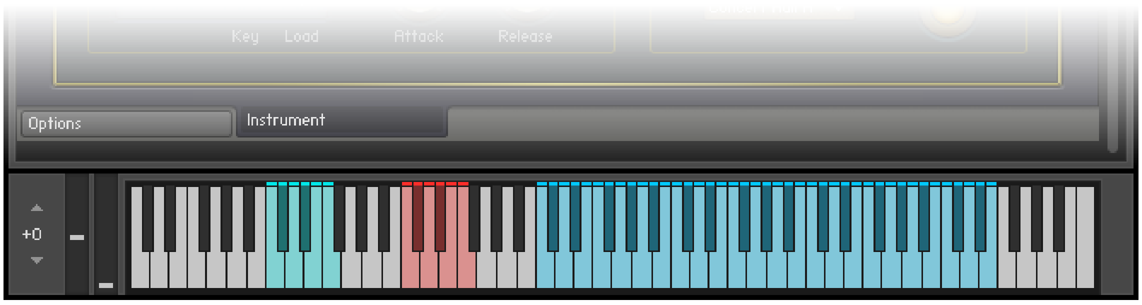


FIGURE 23. Two key switch areas on the left (cyan and red) and a keymapped area on the right (light blue) displayed in Kontakt's virtual MIDI keyboard (Vuolevi 2018)

Keyswitching is an easily accessible way of changing articulations, especially when playing the patches live, but it has a couple downsides:

Without some help from the DAW, key switches usually don't support **chasing** the articulation from the playback position (Stewart & Hearn 2012; Stewart 2014). Chasing means that the previously issued switch will be active at the current position, even if the switching event itself is located earlier in the MIDI sequence. Not supporting this means that in order to have the correct articulation active in an arbitrary playback start position, the keyswitching notes need to be extended to span the whole duration of the notes using that articulation. Even then, the musical note and the key switch note could be registered in the wrong order, and the note would be then played using the previously selected articulation.

The switches being MIDI notes, they can become a distraction when viewed in a score editor, unless the editor supports displaying them as articulation notation or text. The keyswitching notes can also be accidentally transposed along with the rest of the contents of the MIDI sequence. (Stewart & Hearn 2012.)

9.3 Channel switching

In **channel switching**, each articulation is split into its own program/instrument, which are then assigned to MIDI channels. The channel value in the incoming MIDI note messages decides which articulation is used to play those notes. Several articulations can be assigned to a single channel, if they are always played together. (Steinberg 2017, 47.) They can also be layered by playing their corresponding MIDI sequences on different channels simultaneously.

A simple way to switch between articulations using this method is to create sequencer tracks which each are routed into different MIDI channel inputs of the sampler. However, the need to switch between sequencer tracks for articulations can be cumbersome, especially in a large project. (Hosken 2011, 240; Stewart 2017c.) Another method is to change the channel settings of individual notes in the MIDI editor of the DAW, but this might not be very convenient either, depending on the editor's features.

The channel switching method doesn’t require chasing the articulations, because each note contains the switch information in itself. Compared to keyswitching, changing articulations while playing can be more difficult though.

9.4 Program change and CC switching

The program change message is used to change presets in MIDI connected devices like synthesizers (Huber 2007, 26). Software samplers can also utilize it along with the control change message types for articulation switching.

These MIDI messages have the advantage that sequencers usually chase them and process them before note messages, therefore the correct articulation will always be active, regardless of the playback position (Stewart 2014). On the other hand, using them live is not as easy as with key switches, and triggering multiple articulations simultaneously is not necessarily possible (Synthetic Orchestra 2014).

CC switching can be a good option when composing using score editors, because CC messages don’t clutter up the staff with redundant notes. Some score editors allow displaying these messages as articulation notation, so they will be in correct format, ready to be printed out for example, without further clean-up. (Native Instruments 2017, 96; Stewart 2012; Synthetic Orchestra 2014.)

