SMOOTHING OF THE CONTROL SIGNAL WITHOUT CLIPPED OUTPUT IN DIGITAL PEAK LIMITERS

Perttu Hämäläinen

Telecommunications Software and Multimedia Laboratory
Helsinki University of Technology
P.O.Box 5400, FIN-02015 HUT, FINLAND
perttu.hamalainen@hut.fi

ABSTRACT

This paper studies the reduction of nonlinearity of digital peak limiters used for maximizing signal levels. The goal is to control the time-varying gain smoothly enough to avoid frequency artifacts in the output signal. Smoother gain control is traditionally obtained by lowpass filtering the gain or the signal envelope. However, simple filtering causes overshoots and leads to either clipped output or non-maximal signal levels, depending on the gain applied to the limiter output. In order to obtain smooth gain control without clipping, this paper proposes an envelope detection method based on order-statistics filtering.

1. INTRODUCTION

Compressors and limiters are forms of dynamics processing where the dynamic range of the output signal is altered by applying a time-varying gain which depends on the input signal. For a recent treatment on dynamics processing with several examples, the reader is referred to the book by Zölzer [1]. For simplicity, this paper only discusses peak limiters, but the results are also applicable to peak compressors. In peak limiters, the gain control is based on instantaneous signal level as opposed to limiters that analyze for example an RMS value computed in some time window.

A output of a limiter is formed as the product of the input signal and a time-varying gain signal. According to the convolution theorem, this corresponds to convolving of the spectra of the input and the gain and thus produces frequency artifacts. The faster the limiter reacts to signal peaks and the wider the spectrum of the gain signal is, the stronger the artifacts are.

The motivation for this paper was to design a limiter for maximizing signal levels and preventing clipping in interactive computer software such as games. Interactivity implies that the audio content depends on the user's actions and cannot be predicted. Because of this, the gain control should be as smooth as possible for no audible artifacts with any input signal. At the same time, the limiter should react infinitely fast to suppress signal peaks of any kind. To solve this contradiction, an improved envelope detection and gain control method is developed in this paper. Although limiter operation is generally well understood and there apparently are some related proprietary algorithms used for audio mastering, the author has found no previous literature on the topic.

This paper is organized as follows. Section 2 of the paper first reviews basic limiter operation. Section 3 reviews and proposes improvements. Processing blocks are added step by step until the final design is obtained. Finally, the computational efficiency of the proposed design is analyzed.

2. BASIC LIMITER OPERATION

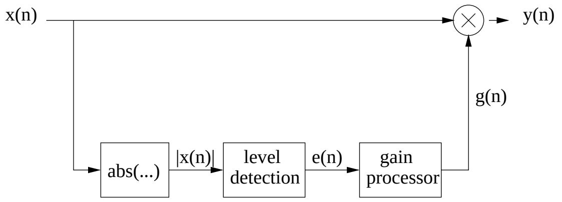

Figure 1: *Block diagram of a compressor/limiter.*

The basic operating principle of a limiter is that if the output signal exceeds a certain threshold level, gain is reduced. If the output signal stays below the threshold, gain is increased towards a certain maximum value, here considered as unity. Several alternatives for limiter design can be found in the literature [1][2][3][4]. One general compressor/limiter configuration is shown in Figure 1, adapted from Orfanidis [2]. Orfanidis suggests that the level or envelope detection block of the figure is implemented as a first order IIR lowpass filter

$$e(n) = a|x(n)| + (1-a)e(n-1). \tag{1}$$

The coefficient a determines how quickly the limiter reacts to signal changes. $e(n)$ is the envelope of the input signal, i.e. the *control signal* that is used to obtain the gain $g(n)$,

$$g(n) = \begin{cases} (e(n)/threshold)^{r-1} & \text{if } e(n) \geq threshold \\ 1 & \text{if } e(n) \leq threshold \end{cases} \tag{2}$$

The exponent r determines the compression ratio. To simplify the following treatment, we set $r = 0$ so that the limiter tries to completely keep $e(n)$ under the threshold and thus prevent clipping of the output signal. The gain can be then expressed as

$$g(n) = \min\left(1, \frac{threshold}{e(n)}\right). \tag{3}$$

Note that the coefficient a in equation (1) is usually not constant to allow different attack and release times. These user defined parameters indicate how quickly the limiter reacts to signal peaks by reducing gain and how quickly the gain is increased after the peaks.

