Denoising image sequences does not require motion estimation

Antoni Buades [*] [†] Bartomeu Coll [*] Jean Michel Morel [†]

Abstract

The state of the art movie restoration methods like AWA, LMMSE either estimate motion and filter out the trajectories, or compensate the motion by an optical flow estimate and then filter out the compensated movie. Now, the motion estimation problem is fundamentally ill-posed. This fact is known as the *aperture problem*: trajectories are ambiguous since they could coincide with any promenade in the space-time isophote surface. In this paper, we try to show that, for denoising, the aperture problem can be taken advantage of. Indeed, by the aperture problem, many pixels in the neighboring frames are similar to the current pixel one wishes to denoise. Thus, denoising by an averaging process can use many more pixels than just the ones on a single trajectory. This observation leads to use for movies a recently introduced denoising method, the NL-means algorithm. This static 3D algorithm outperforms motion compensated algorithms, as it does not lose movie details. It involves the whole movie isophote, including the current frame, and not just a trajectory. Experimental evidence will be given that it also improves the "dirt and sparkle" detection algorithms.

1 Introduction

1.1 Sequences and noise

Let us define a continuous image sequence $u(x, y, \theta)$ as a bounded function defined on a domain $\Omega \subset \mathbb{R}^3$. A discrete sequence is just a sampling of $u(x, y, \theta)$ for a discrete set of times $t_1, \cdots, t_n$. For a fixed time t, the image $u(i, j, t)$ is defined on a finite grid of pixels, each one denoted by $\mathbf{x} = (i, j, t)$.

Figure 1 displays three consecutive frames of a degraded image sequence. These degradations are due to several reasons. When digital video is produced directly from the scene, noise is introduced by the acquisition equipment. Each one of the pixel values $u(i, j, t)$ is the result of a light intensity measurement.

[*]Universitat de les Illes Balears, Ctra. Valldemossa Km. 7.5, 07122 Palma de Mallorca, Spain

[†]Centre de Mathématiques et Leurs Applications. ENS Cachan 61, Av du Président Wilson 94235 Cachan, France

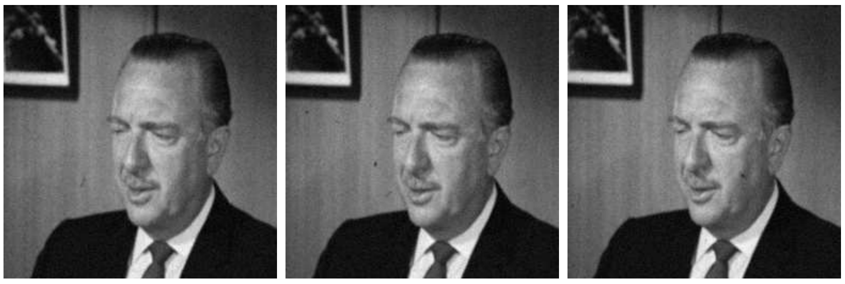


Figure 1: Three consecutive frames of a degraded image sequence. Two types of degradations are distinguished: slight fluctuations of the grey level values (noise) and local artifacts which change abruptly position ("dirt and sparkle"). The sparse time sampling in film sequences makes restoration more difficult than in 3D images.

When the light source is constant, the number of photons received by each pixel fluctuates around its average. In a first rough approximation one can write

$$u(i,j,t) = u_0(i,j,t) + n(i,j,t),$$

where u is the observed sequence, u_0 is the original sequence without noise and $n(i,j,t)$ is the noise perturbation.

Film and videotape material is also subject to degradations with time and repeated use which introduce noise and artifacts. These degradations are even enhanced by the digitization process. The sequence of Figure 1 presents a type of degradation familiarly called "dirt and sparkle". This degradation appears under the form of stains which affect a relatively small zone and whose position is random. Thus, they change abruptly position and are seldom superposed in consecutive frames.

What makes particularly difficult the sequence restoration is the movement. If an image sequence is completely static, then a simple temporal averaging would be an excellent estimation. Unfortunately, if some region of the scene moves, the temporal averaging will blur it. For this reason, recent filtering techniques are adapted to the dynamic character of image sequences. The two main approaches for avoiding the temporal artifacts are the *adaptivity* and the *motion compensation*. Adaptive algorithms implicitly take into account the motion assumption in the design of the method. Motion compensation algorithms compute a motion estimation as a preprocessing step.

1.2 Filtering methods

In local filtering methods, the restoration of a certain pixel $\mathbf{x}$ only depends on the value of its neighboring pixels. The set of pixels involved in the restoration of $\mathbf{x}$ is called the *support of* $\boldsymbol{x}$, $S_{\mathbf{x}}$. When the support $S_{\mathbf{x}}$ involves only pixels belonging to the same frame, we talk about *spatial filters*. In that case, each image of the sequence is filtered independently of the rest. Such an approach neglects the temporal correlation between consecutive frames and does not exploit the huge redundancy of image sequences. *Spatiotemporal filters* take advantage of the correlations that exist in both the spatial and temporal directions. In that case, the support of a pixel involves pixels in the current frame but also in the previous and posterior ones. This fact leads to a more coherent filtering of the sequence. Finally, *temporal filters* exploit only the correlation in the temporal direction.

In this work, we are mainly concerned with spatiotemporal filters. We shall distinguish three classes of filters: *static filters*, *adaptive filters* and *motion compensated filters.* The *static filters* are straightforward extensions of image filters. These filters are obtained by extending the 2D image support to a 3D space-time support. Even though they are the simplest algorithms, they are crucial for the understanding of the sequence filtering problem. The *adaptive filters* are designed specifically to deal with image sequences and take into account the possibility of a motion. The *motion compensated filters* estimate explicitly the motion of the sequence by a motion estimation algorithm. The trajectories obtained by the previous estimation yield a new stationary data. The most recent articles use the motion compensation strategy. This seems a necessary step to obtain good results in sequence filtering. However, the motion estimation is a very difficult problem itself and there is no algorithm able to give a final solution. In fact, the motion compensation can propose inaccurate trajectories and a filtering along these trajectories a blur and an information loss.

We shall sustain the position that, in fact, motion estimation is not only unnecessary, but probably counterproductive. The aperture problem, viewed as a general phenomenon in movies, can be positively interpreted in the following way : there are many pixels in the next or previous frame which can match the current pixel. Now by the law of large numbers, a denoising method works so much the better if the estimation of the current pixel is based on many, and not just a few pixels. Thus, it seems sound to use not just one trajectory, but rather all similar pixels to the current pixel across time and space.

This leads us back to a classical filter, namely the sigma-filter [17], or neighborhood filter [31], or SUSAN [25], more recently renamed bilateral filter [26]. The main idea of this spatial filter is to denoise the current pixel by taking an average of all neighboring pixels whose grey level value is close enough. If one applies a sigma-filter to the space-time data, one noticeably averages on the whole space-time isophote. As we shall see, this static filter gives excellent results, but which can still be improved by piling up some motion compensation.

In this paper, we adapt a recently introduced nontrivial extension of the sigma-filter, the NL-means algorithm [4, 5]. This spatial algorithm defines a generalized isophote on which noise reduction is performed by averaging. In the case of movies, as we shall see, this *static filter is no more improvable by motion compensation.*

The plan of this paper derives from the above remarks. In section 2 we review some classical image denoising algorithms and their extension to the 3D time-space support. In section 3 we review classical adaptive algorithms. In section 4 we introduce motion estimation algorithms and the aperture problem. These algorithms are used in section 5 to compensate the algorithms of previous sections. Experiments show the improvement of motion compensated versions of static filters. In section 6 we deal with the "dirt and sparkle problem". For a sake of completeness, we describe a simple algorithm which removes these local artifacts. In section 7 we introduce the NL-means algorithm and its extension to image sequence filtering. The experiments compare the performance of the various algorithms for noise filtering. Finally, we discuss the combination of NL-means with the removal of "dirt and sparkle".

