J. Vis. Commun. Image R. 20 (2009) 254–274

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

One-iteration dejittering of digital video images

Mila Nikolova

CMLA, ENS Cachan, CNRS, PRES UniverSud, 61 Av. Président Wilson, F-94230 Cachan, France

ARTICLE INFO

Article history:
Received 22 June 2008
Accepted 3 March 2009
Available online 19 March 2009

Keywords:
Error measures
Fast algorithms
Image modeling
Image restoration
Jitter
Matching
Non-convex data-fitting
Non-smooth data-fitting
Restoration
Video

ABSTRACT

We propose several very fast algorithms to dejitter digital video images in one iteration. They are based on an *essential* disproportion of the magnitude of the second-order differences along the columns of a real-world image and all its jittered versions. The optimal row positions are found using non-smooth and possibly non-convex local criteria, applied on the second-order differences between consecutive rows. The dejittering iteration involves a number of steps equal to the number of the rows of the image. These algorithms are designed for gray-value and color natural images, as well as to noisy images. A reasonable version of these algorithms can be considered as parameter-free. We propose specific error measures to assess the success of dejittering. We provide experiments with random and structured jitter. The obtained results outperform by far the existing methods both in quality and in speed (the ours need around 1 second for a 512×512 image on Matlab). Our algorithms are a crucial step towards real-time dejittering of digital video sequences.

1. Intrinsic dejittering

Image jittering consists in a random horizontal displacement of each row of the image. Jittering occurs in video images (frames) when the synchronization signals—carrying the information about the proper location of the rows relative to each other—are corrupted e.g. by noise or degradation of the storage medium; it often occurs in wireless transmission. The visual effect is quite disturbing since each row is randomly displaced within a range of ± 6 pixels or even much more. Then shapes appear to be jagged in the vertical direction—see e.g. Fig. 6(a). Structured (e.g. sinusoidal jitter)—can be provoked by acoustic, electrical or other interferences [9]. The rows of the image are displaced with a number of pixels corresponding to the frequency and the amplitude of the perturbation; then vertical lines are transformed into sinusoids, as in Fig. 11. *Time base corrector* machines process with some success the analogue video signal in order to recover the row synchronization information [8]. In many cases, such an operation is unsuccessful or impossible. The alternative approach—restoring the image frames directly from the observed jittered data, often called *intrinsic dejittering* [9], is much more flexible and widely applicable. It naturally uses prior assumptions on natural images. We focus on such an intrinsic approach.

1.1. State of the art

The very trivial solution to this problem—searching for the shift between consecutive lines that maximizes their correlation—is known to cause a bias towards vertical lines and fails in most of the cases [8]. Intrinsic dejittering was invented by Kokaram et al. in [7]. An improved version of the method, based on a 2D autoregressive model (2D AR) of the image, is explained in the textbook on film and video of Kokaram [8, Chapter 5]. The unknown 2D AR coefficients and row displacements are considered by blocks. Thy are estimated jointly using an iterative algorithm. A drift compensation finalizes the restoration. Laborelli proposes in [9] a different approach where the ℓ_1 norm of the differences between (two or three) consecutive shifted rows is compared. The optimal shifts are recovered during the backward iteration of a dynamic programming method. Later on, Shen proposed a fully Bayesian method using a Total Variation (TV) prior model for the underlying image [14] for joint dejittering and denoising. It requires to minimize a non-smooth function for each frame, which is time consuming. A two-step method, called *Bake and Shake* is proposed by Kang and Shen in [5]. It relies on the idea that a good PDE-based image restoration method, such as the Perona–Malik diffusion, can help to find the right row positions. Recently, the same authors analyze in [6] the slicing moments of images of bounded variation (BV) and derive an original variational method based on Bayesian regularization of the vertical slicing moments. It is numerically cheaper but the according to the authors, it is less efficient under strong jitter than the previous method.

E-mail address: nikolova@cmla.ens-cachan.fr

1047-3203/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2009.03.004

1.2. Our approach and plan of the paper

In Section 2 we exhibit a pertinent prior on the columns of natural images and construct a criterion that efficiently discriminates the wrong row shifts. More precisely, a non-smooth and possibly non-convex local criterion is applied to the magnitude of the 2-nd order vertical differences between consecutive rows. Its rationale and properties are explained as well. Specific error measures to evaluate the quality of dejittering are proposed in Section 3. The dejittering algorithm for gray-value, noise-free (natural) jittered images is presented in Section 4, along with some possible refinements. Numerous illustrations are provided. A large-scale experiment (4×1000) on four images is described in Section 5 and the main conclusions are summarized; the relevant tables are given in Appendix A. Straightforward extensions of the dejittering algorithms to color images, corroborated by illustrations, are presented in Section 6. In Section 7 we extend these algorithms in order to deal with noisy jittered images within a two-stage approach. In all experiments, we consider both independent jitter and structured (sinusoidal) jitter. Conclusions and perspectives are outlined in Section 8.

1.3. Notations

Remind that $\mathbb{Z}$ is the set of all integers and $\mathbb{N}_+$ the subset of all positive integers. For any $m \in \mathbb{N}_+$ and $n \in \mathbb{N}_+$, the rows of a matrix $h \in \mathbb{R}^{m\times n}$ are systematically denoted by $h_i,\ 1 \leqslant i \leqslant m$ and the components of a row h_i by $h_i(j),\ 1 \leqslant j \leqslant n$. The components of any n-length vector u are denoted by $u_i,\ 1 \leqslant i \leqslant n$. For any $q \geqslant 1$, the ℓ_q norm is denoted by $\|.\|_q$. For any $u \in \mathbb{R}^n$ we write $\|z\|_0 \stackrel{\text{def}}{=} \#\{i \in \{1,\cdots,n\} : z_i \neq 0\}$ where $\#$ stands for cardinality. By $\mathbb{1}_n$ we denote the n-length vector composed of ones. Zero-mean Gaussian distribution with standard deviation σ is denoted by $\mathcal{N}(0,\sigma^2)$.

The original (unknown) image, of size $r \times c$, is denoted by f and its degraded version by $g \in \mathbb{R}^{r\times c}$. The jittering vector is denoted by $d \in \mathbb{R}^r$. The restored image and row displacements are denoted by $\hat{f}$ and $\hat{d}$, respectively.

2. Choice of a criterion

Given an original image f the usual model for the production of a jittered image g reads [8]

$$\forall j \in \{1,\cdots,c\}, \forall i \in \{1,\cdots,r\},\quad g_i(j) = \begin{cases} f_i(j+d_i), & \text{if } 1 \leqslant j+d_i \leqslant c \\ \text{any, e.g.} = 0, & \text{otherwise} \end{cases} \quad d_i \in \mathbb{Z}, \quad |d_i| \leqslant M. \tag{1}$$

In practice, M is around 6 pixels or more [8]. Different types of jitter are systematized in [9]:

- Independent line jitter, often modeled using a (truncated and quantized) Gaussian or a Uniform distribution.
- Structured jitter—such as low-frequency or high-frequency sinusoidal jitter.