The operation of a limiter obeying the above equations on a sinusoidal signal is illustrated in Figure 2. In the figure, $a = 0.2$

in attack mode, that is, for $|x(n)| > e(n-1)$, and $a = 0.01$ otherwise. Other limiter variants all work more or less similarly. Practically all limiters feature some kind of low-pass filtering to reduce distortion and produce a smoothly varying gain signal.

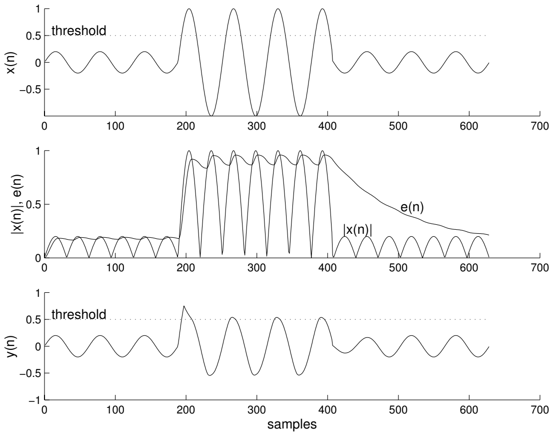

Figure 2: *Limiter operation on a sinusoidal input.*

3. ENVELOPE DETECTION WITHOUT CLIPPING

The limiter design proposed in this paper adds several processing blocks to Figure 1, resulting in the structure shown in Figure 4. The following describes the effect of the added blocks.

3.1. Delay in the Limiter's Output

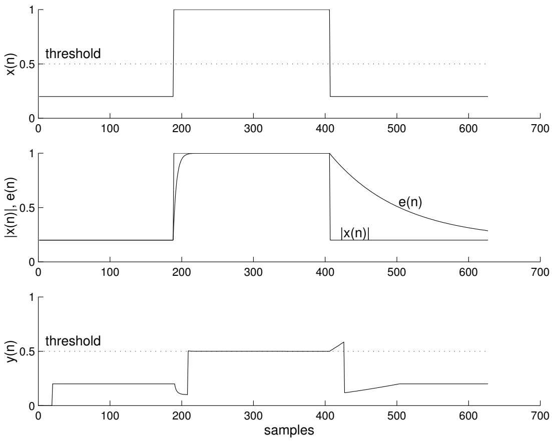

Figure 3: *Trying to prevent clipping using a delay.*

The basic requirement for preventing clipping is that the limiter's output stays below the threshold, $|x(n)g(n)| \leq threshold$. Assuming that $g(n) \geq 0$ and substituting equation (3) yields

$$e(n) \geq |x(n)|. \tag{4}$$

To guarantee this, the attack time should be set to zero, that is $a = 1$ for $|x(n)| > e(n-1)$ in equation (1). However, this produces sharp gain transitions that can be heard as clicks or other artifacts. McNally [3] suggests a feedforward configuration where the output of the limiter is delayed, as denoted by the z^{-N} block in Figure 4. The delay allows the limiter to decrease the gain in advance with a nonzero attack time. The limiter's output is in this case $y(n) = x(n-N)g(n)$ and equation (4) becomes

$$e(n) \geq |x(n-N)|. \tag{5}$$

However, the delay does not completely prevent clipping. The effect of the delay depends on the signal. Figure 3 shows limiter operation with a piecewise constant input, a delay of $N = 20$ samples, $a = 0.2$ in attack mode and and $a = 0.01$ otherwise. The output stays below the threshold when the input signal level increases rapidly, but the output peaks above the threshold when the limiter goes to release mode. Although the test signal is artificial, the case demonstrates that a delay is not a foolproof cure for clipping. Clipping will generally also happen if a peak is shorter than N samples. In that case the gain first begins to decrease, but when the peak ends, gain begins to increase even though the peak has not yet reached the output.

