1. **Auralization**
   1. **Introduction**

The convolution is a mathematical operation on two functions that produces a third function which expresses how the shape of one function is modified by the other. It is defined as the integral of the product of the two functions after one is reversed and shifted. The integral is evaluated for all values of the shift, producing the convolution function. In acoustics these functions are audio signals or room impulses responses. Room impulse responses are generally referred as the transfer functions of the room which is known as system in term of signal processing. If the source signal and the system’s transfer function (impulse response) are obtained separately the resulting output signal can be calculated by convolution. The convolution can be processed in various ways, either directly in the time domain by using FIR filters or by using FFT convolution. However, it should be kept in mind that FFT requires fixed block lengths and is related to periodic signals. Time windows might be required for reducing artefacts from discontinuities. The same holds for framewise convolution in slowly time-variant systems. Also, the technique of convolution or “filtering” (IIR, FIR) is valid for LTI systems exclusively. For time-varying systems, the excitation signal must be processed in frames representing pieces of approximate time invariance. In this case, filters might be adapted while processing and fading must be used to move from frame to frame. There are different types of convolutions which are discussed in the next sections. Before discussing the type of convolution the basic concepts of digital filters is important to understood.



**Figure 1:** Block signal processing for auralization [**1**]

* 1. **Digital filters**

For many applications in audio processing we need modified versions of the sound (which is an audio signal) as per requirements. A non-optimal transfer function, for example, a transfer function with peaks and notches, can be improved using an equalization filters, resulting in a nearly flat magnitude response. Very often in acoustic signal processing we use linear time-invariant (LTI) systems and filters. These filters are commonly used, as the physics of sound propagation behaves nearly linear and time-invariant in the most relevant cases. We process the signals on computers, were they are stored as sampled, discrete-time signals. In other words, sequences of numbers. Hence the filters of interest are time-discrete, linear and time-invariant filters (sometimes referred to as linear shift-invariant). Such filters combine three elementary components: Amplifiers (multiplications), delay units (shifts or accumulators) and summations (additions). Depending on how these elements are combined, a variety of different filter classes are derived. In general, they can be classified as either finite-impulse response (FIR) filters or as infinite impulse response (IIR) filters. In both classes, we find several sub categories. FIR filters are purely feed forward networks and do not contain any loops or circuits in their block diagrams. In contrast, IIR filters make explicit use of such feedback loops. FIR filters offer maximum control over the entire transfer functions (in magnitude and phase). Hence, high quality audio processing usually implements FIR filtering. FIR filters are inherently stable (absence of loops).

The typical block diagram of a FIR filter is shown in Figure **2**. This layout is called direct form I or simpler a tapped-delay-line (TDL). The input samples are stored in delays (which are termed as accumulators). The latest sample and all past samples are weighted by filter coefficients (which are called taps) and then added up (i.e. a simple summation) to form the output samples . The number of filter coefficients (taps) is denoted by . The filter length minus one is called the order of the filter (e.g. for a fourth order filter).

C:\Users\fwefers\Cloud\Lehre\V4 Novo\figures (diss_fwe)\TDL.emf

**Figure 2:** Block diagram of a FIR filter in direct form I (so-called first form) also known as a tapped-delay-line

The output signal of the FIR filter is described by the following equation

|  |  |
| --- | --- |
|  | (1) |

The system function of a FIR filter (in the complex Z-domain) is directly derived from the block diagram in Figure 2.

|  |  |
| --- | --- |
|  | (2) |

* 1. **Fast convolution**

The concept of FFT-based fast convolution was proposed in 1965. Figure **3** depicts the algorithm for computing the linear convolution of a length signal and a length filter impulse response using the FFT.

A transform size is selected (usually the next larger power of two). Both operands and are appended with zeros (padding) to the length . Then these are transformed into the frequency domain using -point FFT. The DFT coefficients are multiplied pair by pair. The result is transformed back into the time domain using a -point IFFT. From the values only the left-most samples are extracted.