2 Static filters

The static filters are obtained by extending the 2D image filter support to the 3D time-space support. We call them static because they do not take into account the dynamic character of image sequences. Many image denoising algorithms have been proposed in the literature. However, only a few ones can be easily extended to the 3D support, namely the local average filters.

The support of a pixel $\mathbf{x} = (i_0, j_0, t_0)$ is given by

$$S_{\mathbf{x}} = \{\mathbf{y} = (i,j,t) \mid |i - i_0| \leq \delta_i,\ |j - j_0| \leq \delta_j \text{ and } |t - t_0| \leq \delta_t\},$$

where $\delta_i, \delta_j, \delta_t > 0$. Then, static filters can be written as

$$\hat{u}(\mathbf{y}) = \frac{1}{C(\mathbf{x})} \sum_{\mathbf{y} \in S_{\mathbf{x}}} w(\mathbf{x}, \mathbf{y}) u(\mathbf{y}), \tag{1}$$

where $\hat{u}$ denotes the restored sequence and $C(\mathbf{x}) = \sum_{\mathbf{y} \in S_{\mathbf{x}}} w(\mathbf{x}, \mathbf{y})$ is the normalizing term. A median filter is just obtained by setting a selected pixel weight to one and all others to zero.

The most classical example is the arithmetic mean. The spatiotemporal extension simply sets all the weights in the support to 1. We shall denote this filter by *MFu*. Like the two dimensional version, the algorithm blurs the edges and removes many details. These artifacts are aggravated on moving regions, see Figure 2.

The blurring of the previous method can be avoided by analyzing the performed mean. When the averaging is performed on a contour or texture, the variance of the grey levels of the averaged pixels can become larger than the variance of the noise. A more conservative estimate can be obtained by the following method, due to Lee [16].

Let us assume that the observed value $u(\mathbf{x})$ is the realization of a random variable X. Let us suppose that $X = X_0 + N$ where X_0 is the true value and N is the noise value. Then, the best linear estimate in a minimum least square error sense is:

$$\hat{X} = EX_0 + \frac{\sigma_{X_0}^2}{\sigma_{X_0}^2 + \sigma_N^2}(X - EX_0),$$

where EX_0 denotes the expectation of the variable X_0, σ_{X_0} its standard deviation and σ_n the noise standard deviation. The expectation EX_0 is replaced by the mean MFu and the variance of the original variable is approximated by

$$\sigma_{X_0}^2 = max(0, \sigma_X^2 - \sigma_N^2),$$

where σ_X^2 is estimated on the sample sequence

$$\sigma_X^2 = \frac{1}{|S_{\mathbf{x}}|} \sum_{\mathbf{y} \in S_{\mathbf{x}}} (u(\mathbf{y}) - MFu(\mathbf{x}))^2.$$

The original noisy values are less altered near the boundaries or textured regions, where σ_X is large. Thus, the noise is mainly reduced in flat zones where σ_X and σ_N take close values, see Figure 2.

The neighborhood (or sigma-) filters [17, 31] also avoid the blurring effect of the mean filter. Neighborhood filters are based on the assumption that pixels belonging to a same region have a similar grey level value. Therefore, one should restrict the average to pixels with a small grey level difference with the one in restoration. The filter is written under the average form (1) by taking the weights

$$w(\mathbf{x}, \mathbf{y}) = e^{\frac{-(u(\mathbf{y}) - u(\mathbf{x}))^2}{h^2}}, \tag{2}$$

where $h > 0$ controls the degree of filtering of the restored sequence. This parameter controls the decay of the exponential in function of the grey level distances, and therefore the decay of the weights. The neighborhood filter is a better denoising tool than Lee's correction. It maintains sharp boundaries, since it averages pixels belonging to the same region as the reference pixel. Unfortunately, this method fails when the standard deviation of the noise exceeds the contrast of edges.

Notice that the three above mentioned classes of static filters are not adapted to the removal of local artifacts such as the "dirt and sparkle". The mean filter reduces these artifacts because it averages all the pixels inside the spatiotemporal support. The cost of this reduction is the blurring of boundaries and details. Since the mean performed at local artifacts has a large variance, the Lee statistical correction performs a more conservative estimate and keeps the original values. Thus, it maintains dirt and sparkle. Since the neighborhood filters average pixels with a similar grey level value, they also keep these artifacts.

The median filter [28] chooses the median value, that is, the value which has exactly the same number of grey level values above and below in a fixed neighborhood. This filter is optimal for the removal of impulse noise on images. Figure 2 shows that this algorithm is able to reduce the additive noise and to remove as well dirt and sparkle. The median filter preserves the main boundaries, but it tends to remove the details and to blur the boundaries of moving regions.

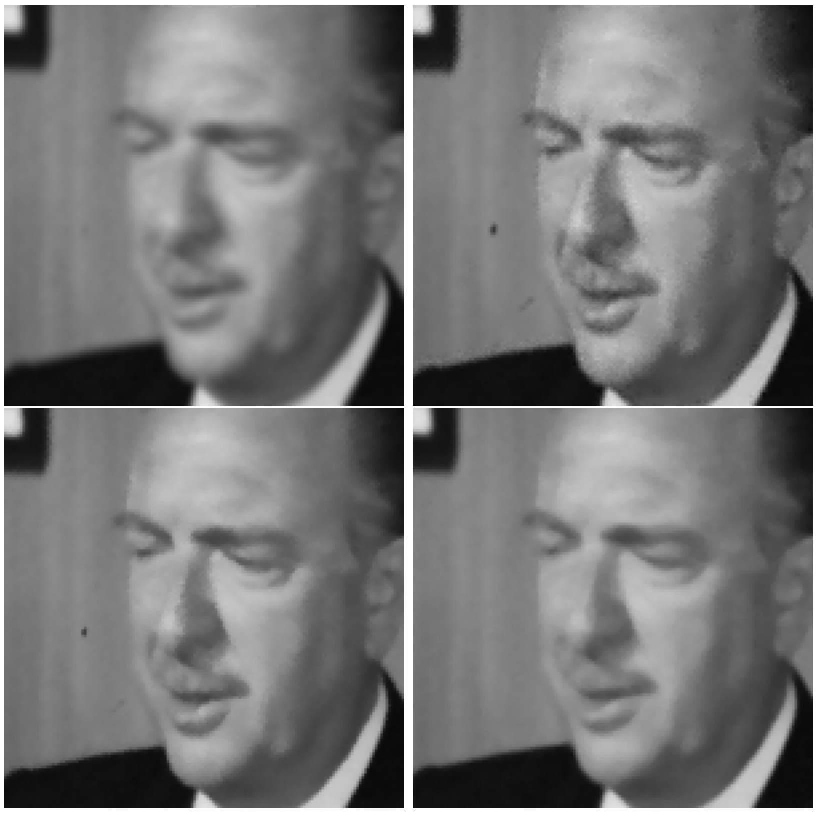


Figure 2: Application of static filters to the sequence of Figure 1 (only the central frame is displayed). From left to right and from top to bottom: mean filter, Lee's statistical correction, the neighborhood filter and the median filter. All the experiments have been performed with a spatiotemporal support of $\delta_i = \delta_j = 3$ pixels and $\delta_t = 2$ frames. The mean filter averages all the pixels in the support, thus blurring the edges and moving regions. The statistical correction performs a more conservative estimate when a large variance of the mean is observed. As a consequence, the edges and boundaries between moving regions are kept noisy. The neighborhood filter does not blur the image as it averages only pixels with a similar grey level. However, many isolated noisy points are still visible. All of these methods reduce the noise but are not able to remove the "dirt and sparkle" effect. The median filter preserves the edges and at the same time reduces the noise and the local artifacts. However, the result is blurred because of the poor adaptation to movement.