10 SAMPLER DATA AND FILE FORMATS

10.1 Bit depth

Digital recording most commonly uses 24 bits of resolution nowadays, while 16 and 20 bits have been the previous standards (Hosken 2011, 78; Huber 2007, 165). Although 16-bit audio can cover the entire dynamic range of human perception in normal listening conditions (taking into account the noise floor of the ear and the environment), signal processing can benefit from more resolution. Increasing the resolution of recordings to 24 bits lowers the digital noise floor, which means that recordings can be made at lower input level which reduces the chance of clipping, and it also leaves more room for dynamic range compression to work with before the noise floor becomes audible. (Hosken 2011, 78; Montgomery 2012.)

10.2 Sample rate

The sample rate determines the highest frequency that the digital signal can represent. It is equal to half of the sample rate, and is known as the **Nyquist frequency** in the Nyquist–Shannon sampling theorem. (Hosken 2011, 74.) The standard for audio playback has long been 44.1 kHz, and for example conventional audio CDs use this rate (Howell 2005a). It is sufficient to represent audio signals that can cover the entire audible frequency range of the ear in all practical conditions (Hosken 2011, 74). For multimedia productions with video, 48 kHz is the conventional format, because it can be synchronized with many common frame rates (Hoffner 2003).

These two rates meet most psychoacoustic and technical specifications, but higher sample rates can be of benefit to sound designers for example. Because a higher sampling frequency shifts the sharp cutoff of the Nyquist frequency upwards, there are higher partials present in the signal which can be brought down into the audible range when the audio is pitched down. High sample rates obviously come with the cost of disk space though.

10.3 Uncompressed (raw) audio data formats

This category includes **formats** like WAV and AIFF, which store audio signal data as a PCM-coded bitstream. The above two standard formats are universally compatible with different software. They can contain metadata, and feature multichannel support, which both can be utilized in samplers. (Huber 2007, 231; Hosken 2011, 79.)

As they most typically contain raw data, they require a lot of storage space compared to formats that use compression. They're commonly used in media production tasks because of their accessibility, speed, and preserved fidelity.

10.4 Audio data compression codecs

These can be separated into two different categories. The first one is **lossless compression**, which is provided by **codecs** like FLAC (Free Lossless Audio Codec) and ALAC (Apple Lossless Audio Codec).

Lossless compression can approximately halve the file size without any negative impact on the integrity of the audio signal, as it's perfectly reconstructed upon decompression (Hosken 2011, 81). The decoding process during playback requires CPU time though, and writing them is slower than raw data formats due to the required coding.

The second category is **lossy compression**. Commonly used codecs include AAC (Advanced Audio Coding), MP3 (MPEG-1/MPEG-2 Audio Layer III) and Vorbis (superceded by Opus) for example. (Huber 2007, 233-234; Hosken 2011, 81-82.) These codecs use **perceptual coding** techniques to alter the audio signal in order to remove information that is likely inaudible due to psychoacoustic phenomena such as **masking** (Hosken 2011, 81). This process achieves higher compression rates than lossless compression, but the artifacts it introduces can become unmasked with further processing. So in order to avoid generation loss, it's not advisable to use lossy files in media production tasks. Lossy compression also requires additional CPU time for the read and write processes.

10.5 Sampler file formats

In addition to supporting common audio file formats, many samplers also feature their own exclusive storage file formats for both audio and patch data. The formats come in two types: monolithic and non-monolithic.

Monolithic files store all the data in a single file, consolidating the sampler parameters and mappings and other patch data, and all the different audio files used by key and velocity zones and round robin layers. This is helpful in keeping all the relevant data together, and simplifies storing and sharing the instruments. (Native Instruments 2017, 41.) Monoliths usually don't support very big sample sets though, because of the limitation of some file systems on the maximum size of individual files (e.g. 4GiB for FAT32). Modifying the data also requires rewriting the whole monolith file, which can be time consuming. (Microsoft 2018; rgc:audio 2004.)

