Edge-preserving noise reduction in digital video sequences

EDGE-PRESERVING NOISE REDUCTION IN DIGITAL VIDEO SEQUENCES

P.J. van Otterloo*, A.J.E.M. Janssen*, C.B. Dekker**

* : Philips Research Laboratories, Eindhoven, The Netherlands
** : D.A. Consultant, Heerhugowaard, The Netherlands

1. INTRODUCTION

A number of techniques have been proposed to remove noise in digital images [1-4]. Most techniques deal with noise removal in single images. Our primary interest is to formulate a noise reduction method for sequences of images, in particular for video applications.

In digital video we have time sequences of two-dimensional images, with the complication that corresponding lines in consecutive fields are shifted over half a vertical sampling distance (2:1 interlacing). Though the method that we propose has been designed in principle for digital video applications, it is sufficiently flexible to allow for its application to still pictures or general 3-D data structures.

Our method differs from the one given in [4], where motion estimation in video sequences plays a crucial role. In the method of the present paper there is no explicit motion detection/estimation device; motion is taken into account implicitly by adapting the coefficients of a spatio-temporal smoothing filter to changes in both temporal and spatial directions.

2. Survey of the noise reduction procedure

In this paper we discuss only the noise reduction in the luminance component of video signals. Extensions to include also the color components in the noise reduction process can be formulated in a straightforward way. The proposed noise reduction process belongs to the class of edge-preserving noise smoothing techniques (cf. [1-3]).

The basic idea of such techniques is to smooth noise in relatively smooth regions while preserving luminance transitions (i.e. edges), which are important in creating the impression of picture sharpness. The method is more easily illustrated in two dimensions, as we will do for this moment. We start by defining a neighborhood $V(k_0,l_0)$ of pixels for a pixel (k_0,l_0), where k_0 and l_0 are the horizontal and vertical coordinate of the pixel (cf. Figure 1). For a pixel (k_0,l_0) we also define a set of neighbors $N(k_0,l_0)$ (cf. Figure 2). Please note that in this case $N(k_0,l_0) \subset V(k_0,l_0)$. This will always be the case: $N(k_0,l_0)$ indicates the set of direct neighbors of the pixel (k_0,l_0), while $V(k_0,l_0)$ comprises all pixels over which the smoothing takes place to compute an output value at position (k_0,l_0).

Our method consists of the following steps.

Step 1: determine for each neighboring pixel pair in $V(k_0,l_0)$ an edge value that indicates the amount of luminance difference (absolute value) between the two pixels constituting that pair.

Step 2: determine for each pixel $(k,l) \in V(k_0,l_0)$ the 'allowed' path from (k_0,l_0) to (k,l) with minimal total edge value. An 'allowed' path from (k_0,l_0) to (k,l) satisfies a no-return condition such that with each step in the path the distance to (k,l) decreases (cf. Figure 3).
The total edge value of a path is the sum of the edge values along that path. The total edge value of the minimal path between (k_0,l_0) and (k,l) is indicated as $a(k,l;k_0,l_0)$. Thus one obtains a kind of altitude chart or contour plot around (k_0,l_0).

Step 3: convert the contour plot values $a(k,l;k_0,l_0)$ to tap weight coefficients $h(k,l;k_0,l_0)$ by the conversion function

$$h(k,l;k_0,l_0) = \frac{b(k_0,l_0)}{1 + \left(\dfrac{a(k,l;k_0,l_0)}{\alpha}\right)^{\beta}} , \qquad (1)$$

where $b(k_0,l)$ is a scaling coefficient chosen such that

$$\sum_{V(k_0,l_0)} h(k,l;k_0,l_0) = 1 . \qquad (2)$$

Step 4: Compute the output pixel $y(k_0,l_0)$ as a weighted sum over the luminance values $x(k,l)$ in $V(k_0,l_0)$:

$$y(k_0,l_0) = \sum_{V(k_0,l_0)} h(k,l;k_0,l_0) \cdot x(k,l) . \qquad (3)$$

In Eq.1 we observe that for increasing values of a(;) the corresponding value of h(;) decreases. In fact, h(;) indicates in some sense to what extent $(k,l) \in V(k_0,l_0)$ and (k_0,l_0) belong to the same region in the picture. The constant α is chosen differently for different signal-to-noise ratio (SNR) levels of the pictures to be filtered. With low SNR levels α should be assigned a high value and with high SNR levels a low value. We shall return to this issue shortly.