3 Adaptive filters

Adaptive filters take into account the dynamic character of image sequences but do not compute explicitly the optical flow. The aim of these filters is to avoid the blurring effect where motion occurs. The objective of this section is not to make a review of adaptive filters, but to give some simple ideas about them. For a more complete description see [3].

Adaptive median filters have been proposed in [1, 2, 15]. First, several median filters are performed on different supports. Each support has a different topology in order to preserve a fixed direction or pattern. Then, a new median operation is performed on a subset of the previous computed medians.

Recursive filters follow a similar strategy to the statistical correction of the mean filter. They are written as

$$\hat{u}_a(i,j,t) = \hat{u}_b(i,j,t) + c(i,j,t)(u(i,j,t) - \hat{u}_b(i,j,t)),$$

where $\hat{u}_b(i,j,t)$ is a rough approximation, "before updating", $\hat{u}_a(i,j,t)$ is the final estimate, "after updating", and the control value $c(i,j,t) \in [0,1]$. These filters perform a more conservative estimate when any movement is detected. The rough approximation is computed using the filtered values of previous frames. The estimate before updating is often chosen as [20, 7, 8]:

$$\hat{u}_b(i,j,t) = \hat{u}_a(i,j,t-1)$$

which implies a recursion in the temporal direction. In that case, when $c(i,j,t) = 0$ the filtered value of the previous frame is forwarded while when $c(i,j,t) = 1$ the actual noisy value is kept. The differences between the several approaches lie in the choice of the control parameter $c(i,j,t)$. Intuitively the value $c(i,j,t)$ should depend on the difference between the actual noise value and the forwarded value $e = |u(i,j,k) - \hat{u}_b(i,j,t)|$. A large difference may be due to motion or scene changes. Different functions of e have been proposed: a threshold function [20], a truncated linear function [8] or a non linear function [7].

Recursive filters can be improved by using *a priori* information of the image sequence model. Katsaggelos [13] and Triplicane [27] proposed the following auto regressive model for the original sequence:

$$u(i,j,t) = \sum_{(p,q,l)\in A} a(p,q,l)u(i-p,j-q,t-l) + n(i,j,t),$$

where n is a signal independent white noise and $a(p,q,l)$ are the fixed model coefficients. This model means that each sequence value is written as a combination of the values in previous frames plus a signal independent noise. The model is used for the computation of the rough approximation $\hat{u}_b$ which is estimated as

$$\hat{u}_b(i,j,t) = \sum_{(p,q,l)\in A} a(p,q,l)\hat{u}_a(i-p,j-q,t-l).$$

Then, the control parameter $c(i,j,t)$ is chosen to be optimal in a mean square error sense.

Recursive filters estimate the value of a pixel $\mathbf{x}$ depending on the filtered values of previous frames. This approach is the only one available when dealing with real time applications where we only know the previous frames. However, if we already handle a whole sequence or if some delay is allowed, this approach doesn't take advantage of the information in the forward direction.

Adaptive filters may also be preferred when speed is required or the sequence is highly corrupted. The motion compensation filters have a high computational cost and the accuracy of motion estimation algorithms decreases with noise [3]. However, when dealing with moderated noise, motion compensation filter are usually preferred. In fact, many papers dealing with adaptive filters conclude that a motion compensation is required to overcome the temporal artifacts. In the next section, we introduce the motion estimation problem and two different approaches. The availability of an accurate motion estimation is crucial for the performance of the motion compensation filters.

4 Motion estimation

In this short review, we shall deal with two different approaches, the *optical flow constraint (OFC)* based methods and *the block matching* algorithms. The first class of methods assumes that the grey level value of the objects during their trajectory is nearly constant (Lambertian assumption). The second strategy computes the displacement at each pixel by comparing the grey level values in a whole block around it. One of the major difficulties in motion estimation is the ambiguity of trajectories, the so called *aperture problem.* This problem is illustrated in Figure 3. In many pixels we have several options for the displacement vector. All of these options have a similar grey level value and a similar block around them. Now, motion estimations are forced to select one by some additional criterion.

The OFC based methods assume that the grey level of a pixel is nearly the same on each trajectory. This assumption leads to the optical flow equation

$$u_x v + u_y w + u_\theta = 0$$

where u_x and u_y denote the partial derivatives of u in the spatial directions and u_θ the temporal derivative. The vectors v and w denote respectively the displacement functions in the horizontal and vertical directions. This single equation is not sufficient to determine the unknown functions v and w (the aperture problem). In order to obtain a unique flow field, a second constraint is needed. Such a constraint may impose the flow field to vary smoothly in space. The flow is obtained by minimizing the energy

$$E(v,w) = \int_\Omega (u_x v + u_y w + u_\theta)^2 + \alpha \int_\Omega \Psi(|\nabla v|^2 + |\nabla w|^2)\, dx\, dy$$

where the parameter $\alpha > 0$ controls the tradeoff between the OFC constraint and the regularization term. Larger values of α give more weight to the regularization term leading to a smoother optic flow. The original work of Horn and

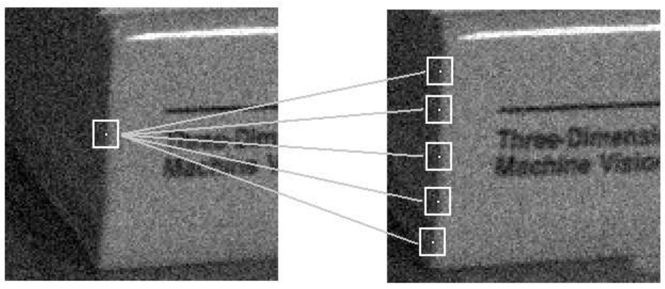


Figure 3: Aperture problem: The ambiguity of trajectories is the most difficult problem in motion estimation. Many good candidates are possible and the motion estimation algorithms must choose one.

Schunk [11] considered the function $\Psi(x) = x$ which leads to an spatial isotropic smoothing of the flow. As a consequence, the flow field is smoothed across the boundaries and therefore blurred. Many modifications have been proposed to avoid this smoothing across the boundaries, see for instance [21]. Non linear regularizing functionals Ψ have also been considered, see for instance [29]. This type of algorithms is local in time, and two frames are in general enough to compute the optical flow. In this work, we shall use the method developed by Weickert and Schnorr [30]. The spatial regularization term is replaced by a spatiotemporal one. The minimized energy is now a three dimensional integral whose solution is the optical flow for all the frames $t \in [O, T]$:

$$\begin{aligned} E(v,w) &= \int_{\Omega\times[0,T]} (u_x v + u_y w + u_\theta)^2 \\ &\quad + \alpha \int_{\Omega\times[0,T]} \Psi(|\nabla_3 v|^2 + |\nabla_3 w|^2)\, dx\, dy\, d\theta. \end{aligned}$$

The spatiotemporal regularization improves the vector fields significantly, smoothes out background noise, and preserves the true motion boundaries. In the experiments, we shall use the nonlinear convex function proposed in the mentioned work,

$$\Psi(x) = \epsilon x + (1-\epsilon)\lambda^2\sqrt{1 + s/\lambda^2},$$

where $0 < \epsilon << 1$ and $\lambda > 0$.