Non-monolithic storage method uses separate file types for the patch and audio data. The patch file contains the various parameters of the instrument, and also references to the audio files as absolute or relative paths in the file system. (Native Instruments 2017, 40-41, Hosken 2011, 241.) The files can be edited and saved individually, and multiple patches can reference the same audio files.

If the files are moved when using absolute path, or the relative path of the patch file and audio files changes, the sampler usually cannot find them and prompts for a location for resolving the file paths (Steinberg 2017, 49).

Many different proprietary formats exist for the data types. Some sampler software developers make the effort to support loading other developers' formats, but due to the variety of parameters that different samplers offer, total conversion of patch parameters included in various formats is not always possible. (Hosken 2011, 242; Howell 2006b.)

For the audio files, the commercial samplers and sample library developers typically employ their own proprietary data formats to encrypt them in order to prevent illegal file sharing (Howell 2006b). Some of the sample audio formats feature lossless compression. This can cut the file size approximately by half, depending on the codec used, but prevents accessing the files outside the sampler for tasks like editing. Besides yielding more compact sample sets, compressing the samples can have the benefit of faster loading times and **disk streaming** rates. Even though decompression requires additional CPU time, the retrieval of data from the hard disk might actually end up being the performance bottleneck during the loading process. (Native Instruments 2017, 41, 224).

10.6 Memory considerations

Historically, the memory capacity of the different devices used in sampling has always imposed limitations. As the technology advanced, so did the sample libraries, which usually meant more memory consumption.

The limits have been circumvented by adaptations such as using short loops instead of long sustain samples, or by playing the samples back from multiple external devices like racks of hardware samplers or remote-controlled slave computers (Howell 2005d; Vincent 2010; Wherry 2007).

With software samplers, a breakthrough on this front was made by Nemesys, the developers of the now discontinued Gigasampler. The first version, which was released in 1998, featured an innovative new technique that had been developed to get around the memory limitations: disk streaming (also referred to as direct-from-disk streaming or DFD). (Sound On Sound 2010; Vincent 2010.)

Instead of loading all the samples to RAM during initialization, the sampler streams the triggered samples directly from the hard drive. Unfortunately, hard drives are one of the slower forms of computer memory, and instantaneous access required by live playing is not possible. (Sound On Sound 2010; Wherry 2007.) However, DFD gets around this problem by buffering a small part of the beginning of every sample into RAM. The buffers are then played back on demand, while the computer looks up the rest of the audio file on the hard drive, and reads the remaining data into memory before the buffer has finished playing. The memory required by the buffer per sample is relatively small (measured typically in kilobytes) compared to loading the whole samples into RAM. (Native Instruments 2017, 223-224; Walker 2003.)

Even with the sample buffers, DFD still has the disadvantage that hard drive throughput can limit the number of simultaneous voices. The buffer size can usually be customized by the user to find a balance between performance and RAM usage - factoring in the speed of the hard drive (HDD vs. SSD), required voice count of the sampler patch in the arrangement, and the available RAM for buffering. Buffer underrun leads to dropouts in audio playback, so the buffer size is usually better set on the safe side. (Native Instruments 2017, 79; Walker 2003.)

Regarding the memory limitations of computers, older 32-bit systems could only allocate around 2GB memory for individual applications' usage. 64-bit systems were a significant development which expanded this virtual address space available to applications by orders of magnitude, practically all but eradicating this limit imposed by the architecture (the limiting factor now being the number of RAM modules one can fit in the motherboard). This development brought back the option to load large sample sets directly into RAM for extended polyphony capabilities. (Walker 2005; Wherry 2007.)

Nonetheless, the memory footprint of some very large orchestral libraries extends into the range of tens of gigabytes (Stewart 2017c). For these, disk streaming might still be the only reasonable option, especially if several libraries are used at the same time. Another consideration is that loading these massive sample sets into RAM can take quite a while.

11 CONCLUSIONS AND DISCUSSION

Overall, the research managed to provide the overview by logically structuring the different parts of the programming process. Additionally, it reviewed useful knowledge about low-level technical concepts relevant to the topic.

