An adaptive spatial filter for additive Gaussian and impulse noise reduction in video signals

Yong Huang; Lucas Hui

STMicroelectronics Asia Pacific Pte Ltd

Abstract

This paper presents a new method for suppressing noise in video sequences. The filter is an extension of the sigma filter. It uses local measurements to obtain noise reduction while preserving the image edges and details. By using a special defined weighting function, this filter can reduce both Gaussian and impulse noise effectively.

1. Introduction

Video signals are often corrupted by noise in many different ways. Sources of noise are for example storage media, image acquisition (e.g cameras, digitizers), transmission channels (e.g broadcasting, satellite links and cable networks) and receiving and recording equipment. The noise has different characteristics, for example: additive white Gaussian noise from electronic equipment, impulse noise from over-the-air transmissions and packet-loss or frame loss in digital transmissions (e.g satellite transmission) over unreliable networks.

Noise reduction techniques have been applied to video signals to improve subjective viewing quality of the signals. However, most of the existing techniques are optimized for one specific type of noise, or have specific optimal parameter values for different types of the noise. Linear filters such as box filter and Gaussian filters are good at Gaussian noise reduction, but they tend to blur edges of objects and details. Non-linear filters use the order statistic neighbor operators such as median to replace the noisy pixels. Such filters cause no edge sharpness loss in the smoothed image. They are good at reducing impulse. In the recent past, a number of adaptive spatial filters have been proposed to minimize the conflict between sharpness and smoothing. The K-nearest neighbor filter [1] takes the mean of the K nearest neighbors from within the ordered list. The alpha-trimmed filter [2] takes the mean of the remaining entries in the ordered list once the first and last entries have been thrown away. The sigma filter [3] takes an average of only those neighboring pixels whose values lie within 2σ of the central pixel value, where sigma is found once for the whole image. This attempts to average a pixel with only those neighbors which have values “close” to it, compared with the image noise standard deviation. These adaptive filters are good at suppress the Gaussian noise while preserve the image edges. But they are still problem for impulse noise reduction.

Temporal noise reduction is usually performed using a temporal lowpass filter [4]. However such systems require complex hardware and huge memory for storage of the previous frames.

Adaptive filters that can reduce both the Gaussian and impulse noises had been studied in literature [5][6][7]. In this paper, we proposed an intraframe-based adaptive spatial filter. This filter is an extension of the sigma filter. It uses local measurements to obtain noise reduction while preserving the image edges and details. By using a special defined weighting function, this filter can reduce both Gaussian and impulse noise effectively.

2. The proposed filter

The architecture of the spatial filter is given in Figure 1. It is an extension of sigma filter.

![Fig. 1 The overview architecture of the proposed filter](image-url)

In Fig. 1, NLE is the noise level estimator. It is used to estimate the noise power for every frame processed. This parameter will be used in the noise reduction block to generate the filter parameters. SNF is the intrafield-based spatial filter used to reduce the noise from the chrominance signals. It is a general spatial filter which using the correlation between the current pixel and its neighborhoods.

The spatial filter is shown in Fig. 2. It is a recursive spatial filter and includes a working window and weighting block. The working window consists of the pixels X(i) (i=0,1,…,14) where X(0) is the central pixel that will be filtered. X(1)-X(14) are the neighborhoods of X(0). The 2D window is shown in Fig. 3.
In the weighting block, the output pixel can be calculated by:

\[
X_{out} = \frac{\sum_{i \in N_1} W_1(i) \cdot X(i) + \sum_{i \in N_2} W_2(i) \cdot X(i)}{\sum_{i \in N_1} W_1(i) + \sum_{i \in N_2} W_2(i)} \tag{1}
\]

where \( N_1 = \{1,2,...,7\} \) and \( N_2 = \{8,9,...,14\} \) are sets of vectors defining a neighborhood. \( W_1(i) \) and \( W_2(i) \) are weighs of each weighted pixel. They are defined as:

\[
W_1(i) = a_1 K(i) \quad i \in N_1 \tag{2}
\]

\[
W_2(i) = a_2 K(i) \quad i \in N_2 \tag{3}
\]

where \( a_1 \) and \( a_2 \) are the parameters related to the position of the weighted pixel. To use more information of filtered pixels, we have \( a_1 = 2^* a_2 \).

\( K(i) \) is a weighing function related to the noise power and the absolute difference between the weighted neighborhood pixel \( X(i) \) and the current input pixel \( X(0) \). This function is defined by:

\[
K(i) = e^{-\frac{(X(i) - (X(0) + \lambda))^2}{\text{sigma}}} \tag{4}
\]

where \( \lambda \) is a parameter related to the local noise power of current input pixel. It is ranged at \( -\text{sigma} \leq \lambda \leq \text{sigma} \). This parameter can be estimated by

\[
\lambda = \arg \min_{\lambda_j} \left\{ \sum_{i \in N_1} |X'(i) - (X(0) + \lambda_j)| + \right. \\
 \left. \sum_{i \in N_2} |X(i) - (X(0) + \lambda_j)| \right\} \tag{5}
\]

To reduce the computational complexity, we have \( \lambda_j = \{-\text{sigma}, 0, \text{sigma}\}, j=1,2,3 \).