Both types of jitter are considered only in [9], as well as in our paper.

2.1. Local criteria based on consecutive lines

For each jittered row g_i we wish to estimate its displacement $\hat{d}_i$ based on the previously restored rows $\hat{f}_{i-1}, \hat{f}_{i-2}, \ldots$. Let us have a look at Fig. 1(b): the left side shows a column of a natural image while its right side shows the "same" column of the jittered image. See also Fig. 3, p. 4.

Remark 1. One observes that the gray-value of the columns of natural images can be seen as pieces of 2nd or 3rd order polynomials which is hard to claim for their jittered versions. This is a sound basis to discriminate a natural image from its jittered versions.

Suppose that $\hat{d}_1, \ldots \hat{d}_{i-1}$ (and hence $\hat{f}_1, \ldots \hat{f}_{i-1}$) are already recovered. Based on Remark 1, we are going to estimate the next row displacement $\hat{d}_i$ by comparing several previously dejittered rows $\hat{f}_{i-1}, \hat{f}_{i-2}, \ldots$ with all possible shifts of the next data row $g_i(j+d_i)$ for $d_i \in [-N, N]$ where $N \geqslant M$ is an overestimate of M. Once we obtain an estimate $\hat{d}_i$ for the displacement d_i, (1) tells us that the restored row reads

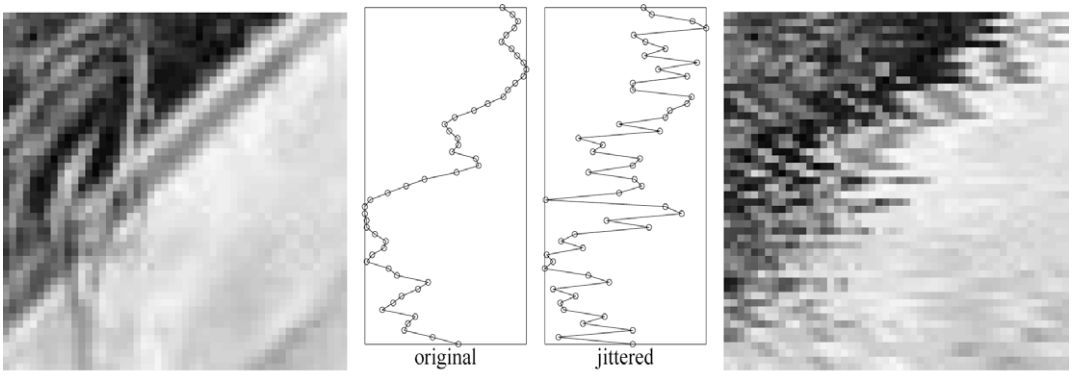
$$\hat{f}_i(j) = g_i(j-\hat{d}_i) \text{ if } 1 \leqslant j \leqslant c \text{ and } 1 \leqslant j-\hat{d}_i \leqslant c. \tag{2}$$

Let us now focus on the left-hand and the right-hand side boundaries of a jittered image, see Fig. 5, p. 6.

Remark 2. Each row of g has at one of its extremes a certain number of undetermined pixels (zero-valued in practice) dues to the jitter. We know that their number is at most equal to N. Involving such pixels in our criterion is risky since this can distort its meaning. For this reason, our criterion includes only columns of g indexed by $\{N+1,\ldots,c-N\}$ since they are guaranteed to contain information on the true image.

By Remark 2 and Eq. (2), we involve into our criterion only portions of the previously restored rows $\hat{f}_n$:

$$\text{for } n = k-1,\ k-2, \cdots, \hat{f}_n(j) \text{ for } j \in \{N+1+\hat{d}_n, \cdots, c-N+\hat{d}_n\}. \tag{3}$$



(a) Original (b) One column (c) Jittered

Fig. 1. A 50×50 zoom of Lena (512×512). (a) Original. (b) One column of the original against one column of the jittered image. (c) The same zoom of the jittered image (± 6 pixel random displacements).

By Remarks 1 and 2, and the theoretical results [11–13], we suggest several criteria in (4)–(6) below.

$$\hat{d}_i = \arg\min \{J_{1,\alpha}(d_i) : |d_i| \leqslant N\} \text{ for}$$
$$J_{1,\alpha}(d_i) = \frac{1}{n-m+1}\sum_{j=m}^{n} |g_i(j-d_i) - \hat{f}_{i-1}(j)|^{\alpha}, \quad \alpha \in (0,1], \tag{4}$$

where

$$m = N + 1 + \max\{d_i, \hat{d}_{i-1}\},\ n = c - N + \min\{d_i, \hat{d}_{i-1}\}.$$

The constants m and n here, as well as in (5), (6), ensure that the recommendations of Remark 2 are satisfied; they are easily derived from (3). The factor $\frac{1}{n-m+1}$ (here and in (5), (6)) accounts for the fact that the number of terms in the criterion varies according to the value of d_i.

For any $\alpha \in (0,1]$, criterion $J_{1,\alpha}$ favors the recovery of nearly *constant* gray-value vertical pieces, i.e. $g_i(j+\hat{d}_i) \approx \hat{f}_{i-1}(j)$. Fig. 1(b) left and Fig. 3 suggest that this is quite exceptional for natural images.

Consider now second-order vertical differences between three consecutive lines:

$$\hat{d}_i = \arg\min \{J_{2,\alpha}(d_i) : |d_i| \leqslant N\}$$
$$J_{2,\alpha}(d_i) = \frac{1}{n-m+1}\sum_{j=m}^{n} |g_i(j-d_i) - 2\hat{f}_{i-1}(j) + \hat{f}_{i-2}(j)|^{\alpha}, \quad \alpha \in (0,1], \tag{5}$$

where
$m = N + 1 + \max\{d_i, \hat{d}_{i-1}, \hat{d}_{i-2}\}$
and $n = c - N + \min\{d_i, \hat{d}_{i-1}, \hat{d}_{i-2}\}$.

$J_{2,\bullet}$ promotes the recovery of vertical pieces with nearly linearly varying gray-value, i.e. $\hat{d}_i$ is such that for numerous pixels j of the *ith* row, $g_i(j+\hat{d}_i) \approx 2\hat{f}_{i-1}(j) - \hat{f}_{i-2}(j)$. This is true for numerous subsets along each columns of a natural image (see Fig. 3) but it is false for the arbitrary displacements in g, such as in Fig. 1(b).

Remark 3. The ℓ_1 norm of the first-order and the second-order differences between consecutive rows was used in [9]. Note that *all* pixels were involved in the criterion (which does not take into account Remark 2). In the 3-row method in [9], fixing the optimal shifts by dynamic programming during the backward iteration does not prevent the algorithms from recovering nearly constant gray-values vertical pieces.