The extension of the method just described to 3-dimensional (e.g. spatio-temporal) data is fairly straightforward. For each pixel (k_0,l_0,t_0) we choose a 3-D neighborhood $V(k_0,l_0,t_0)$ (cf. Figure 4) and a set of 3-D neighbors $N(k_0,l_0,t_0)$ (cf. Figure 5).

The altitude chart is constructed around (k_0,l_0,t_0) in the same way as just described under Step 2, using the same no-return condition to define 'allowed' paths. Subsequently Steps 3 and 4 are executed to yield the result $y(k_0,l_0,t_0)$. So far we have left the operators to measure the edge values unspecified. These are discussed in the following section.

3. Measurement of edge values

We have left the edge detectors, to measure the amount of luminance transition between 3-D neighbors in $V(k_0,l_0,t_0)$, still unspecified. For noise-free pictures one could simply use absolute pairwise luminance value differences between neighboring pixels. However, since we deal with noisy pictures, we must use edge detectors that exhibit some noise resistance. Based on experience with 2-D smoothing edge detectors [5] we have chosen the edge detectors e_H, e_V, e_D, e_{TV} and e_T in horizontal, vertical, diagonal, temporal-vertical and temporal direction, respectively, as described in Table I. Note in Table I that the chosen indexing in vertical direction takes care of the interlacing of the sequence of video fields. The locations of the edge detectors e_H, e_V, e_D, e_{TV} and e_T with respect to the sample position (k_0,l_0,t_0) have been indicated in Figure 5.

4. Choice of parameters

The altitude values a (;) are converted to tap weight coefficients h (;) by means the expression in Eq. 1. The coefficients h(;) are decreasing as a function of in creasing a(;). The parameter β determines how fast h(;) approaches zero as a function of a(;)/α. We have chosen $\beta = 2$. As explained earlier, the choice of the parameter $\alpha > 0$ depends on the SNR of the pictures to be filtered. We have performed experiments with the test sequence 'Wendt', which was also used in the COST-211 videoconference project. In our experiments we used additive white noise:

$$x(k,l.t) = s(k,l,t) + n(k,l,t) \ , \qquad (4)$$

where s( ) is the noise-free signal and n( ) the additive noise component. The noise satisfies $E(n(k,l,t)) = 0$ and $E(n\ (k,l,t)) = \sigma^2$. The noise-free pictures were quantised to 8 bits. As a measure for the SNR, we use

$$\mathrm{SNR}_{in}(t) = 10 \log_{10}\left[\frac{255^2}{\frac{1}{m}\sum_{k,l} n^2(k,l,t)}\right] , \qquad (5)$$

where the summation takes place over all pixels in the field at time t, and m is the total number of pixels in this field. We judge the performance of the algorithm by the gain in SNR:

$$\mathrm{SNR}_{gain}(t) = 10 \log_{10}\left[\frac{\sum_{k,l} |x(k,l,t) - s(k,l,t)|^2}{\sum_{k,l} |y(k,l,t) - s(k,l,t)|^2}\right] . \qquad (6)$$

In Figure 6a we have displayed the value of α_{opt} yielding the highest value of Eq. 6 as a function of SNR_{in} for field # 161 of the sequence 'Wendt'. In Figure 6b we have displayed SNR_{gain} for α_{opt} as a function of SNR_{in}. It was found that, for a fixed SNR_{in}, the fluctuation of α_{opt} for different video fields is quite modest, while also SNR_{gain} turned out to be rather insensitive to modest changes in α when SNR_{in} and the processed field are kept fixed. $\mathrm{SNR}_{gain}(t)$ varies from field to field within a range of about 15%. These variations are mainly due to the field dependent fraction of moving pixels. From Figure 6a we observe that α_{opt} varies considerably with SNR_{in} (displayed in the range 10-35 dB). It is important to know how α_{opt} varies with SNR_{in} since this offers the opportunity to adapt α in the conversion function h(;), Eq. 1, to the noise level presently measured.

