The Lost Art Of Sampling: Part 2

By Steve Howell

As we saw in ***Part 1***, sampling is really just a form of digital recording — but to use short recordings to emulate real instruments, you soon need to understand new concepts like multisampling, looping, and velocity switching. We explain all...

In the ***first Part about The Lost Art Of Sampling***, we saw how digital recording lies at the heart of sampling and explained some basic digital recording concepts, including that of analogue-to-digital conversion and sampling rate. During the digital conversion process, an incoming audio waveform is sliced up into tiny sections and its amplitude in each of these slices (or samples) is measured and stored as a series of binary numbers that can be understood and manipulated by a digital audio processor and/or a computer. As we have seen, to reproduce any given frequency accurately with minimal side-effects, it is necessary to slice at the minimum of twice the rate of the highest frequency you wish to record digitally. As the upper limit of human hearing is supposedly 20kHz, the industry-standard sampling rate for CD-quality digital audio was set at 44,100 times per second (or 44.1kHz, as it is more usually written).

Bits & Pieces

However, there's more to digital recording than sampling rates. Whilst this rate determines the maximum *frequency* that can be accurately recorded, what determines the maximum amplitude, or volume, of the signal being sampled? This is determined by the analogue-to-digital converter's 'bit depth' or 'quantisation' (not to be confused with the timing quantisation found on sequencers). Fortunately, this concept is much less complicated than it sounds — basically, the bit depth refers to the number of binary bits (values of either 0 or 1) which are used to digitally store the waveform amplitude measurements in each of the thousands of samples made per second.

Just as the highest frequency it's possible to accurately record in a digital system increases as you up the sampling rate, so the accuracy with which you can digitally store the volume changes in an analogue signal increases as you up the number of binary bits used to store the amplitude measurements. At one extreme, if you use only a single bit, this can only have a value of 0 or 1 — so the signal is either on at maximum volume,

or off completely. Clearly, this isn't ideal for measuring subtle changes in the dynamics of a waveform! If you use two binary bits, you can store four different amplitude levels (00, 01, 10 and 11), and enjoy a system that recognises 'completely off' 'a third of maximum volume', 'two-thirds of maximum volume', and 'maximum volume'. Clearly, this isn't ideal either, but fortunately, as you increase the number of bits, the range of possible amplitude values (and thus the potential dynamic range of your digital recording system) increases *exponentially*. A three-bit recording system can store eight possible amplitude values, a four-bit system can store 16 possible values, and so on.

In the very early days of sampling technology (the late '70s and early '80s), samplers used eight-bit analogue-to-digital converters, which could assign only 256 (two to the power of eight) different amplitude values to any incoming audio. The use of these converters meant that these early samplers sounded pretty 'lumpy', particularly as they also worked at low sampling rates, which further affected the quality of the sampled sounds in an adverse way. You can see why, by taking a look at the graphs. If sampling rate determines the *horizontal* resolution on this graph (the accuracy with which the frequencies are depicted), then the bit depth or quantisation represents the *vertical* resolution, or the accuracy with which the amplitude is represented. An eight-bit system has, at its theoretical best, a dynamic range/signal-to-noise ratio of 49dB, which is less than that of a typical cassette recorder.

The effects of converting a sine waveform to digital at a low bit rate. A higher sampling rate wouldn't help this blocky waveform: this would mean there would be more individual samples, but the same low number of amplitude values (vertical values on this graph) to which the samples could be assigned.

Fortunately, as the '80s progressed, we saw the arrival of 12-bit samplers and then, of course, 16-bit samplers, and these were the standard through most of the 1990s. A 12-bit sampling system can handle 4096 different amplitude values and has a dynamic range of 60dB, while a 16-bit system can handle 65,536 amplitude values, and has a pretty respectable theoretical dynamic range of 96dB. Clearly, this is much more well suited to the level of dynamics in full-range musical or audio recordings, which is doubtless why the 16-bit, 44.1kHz sampling rate specification for CD digital audio survived for over a decade unchallenged.

With a higher bit rate, there are more possible amplitude values, so there's more vertical resolution to the converted digital waveform, and it resembles the input waveform more closely. With some filtering to smooth out the rough edges, the output here will sound almost indistinguishable from the original input waveform.

However, just as faster sampling rates of 96kHz and beyond are now being used, 24-bit converters and systems are now all the rage. And on the face of it, you can see why. A 24-bit sampling system can distinguish between over 16 *million* different amplitude levels (16,777,216, to be precise), and has a theoretical maximum dynamic range of 144dB — surely a massive improvement over a 16-bit system?