Unless the convolutions are very short (e.g. less than 32 samples), this algorithm is much faster than direct time-domain filtering. The FFT transform size is often chosen to be the next larger power of two. However, non-power-of-two sizes can be of an advantage if the transform size has a few simple prime factors in its prime factorization (e.g. ). The fastest size can only be determined by benchmarks on the target computer.



**Figure 3:** Block diagram of fast convolution using the Fast Fourier Transform (FFT)

* 1. **Real-time filtering**

For interactive acoustic virtual reality simulations the filtering process needs to be performed in real-time with low delays (which are also known as latencies). The audio processing on ordinary computer systems cannot be done in a sample-wise manner. Several stages of buffering exist between the CPU and the digital to analogue (DA) converters in the audio devices. This makes it necessary to process blocks of samples rather than individual samples. The data is exchanged with the audio hardware is performed using streams. An audio stream has a fixed sampling rate, for example , and consists of a continuous sequence of sample blocks, one sample block for each channel. Here the channel is representing the number of output streams and it depends on the reproduction system. For example, for headphone reproduction the number of channels are two (left channel and right channel). Each block contains the same number of samples which referred to the streaming block length or buffer size. Typical values for real-time applications lie down in the range of 64 to 512 samples. Given a sampling rate and a block length , the block duration is given by . The audio device continuously demands the next output block in intervals of seconds. Hence, all processing of the samples of length must be finished within this amount of time, otherwise, dropouts occur in the audio output stream.

* 1. **Running convolutions**

In real-time FIR filtering, a continuous audio stream is filtered with a finite impulse response of length . Here, the signal is provided in blocks of samples (see figure 4). All blocks have a fixed length B, which is initially adjusted with respect to the admissible latency. Typical values are 64, 128 or 256 samples. A real-time filter processes the signal block by block. For each length- input block it computes a length- output block . The introduced FFT technique computes a convolution in one step and requires that both operands (signal and filter) are available at entire length. Hence, it cannot be directly used for real-time filtering. It needs to be adapted, so that the convolution can be realized block by block as well. Therefore, two well-known principles can be used: The Overlap-Add (OLA) or the Overlap-Save (OLS) technique. Both are not restricted to the FFT approach and must be understood as an add-on to a conventional fast convolution.



**Figure 4:** Blocks of an audio stream (continuous audio signal).

* + 1. **Overlap-Add convolution**

This approach convolves each block of length of the signal with the filter of length . This yields intermediate results of lengths , which is longer than . The exceeding samples are accumulated in a buffer and added to later output blocks . Figure **5** illustrates the Overlap-Add (OLA) technique in conjunction with FFT convolution. To summarize it, The OLA technique takes the input blocks as they are and assembles the output from the intermediary convolution results. As shown in the Figure, the Filter with length N and the audio block with B samples are both zero-padded transformed into frequency domain by FFT, then they are multiplied and converted back to time domain by using IFFT. The result will be added to the cyclic buffer as shown in the figure.



**Figure 5:** Block diagram of Overlap-Add (OLA) convolution using the FFT.

* + 1. **Overlap-Save convolution**

The Overlap-Save (OLS) technique works exactly the other way around. Here, the input to the intermediary convolutions is prepared in a way that each convolution delivers a ready output block. The technique incorporates time-aliasing but directs it into ranges where it does not invalidate the convolution. It is depicted in Figure **6** and works as follows. The last input samples (which is known as sliding window) are convolved with the filter of length . This yields a result of samples. As the convolution is realized using a -point FFT, which implies that samples wrap around the period and consequently corrupt samples (which is shown as marked area in Figure **6**). The technique is called Overlap-Save, as these overlapping samples are discarded or in other words “saved”. The other samples mark a valid output block and they already contain the turn-off transients (which are the ‘decay’ from earlier input samples) of the preceding samples.



**Figure 6:** Block diagram of Overlap-Save (OLS) convolution using the FFT