Yet another option that seems reasonable with respect to Fig. 1(b) is to consider third-order vertical differences,

$$\hat{d}_i = \arg\min_{|d_i|\leqslant N} J_{3,\alpha}(d_i) \text{ for}$$
$$J_{3,\alpha}(d_i) = \sum_{j=N+1}^{c-N} |g_i(j-d_i) - 3\hat{f}_{i-1}(j) + 3\hat{f}_{i-2}(j) - \hat{f}_{i-3}(j)|^{\alpha}, \quad \alpha \in (0,1], \tag{6}$$

where
$m = N + 1 + \max\{d_i, \hat{d}_{i-1}, \hat{d}_{i-2}, \hat{d}_{i-3}\}$
and $n = c - N + \min\{d_i, \hat{d}_{i-1}, \hat{d}_{i-2}, \hat{d}_{i-3}\}$.

This criterion favors the recovery of columns where numerous sets of 4 neighboring pixels in the vertical direction have a gray-value following a nearly quadratic shape. This may also hold for some slightly wrong estimates of the row displacements, so this criterion should be less effective than $J_{2,\bullet}$.

Remark 4. (**Choice of *N***). Suppose that we choose $N = M$ and that a wrong $\hat{d}_i$ reads either $\hat{d}_i = -M$ or $\hat{d}_i = M$. Then the possible optimal shifts for the next row $i+1$ are restricted only to $\hat{d}_{i+1} \geqslant -M$ in the first case and $\hat{d}_{i+1} \leqslant M$ in the second case. It can happen that the true position of row $i+1$ with respect to row i is in the opposite side, in which case $\hat{d}_{i+1}$ is erroneous. For this reason why *we strongly recommend to choose* $N > M$. Taking $N = M + 1$ is usually enough since the method is quite precise.

The image in Fig. 2(b) is corrupted with jitter which is uniform on $\{-5,\dots,5\}$, so the displacements are important with respect to the size of the image and the details it contains. We observe that $J_{1,\alpha}$ works badly—it tends to recover vertically constant pieces. Both criteria $J_{2,0.5}$ and $J_{2,1}$ yield the original image with no error, i.e. $\hat{d} = d$. Obviously, $J_{3,0.5}$ and $J_{3,1}$ perform less well: they cannot discriminate well enough between the true image and its slightly shifted versions. *According to our numerous experiments,* $J_{2,\bullet}$ *gives systematically much better results than the other suggested criteria. Hence we focus basically on* $J_{2,\bullet}$.

2.2. Interpretation of $J_{2,\bullet}$

Since [12,13], the minimizer of $J_{2,\alpha}$, for $\alpha \leqslant 1$, looks to recover $\hat{d}_i$ such that $g_i(j+\hat{d}_i) \approx 2\hat{f}_{i-1}(j) - \hat{f}_{i-2}(j)$, i.e. that $g_i(j+\hat{d}_i)$ nearly lies on the gray-value line determined by the previously recovered

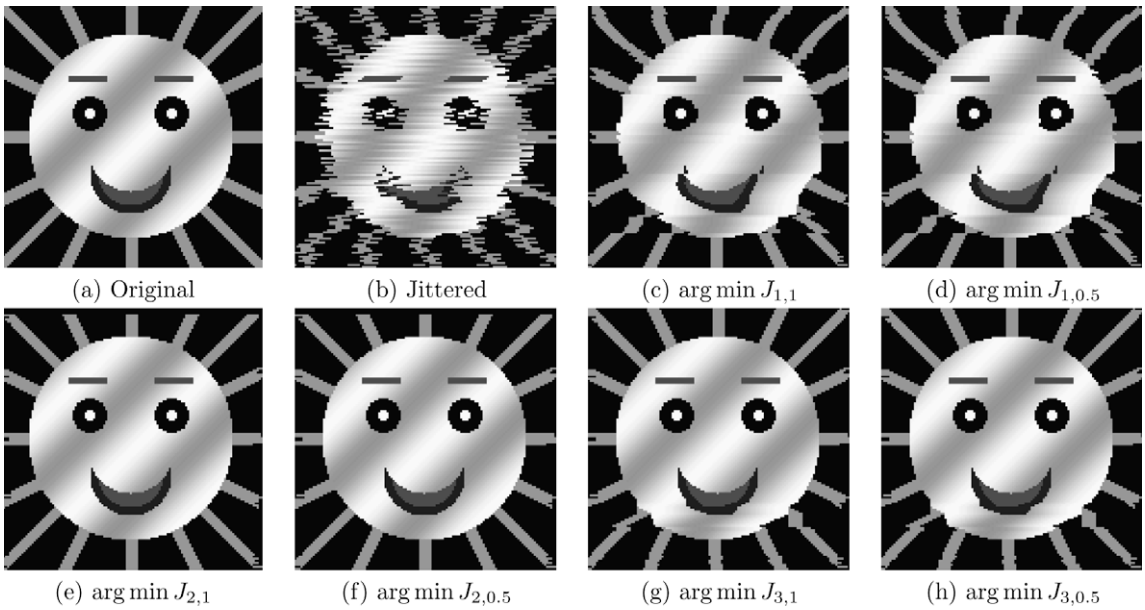


Fig. 2. A toy image—My Sun—128 × 128 in (a), uniform jitter on $\{-5,\dots,5\}$ in (b).

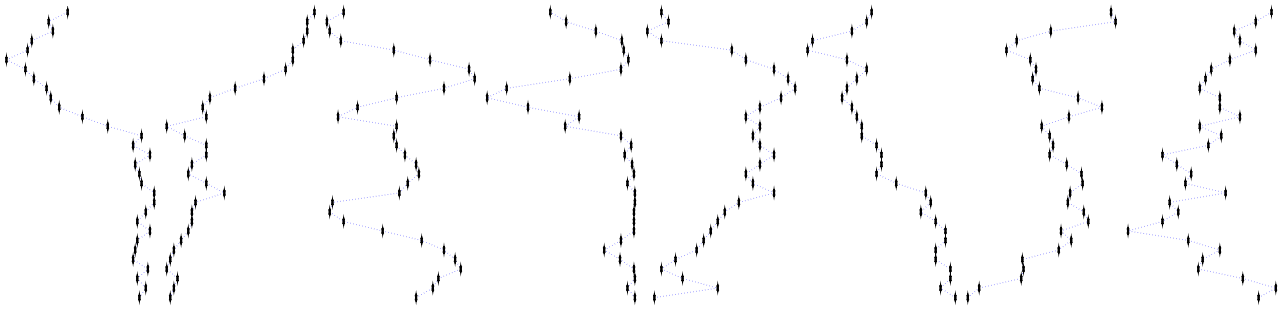


Fig. 3. The gray value of pieces of columns from different natural images used in this paper. The gray value of each pixel is emphasized.