Well, theoretically, this *is* true, but as I did with higher sampling rates in ***Part 1***, I feel bound to point out a few things about greater bit depths. A symphony orchestra's dynamic range is around 110dB (from the quietest whisper of a mildly bowed violin to all the performers going at full tilt), so clearly, if you make your living from recording orchestral music, you may experience better results by using a 24-bit system. There are those that argue that the higher quantisation resolution of a 24-bit system is going to give a more accurate representation of the audio you're recording, and that's fair enough. I can certainly understand why someone would want to record and mix complete performances using 24-bit digital systems, especially when recording sources with a high dynamic range. And having a higher bit depth certainly offers greater resolution for digital signal processing and mixing. But in the context of *samples* — recordings of single notes triggered in different tonal combinations over MIDI, which can itself only offer a maximum of 128 possible velocity values, there is less to be gained. Instrument samples recorded at (or normalised to) 0dB in a 24-bit system and

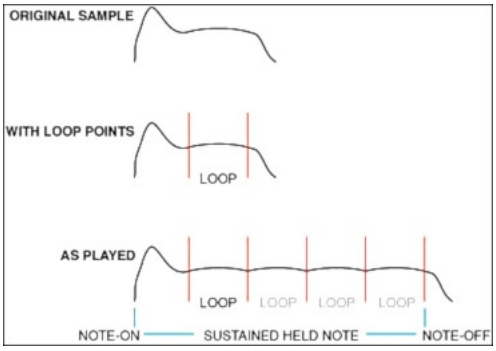
triggered via MIDI will sound pretty much exactly the same as their 16-bit counterparts! So recording a highly dynamic performance on a full drum kit at 24-bit resolution is perhaps a good idea, but recording a set of full-level samples of single drums from the kit will yield fewer benefits, in my opinion.

What's more, if you examine the real-world specs of a typical 24-bit D-A converter, the quoted dynamic range may be (at best) in the region of 118dB, which equates to a real-world performance a little less than that of a 20-bit system. And many D-As offer a dynamic range of only 102dB, which is not a huge improvement over the theoretical range of a 16-bit system!

Nevertheless, as with the argument over higher sampling rates, I can't deny that there are plenty of people who have listened to 24-bit samples and declare the sound superior to that of samples recorded on 16-bit systems. However, I'd be failing in my duty to fully explain the world of sampling if I didn't point out that sampling at 24-bit resolution, or playing back pre-recorded 24-bit libraries, can reduce the polyphony available in hardware samplers (assuming you can find one of the few that can cope with high-resolution audio) and/or put more strain on the CPU host in the case of software samplers. And 24-bit sample files themselves are larger, which can lead to increased loading times and fill your sample storage drives faster. Those caveats aside, the choice is yours.

Why Loop Instrument Samples?

Many people now regard the need to loop instrument samples as a thing of the past, and it's certainly true that it came about in the early days because of technical restrictions. When samplers first appeared, sample RAM was hideously expensive, and so most early samplers offered only a few seconds of sampling time, at most — figures of 512K or 750K of memory were touted as cutting-edge specs, and I recall trying to fit entire sampled grand pianos into 750K of RAM. Sampling short sounds, like bass-drum or snare hits, wasn't a problem, but sustained piano notes or long trumpet blasts were right out.Looping a sample to create a sustaining sound without having to take up additional sample memory.



The technique developed to get around this problem was looping. Most sounds, once they've completed the initial part of their development, or 'attack', settle into a sustaining phase of roughly even loudness, and by taking a section from this sustain phase and repeating it for as long as the sample was being triggered, sample designers could make their sounds sustain indefinitely without having to take up too much sample memory to achieve this (see the diagram on the right, which should make this clearer). Of course, the points at which the loop began and ended had to be chosen carefully, or the note would glitch or click each time it entered the loop cycle, so samplists developed a number of techniques (and sample-editing tools) for finding smooth-sounding loop points.

Of course, the memory restrictions of the early '80s are no longer with us, and these days, we have software samplers that can use the host computer's complement of RAM, meaning that we can now have several *Gigabytes* of sample RAM at our disposal if we wish. For the past few years, too, we've had software samplers that can stream samples directly from hard disk, so your samples can theoretically be as long as a hard disk can hold. As a result, many modern samplists make samples that are longer than they'll ever need them to sustain, and never have to create looped samples. Nevertheless, the technique can still be put to good use.