3.2. Adding a Max Filter

Smoothing of the limiter gain without clipping can be formulated as producing a control signal that satisfies equation (5) and that varies smoothly enough to prevent audible artifacts. Basically the function would be similar to a “cloth” hung on the peaks of $|x(n-N)|$ and solutions could be found in physical modelling of soft materials for computer graphics. The problem could also be viewed as an optimization problem of finding an $e(n)$ that maximizes some smoothness criteria and minimizes the error $|x(n-N)| - e(n)$, giving the error infinite weight if it is positive. The optimal curve would be sought observing the samples in the delay line of a feedforward limiter.

Having to optimize gain according to the contents of the delay line can cause a heavy computational load if the delay line is long. On the other hand, the shorter the delay, the less time the limiter has to react and the less smoother the gain control is.

The key idea used in this paper is to add a max filter (running max selection) and a clipping control block to the limiter sidechain before the level detection, as shown in Figure 4. The max filter is a special case of order statistics filtering, of which median filtering is probably more usual in audio signal processing. The main benefit of using a max filter is that the control signal can be determined based on the filter's output, instead of considering all the samples in the delay line at each n.

The max filter is defined here as operating on a history of input samples,

$$x_{max}(n) = max\left[c(n-N), ..., c(n)\right], \tag{6}$$

where c is the output of the clipping control block, explained later, and N is the filter order, here same as the delay length. The level detector equation (1) now becomes

$$e_{max}(n) = a x_{max}(n) + (1-a) e_{max}(n-1). \tag{7}$$

Considering that

$$x_{max}(m) \geq c(n) \quad \text{for } m = (n-N), ..., n, \tag{8}$$

and $a > 0$, we may write

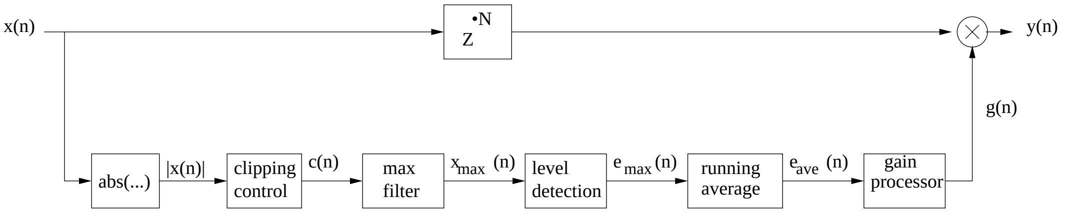

Figure 4: *Block diagram of the improved limiter, adding the clipping control, max filter and averaging.*

$$e_{max}(m) \geq e'(m), \tag{9}$$

$$e'(m) = \begin{cases} e_{max}(m) & \text{if } m = 0, ..., (n-1) \\ ac(n) + (1-a)e'(m-1) & \text{if } m = n, ..., (n+N) \end{cases} \tag{10}$$

In the latter case, the $c(n)$ term is constant and $e'(m)$ can be expressed in closed form using the inverse z-transform, resulting in

$$e'(m) = c(n) + \left[e_{max}(n-1) - c(n)\right](1-a)^{m-n+1}. \tag{11}$$

According to equations (9) and (5), requiring that $e'(n+N) \geq |x(n)|$ prevents clipping of the limiter's output. Solving for $c(n)$ yields

$$c(n) \geq \frac{|x(n)| - \beta e_{max}(n-1)}{1-\beta}, \tag{12}$$

$$\beta = (1-a)^{N+1}. \tag{13}$$

To allow the level detector block to determine the release behavior of the limiter, $c(n)$ should be at least equal to $|x(n)|$. Without violating equation (12), we may write

$$c(n) = max\left[|x(n)|, \frac{|x(n)| - \beta e_{max}(n-1)}{1-\beta}\right]. \tag{14}$$