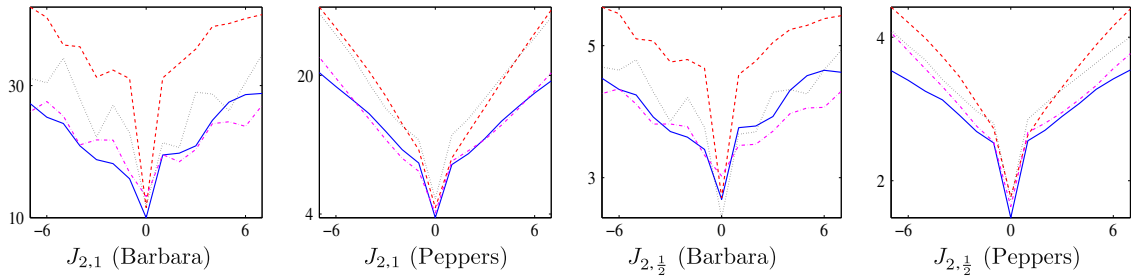


Fig. 4. X-axis: $d_i \in \{-M,\dots,M\}$ for $M=7$. Y-axis: $J_{2,\alpha}(d_i) = \frac{1}{n-m+1}\sum_{i=m}^{n} |f_i(j+d_i) - 2f_{i-1}(j) + f_{i-2}(j)|$ for $i \in \{173, 298, 419, 478\}$ (Barbara) and $i \in \{23, 123, 273, 477\}$ (Peppers). The true solution is naturally $\hat{d}_i = 0$.

$\hat{f}_{i-1}(j)$ and $\hat{f}_{i-2}(j)$. The latter corresponds to monotone gray value in the vertical direction, which is fully coherent with Fig. 3.

For a fixed i, let us set $\hat{h}_i(j) \stackrel{\text{def}}{=} 2\hat{f}_{i-1}(j) - \hat{f}_{i-2}(j)$, which is a constant at the step when we estimate the optimal $\hat{d}_i$. Let us also set $I \stackrel{\text{def}}{=} \{m,\dots,n\}$. Problem (5) can be reformulated as

$$\text{find } \hat{a} = \arg\min_{a \in D} J(a) \text{ where} \tag{7}$$

$$J(a) = \sum_{j \in I} |a - \hat{h}_i(j)|^{\alpha} \text{ and } D = \{g_i(j-\delta) : \delta \in \{-N,\dots,N\}\} \subset \mathbb{R}.$$

The optimal solution $\hat{a}$ can be seen as an M-estimator [15,18] constrained to the discrete, finite set D. For $\alpha = 1$, the optimal $\hat{a}$ is the median of $\{\hat{h}_i(j) : j \in I\}$ constrained to D. For $0 < \alpha < 1$, J is non-convex and has numerous local minima and in our case—one for each element of D. The attractive properties of non-convex M-estimators—in terms of robustness and edge-enhancement—are discussed in [16] and in [17, Chapter 3]. Note that this case is not well understood in the literature, even without constraints. The global solution $\hat{a}$ of problem (7) is easy to compute by exhaustive search since the cardinality of D is only $2N+1$. The formulation provided in (7) reveals several important facts:

1. Problem (7) promotes solutions $\hat{a} \in D$ such that $|\hat{a} - \hat{h}_i(j)|^{\alpha}$ is small, i.e. $\hat{f}_i(j+\hat{d}) \approx 2\hat{f}_{i-1}(j) - \hat{f}_{i-2}(j)$, for a maximum number of components j.
2. The weight of all terms such that $|\hat{a} - \hat{h}_i(j)|^{\alpha} \gg 0$, i.e. $|\hat{f}_i(j+\hat{d}_i) - 2\hat{f}_{i-1}(j) + \hat{f}_{i-2}(j)|^{\alpha} \gg 0$ is small and it decreases as far as $\alpha \leqslant 1$ decreases. Such terms correspond to edge points between $\hat{f}_i(j+\hat{d})$ and the past $2\hat{f}_{i-1}(j) - \hat{f}_{i-2}(j)$, so they are less penalized when $\alpha \leqslant 1$ decreases. This is coherent with Fig. 3.
3. Both items 1 and 2 suggest that a pertinent choice for α should be $\alpha \in (0,1)$. For stability reasons the function $|.|^{\alpha}$ should be increasing enough, so we consider only $\alpha \in \left[\frac{1}{2}, 1\right]$.

Criteria $J_{2,\alpha}$ for $\alpha = 1$ and $\alpha = 1/2$, corresponding to several "difficult" rows of Barbara and Peppers are illustrated in Fig. 4. In all cases, right answer ($\hat{d}_i = 0$) is a neat global minimizer.

3. Error measures for dejittering

The standard tools to assess the quality of a dejittered image with respect to the original one—e.g. SNR, PSNR, MAE, etc.—cannot be applied directly since the restored image $\hat{f}$ is shifted with respect to f and the extremities of its rows are unknown (zero-valued in practice), because of the jitter. A possible way out is described next. We shrunk $\hat{f}$ to $\hat{f}^s$ according to

$$\hat{f}^s_i(j) = \hat{f}_i(j+N), \quad 1 \leqslant j \leqslant c-2N, \quad \forall i \in \{1,\dots,r\},$$

so that $\hat{f}^s_i$ contains only proper image information (and has no gaps at the ends of its rows). Then we choose a shrunk by $2N$ columns version of the original f, say f^s, that matches $\hat{f}^s$ the best so that we can evaluate the error of $f^s - \hat{f}^s$. Different criteria can be used to define an optimal matching for f^s. Note that any error measure on $f^s - \hat{f}^s$ is sensitive to the choice of f^s. Having in mind that the ℓ_1 norm is well suited for image evaluation, we seek for an $f^s \in \mathbb{R}^{r\times(c-2N)}$ such that

$$\|f^s - \hat{f}^s\|_1 = \min_{0 \leqslant k \leqslant 2N} \sum_{i=1}^{r} \sum_{j=1}^{c-2N} |f_i(j+k) - \hat{f}^s(j)|.$$

For the same reason, we use the ℓ_1-based *mean absolute error* (MAE), defined by

$$\text{mae}(\hat{f}, f) = \frac{1}{r(c-2N)} \|f^s - \hat{f}^s\|_1. \tag{8}$$

The dynamic range of $(\hat{f}^s, f^s)$ reads $\delta \stackrel{\text{def}}{=} \max\left\{\max_{i,j}\{f^s_i(j)\}, \max_{i,j}\{\hat{f}^s_i(j)\}\right\} - \min\left\{\min_{i,j}\{f^s_i(j)\}, \min_{i,j}\{\hat{f}^s_i(j)\}\right\}$. Then we consider the *peak signal to noise ratio* (PSNR), defined by:

$$psnr(\hat{f}, f) = 10\log_{10} \frac{\delta^2 r(c-2N)}{\|f^s - \hat{f}^s\|_2^2}. \tag{9}$$

Remind that PSNR $= \infty$ if $f^s = \hat{f}^s$.

The quality of dejittering can also be evaluated based on $d - \hat{d}$ where we remind that $d \in \mathbb{R}^r$ is the vector of the horizontal displacements that degrade the original image f in (1) and that $\hat{d}$ is its estimate. We will use error measures for $d - \hat{d}$, based on the ℓ_∞-norm, the ℓ_1-norm, or the ℓ_0 "norm".