CONCLUSIONS

In this paper an edge-preserving method for the enhancement of noisy sequences of pictures has been described and some experimental results have been reported. For a more detailed experimental analysis of the method we refer to [5]. The experimental results give rise to the following remarks.

Using the 5-point scale for picture quality [6], the method is able to restore noisy pictures of quality level 4 to a picture quality level 5 and of quality level 2 to quality level 4. For pictures with a very low SNR_{in} the method manages to make details, that were lost in the noise, recognizable.

An objection against the method is its complexity and its use of as much as 9 consecutive fields. It turns out that simplification of the 3-D edge detectors heavily affects the performance of the method. A reduction of the pixel neighborhood V(;) to 5 consecutive fields leads to a drop in SNR_{gain} of about 10%.

REFERENCES

1 M. Nagao and T. Matsuyama, Edge Preserving Smoothing, *Computer Graphics and Image processing* **9** (1979), pp. 394-407.

2 R.T. Chin and C.L. Yeh, Quantitative Evaluation of Some Edge-Preserving Noise-Smoothing Techniques, *Computer Vision, Graphics, and Image Processing* **23** (1983), pp. 67-91.

3 J.-S. Lee, Digital Image Smoothing and the Sigma Filter, *Computer Vision, Graphics, and Image Processing* **24** (1983), pp. 255-269.

4 T.S. Huang and Y.P. Hsu, *Image Sequence Enhancement, in: Image Sequence Analysis* (Ch.4), ed. T.S. Huang, Springer, 1981.

5 A.J.E.M. Janssen, P.J. van Otterloo and C.B. Dekker, The Contour Plot Method for Noise Reduction in Digital Video (accepted for publication in *Acta Electonica,* 1986).

6 W.K. Pratt, *Digital Image Processing,* Wiley, New York, 1978.

$$
\begin{aligned}
e_H(k,l,t) = 1/5 \; \Big| &\{x(k+1,l-1,t-1) + x(k+1,l+1,t-1) + x(k+1,l,t) + x(k+1,l-1,t+1) + x(k+1,l+1,t+1) \\
&- \{x(k,\, l-1,\, t+1) + x(k,\, l+1,\, t-1) + x(k,l,t) + x(k,\, l-1,\, t+1) + x(k,\, l+1,\, t+1)\}\Big|
\end{aligned}
$$

$$
\begin{aligned}
e_V(k,l,t) = 2/9 \; \Big| &\{x(k,l-3,t-1) + x(k,l-3,t+1) + x(k-1,l-2,t) + x(k,l-2,t) + x(k+1,l-2,t)\} \\
&- \{x(k,l+1,t-1) + x(k,l+1,t+1) + x(k-l,l,t) + x(k,l,t) + x(k+1,l,t)\}\Big|
\end{aligned}
$$

$$
\begin{aligned}
e_D(k,l,t) = 1/5 \; \Big| &\{x(k+1,l-3,t-1) + x(k+1,l-3,t+1) + x(k+1,l-2,t) + x(k+1,l-1,t-1) + x(k+1,l-1,t+1)\} \\
&- \{x(k,l-1,t-1) + x(k,l-1,t+1) + x(k,l,t) + x(k,l+1,t-1) + x(k,l+1,t-1)\}\Big|
\end{aligned}
$$

$$
\begin{aligned}
e_{TV}(k,l,t) = 1/5 \; \Big| &\{x(k,l-2,t) + x(k-1,l-1,t+1) + x(k,l-1,t+1) + x(k+1,l-1,t+1) + x(k,l,t+2)\} \\
&- \{x(k,l-1,t-1) + x(k-1,l.t) + x(k,l,t) + x(k,l+1,t+1)\}\Big|
\end{aligned}
$$

$$
\begin{aligned}
e_T(k,l,t) = 1/9 \; \Big| &\{x(k-1,l-2,t+2) + x(k,l-2,t+2) + x(k+1,l-2,t+2) + x(k-1,l,t+2) + x(k,l,t+2) \\
&+ x(k+1,l,t+2) + x(k-1,l+2,t+2) + x(k,l+2,t+2) + x(k+1,l+2,t+2)\} \\
&- \{x(k-1,l-2,t) + x(k,l-2,t) + x(k+1,l-2,t) + x(k-1,l,t) + x(k,l,t) + x(k+1,l,t) \\
&+ x(k-1,l+2,t) + x(k,l+2,t) + x(k+1,l+2,t)\}\Big|
\end{aligned}
$$