The main takeaway is that there are multiple dimensions to the sounds of acoustic instruments, and samplers try to emulate these properties with snapshots captured from the original source. For this they use rules that are designed to correlate the samples to musical information representing the different dimensions.

However, problems are often encountered due to the fact that the discrete set of snapshots is inadequate to represent the continuous variance in those different dimensions, if for example there are too few of them and/or they do not blend well together. Other possible considerations include the capability to adapt to different kinds of inputs like varying note lengths, and the necessity of providing the user ways with which to further manipulate the sound to emulate the flexibility of real instruments.

The techniques used in sampling solve these problems to various degrees and offer a number of sound quality benefits, but usually come with their own compromises like increased memory usage, increased difficulty during recording, phase cancellation problems, and undesirable artifacts like inconsistencies, unnaturalness or repetitiveness.

Awareness of these consequences is important when deciding whether to use the techniques. Also, the need for their application depends heavily on the source sounds. For some type of sounds the need for techniques like dynamic crossfading is easily justified. For others, it depends on various factors such as the tolerance of the sound of the instrument to the introduced compromises, aesthetic considerations, technical requirements of the intended application (e.g. music genre), and the need to impose limitations to the memory requirements of the patch for example.

The nature of the subject of sampling is such that many sources that go into details about the workflows and techniques in the different categories can include practical know-how, personal experiences and anecdotal information instead of theoretical approaches based on research and formalized knowledge, so their reliability is sometimes questionable. Also, arguments about sound quality for example don't always have an objective base, and many of the introduced concepts could benefit from more rigorous experimentation.

In the future, more work could be done in an attempt to generalize the different phases in the practice of sampling. One approach could be to use the Hornbostel-Sachs instrument classification method for example, to group together instruments with similar features, and to look for common ground in the workflows. When the process becomes better understood and formalized, it can speed up the creation of sample libraries and have a positive effect on the quality of the end results. It can also open new avenues for technology like automation, and lead to more intelligent features for the tools, which can benefit both the sample content creators and the end users.

REFERENCES

Accusonus. 2017. Drum bleed: what you need to know. Accessed 10.05.2018. https://accusonus.com/blog/drum-bleed-what-you-need-know

AKAI. 2000. s5000 / s6000 STEREO DIGITAL SAMPLER. Accessed 26.05.2018. https://www.manualslib.com/manual/629022/Akai-S6000.html?page=132

Baer, D. 2018. Review – HALion 6 from Steinberg. Accessed 10.05.2018. http://soundbytesmag.net/review-halion-6-steinberg/

Brazil, E. 2002. Cue Points: An Examination of Common Sound File Formats. Accessed 10.05.2018.
https://pdfs.semanticscholar.org/a8b1/32f270600e5dbcceacb919a26417d4100d03.pdf

Computer Music. 2016. 10 ways to get more out of velocity. Accessed 16.05.2018. https://www.musicradar.com/tuition/tech/10-ways-to-get-more-out-of-velocity-641607

Couchman, D. & Dearcangelis, C. 2016. The Production of Public Enemy: Gear, Sampling and Embracing Distortion. Accessed 25.04.2018.
https://reverb.com/news/the-production-of-public-enemy-gear-sampling-andembracing-distortion

Dunkley, J. & Houghton, M. 2011. Replacing & Reinforcing Recorded Drums. Accessed 10.05.2018.
https://www.soundonsound.com/techniques/replacing-reinforcing-recorded-drums

Elsea, P. 2013. The Art and Technique of Electroacoustic Music. Middleton: A-R Editions Inc.

Epand, L. 1976. A Phantom Orchestra at Your Fingertips. Accessed 12.04.2018. http://egrefin.free.fr/images/Chamberlin/HCInterview.pdf

FXpansion Audio UK Ltd. 2017. BFD3 Manual. Drum editor. Accessed 10.05.2018. https://www.fxpansion.com/webmanuals/bfd3/operationmanual/

Gardner, J. 2012. Interview - David Cockerell. Accessed 15.05.2018.
http://www.radionz.co.nz/concert/programmes/hopefulmachines/audio/201812323/interview-david-cockerell

Hoffner, R. 2003. Digital Audio Sample Rates: The 48 kHz Question. Accessed 29.04.2018.
https://www.tvtechnology.com/opinions/digital-audio-sample-rates-the-48-khz-question

Hosken, D. 2011. An Introduction to Music Technology. New York: Routledge.