The second model we shall involve in comparisons is the block-matching. The image is divided into blocks of a general rectangular shape. For each block in the current frame the block of pixels in the reference frame which is the most similar is searched for. The similarity is measured by l^1 or a l^2 distance. Then, a constant displacement is assumed in each block. In order to achieve the most accurate motion estimation at each pixel, we have implemented a slower version of this algorithm. For every pixel $\mathbf{x} = (i_0, j_0, t_0)$, we take a block centered at $\mathbf{x}$ and size $N \times N$. Then, we search the pixel in the reference frame which has

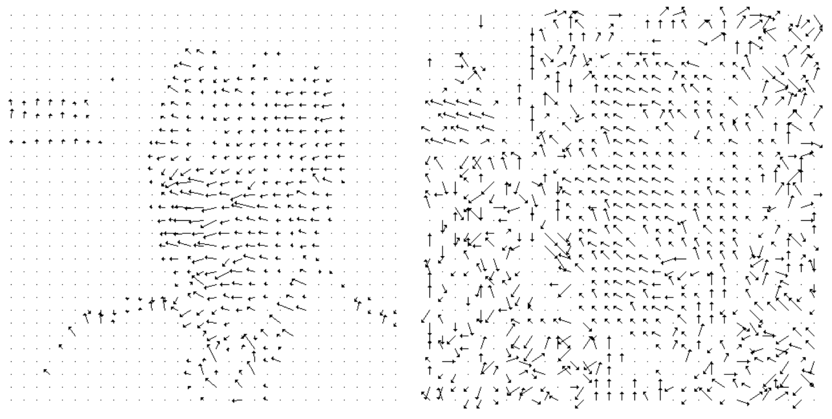


Figure 4: Comparison of the optical flow obtained by the Weickert and Schnorr algorithm (left) and the block matching algorithm (right). The regularity condition of the OFC based methods yields a smooth optical flow. The non-presence of this regularity term in the block matching can lead to a chaotic optical flow field. Indeed, by the aperture problem, trajectories on isophotes are essentially ambiguous and therefore the choice performed by block matching close a random walk.

a more similar block to the one centered in $\mathbf{x}$. The displacement vector at $\mathbf{x}$, (v, w), minimizes

$$\sum_m \sum_n (u(i_0 + n, j_0 + m, t_0) - u(i_0 + v + n, j_0 + w + m, t_0 + \Delta t))^2$$

where Δt represents the time interval between the current frame and the reference one.

As Figure 3 shows, there can be many blocks with similar configurations in the reference frame. Then, the algorithm chooses the closest one. That does not ensure we choose the right pixel, particularly when the sequence is noisy. In contrast with the (OFC) based algorithms, there is no constraint on the regularity of the obtained flow. Thus, pixels of the same region can have very different displacements vectors.

Figure 4 displays the motion estimation of the sequence in Figure 1. The flow obtained by the Weickert and Schnorr algorithm is more coherent while the block matching flow seems to be less accurate and chaotic.

5 Motion compensated filters

Nearly all the recently proposed image sequence filters are motion compensated. It is assumed that, in order to deal with the dynamic character of sequences and to obtain high quality results, a motion estimation is necessary. The underlying idea is the existence of a "ground true" physical motion, which motion estimation algorithms should be able to estimate. Legitimate information should exist only along these physical trajectories.

A motion estimate of the sequence is computed previously to the application of the algorithm. For a pixel $\mathbf{x} = (i_0, j_0, t_0)$, let v^t and w^t denote the horizontal and vertical displacements between the pixel $\mathbf{x}$ and the corresponding pixel in the frame t. Then, the new support is obtained by centering the spatial window at each frame t at $(i_0 + v^t, j_0 + w^t)$

$$S_{\mathbf{x}} = \{(i,j,t) \mid |i - (i_0 + v^t)| \le \delta_i,\ |j - (j_0 + w^t)| \le \delta_j \text{ and } |t - t_0| \le \delta_t\},$$

where $\delta_i, \delta_j, \delta_t > 0$. Following this scheme, all of the previous static filters can be compensated. Many of these motion compensated denoising algorithms have been directly proposed in the literature. Samy [23] and Sezan et al. [24] proposed the LMMSE filter which is a motion compensation of the statistical correction of the mean filter. Ozkan et al [22] proposed the AWA filter which is in fact a motion compensation of the neighborhood filter. These authors do not use the exponential function as in (2) but another decreasing function of the grey level differences. Huang [12] and Martinez [19] implemented motion compensated median filters. Motion compensated Wiener filters were also proposed by Kokaram [15].

The experiments illustrate the improvement of motion compensated algorithms compared with static and adaptive filters. Figure 5 compares the mean and median static filters and their motion compensated versions. The motion compensated mean filter uses the optical flow computed by the Weickert and Schnorr algorithm. The motion compensated median filter uses the block matching estimation. In both cases, the motion compensation improves the static version, avoiding the excessive blurring of the static filters. The details are also better preserved with motion compensation. This explains why most recent papers propose motion compensated algorithms.

Figure 6 compares the motion compensated versions of the neighborhood filter and the statistical correction of the mean. We compare the estimates obtained by the two motion estimation algorithms. Both motion compensation versions improve their static version of Figure 2 and lead to high quality restored sequences. However, the filtered sequence using the block matching algorithm better preserves the details and is less blurred than the filtered with the Weickert and Schnorr algorithm. The block matching motion estimation is more accurate near the edges. As a consequence, the statistical correction is able to reduce more noise when the block matching is used. When using the Weickert and Schnorr method the algorithm is forced to be more conservative and keeps much more noise. When dealing with the neighborhood filter the block matching also better preserves the details. In spite of its chaotic optical flow (Figure 4) it seems that the block comparison is more robust when dealing with noisy sequences.

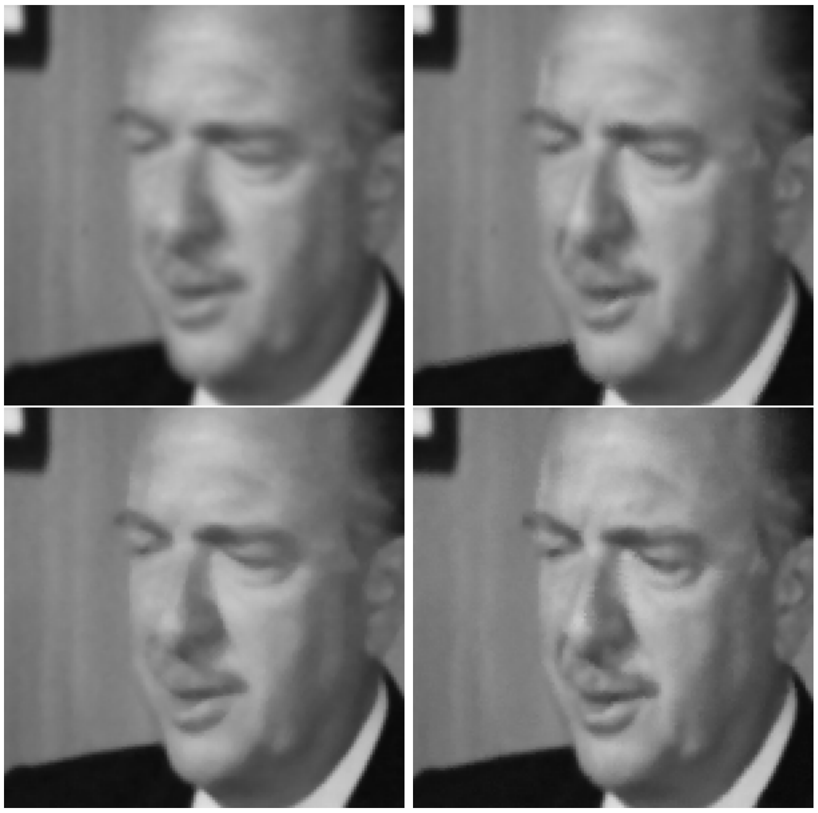


Figure 5: Comparison of static and motion compensated filters with the sequence of Figure 1 (only the central frame is displayed). Top: mean and motion compensated mean filter with the WS estimation. Bottom: median and motion compensated median filter with the BM estimation. All the experiments have been performed with a spatiotemporal support of $\delta_i = \delta_j = 3$ pixels and $\delta_t = 2$ frames. The application after motion compensation improves the static version with both motion estimation algorithms. The blurring is reduced and many more details are preserved. This experiment illustrates why most recent papers propose motion compensated algorithms.