Sample libraries consisting entirely of full-length samples with no loops can be very large, and consequently very slow to load into RAM, and hard-disk streaming (sometimes known as a 'direct from disk' facility in some software samplers) places further strain on your host computer's CPU, not to mention the disk drive itself. Accessing sample data on a hard drive is also slower than accessing it from RAM, so polyphony (in other words, the number of sampled notes you can access simultaneously) can often be compromised and/or unpredictable. Acquiring good looping techniques can do much to optimise your sound library, making for faster load/save times and lessening storage requirements, as well as reducing demands on your computer's CPU. We'll look at these techniques later in this series.

What Else Is There To Know?

So now you understand the basic concepts behind digital recording, bit depths and sample rates, what more do you need to know? Well, if you're creating abstract and/or unpitched samples (for use as atmospheric washes or in *musique concrète*-type compositions), there isn't all that much more to know, other than how to capture the audio on (or import it to) the sampler of your choice, and how to set up the recordings so that they can be triggered from a keyboard, or other controller of your choice. I'll say more about all of those procedures in ***Part 3***. However, if, like a lot of people, you're interested in sampling as a means of gaining access to the sound of real instruments, by triggering short recordings of said instruments from a keyboard, then I should introduce a few other concepts before the end of this time's instalment, such as multisampling, looping (see the box below), and velocity switching.

The concept of being able to trigger digitally recorded instrument sounds from a keyboard was what really excited hi-tech musicians about samplers in the early '80s, and thinking back to some of the music technology history I covered in ***Part 1***, it's not hard to see why. The 1950s idea of being able to link recordings of real instruments at different pitches to the appropriate keys of a keyboard was tremendous, in theory giving keyboard players access to the sound of any instrument. But the analogue and mechanical attempts to realise this idea, such as the Chamberlin and Mellotron, had been a disappointment, broadly speaking (although they inspired a great deal of affection in those who could get access to them). Because they required different recordings, different tapes and therefore different tape-playback mechanisms to be associated with *each key*, they were, mechanically, tremendously complex beasts. They were therefore expensive, and prone to mechanical faults. But because the Fairlight stored its instrument recordings digitally, processing the recordings so that they could play back at different pitches was a relatively simple matter. Unlike the Mellotron, which required dozens of tapes of, say, a cello at different pitches, with the Fairlight, it was simple. Just sample the cello playing a middle 'C', and then slow the sound down or speed it up to play it at any pitch from the keyboard. The result — a perfect, digitally stored, keyboard-playable cello!

Except, of course, it wasn't. You only have to listen to some of the Fairlight string sounds on Kate Bush's 1980 album *Never For Ever*, made in the days when producers were in the first flush of their Fairlight infatuations, to realise that this idea was a mistake ('Army Dreamers' is particularly bad). And there are several reasons why...

Of Formants & Munchkins

Firstly, simply speeding up a sample to pitch it higher, or slowing it down to pitch it lower, creates exactly the same problems a tape-based recording has when it's varispeeded. We all know what happens when you speed up or slow down a recording of your voice by a large amount — the result sounds very unnatural and unrealistic, and the same is true of recordings of instruments.

The reason for this is that the sound of the human voice (and the sound of most instruments) contains what are known as 'formants': fixed-frequency 'resonances' that stay constant even when the pitch of the instrument or voice changes. For example, the sound of the oboe exhibits two fixed formant frequencies of 1400Hz and 3000Hz, which remain constant regardless of the note being played. When you speed up or slow down a tape-based or digital recording with no further processing, the pitch of everything, including the formants, is shifted, and it's the shifted formants that give the end result its unnatural character. In the context of sampling, this means that the sampled instrument sounds less and less like the real instrument the further up or down the keyboard you play from the pitch at which the note was originally sampled. This problem is known as 'munchkinsation', after the Munchkins, the little people in the classic film *The Wizard Of Oz*, whose peculiar squeaky voices were created by recording their voices on tape at one speed and playing the lines back faster.

But the problems don't end there. The timbre of many common acoustic instruments varies, sometimes quite dramatically, over their range. The piano is a prime example. When played normally, it sounds like a balanced instrument, but sit down and just play individual notes in isolation, from the lowest key to the highest, and you can hear the wide range of tones available. Low notes are deep and full, and last a long time. High notes, on the other hand, are short and 'plinky' in nature, lasting less than a second. The same can be said of a guitar. Pluck the lowest note and then the highest note to hear the difference. Also, just pluck the different open strings to hear tonal differences between them. Most instruments are the same, and exhibit tonal change across their range. Simply taking a piano sample at one note and pitching it up two octaves is not going to produce a result anything like that of a real piano being played two octaves further up, even allowing for munchkinisation.