The effect of feeding $c(n)$ to the max filter instead of $|x(n)|$ is to make the level detector “aim” a little higher than the actual input signal. This ensures that the limiter's output is not clipped. However, depending on the coefficient a, this may lead to overshoots at the envelope detector and thus non-maximal gain at the limiter's output. Let $\alpha = c(n)/|x(n)|$ denote the overshoot relative to the input signal. The maximum overshoot happens when $e_{max}(n-1) = 0$, that is,

$$\alpha_{max} = \frac{1}{1-\beta} \tag{15}$$

This gives us an estimate of the overshoot. Solving for a yields

$$a = 1 - 10^{\frac{log_{10}\frac{\alpha_{max}-1}{\alpha_{max}}}{N+1}}. \tag{16}$$

Now a can be chosen to limit the overshoot to a small value. Figures 5 and 6 show examples of limiting signals using the max filter. In the figures, $N = 20$ and $\alpha_{max} = 1.01$, resulting in $a = 0.1973$. Figure 7 offers a closer look at $|x(n-N)|$, $x_{max}(n)$ and $e_{max}(n)$.

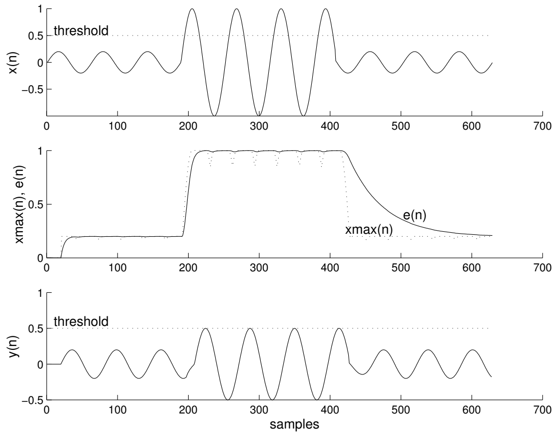

Figure 5: *Limiting sinusoidal input using a max filter.*

3.3. Filtering the Envelope with an Averaging Filter

Orfanidis [2] suggests that the control signal can be smoothed using a lowpass filter, for example a M point moving average. The averaging filter can also be used together with the max filter. It is inserted after the level detection, as shown in Figure 4. The output of the filter is

$$e_{ave}(n) = \frac{1}{M}\sum_{i=0}^{M-1} e_{max}(n-i). \tag{17}$$

Combining this with equation (9), we obtain

$$e_{ave}(m) \geq \frac{1}{M}\sum_{i=0}^{M-1} e'(m-i). \tag{18}$$

Similarly as before, equations (18) and (5) imply that the limiter's output is not clipped if $e'(n+N) \geq |x(n)|$. Substituting equation (11), solving for $c(n)$ and incorporating the max function as in equation (14) yields

$$c(n) = max\left[|x(n)|, \frac{|x(n)| - \beta e_{ave}(n-1)}{1-\beta}\right], \tag{19}$$

$$\beta = \frac{1}{M}\sum_{i=0}^{M-1}(1-a)^{N+1-i}. \tag{20}$$

Note that now a cannot be generally solved from equation (15), but in practice the equation (16) can be used or the value can be determined via experimentation. a can be fixed unless the user is allowed to adjust N.