Remark 5. Dejittering an isolated video frame inevitably produces a *shifted* version of the displacement estimate, say $\hat{p} = \hat{d} + C$. When

d is known, we can estimate $\hat{d}$ by selecting C so that $\hat{d}_i = d_i$ for a maximum number of samples, i.e.
$C = \arg\min_{C\in\mathbb{Z}} \|\hat{p} - C - d\|_0$.

More details are given in Section 4—*Algorithm: Shift recovery*, p. 6.

The following ℓ_1-based error measure e_1

$$e_1(\hat{d}, d) = \frac{1}{r}\|d - \hat{d}\|_1, \tag{10}$$

evaluates the *average displacement of the pixels along each column.* It is the same for the pixels along any column of the image. It is quite relevant for the quality of dejittering.

The ℓ_0-based error measure e_0 below

$$e_0(\hat{d}, d) = \frac{100}{r}\|d - \hat{d}\|_0\% \tag{11}$$

gives the *percentage of displaced rows* in the restored image with respect to the original one.

The following two error measures are quite interesting:

$$e_\infty(\hat{d}, d) \overset{\text{def}}{=} \frac{100}{c}\|d - \hat{d}\|_\infty\%; \tag{12}$$

$$e_0^A(\hat{d}, d) \overset{\text{def}}{=} \frac{100}{r-1}\#\Big\{(\hat{d}_i - d_i) - (\hat{d}_{i+1} - d_{i+1}) \neq 0, \quad 1 \leqslant i \leqslant r-1\Big\}\%. \tag{13}$$

The first one e_∞ measures the maximum horizontal error with respect to the width c of the image. The second one e_0^A measures the number of changes in $d - \hat{d}$ with respect to the height r of the image. Qualitatively speaking, e_0^A assess up to what degree our model for the underlying image, involved in the criterion that is used, fits the image that is considered.

Remark 6. When both e_∞ and e_0^A are small (e.g. $e_\infty \leqslant 0.4\%$ and $e_0^A \leqslant 0.8\%$), we are guaranteed that dejittering is nearly perfect, independently of any other error measure (see Figs. 9, 10, 13 and 16). Indeed, for a 512×512 image, the proposed error bounds mean that no more than 4 rows have a horizontal erroneous shift which is no more than 2 pixels. For a natural image, such an error is invisible to the naked eye. However, if one of these values is larger, their value is meaningless.

For instance, Fig. 9 is almost perfectly dejittered and we have $e_\infty(d, \hat{d}) = 0.39\%, e_0^A(d, \hat{d}) = 0.6\%$ whereas PSNR is not very favorable. Note that if an image is homogeneous on wide horizontal regions, even though dejittering is fine, these errors can be high. Fig. 12 is visually nicely restored but we have $e_\infty = 3.76\%$ and $e_0^A = 4.19\%$; for this image, MAE and e_1 are small and PSNR is very high. Yet another example is Fig. 7 which is nicely dejittered (e.g. PSNR is high and MAE is small) but $e_\infty = 4.49\%$ and $e_0^A = 1.96\%$. Indeed, both images involve wide homogeneous regions.

For jitter degradation, the visual convenience still remains of paramount importance. Let us notice that the error measures given above do not take into account the content of the image f.

4. Algorithms for gray-value jittered natural images

The goal of this section is to furnish a detailed implementable algorithm enabling to find $\hat{d}$ and $\hat{f}$ according to $J_{2,\bullet}$—as given in (5)—along with some possible refinements.

Additional notations. We systematically denote by θ_n the zero-valued row-vector of length n:

$$\theta_n = [\underbrace{0, \ldots, 0}_{n}]. \tag{14}$$

The classical notation $[a\vdots b\vdots c]$ means that we concatenate horizontally a, b and c (which can be row-vectors or matrices). We write $a \leftarrow b$ when we replace a by b. Remind that the rows of a matrix γ are denoted by γ_i.

4.1. Main algorithm with explanations

Before to run the algorithm, we have to fix the following choices:

- Fix $N > M$, e.g., $N = M + 1$ (see Remark 4);
- Fix $\alpha \in (0, 1]$, e.g. $\alpha = 1$ or $\alpha = 0.5$ (especially if the image involves regions with well organized texture).

Algorithm 1. (*Gray value images*)

1. Put $\tilde{f}_1 = [\theta_N \vdots g_1 \vdots \theta_N] \in \mathbb{R}^{1\times(c+2N)}$.
2. Split data g into 3 sub-matrices $g = [g^L \vdots \gamma \vdots g^R]$ where $g^L \in \mathbb{R}^{r\times N}, \gamma \in \mathbb{R}^{r\times(c-2N)}$ and $g^R \in \mathbb{R}^{r\times N}$.
3. Put $\hat{p}_0 = \hat{p}_1 = N + 1$ and $\phi_1 = \phi_2 = [\theta_N \vdots \gamma_1 \vdots \theta_N] \in \mathbb{R}^{1\times c}$.
4. For any $i = 2, \ldots, r$ do the following:
   (a) for any $k = 1, \ldots, 2N + 1$
      i. Put $h^k = [\theta_{k-1} \vdots \gamma_i \vdots \theta_{2N-k+1}] \in \mathbb{R}^{1\times c}$;
      ii. Find $m = \max\{k, \hat{p}_{i-1}, \hat{p}_{i-2}\}$ and $n = \min\{k, \hat{p}_{i-1}, \hat{p}_{i-2}\} + c - 1$;
      iii. calculate $\mathscr{J}(k) = \frac{1}{n-m+1}\sum_{j=m}^{n} |h^k(j) - 2\phi_1(j) + \phi_2(j)|^\alpha$;
   (b) find $\hat{p}_i = \arg\min\{\mathscr{J}(k) : \ 1 \leqslant k \leqslant 2N + 1\}$;
   (c) Substitute: $\phi_2 \leftarrow \phi_1$ and $\phi_1 \leftarrow h^{\hat{p}_i} = [\theta_{\hat{p}_i - 1} \vdots \gamma_i \vdots \theta_{2N+1-\hat{p}_i}]$;
   (d) Put $\tilde{f}_i = [\theta_{\hat{\mathbf{p}}_i - 1} \vdots g_i \vdots \theta_{\mathbf{2N}-\hat{p}_i+1}] \in \mathbb{R}^{1\times(c+2N)}$;
5. Extract $\hat{f} \in \mathbb{R}^{r\times c}$ from $\tilde{f} \in \mathbb{R}^{r\times(c+2N)}$ by eliminating $2N$ columns at the extreme left and right ends that contain the largest number of zeros or jitter.