And there are further problems. The very nature of playing back a sample slower or faster means that natural characteristics of the instrument, such as its attack and/or vibrato, or its decay, or release will also slow down and speed up the further any sample is transposed away from the original sampled note (the diagram opposite should make this clearer).

Creative Transposition

Sometimes the perfection of a beautifully multisampled instrument is not what you seek; the hideous distortions caused by transposing a sample far from its original pitch produce some delightfully creative sounds. Sometimes you don't even have to take things that far. For example, a single, sustained voice sample can sound fantastic played across the keyboard — it can be deep and menacing down low, or thin and ethereal higher up. Many of the vocal pad sounds you find on synths are just that — a single 'Ooooh' or 'Aaaah' looped and played across the keyboard. Possibly the most famous of these was the Fairlight IIx's 'Arr1', which featured on famous records of the '80s by artists such as Tears For Fears, Tom Dolby, Jan Hammer and others, and has been copied by manufacturers ever since.

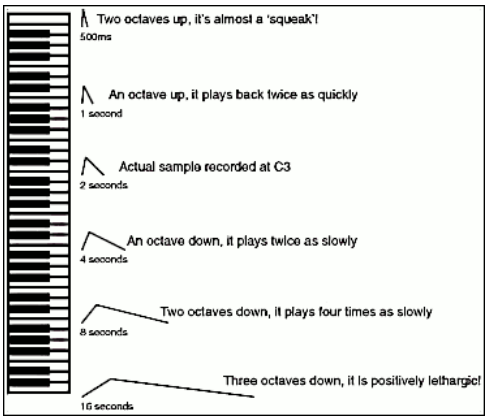
Other sounds, too, can sound great when played out of their natural range. More often than not, samples take on a better character when played lower (when played higher, everything speeds up, often with comic results). A middle 'C' on a piano pitched down three octaves and run through some reverb can be a powerful bass sound. Other sounds can also take on a life of their own when pitched down, and the most innocuous item can be transformed. For example, my mother has a stainless-steel washing-up bowl which I tapped with a beater and sampled some years ago. At its normal pitch, it sounded nice enough, with a pleasant tubular bell-like quality about it, but pitched down an octave or two, it turned into a surreal gong sound, especially when augmented by subtle flanging and reverb. I also partially filled the bowl with water and, as I struck it, I moved the bowl so that the water shifted and changed the pitch as the sound decayed. A few octaves down, this became an eerie, sinister sound that wouldn't be out of place in a Hollywood horror movie.

In fact, sound effects for films and TV programmes are often made like this — for example, a tub of Swarfega (a jelly-like substance for cleaning oil off your hands), when squelched about, sampled and pitched down a few octaves, can suddenly become a primordial swamp bubbling with volcanic activity. Film sound effects are also created using layers of different samples played at different pitches. In other words, unlikely objects can become quite interesting or even inspirational sound sources, especially when transposed up or down. Just think... your vacuum cleaner could be transformed into a menacing bass sound, or a squirt of your air freshener could be a good cabasa... and slamming your studio door could be just the snare-drum substitute you're looking for!

The Need For Interpolation

As if all these problems inherent in the sound weren't bad enough, the sampler you're using can itself contribute to the difficulties with transposition. In the very early days of sampling, samples would be repitched simply by altering the rate of the sampler's digital playback clock, and if you listen to 1970s samples, you can hear this. For example, if a sample was recorded at a sample rate of, say, 32kHz (a popular sampling rate in the early days, giving a bandwidth of around 15kHz), then when the sample was transposed down an octave, the sample clock would be running at 16kHz, in the upper reaches of our hearing range. If you played the sample a further octave down, the sample clock would operate at 8kHz, well within the range of human hearing, and exhibit itself as a high-pitched whistle. All sorts of filters were employed to keep this to a minimum, but you can hear it nevertheless. I have some original Fairlight samples, and you can hear the whistle gradually creeping in as you play lower down the keyboard.

To get around this, later samplers (and indeed all modern samplers) used a fixed sample-playback rate, and employed a process known as interpolation to allow samples to be transposed. When samples are played out slower, the interpolation process has to fill in the gaps to reconstruct the waveform as accurately as possible at lower pitches, and when played back higher, it has to seamlessly remove tiny snippets of data in order for the sound to be played back faster. And all of this takes place in real time!