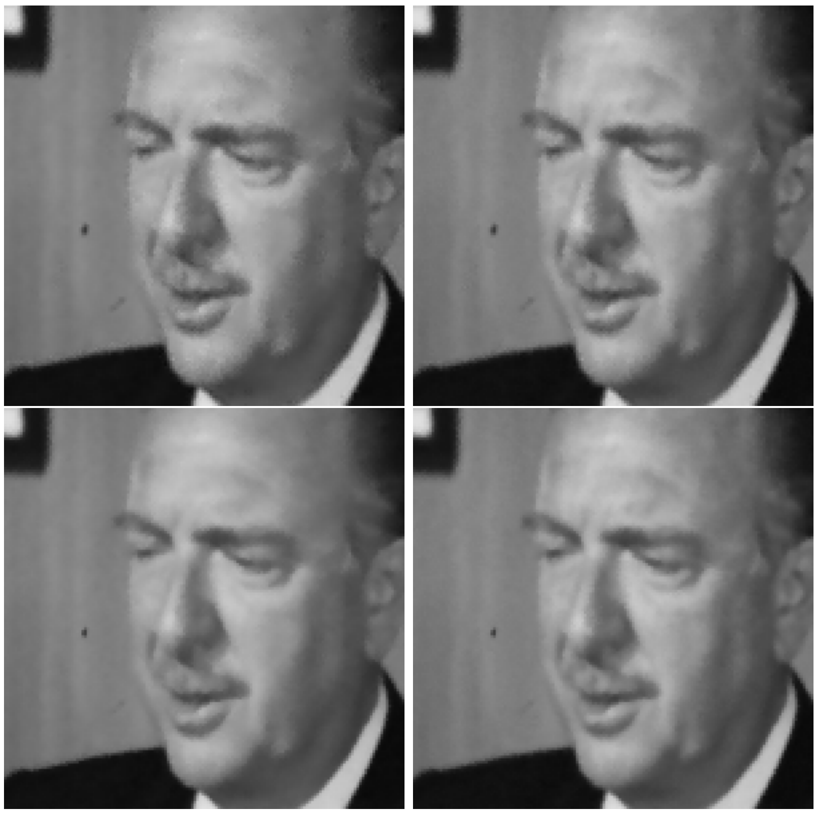


Figure 6: Comparison of motion compensated filters by the WS and the BM algorithms. Top left: statistical correction with WS. Top right: statistical correction with BM. Bottom left: neighborhood filter with WS. Bottom right: neighborhood filter with BM. In spite of the chaotic optical flow of Figure 4, the BM algorithm better preserves the details and creates less blur. The block matching motion estimation is more accurate near the edges. As a consequence, the statistical correction is able to better reduce the noise. When using the WS method, the algorithm is forced to be more conservative and keeps much more noise. When dealing with the neighborhood filter, the block matching also better preserves the details.

6 Dirt and sparkle

The local artifacts are of relatively small size and remove the original information. The positions of these artifacts change abruptly from one frame to another. Thus, a removal algorithm should take advantage of the previous and next frames where artifacts occur, but it should not modify the value of other pixels. The previously mentioned algorithms are unable to remove these artifacts without blurring the whole sequence.

The local artifacts are easily recognized by the eye as large oscillations of the grey level values in consecutive frames. Thus, they should be easily recognized by detecting these oscillations [15]. Let us present, for a sake a completeness, a simple strategy to remove them. We shall use this simple algorithm for comparisons hereafter. For each pixel, let us look for the pixel with the closest value in the previous and next frames and at a distance less or equal than a fixed parameter. Then, let us take the median value of the triple made by the current pixel and the two selected ones. Such an algorithm is very conservative. If no local artifacts are present, it should always find pixels with a similar grey level value in the previous and next frames. Guichard's movie filtering PDE [10] can be viewed as an iterative extension of this algorithm, and yields similar results.

Figure 7 displays an application of the previous, classical, strategy. We have applied it to the noisy sequence of Figure 1. The local artifacts are removed and the rest of the image is nearly not modified. In particular, the noise is generally kept. We also display a threshold of the difference between the noisy and filtered sequences. With this simple threshold, one is able to single out the local artifacts.

In contrast to structured dirt and sparkle, white or nearly white noise is mixed with the sequence features and cannot be detected by such simple devices. In the next section, we deal with this problem and present an extension to movies of a static image denoising algorithm, the NL-means.

7 The NL-means algorithm

The NL-means is a recently introduced image denoising algorithm [4, 5]. This algorithm tries to take advantage of the high degree of redundancy of any natural image. Most small windows in a natural image have many similar windows in the same image. In a very general sense inspired by the neighborhood filters, one can define as "neighborhood of a pixel i" any set of pixels j in the image such that a window around j looks like a window around i. All pixels in that neighborhood can be used for predicting the value at i, as was first shown in Efros et al. [9] for texture synthesis. This prediction was shown to be consistent by Levina [18].

Let us take first u to be a single image defined on a bounded domain $\Omega \subset \mathbb{R}^2$. The NL-means algorithm is defined as

$$NLu(\mathbf{x}) = \frac{1}{C(\mathbf{x})} \int_\Omega e^{-\frac{(G_a * |u(\mathbf{x}+.)-u(\mathbf{y}+.)|^2)(0)}{h^2}} \, u(\mathbf{y}) \; d\mathbf{y},$$

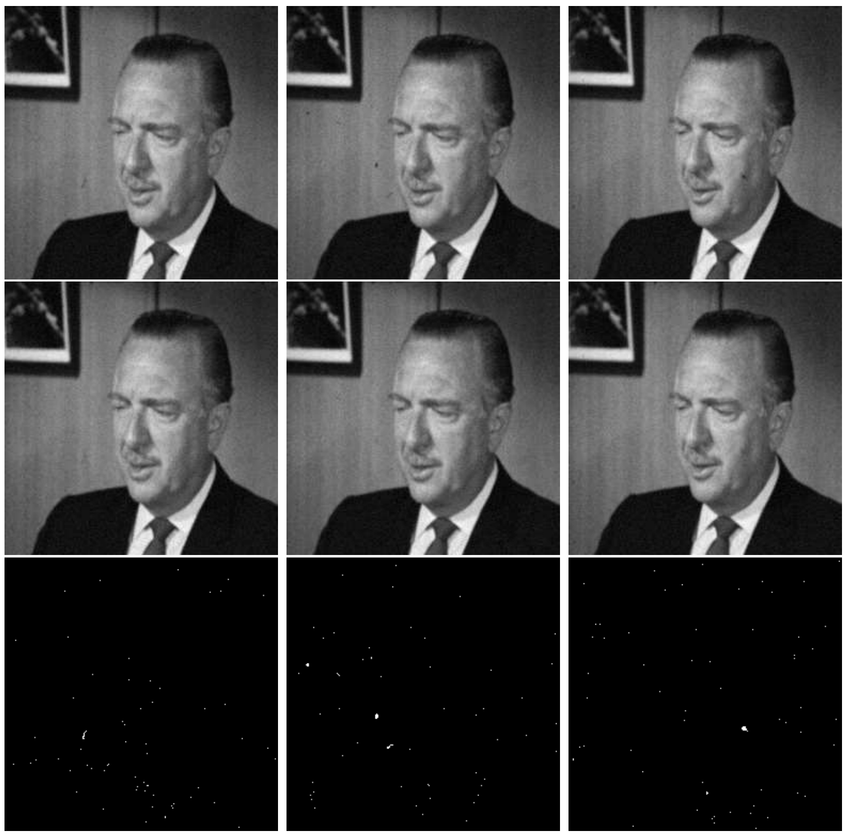


Figure 7: Removal of “dirt and sparkle’ artifacts by a simple classical algorithm. Top: three consecutive frames of a degraded sequence. Middle: sequence after the application of the algorithm for the removal of local artifacts. Bottom: Threshold of the difference of both sequences. Pixels with a difference larger than 5 grey level values are shown in white. The ’Dirt and Sparkle’ effect is removed by this simple algorithm. The rest of the image is nearly not modified, including the noise. The $l1$ differences between the three degraded frames and its filtered versions are: 0.174, 0.177, 0.179 (we note that grey level values vary from 0 to 255). Thus, the image is practically unaltered. With a simple threshold of the difference (actually 5), one is able to detect the stains. Once the detection is performed, more sophisticated algorithms can of course be applied if desired. White noise instead is not detectable in that way, as it is mixed with the movie features.