The shift of the top line can be assigned arbitrarily (since we have a single image). The algorithm constructs a wider $r \times (c + 2N)$ matrix $\tilde{f}$; at step 1, we insert g_1 in the middle of its first row $\tilde{f}_1$; hence $\hat{p}_1 = N + 1$. Following Remark 2, γ at step 2 is the $r \times c - 2N$ submatrix of g containing only image data. Row displacements $\hat{p}_i$ are estimated based on γ only. In this version, we initialize $\hat{p}_0 = \hat{p}_1$ (step 3). At step 4, we successively estimate the positions $\hat{p}_i$, for $i \geqslant 2$, according to Eq. (5). At each sub-step, ϕ_1 and ϕ_2 are c-length row vectors corresponding to the estimates of γ_{i-1} and γ_{i-2}, respectively. In the inner iteration 4a, h^k realizes all possible shifts ($k = 1, \cdots, 2N + 1$) for the row γ_i. The constants m and n have the same meaning as in Eq. 5 and $\mathscr{J}$ is calculated accordingly. Once we get the minimizer $\hat{p}_i$, the estimate for γ_i is precisely $h^{\hat{p}_i}$. We can hence update ϕ_1 and ϕ_2, as specified at sub-step 4c. At sub-step 4d we insert in $\tilde{f}_i$ the whole observed g_i (and not only its restriction γ_i as we did in 4(a)i). When $i = r$, the matrix $\tilde{f}$ is full. The dejittered image $\hat{f}$ is an $r \times c$ inner sub-matrix of $\tilde{f}$ that we extract as described in step 5.

Some preliminary illustrations of Algorithm 1 can be seen in Fig. 2(e)–(f) as well as in Fig. 5.

As noticed in Remark 5, the vector $\hat{p} \in \mathbb{R}^r$ is shifted with respect to the estimate of the displacement $\hat{d}$, namely $\hat{d}_i = \hat{p}_i - C$, $\forall i \in \{1, \ldots, r\}$. The constant C satisfies $1 - N \leqslant C \leqslant 3N + 1$. Indeed, $\forall i \in \{1, \ldots, r\}$ we have $|d_i| \leqslant N$ and $1 \leqslant \hat{p}_i \leqslant 2N + 1$, hence $1 - N \leqslant \hat{p}_i - d_i \leqslant 3N + 1$. In order to compute the the error measures defined in (10), (11), (12) and (13), we have to find the shift C. Given the true displacement d, we find C using the criterion given in Remark 5. This can be done using the numerical scheme below.

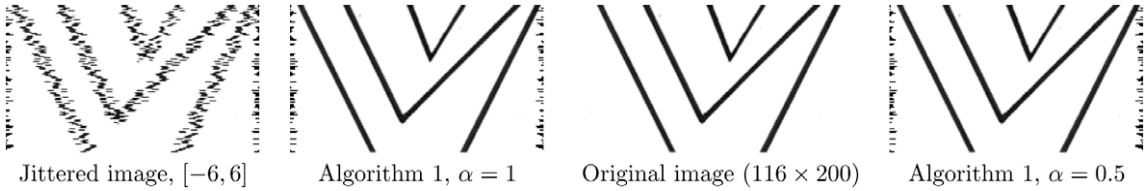


Fig. 5. The original image is contaminated with independent jitter uniformly distributed on $\{-6,\dots,6\}$.

Algorithm. *Shift recovery*

1. Define the set of integers $I=\{-N+1,\dots,3N+1\}$.
2. Let $\mathscr{H}$ be the histogram of $\hat{p}-d$ on I—i.e. $\mathscr{H}(n)=\#\{j\in I:\hat{p}(j)-d(j)=n\}$, for $n\in I$.
3. Obtain $C=\arg\max_{n\in I}\mathscr{H}(n)$.Then $\hat{d}_i=\hat{p}_i-C, \forall i\in\{1,\dots,r\}$.

Computation times. The computation time depends clearly on the size of the image, on the value of N, and it is higher if we choose $\alpha=0.5$ instead of $\alpha=1$. We did some comparisons using Matlab 7.2 on a PC with a Pentium 4 CPU 2.8GHz and 1GB RAM, running on Windows XP Professional service pack 2. For a 512×512 size gray-value image and $N=7$ we got the solution in 0.62 second for $p=1$ and in 1 second for $p=0.5$. Let us mention that our Matlab implementation is not optimal.

4.2. Refinements

4.2.1. Boundary condition

The boundary condition in step 3 amounts to estimate $\hat{d}_2$ using the first-order criterion given in (4). This is not fully satisfying. In some cases, a mirror boundary condition can produce better results. Such a condition cannot be applied directly to a jittered image: we have first to align in some way the first two rows of the image. We propose a modification of Algorithm 1 where we first align γ_1 and γ_2 (using $J_{1,\alpha}$ in (4)) and take for ϕ_1 the aligned version of γ_2 and for $\hat{p}_0$ its estimated position. This change concerns only step 3 in Algorithm 1.

Algorithm 1(a)

Steps 1 and 2 are the same as in Algorithm 1. Step 3 is replaced by

3. Set $\hat{p}_1=N+1$ and $\phi_1=[\theta_N,\gamma_1,\theta_N]$;
   (a) for any $k=1,\dots,2N+1$, calculate:
      i. $h^k\stackrel{\text{def}}{=}[\theta_{k-1},\gamma_2,\theta_{2N-k+1}]$;
      ii. $m=\max\{k,N+1\}$ and $n=\min\{k,N+1\}+c-1$;
      iii. $\mathscr{J}(k)=\frac{1}{n-m+1}\sum_{j=m}^{n}|h^k(j)-\phi_1(j)|^\alpha$;
   (b) set $\hat{p}_0=\arg\min_{1\leqslant k\leqslant 2N+1}\mathscr{J}(k)$;
   (c) set $\phi_2=\left[\theta_{\hat{p}_0-1},\gamma_2,\theta_{2N-\hat{p}_0+1}\right]$.

   Then continue with the steps 4 and 5 of Algorithm 1.

Algorithm 1(a) does not completely prevent from constant vertical transitions between the first two rows.

Yet another possibility is to fix $\hat{p}_1$ as in Algorithm 1 and then to find jointly $\hat{p}_2$ and $\hat{p}_3$ by considering all possible shifts for γ_2 and γ_3. Then we choose $\hat{p}_2$ such that $\mathscr{J}_{2,\alpha}$ is minimal for $(\hat{p}_2,\hat{p}_3)$.

Algorithm 1(b).

Steps 1 and 2 are the same as in Algorithm 1. Step 3 is replaced by

3. Set $\hat{p}_1=N+1$ and $\phi_2=[\theta_N,\gamma_1,\theta_N]$;
   (a) for any $k=1,\dots,2N+1$
      i. $h_1^k=[\theta_{k-1},\gamma_2,\theta_{2N-k+1}]$;
      ii. for any $\ell=1,\dots,2N+1$, calculate
         A. $h^\ell=[\theta_{\ell-1},\gamma_3,\theta_{2N-\ell+1}]$;
         B. $m=\max\{k,\ell,N+1\}$ and $n=\min\{k,\ell,N+1\}+c-1$;
         C. $J(k,\ell)=\frac{1}{n-m+1}\sum_{j=m}^{n}|h^\ell(j)-2h_1^k(j)+\phi_2(j)|^\alpha$;
      iii. find $\mathscr{J}(k)=\min_{1\leqslant\ell\leqslant2N+1}J(k,\ell)$;
   (b) find $\hat{p}_2=\arg\min_{1\leqslant k\leqslant2N+1}\mathscr{J}(k)$;
   (c) Set $\phi_1=\left[\theta_{\hat{p}_2-1},\gamma_2,\theta_{2N-\hat{p}_2+1}\right]$.