A simple diagram showing how a single percussive sample, originally taken at C3, would speed up and down, and shorten and lengthen, as you play it over the range of the keyboard. Of course, in addition to the amplitude envelope being affected, other qualities inherent in the sound would also be altered by the transposition, such as vibrato and attack characteristics (scrape or rasp sounds, for example).

If this seems confusing, think of resizing a photo in an image editor such as Photoshop, which uses a similar process. If you enlarge the photo, the image editor has to somehow fill in the pixels so that the enlarged image isn't horribly distorted. It can't enlarge the actual pixels that make up the image; it has to interpolate and add new pixels that fit with the existing image and don't look out of place. It's similar when you're playing a sample lower in pitch — audio 'pixels' have to be inserted to create the longer transposed samples. And when you reduce the size of a picture in *Photoshop*, pixels have to be removed, because they can't be made smaller. It's much the same when transposing a sample upwards — audio 'pixels' somehow have to be removed. All of this is taken care of with a real-time interpolation algorithm.

Anyone who has used an image-editing package such as Photoshop will have noticed that there are often different interpolation algorithms that can be used to fill in/add or remove pixels with differing levels of quality. Different samplers also use different interpolation methods to play back sounds at different pitches. Some samplers use the most basic interpolation, and, as a result, it is not possible to transpose a sample far beyond the note at which it was sampled without interpolation distortion being quite evident. In cases like

these, transposing sounds several octaves up or down can render them almost unrecognisable, although these side-effects can be used in a positive way for creative purposes (see the box opposite). A good sampler should use high-quality interpolation algorithms that are far kinder on samples, even when they are transposed in either direction some distance from their base pitch.

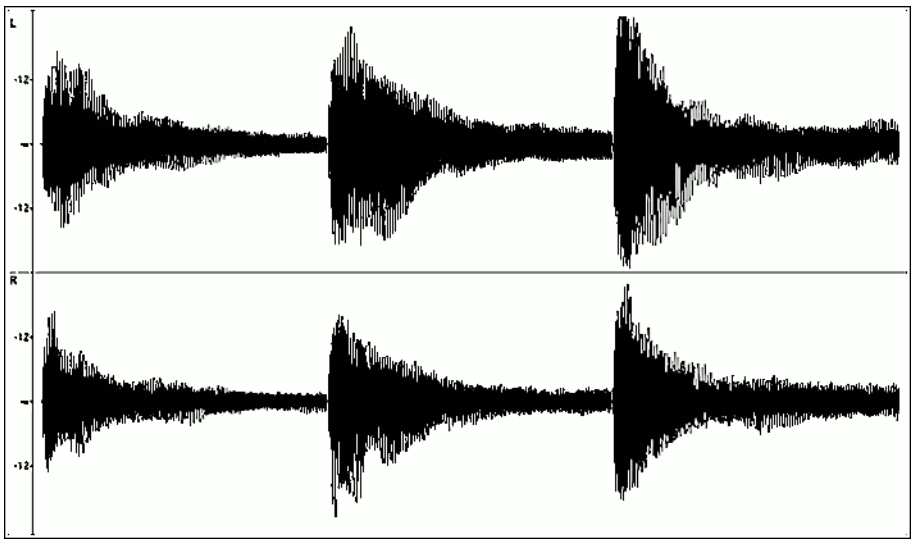
As samplers developed, though, the progression wasn't always smooth — for example, the Akai S1000 and S1100 used so-called 'eight-point windowed sinc interpolation', which was a good algorithm allowing a good deal of transposition in either direction, and which introduced artefacts only with extreme transpositions. But the later S2000 and S3000 family used linear interpolation, one of the most basic methods available, as a cost-cutting exercise to make the range of samplers more affordable. In practice, this meant that samples couldn't be transposed too far away from their base pitch without transposition artefacts being heard (a kind of metallic 'mush'). In my experience, hardware samplers seem to handle transposition better than software ones, perhaps because hardware samplers have dedicated circuitry built into them devoted to interpolation, and maybe also because the software that drives this hardware will often be written in the lowest level of machine code to ensure optimal performance under all circumstances, unlike the software interpolation 'emulators' responsible for transposition in a software sampler. Of course, low-quality interpolation will have no effect on recordings when they are played at their sampled pitch, but the usefulness of a sampler is reduced if it can't transpose audio too far away from its original pitch.