where $\mathbf{x} \in \Omega$, G_a is a Gaussian kernel of standard deviation a, h acts as a filtering parameter and $C(\mathbf{x}) = \int_\Omega e^{-\frac{(G_a * |u(\mathbf{x}+.)-u(\mathbf{z}+.)|^2)(0)}{h^2}} d\mathbf{z}$ is the normalizing factor. In order to make clear the previous definition, we recall that

$$(G_a * |u(\mathbf{x}+.) - u(\mathbf{y}+.)|^2)(0) = \int_{\mathbb{R}^2} G_a(\mathbf{t})|u(\mathbf{x}+\mathbf{t}) - u(\mathbf{y}+\mathbf{t})|^2 d\mathbf{t}.$$

This amounts to say that $NLu(\mathbf{x})$, the denoised value at $\mathbf{x}$, is a mean of the values of all pixels whose gaussian neighborhood looks like the neighborhood of $\mathbf{x}$.

The NL-means algorithm is non-local since all pixels in the image are used for the estimation at a pixel $\mathbf{x}$. NL-means take advantage of the huge redundancy present in natural images. As this redundancy is even larger in image sequences, it seems obvious to directly extend the $2D$ support to a $3D$ spatiotemporal one.

7.1 Sequence filtering algorithm

Every detail or small window usually has many similar windows through the sequence. However, the comparison of a window with all possible windows of the sequence is a prohibitive amount of computation. For this reason, the NL-means algorithm is usually applied in a fixed, large enough neighborhood,

$$S_{\mathbf{x}} = \{\mathbf{y} = (i,j,t) \mid |i - i_0| \leq \delta_i,\ |j - j_0| \leq \delta_j \text{ and } |t - t_0| \leq \delta_t\},$$

where $\delta_i, \delta_j, \delta_t > 0$ and $\mathbf{x} = (i_0, j_0, t_0)$. The estimated value $NL(u)(\mathbf{x})$ is computed as a weighted average of all the pixels in the support of $\mathbf{x}$,

$$NLu(\mathbf{x}) = \frac{1}{C(\mathbf{x})} \sum_{\mathbf{y} \in S_{\mathbf{x}}} w(\mathbf{x}, \mathbf{y}) u(\mathbf{y}), \tag{3}$$

where the weights $w(\mathbf{x}, \mathbf{y}) \geq 0$ depend on the similarity between the pixels $\mathbf{x}$ and $\mathbf{y}$ and $C(\mathbf{x})$ is the normalizing factor.

The similarity between pixels $\mathbf{x}$ and $\mathbf{y}$ depends upon the similarity of the intensity gray level vectors $u(\mathcal{N}_{\mathbf{x}})$ and $u(\mathcal{N}_{\mathbf{y}})$, where $\mathcal{N}_{\mathbf{z}}$ denotes a two dimensional square neighborhood of fixed size and centered at the pixel $\mathbf{z}$. Square neighborhoods of fixed size are used for simplicity.

The similarity of the intensity gray level vectors $u(\mathcal{N}_{\mathbf{x}})$ and $u(\mathcal{N}_{\mathbf{y}})$ can even be computed as an L^2 distance, $\|u(\mathcal{N}_{\mathbf{x}}) - u(\mathcal{N}_{\mathbf{y}})\|_2^2$. Efros and Leung [9] showed that the L^2 distance is a reliable measure for the comparison of image windows in a texture patch. Now, this measure is so much the more adapted to any additive white noise as such a noise alters the distance between windows in a uniform way. Indeed,

$$E||u(\mathcal{N}_{\mathbf{x}}) - y(\mathcal{N}_{\mathbf{y}})||_2^2 = ||u_0(\mathcal{N}_{\mathbf{x}}) - u_0(\mathcal{N}_{\mathbf{y}})||_2^2 + 2\sigma^2$$

where the observed sequence u is supposed to be obtained by the addition of a signal independent white noise of standard deviation σ to the true sequence u_0.

This equality shows that, in expectation, the Euclidean distance preserves the order of similarity between pixels. So the most similar pixels to $\mathbf{x}$ in u also are expected to be the most similar pixels to $\mathbf{x}$ in u_0. The weights associated with the quadratic distances are defined by

$$w(\mathbf{x}, \mathbf{y}) = e^{-\frac{||u(\mathcal{N}_{\mathbf{x}})-u(\mathcal{N}_{\mathbf{y}})||_2^2}{h^2}},$$

where h controls the decay of the exponential function and therefore the decay of the weights as a function of the Euclidean distances.

7.2 Motion adaptation and the aperture problem.

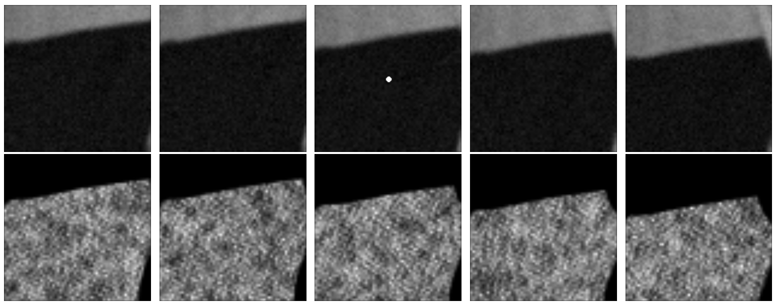
The NL-means algorithm applied to movies is a static filter, that is, a straightforward extension of a two dimensional filter. It does not directly take into account the dynamic character of image sequences. The aim of this section is to experimentally show that motion compensation is not necessary and even counterproductive.

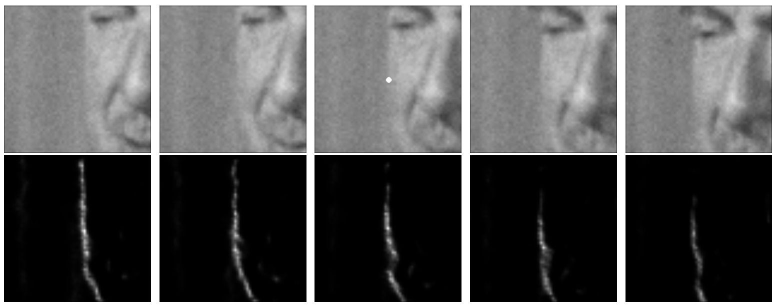
Motion estimation algorithms try to solve the aperture problem. The block matching algorithm chooses the pixel with the more similar configuration, thus loosing many other interesting possibilities, as displayed in Figure 3. Algorithms based on the OFC must impose a regularity condition of the flow field in order to choose a single trajectory. Thus, the motion estimation algorithms are forced to choose a candidate among all possible equally good choices. However, when dealing with sequence restoration, the redundancy is not a problem but an advantage. Figure 3 shows that we could choose anyone of the possible candidates for the averaging, so why not take them all.

In Figure 8, we display the probability distributions computed by the NL-means for three different cases. We display the support and the probability distributions used to estimate the central pixel of the middle frame. The support contains the two previous and posterior frames. In a flat zone a) we see that the NL-means gives a large weight to all the pixels belonging to the same region. In straight and curved edges the algorithm favors pixels belonging to the same space time contour (b) and c)). The algorithm favors pixels with a similar local configuration even if they are far away from the reference pixel. As the similar configurations move, so do the weights. Thus, the algorithm is able to follow the similar configurations when they move but without an explicit motion computation. So, there is no need to solve any aperture problem ; let us just average all pixels with a similar local configuration.