Table I. Spatio-temporal edge detectors

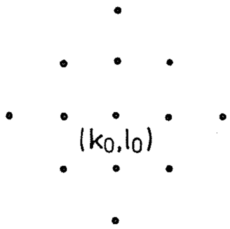

Figure 1.
Neighborhood $V(k_0,l_0)$ of the pixel with coordinates (k_0,l_0)

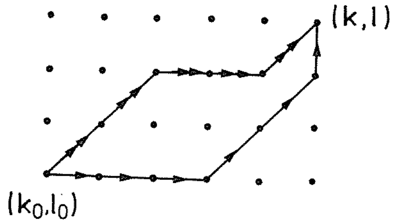

Figure 3.
Two 'allowed' paths from (k_0,l_0) to (k,l)

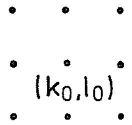

Figure 2.
Neighbors $N(k_0,l_0)$ of the pixel with coordinates (k_0,l_0)

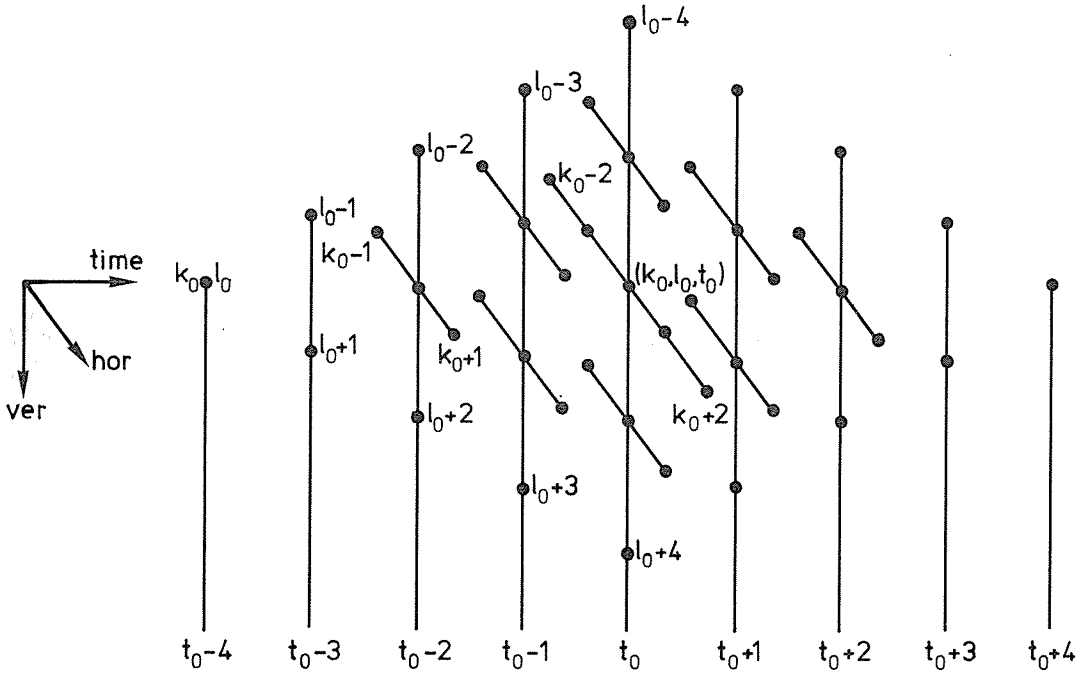

Figure 4.
Neighborhood $V(k_0,l_0,t_0)$ of the pixel with coordinates (k_0,l_0,t_0)