Multisampling

To overcome all these transposition-related problems, the concept of 'multisampling' was devised. Ironically, this involved returning to a situation more like that on the pre-digital Mellotron, where each note had its own associated tape recording. When multisampling an instrument, you make recordings of the instrument playing at several different pitches across its range, and then map the right ones to the appropriate keys of your keyboard. Fortunately, with decent transposition/interpolation algorithms, you don't usually have to go as far as taking a sample for *every* note, although you can if your sampler's memory allows this. It depends on the instrument you're sampling, and how realistic you want the sampled instrument to sound. With some instruments, two samples every octave — on 'C' and 'G', say — might suffice, although you might notice envelopes and other characteristics of the sound speeding up and slowing down as you move further from the original pitches of the samples. Of course, in these days of plentiful memory, it's tempting to have a separate sample for every note, although as with my comments in the box on looping a couple of pages back, I think such 'modern' practices make for very memory-intensive sample libraries, which can take a long time to load, and which can put a lot of strain on the host CPU in the case of software samplers.

When multisampling, I find a good compromise is to have a sample every minor third in every octave the instrument covers. This way, a sample is never transposed more than one semitone up or down, so you're unikely to hear any serious 'munchkinisation' or transposition distortion, and for the vast majority of instruments, this will be more than enough to give an accurate representation. Admittedly, if it's an instrument that will be featured in isolation, such as solo piano, this might not be sufficiently realisitic, but in a mix with other instruments, you'll find that this compromise works well enough even with potentially troublesome sound sources like the human voice, acoustic piano, saxophone and so on. From a practical and technical point of view, an instrument sampled this way is also going to be a third of the size of the same instrument sampled on every note. Even if you're not doing your own sampling, it can be a worthwhile exercise going through some of the enormous sample libraries currently available and trimming them down, using samples only every minor third to create your own 'lite' versions. These can be useful for situations where absolute realism isn't required — for use in a busy mix or for when playing live, for example.

Whilst recording multisamples seems to make a great deal of sense, it brings with it many considerations, not least of which is maintaining an even playing technique for each of the notes you sample. Obviously, if some notes in your multisample were played more loudly when they were sampled than others, then the sampled instrument will sound very strange when played up and down the keyboard, with different notes potentially sounding louder than others. Actually, overall level is less of a problem, as samples can all be normalised or otherwise level-balanced in the sampler once they've been taken, but the source instrument's tone will invariably be brighter or darker when played more loudly or softly, so it is important to play the instrument such that the recordings making up the multisample sound consistent.



One demonstration of why velocity switching is necessary. Shown here in a stereo editor are three separate performances of the same combination of notes (three G#s in different octaves), each louder than the last, from Alfred Brendel's classic 1972 piano performance of Movement 3 of Beethoven's 'Moonlight Sonata'. As you can see from the shape of the waveforms, even the same notes played more loudly or quietly on a grand piano produce very different waves, with the louder notes creating more high frequencies. You certainly can't achieve a realistic sampled piano sound by sampling the notes at one velocity, and making these samples play back more loudly with increasing velocity.

Velocity Switching

This brings us neatly to the final concept I'll introduce this time, namely velocity switching. As already mentioned, most instruments vary in the brightness (or otherwise) of their tone, but this is not not only the case across their frequency range — they can vary tremendously within one note, depending on how this is played. Once again, the piano is a good example of this; stroke a key lightly, and the hammer will barely hit the strings, creating a soft attack and a muted tone with few harmonics, because the strings haven't been 'excited' too much. However, hit the same note hard and you'll get a highly pronounced, percussive attack as the hammer smacks those strings, producing a brighter sound rich in upper harmonics which will also tend to sustain for longer. Sometimes, even the same notes played at similar volumes can sound very different, as the piano chords depicted in the screenshot opposite show.

The same is true of many other instruments. Guitar, bass guitar, electric piano... all have a lot of tonal variation within their dynamic range. And when stringed instruments such as the violin and cello are played softly, their attack is slow and languid, and the tone is smooth and somewhat muted — but when they're played aggressively, there is a pronounced 'scrape' during the attack and a rich, bright tone during the sustain portion of the sound. Flutes or pan pipes are also good examples — they produce a soft, almost sine-wave-like sound when played gently, with just a hint of breath noise, but they can be percussive and 'chiffy' when blown hard. You get the picture. For maximum realism, therefore, you should ideally sample any given instrument not only at multiple pitches, but also softly and loudly at all those pitches, in order to capture these different tonal characteristics. These samples at different 'velocities' can then be triggered from your controller by appropriate MIDI velocities, so that more softly played notes on your MIDI controller keyboard trigger suitably 'soft' samples, and so on. Hence the term velocity switching.