8 Experimentation and comparison

In the previous sections we have compared the performance of static filters and motion compensated filters. The comparison of these algorithms leads to conclude that the motion compensation improves a lot the static versions of the algorithms. In this section, we limit ourselves to the comparison of motion compensated filters with the NL-means algorithm.


a)


b)

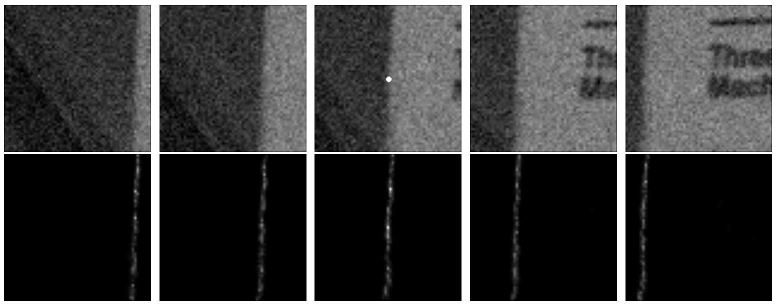

c)

Figure 8: Display of the probability distributions given by the NL-means algorithm. We display the original noisy values of the support and the weight distribution used to estimate the central pixel (in white) of the middle frame. The weight configuration is spatially adapted to the local configuration of each frame. The algorithm looks for the pixels with a more similar configuration even they have moved. This algorithm is adapted to moving pictures without the need of an explicit motion estimation.

In Figures 9 and 11 we compare the NL-means algorithm with the motion compensated statistical correction and neighborhood filter. The motion estimate has been obtained using the block matching algorithm. We display a filtered image of the sequence and the noise removed by the three algorithms. Ideally, the removed noise should not contain any noticeable structure and it should look like the realization of a white noise. The statistical correction and the sigma filters present many structures on the residual noise. This implies that these structures have been removed from the original sequence. The NL-means residual noise does not present any noticeable structure. As a consequence, the filtered image has kept more details and is less blurred.

In Figure 12 we combine the NL-means algorithm and the removal of dirt and sparkle. The NL-means algorithm has been applied first to remove white noise. Then, the strategy described in section 6 has been applied to the filtered sequence. The combination of both methods permits the removal of the two artifacts without blurring the sequence. As shown in Figure 13, it is better to denoise first and thereafter remove the “dirt and sparkle”. The local artifacts are not totally removed by first applying the dirt and sparkle removal. Applying instead first the NL-means algorithm leads to a significantly better removal.

References

[1] M.B. Alp, and Y. Neuvo, ”3-dimensional median filters for image sequence processing”, IEEE Proc. Int. Conf. Acoustics, Speech, and Signal Proc., vol. 4, pp. 2917-2929, 1991.

[2] G. R. Arce, ”Multistage order statistic filters for image sequence processing” IEEE Trans. Signal Processing, vol 39, pp. 1147-1163, 1991.

[3] J. C. Brailean, R. P. Kleihorst, S. Efsratiadis, A. K. Katsaggelos, and R. L. Lagendijk, ”Noise reduction filters for dynamic image sequences: a review”, Proceedings of the IEEE, vol. 83, pp. 1272-1292, 1995.

[4] A. Buades, B. Coll and J.M. Morel, ”A review of image denosing methods, with a new one”, accepted in Multiscale Modeling and Simulation (SIAM). Preprint avalaible at http://www.cmla.ens-cachan.fr/Cmla/Publications/2004.

[5] A. Buades, B. Coll and J.M. Morel, ”A non-local algorithm for image denoising”, IEEE International Conference on Computer Vision and Pattern Recognition, 2005.

[6] A. Buades, B. Coll and J.-M. Morel, ”Procédé de traitement de données d’image, par réduction de bruit d’image, et caméra intégrant des moyens de mise en oeuvre du procédé (Image data process by image noise reduction and camera integrating the means for implementing this process)”, European patent pending, May 5, 2004.

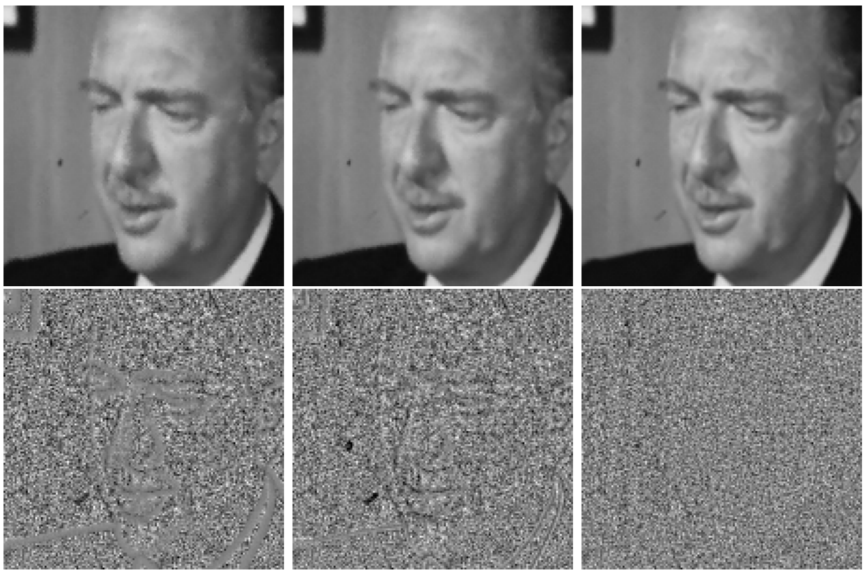


Figure 9: Comparison experiment. From top to bottom and left to right: one frame extracted from a sequence filtered by the motion compensated statistical correction of the mean, the motion compensated neighborhood filter and the NL-means. The motion estimation has been obtained by the block matching algorithm. Bottom: the residual noise for each one of the three methods. The residual noise is the noise removed by the algorithm. Ideally, it should not contain any noticeable structure and it should look like the realization of a white noise. The statistical correction and the neighborhood filter present many structures on the residual noise. These structures are removed from the original image. The NL-means residual noise does not present any noticeable structure. As a consequence, the filtered image preserves more details and is less blurred. Dirt and sparkle is not removed, rather restored by these algorithms (see below).

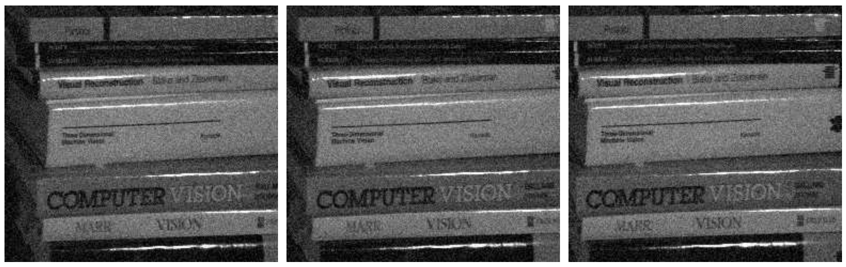


Figure 10: Three consecutive frames of a noisy image sequence. The noisy sequence has been obtained by the addition of a Gaussian additive white noise ($\sigma = 15$) to the original sequence.