Howell, S. 2005a. The Lost Art Of Sampling: Part 1. Accessed 10.05.2018. https://www.soundonsound.com/techniques/lost-art-sampling-part-1

Howell, S. 2005b. The Lost Art Of Sampling: Part 2. Accessed 10.05.2018. https://www.soundonsound.com/techniques/lost-art-sampling-part-2

Howell, S. 2005c. The Lost Art Of Sampling: Part 3. Accessed 13.05.2018. https://www.soundonsound.com/techniques/lost-art-sampling-part-3

Howell, S. 2005d. The Lost Art Of Sampling Part 4: Looping & Time-stretching. Accessed 14.05.2018.
https://www.soundonsound.com/techniques/lost-art-sampling

Howell, S. 2005e. Q. Did Mellotrons use tape loops or not? Accessed 15.05.2018. https://www.soundonsound.com/sound-advice/q-did-mellotrons-use-tape-loops-or-not

Howell, S. 2005f. The Lost Art Of Sampling: Part 5. Accessed 14.05.2018. https://www.soundonsound.com/techniques/lost-art-sampling-part-5

Howell, S. 2006b. The Lost Art Of Sampling: 7. Accessed 14.05.2018 https://www.soundonsound.com/techniques/lost-art-sampling-7

Huber, D. 2007. The MIDI Manual. Third edition. Burlington: Focal Press.

Jensen, B. 2016. Endless possibilities – in sound and music. Accessed 17.05.2018. https://nmh.no/en/about_nmh/news/endless-possibilities-in-sound-and-music/

Johnson, D. & Poyser, D. 1996. Steinberg Cubase VST. Accessed 15.04.2018. https://web.archive.org/web/20141102005500/http://www.soundon-sound.com/sos/1996_articles/jul96/steinbergcubase3.html

Langford, S. 2014. Digital Audio Editing. Burlington: Focal Press.

Magnus, N. 2016. NI Symphony Series Brass Collection. Accessed 13.05.2018. https://www.soundonsound.com/reviews/ni-symphony-series-brass-collection

Mantione, P. 2018. The Fundamentals of Sampling Instruments and Libraries. Accessed 26.05.2018.
https://theproaudiofiles.com/fundamentals-sampling-and-instruments-libraries/

Messitte, N. 2018. Live Drums vs. Sampled Drums: When to Use Which. Accessed 10.05.2018.
https://www.izotope.com/en/blog/music-production/live-drums-vs-sampled-drumswhen-to-use-which.html

Microsoft. 2018. File System Functionality Comparison. Accessed 18.05.2018. https://msdn.microsoft.com/enus/library/windows/desk-top/ee681827(v=vs.85).aspx#limits

Montgomery, C. 2012. 24/192 Music Downloads ...and why they make no sense. Accessed 29.04.2018.
https://people.xiph.org/~xiphmont/demo/neil-young.html

MOTU Inc. 2013. MachFive 3 User Guide. Accessed 10.05.2018.
https://s3.amazonaws.com/motu-wwwdata/manuals/software/vi/Mach-Five+3+User+Guide.pdf

Native Instruments GmbH. 2006. BATTERY 3 Operation Manual. Accessed 16.05.2018. https://www.nativeinstruments.com/fileadmin/ni_media/downloads/manuals/Battery_3_Manual_English.pdf

Native Instruments GmbH. 2017. Kontakt 5 Manual. Accessed 28.04.2018. https://www.nativeinstruments.com/fileadmin/ni_media/downloads/manuals/KONTAKT_5_6_8_Manual_English.pdf

Reid, G. 2002. Mellotron MkVI. Accessed 15.05.2018. https://www.soundonsound.com/reviews/mellotron-mkvi

rgc:audio. 2004. The sfz Format: Basics. Accessed 18.05.2018. http://www.sfzformat.com/legacy/

Russ, M. 2009. Sound Synthesis and Sampling. Third edition. Kidlington: Elsevier Ltd.