   Then continue with steps 4 and 5 of Algorithm 1.

The computational cost of Algorithm 1(b) is slightly higher than Algorithms 1 and 1(a). Even though the experimental results are very encouraging, more research is needed to find in a fast and efficient initialization.

4.2.2. Compound models for images involving a noticeable vertical trend

In case when an image exhibits a *dominant vertical structure* (e.g. Figs. 14 and 15), better results can be obtained by using a compound criterion that mixes first and second-order differences via an additional parameter $\beta>0$.

Algorithm 1(c)

Do Algorithm 1 (or 1(a) or 1(b)) by replacing step 4(a)iii by the following expression:

$$\mathscr{J}(k)=\frac{1}{n-m+1}\sum_{j=m}^{n}(|h^k(j)-2\phi_1(j)+\phi_2(j)|+\beta|h^k(j)-\phi_1(j)|)^\alpha. \quad (15)$$

Experimentally, the algorithm is quite stable with respect to β.

4.3. Illustrations

In Figs. 6, 7, 10 and 21, we compare our method with the Bayesian TV (B-TV) method of Shen [14] and with the Bake and Shake (B & S) method of Kang and Shen [5]. All these restorations were realized by Dr. Sung Ha Kang using the codes of Dr. Jackie Shen for the B-TV method, and using her own codes for the B & S method. Thus the parameters for these competitors are tweaked to reach their best level of performance.

For all images, our dejittering algorithms are systematically applied with $N = M + 1$, in accordance with Remark 4.

In Fig. 6, the picture of Lena (256×256), is degraded with independent uniform jitter on $\{-6, \dots, 6\}$. The width of the image being small, the jittering effect is very strong. This jittered image is first restored using the B-TV method in [14] and by the B & S method [5]. The second one clearly outperform the B-TV method both visually and in terms of the error measures, thus yielding MAE $= 7.37$ and PSNR $= 23.06$. Then we applied successively Algorithms 1 and 1(a) for $\alpha = 1$ and $\alpha = 0.5$. In all cases, we obtained perfect reconstructions where all errors are zero except PSNR $= +\infty$. It was our first trial and we kept the result since it is perfect.

Peppers in Fig. 7 is degraded with independent uniform jitter on $\{-10, \dots, 10\}$ (i.e. $M = 10$). The restoration obtained by B-TV [14] is poor. The result of B & S [5] is much better. For $\alpha = 1$, Algorithms 1, 1(a) and 1(b) yields the same result with error measures MAE $= 1.38$, PSNR $= 31.27$, $e_1 = 0.43, e_0 = 23.63\%, e_\infty = 5.1\%$ and $e_0^A = 2.35\%$. It is not displayed since visually it is very close to result for $\alpha = 0.5$. For $\alpha = 0.5$, Algorithms 1 and 1(a) give the same result—displayed in the figure—whose errors read MAE $= 1.35$, PSNR $= 31.51$, $e_1 = 0.4$, $e_0 = 23.63\%$, $e_\infty = 4.49\%$ and $e_0^A = 1.96\%$. The image of the error—where the central $c - 2N$ part of the restored and the original images are matched as explained in Section 3—shows a slight displacement of several pixels. However, our dejittered image is hard to distinguish from the original image to the naked eye.

The picture of Barbara (512×512) is challenging for many image processing tasks since it contains a lot of small details with a strong local geometry. In Fig. 8, the picture is contaminated with independent jitter obtained from $\mathcal{N}(0, \sigma^2)$ for $\sigma = 3$, quantized on $\mathbb{Z}$ and constrained to $\{-6, \dots, 6\}$. Here we compare the role of α. For $\alpha = 1$, there are wrongly estimated displacements visible on the leg of the table. For $\alpha = 0.5$, the result is nearly perfect since $e_\infty = 0.39\%$ and $e_0^A = 0.59\%$ (see Remark 6).

The next Fig. 9 considers the same image contaminated with independent uniform jitter on $\{-6, \dots, 6\}$. The dejittered image is impossible to distinguish visually from the original; indeed, $e_\infty = 0.39\%$ and $e_0^A = 0.59\%$, even though the other errors—MAE, PSNR and e_1—are less favorable. The error image $\hat{f}^s - f^s$ shows that the dejittering is perfect in the central rows of the image but that it is slightly wrong somewhere in the middle of the face of Barbara, as well as in the bottom part, at the level of row 430. On the zoom of the face of Barbara, it is hard to find where is the wrong displacement since it is only one pixel.

The first image in Fig. 10 is degraded with strong uniform jitter on $\{-10, \dots, 10\}$ (i.e. $M = 10$). The restoration obtained using B-TV [14] contains a lot of jitter. The restoration using B & S [5] is better.

Fig. 6. Lena (256×256), corrupted with a jittering vector d uniform on $\{-6, \dots, 6\}$. Different dejittering methods. We do not display the original image since it is perfectly recovered by our Algorithm 1.