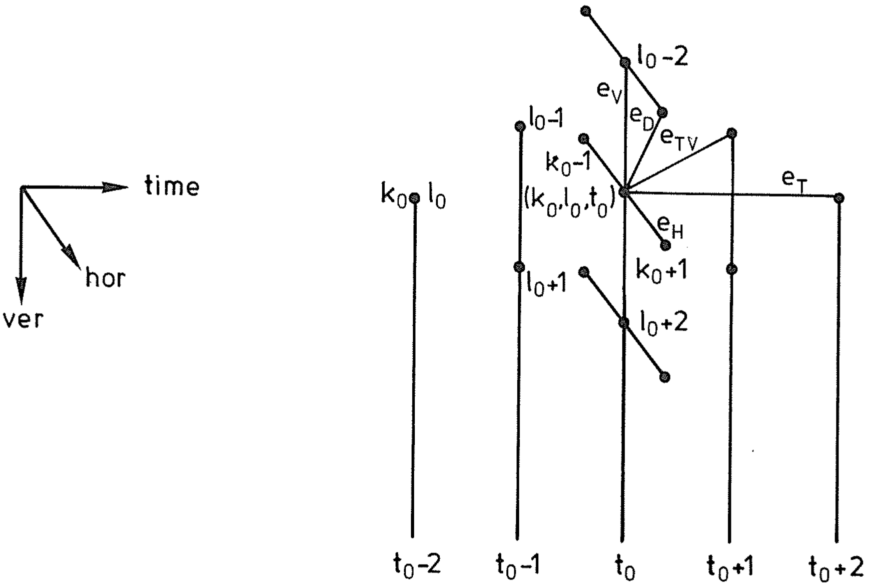

Figure 5.
Neighbors $N(k_0,l_0,t_0)$ of the pixel with coordinates (k_0,l_0,t_0)

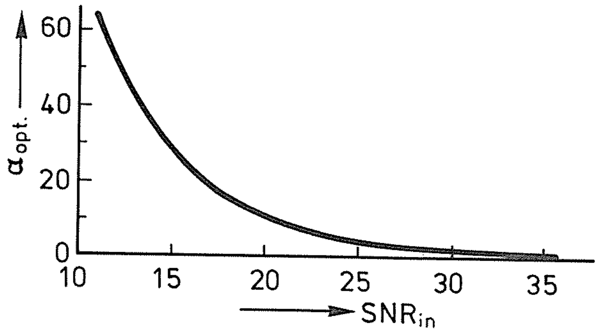

Figure 6a.
α_{opt} as a function of SNR$_{in}$ for field # 161 of the sequence 'Wendt'

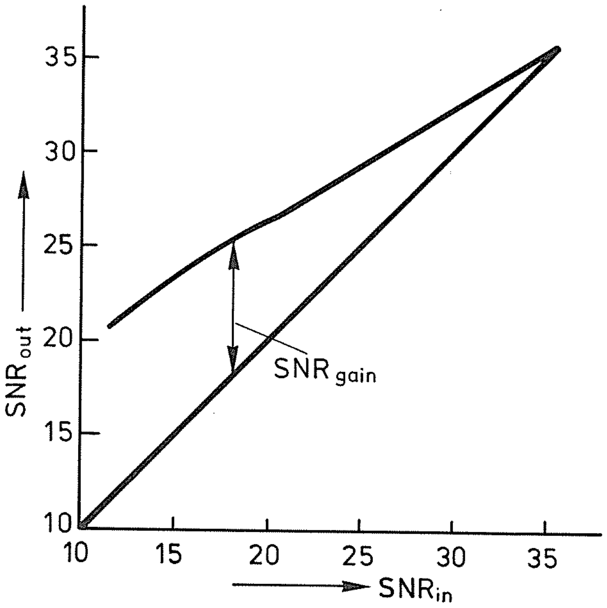

Figure 6b.
SNR$_{out}$ and SNR$_{gain}$, using α_{opt} as a function of SNR$_{in}$ for field # 161 of the sequence 'Wendt'