The weighting function \( K(i) \) indicates that the contribution of a neighborhood pixel to the current center pixel is exponentially related to the difference between them. The smaller the difference, the more contribution the neighborhood pixel has. This property can eliminate the Gaussian noise effectively. In the case of the central pixel corrupted by impulse noise, due to the sum of the weights is just taken over the local neighborhood excluding the center pixel itself, this allows good reduction of impulse noise and do not blur the edges.

To reduce the implementation complexity, the weighting function \( K(i) \) can be simplified as:

\[
K(i) = \begin{cases} 
64 & \text{if } |X(i) - (X(0) + \lambda)| < Th1 \\
16 & \text{if } |X(i) - (X(0) + \lambda)| < Th2 \\
8 & \text{if } |X(i) - (X(0) + \lambda)| < Th3 \\
1 & \text{if } |X(i) - (X(0) + \lambda)| > Th3 
\end{cases} \tag{6}
\]

where \( Th1 = 1.5 \times \text{sigma} \); \( Th2 = 2.0 \times \text{sigma} \); \( Th3 = 2.5 \times \text{sigma} \).

### 3. Noise Level Estimator

The NLE is used to estimate the noise power (sigma) for every frame processed. Sigma is to be used in the noise reduction block and its value ranges from 0 to 31. The difference of every pixel with its previous and next pixel is calculated and the accumulated differences are delayed for 4 pixels and further accumulated into a register (SAD) (see Fig 4, Fig 5.). This value is compared with a noise range. Noise range is from half noise power to \( \frac{3}{4} \) noise power. For every frame processed, \( cnt \) (counter) will increment by 1.
whenever the SAD is within the noise range. Another counter is incremented by 1 whenever the cnr is less than the threshold; else it will be decremented by 1. This threshold is defined as:

\[ \text{threshold} = \frac{(\text{field width} \times \text{field height} \times 0.28)}{100} \quad (7) \]

Sigma is the mean of the last two values of this counter. The block diagram is shown in Fig. 6. For more detail of NLE process, please refer to [8].

\[ \text{previous pixel} \rightarrow \text{current pixel} \rightarrow \text{next pixel} \]

Fig. 4. The difference of every pixel with its previous and next pixel

\[ \text{absdiff} \rightarrow \text{Delay} \rightarrow \text{Delay} \rightarrow \text{Delay} \rightarrow \text{SAD} \]

Fig. 5. Block diagram of the SAD block used in NLE.

\[ \text{SAD} \rightarrow \text{Comparator A<SAD<B} \rightarrow \text{Counter cnt} \rightarrow \frac{1}{2} \rightarrow \frac{3}{4} \rightarrow \text{Sigma} \rightarrow \text{Counter} \rightarrow \text{Comparator cnt>threshold} \]

Fig 6. Block diagram of NLE

4. Experiment results

To evaluate the performance of the proposed method, we use a “clean” sequence and have corrupted it with the pure Gaussian noise (\(\sigma=20\)) and mixture of Gaussian and impulse noise (10%), respectively. The PSNRs of the noisy sequence, filtered sequence using sigma filter and the filtered sequence using proposed filter are shown in Figure 7 and 8. These figures display the PSNR of each frame in the sequence. Figure 7 is the PSNR results of the filtered sequence with pure Gaussian noise. Figure 8 is the PSNR results of the filtered sequence with the mixture noise. It can be seen from the results that the proposed filter performs better than sigma filter for all frames in the sequence.

Fig 7: PSNR of the noise sequence with pure Gaussian noise and the filtered sequences. (red line: noise sequence; green line: filtered sequence using sigma filter; blue line: filtered sequence using proposed filter).

Fig 8: PSNR of the noise sequence with pure Gaussian noise and the filtered sequences. (red line: noise sequence; green line: filtered sequence using sigma filter; blue line: filtered sequence using proposed filter).
Fig 9 to Fig 11 give the visual effects of one frame in the filtered sequences. Fig 9 is the noisy frame corrupted by the Gaussian and impulse noises. Fig 10 is the frame image after noise reduction using sigma filter. From this image we can see that the Gaussian noise is removed. However the impulse noise is still there. Fig 11 is the filtering result using the proposed method. It can be seen that both the Gaussian and impulse noise are reduced effectively.

5. Conclusion

An intraframe-based adaptive spatial filter was proposed in this paper. This filter uses local measurements to obtain noise reduction while preserving the image edges and details. By using a special defined weighting function, this filter can reduce both Gaussian and impulse noise effectively.

References