Solida, S. 2017. 20 hardware samplers that changed music production forever. Accessed 14.04.2018. https://www.musicradar.com/news/tech/20-hardware-samplers-thatchanged-music-production-forever-631613

Sound On Sound. 2010. 25 Products That Changed Recording. Accessed 29.05.2018. https://www.soundonsound.com/reviews/25-products-changed-recording

Steinberg Media Technologies GmbH. 2017. Halion 6 Operation Manual. Accessed 10.05.2018. https://download.steinberg.net/downloads_software/VSTi_HALion_6/HALion_6_Operation_Manual_en.pdf

Stewart, D. 2012. Arranging For Strings: Part 2. Accessed 17.05.2018. https://www.soundonsound.com/techniques/arranging-strings-part-2

Stewart, D. & Hearn, D. 2012. Arranging For Strings: Part 3. Accessed 17.05.2018. https://www.soundonsound.com/techniques/arranging-strings-part-3

Stewart, D. 2014. Spitfire Audio Mural Symphonic Strings. Accessed 17.05.2018. https://www.soundonsound.com/reviews/spitfire-audio-mural-symphonic-strings

Stewart, D. 2017a. The Sampled Orchestra: Part 1. Accessed 10.05.2018. https://www.soundonsound.com/techniques/sampled-orchestra-part1

Stewart, D. 2017c. The Sampled Orchestra: Part 3. Accessed 17.05.2018. https://www.soundonsound.com/techniques/sampled-orchestra-part3

Stewart, D. 2018c. The Sampled Orchestra: Part 9. Accessed 13.05.2018. https://www.soundonsound.com/techniques/sampled-orchestra-part9

Sweetwater. 2014. Sample Library. Accessed 25.05.2018. https://www.sweetwater.com/insync/sample-library/

Synthetic Orchestra Ltd. 2014. Tutorials » Spitfire Audio » Changing articulations. Accessed 17.05.2018.
http://syntheticorchestra.com/blog/?10

The Pennsylvania Gazette. 2013. Synthesizing Music and Science. Accessed 25.04.2018.
http://www.upenn.edu/gazette/0913/arts02.html

Thornton, M. 2007. Using Fades & Crossfades. Accessed 27.04.2018.
https://www.soundonsound.com/techniques/using-fades-crossfades

Toontrack Music AB. 2017. Superior Drummer 3 Manual. Accessed 10.05.2018.
https://www.toontrack.com/manual/superior-drummer-3/11/11-3-voice-and-layerproperty-box

UVI. 2018. UVI Script. Accessed 10.05.2018.
https://www.uvi.net/uviscript/

Vincent, R. 2010. Should Musicians Move To A 64-bit OS? Accessed 29.05.2018.
https://www.soundonsound.com/techniques/should-musicians-move-64-bit-os

Walker, M. 2002. Native Instruments Kontakt. Accessed 10.05.2018.
https://www.soundonsound.com/reviews/native-instruments-kontakt

Walker, M. 2003. Specifying A PC For Your Needs. Accessed 18.05.2018.
https://www.soundonsound.com/techniques/specifying-pc-your-needs

Walker, M. 2005. From 32-bit To 64-bit. Accessed 29.05.2018.
https://www.soundonsound.com/techniques/32-bit-64-bit

waveforms.fairlyconfusing.net. 2014. Fake random round robin script for Kontakt. Accessed 13.05.2018.
http://waveforms.fairlyconfusing.net/2014/09/fake-random-round-robin-script-in.html

Wherry, M. 2007. Scoring Pirates Of The Caribbean III. Accessed 29.05.2018.
https://www.soundonsound.com/techniques/scoring-pirates-caribbean-iii