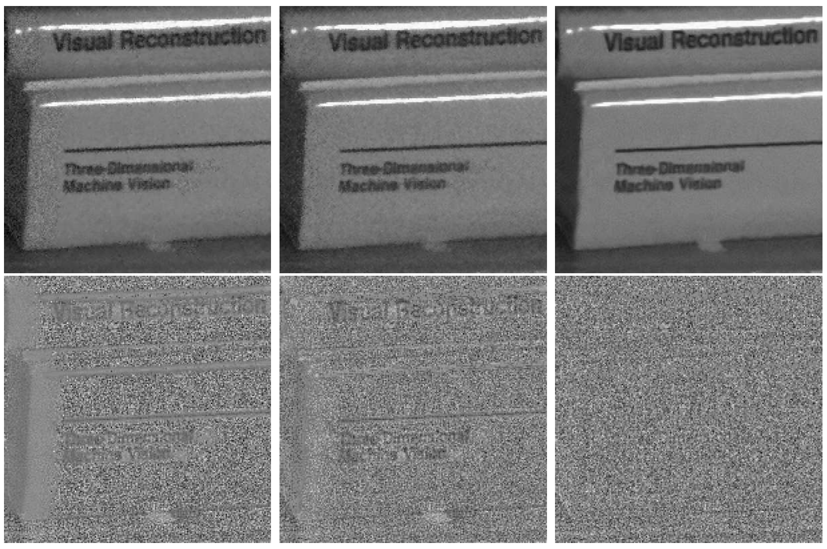


Figure 11: Comparison experiment with the sequence of Figure 10. Top to bottom and left to right: the same filtered frame extracted from a sequence after three different algorithms have been applied : the motion compensated statistical correction of the mean, the motion compensated neighborhood filter and the NL-means. The motion estimation has been obtained by the block matching algorithm. Bottom: the residual noise from the three algorithms above. This experiment corroborates the observations of Figure 9. The statistical correction residual noise is nearly zero on the strong boundaries. Now, these boundaries are kept noisy on the filtered sequence. We can read the titles of the books on the residual noise of the neighborhood filter. Therefore, that much information has been removed from the original. Finally, the NL-means algorithm does not have any noticeable structure in its residual noise.

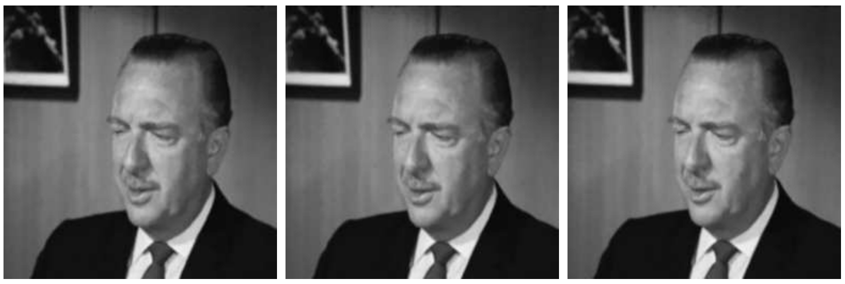


Figure 12: Combination of the NL-means and the dirt and sparkle removal on the sequence of Figure 1. Are displayed three consecutive frames of the restored movie, after applying: First, the NL-means algorithm to remove the additive noise. Second, the strategy described in section 6 for the removal of local artifacts. As shown in Figure 13 this order of application performs better in the removal of the dirt and sparkle.

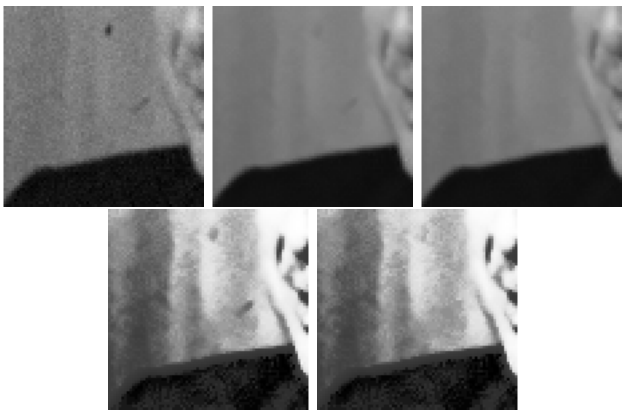


Figure 13: Comparison of the filtered sequences by the application of the NL-means algorithm and the removal of dirt and sparkle in different order. Top: Crop of the third frame, filtered crop by first applying the removal of dirt and sparkle, filtered crop by first applying the NL-means algorithm. The local artifacts are not totally removed by first applying the dirt and sparkle removal. Applying first the NL-means algorithm, they are no more visible. On the bottom, the histogram equalization of both images corroborates the first visual observation.

[7] D.I. Crawford, "Spatio-temporal prefiltering for a video conference coder", Proc. Int. IEEE Conf. on electronic image processing, pp 236-242, 1982.

[8] E. Dubois, "Motion compensated filtering of time varying images", Multidimensional Systems and Signal Processing, vol. 3, pp. 211-239, 1992.

[9] A. Efros and T. Leung, "Texture synthesis by non parametric sampling," Proc. Int. Conf. Computer Vision (ICCV 99), Vol. 2, pp. 1033-1038, 1999.

[10] F. Guichard, "A morphological, afine, and galilean invariant scale-space for movies", Trans. on Image Processing, vol. 7(3), pp. 444-456, 1998.

[11] B. Horn and B. Schunck, "Determining optical flow," Artif. Intell., Vol. 17, pp. 185-203, 1981.

[12] T. Huang, "Image Sequence Analysis", Springer-Verlag, 1981.

[13] A.K. Katsaggelos, J.N. Driessen, S.N. Efstratiadis, and R.L. Lagendijk, "Temporal motion compensated filtering of image sequences", Proc. SPIE Conf. Visual Commun. and Image Processing, vol. 1119, pp. 61-70, 1989.

[14] R.P. Kleihorst "Noise Filtering of Image Sequences", PhD thesis, Delft Univerity of Technology, 1994.

[15] A. C. Kokaram, "Motion Picture Restoration", PhD thesis, Cambridge University, 1993.

[16] J.S. Lee, "Digital image enhancement and noise filtering by use of local statistics", IEEE Trans. Patt. Anal. Machine Intell., vol. 2, pp. 165-168, 1980.

[17] J.S. Lee, "Digital image smoothing and the sigma filter", Computer Vision, Graphics and Image Processing, vol. 24, pp. 255-269, 1983.

[18] E. Levina, *Statistical Issues in Texture Analysis.* PhD dissertation, UC Berkeley, 2002.

[19] D. M. Martinez, "Model-based motion estimation and its application to restoration and interpolation of motion pictures", PhD thesis, Massachusetts Institute of Technology, 1986.

[20] R. H. McMann, "Digital noise reducer for encoded NTSC signals" SMPTE journal, vol. 87, pp. 129-133, 1978.

[21] H.H. Nagel, "Constraints for the estimation of displacement vector fields from image sequences," Proc. Eighth Int. Joint Conf. on Artificial Intelligence (IJCAI '83), pp. 945-951, 1983.

[22] M.K. Ozkan, M.I. Sezan, and A.M. Tekalp, "Adaptive motion compensated filtering of noisy image sequences", IEEE Trans. Circuits and Systems for Video Technology, vol. 3, pp. 277-290, 1993.

[23] R. Samy, "An adaptive image sequence filtering scheme based on motion detection", SPIE, Vol. 596, pp. 135-144, 1985.

[24] M.I. Sezan, M.K. Ozkan and S.V. Fogel, "Temporally adaptive filtering of noisy sequences using a robust motion estimation algorithm", Proceedings of the Int. Conf. Acoustics, Speech, Signal Processing 91, pp. 2429-2432, 1991.

[25] S.M. Smith and J.M. Brady, "Susan - a new approach to low level image processing," International Journal of Computer Vision, Volume 23 (1), pp. 45-78, 1997.

[26] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," Sixth International Conference on Computer Vision, pp. 839-46. 1998.

[27] R.C. Triplicane, "Recursive restoration of noisy image sequences" M.Sc. thesis, Northwestern University, 1989.

[28] J. Tukey, "Exploratory Data Analysis", Addison-Wesley, 1977.

[29] J. Weickert, "On discontinuity-preserving optic flow," Proc. Computer Vision and Mobile Robotics Workshop, pp. 115-122, 1998.

[30] J. Weickert and C. Schnörr, "Variational optic flow computation with a spatio-temporal smoothness constraint", Journal of Mathematical Imaging and Vision, vol. 14, pp. 245-255, 2001.

[31] L.P. Yaroslavsky, "Digital Picture Processing - An Introduction", Springer Verlag, 1985.