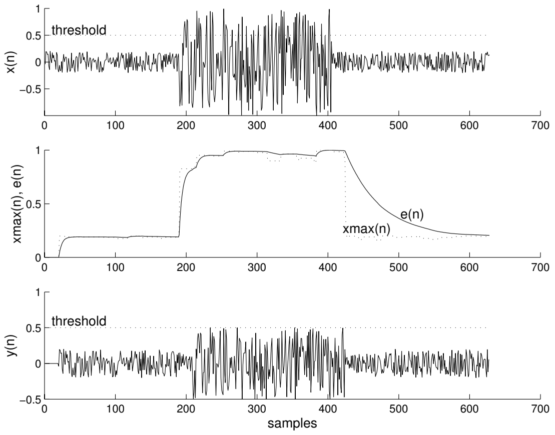

Figure 6: *Limiting random input using a max filter.*

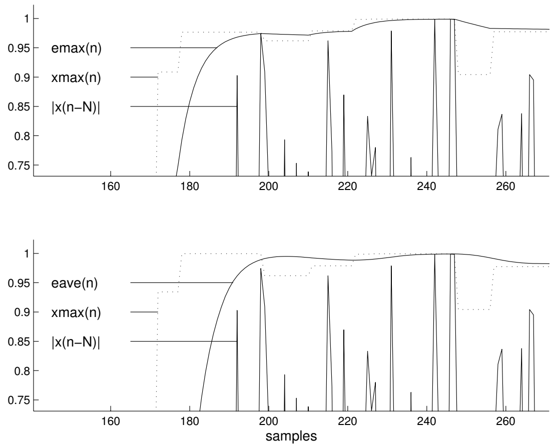

Figure 7: *Level detection using a max filter (above) and using a max filter and averaging (below).*

Figure 7 shows a close-up of level detection using a 12 point averaging filter.

3.4. Computational Cost

A direct implementation of the max filter requires $N - 1$ comparisons for each output sample. This may not be feasible if N is large. Fortunately, Pitas [5] describes fast algorithms that only need $\log_2 N$ comparisons or less, depending on the signal.

The author has not carried out listening tests to research suitable delay lengths, but as a personal opinion, $N = 64$ already gives good results.

In principle, the averaging also requires N operations. However, it is usually implemented efficiently in a recursive form,

$$e_{ave}(n) = e_{ave}(n-1) + \frac{1}{M}\left[(e_{max}(n) - e_{max}(n-M)\right]. \tag{21}$$

Each new sample is added to the average and the samples that come out of a M sample delay line are subtracted.

3.5. Design Options

Instead of the absolute value of the input signal, the envelope detection can operate on $max(threshold, |x(n)|)$. In this case, the min operator is removed from the gain processing stage, i.e. equation 3. This produces slightly different attack/release curves.

Another alternative is to reverse the order of the envelope detection and gain processing in Figure 1, as done by Lu [4]. In this case, $|x(n)|$ is substituted for $e(n)$ in equation 3. The result is the maximum allowed instantaneous gain g_{max}. The level detection block should now produce a smoothed gain so that $g(n) \leq g_{max}(n)$. In principle, this could be done using a similar structure as in Figure 4. However, the gain processing block should be moved right before the clipping control block, the max filter should be replaced with a min filter and all the inequalities should be reversed. One must also be aware that because of undershoots, gain may decrease below zero with low threshold values, especially if the averaging filter is used.

4. CONCLUSION

This paper described an envelope detection method that allows smooth gain control in peak compressors/limiters without the risk of signal clipping. The key improvement to traditional limiter design was to add a max filter, a special case of order-statistics filters, and a clipping control block to the envelope detector.

The main benefit of the proposed design is that it is fail-safe: the digital output signal can not be clipped no matter what the input signal is. This makes the proposed limiter design suitable for interactive computer applications, where the human computer interaction affects the soundscape, making it impossible to predict. The computational cost is also low, especially if a fast algorithm is used for the max filter.

The Matlab code and other resources used in this paper can be found at the author’s homepage, http://www.tml.hut.fi/~pjhamala.

5. ACKNOWLEDGEMENTS

The author is supported in the form of a graduate assistantship in the HeCSE graduate school of the Academy of Finland. The author would also like to thank Tommi Ilmonen, Tapio Lokki and Lauri Savioja for their comments.

6. REFERENCES

[1] Udo Zölzer, Ed., *Digital Audio Effects*, J. Wiley & Sons, 2002.

[2] Sophocles J. Orfanidis, *Introduction to Signal Processing*, Prentice Hall, 1995.

[3] G. W. McNally, “Dynamic Range Control of Digital Audio Signals,” *Journal of the AES*, vol. 32, no. 5, pp. 316–327, May 1984.

[4] Luzheng Lu, “A Digital Realization of Audio Dynamic Range Control,” in*Proceedings of Fourth International Conference on Signal Processing (ICSP’98)*, 1998, pp. 1424–1427.

[5] Ioannis Pitas, “Fast Algorithms for Running Ordering and Max/Min Calculation,” *IEEE Transactions on Circuits and Systems*, vol. 36, no. 6, pp. 795–803, 1989.