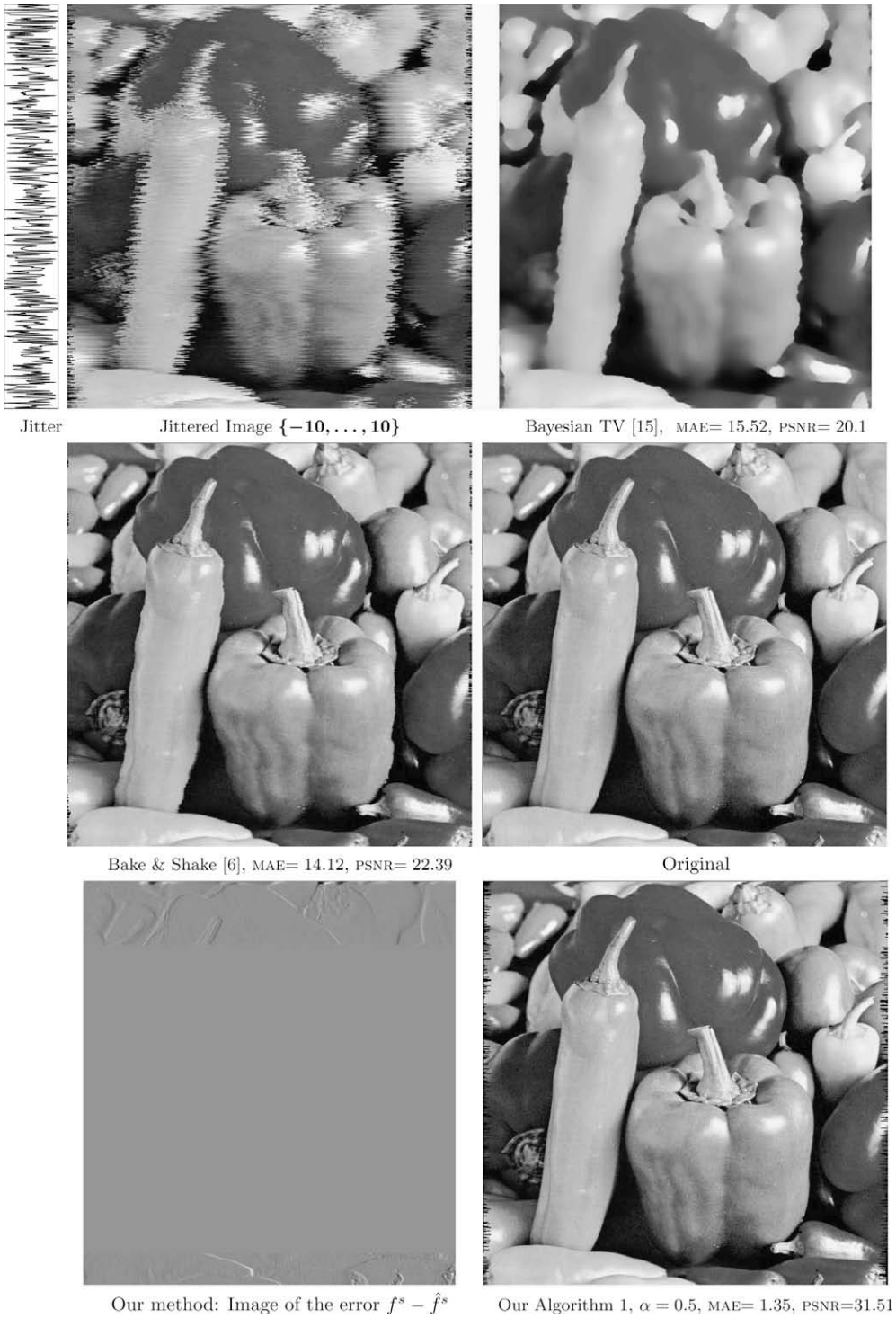


Fig. 7. Peppers (512×512). The jitter is uniform on $\{-10,\ldots,10\}$. Dejittering using different methods.

Algorithms 1 and 1(a) for both $\alpha = 1$ and $\alpha = 0.5$ lead to the same dejittering shown on the right. The latter is nearly-perfect since $e_\infty = 0.39\%$ and $e_0^\Delta = 0.25\%$. The original Boat image can be seen on the right side of Fig. 11 where the restorations are exact (i.e. $\hat{f}^s = f^s$, as defined in Section 3).

Fig. 11 illustrates the cases of structured jitter as mentioned in the Introduction—in particular low-frequency and high-frequency sinusoidal jitter. Restoration from both the low-frequency and the high-frequency sinusoidal jitter using Algorithm 1 for $\alpha = 1$ yields the original image.

5. Large-scale experiment

Even if our method relies on a good rationale, we cannot justify it in a fully theoretical way. Instead, we check its stability and its

Fig. 8. Barbara 512×512. The jitter corresponds to $\mathcal{N}(0, \sigma^2\mathbb{1})$, $\sigma = 3$, truncated and quantized on $\{-6, \ldots, 6\}$. The error measures for $\alpha = 1$ are: MAE $= 10.57$, PSNR $= 21.41$, $e_1 = 1.92$ and $e_\infty = 5$. The restoration for $\alpha = 0.5$ is nearly perfect: $e_\infty = 0.39\%$ and $e_0^A = 0.59\%$; note also that MAE $= 4.16$, PSNR $= 25.53$ and $e_1 = 0.52$.

performance using a very large number of experiments. These help us to make relevant choices for the boundary conditions as well as for α.

For each one of the following popular and "difficult" gray-value images—Lena, Peppers, Barbara and Boat—we realized 1000 independent experiments as described below. These four images present completely different features: Lena involves a lot of smooth textures along with regular homogeneous parts; Peppers is composed out of nicely homogeneous regions separated by regular contours; Barbara involves textures with a strong local geometry with lots of small edges as well as nicely homogeneous regions; Boats is very geometrical without important texture.

- UNIFORM JITTER.
  The components of the jittering vector $d \in \mathbb{R}^r$ are independent and uniformly distributed on the set of integers $\{-M, \ldots, M\}$ for $M = 6$. All results relevant to uniform jitter are indicated in Tables 1–3 given in Appendix A with the letter "U".Each original image was degraded using such a jittering vector d, according to 1. The same jittered image was then restored successively using Algorithm 1 for $\alpha = 1$ and $\alpha = 0.5$, then Algorithm 1(a) for $\alpha = 1$ and $\alpha = 0.5$ and last Algorithm 1(b) for $\alpha = 1$ and $\alpha = 0.5$.
- TRUNCATED GAUSSIAN JITTER.
  The components of the jittering vector $d \in \mathbb{R}^r$ are integers belonging to the set $\{-M, \ldots, M\}$ for $M = 6$. They are obtained as described next. We generate $2r$ or $3r$ independent reals $\tilde{d}_i \sim \mathcal{N}(0, \sigma^2)$ for $\sigma = 3$. Then we select r elements that belong to $[-M - 0.5 + \varepsilon, M + 0.5 - \varepsilon]$ with $\varepsilon = 2 \times 10^{-16}$ and approximate them to the nearest integer. Thus we get a sequence of independent samples following a quantized and truncated centered Gaussian distribution. The results corresponding to this kind of jitter are indicated in Tables 1–3 with the letter "G".The original image f was jittered with the so obtained vector d, following 1. The same jittered image was then restored as in the previous case: Algorithm 1 for $\alpha = 1$ and $\alpha = 0.5$, then Algorithm 1(a) for $\alpha = 1$ and $\alpha = 0.5$ and last Algorithm 1(b) for $\alpha = 1$ and $\alpha = 0.5$.

For each restoration, we calculated the errors MAE, e_1, $e_0\%$, $e_\infty\%$ and $e_0^A\%$, as defined in Eqs. (8), (10)–(12) and (13) respectively. (We cannot find any mean value for the PSNR error measure (9): we already observed that in some cases dejittering is exact ($\hat{f}^s = f^s$) which leads to PSNR $= \infty$. For each image, each case of jitter, each algorithm and each parameter, we thus obtained 1000 measures of these errors. Table 1 given in Appendix A summarizes their means, Table 2 gives the percentage of the cases when $e_\infty \leqslant 0.4\%$ and $e_\infty \leqslant 0.6$, jointly with $e_0^A \leqslant 0.8\%$ while Table 3 gives the variances of MAE and e_1. These results should be considered with some distance—in the body of the paper we present several cases where visually better restorations do not match the minimum of these errors, except when $e_\infty \in [0, 0.4]\%$ and $e_0^A \in [0, 0.8]\%$. Nevertheless, the mean of these error measures should be relevant. The results

Fig. 9. Barbara 512×512. The jitter is uniform on $\{-6, \ldots, 6\}$. The errors for the restoration read MAE = 4.16, PSNR = 25.53, $e_1 = 0.52$; note that $e_\infty = 0.39\%$ and $e_0^d = 0.59\%$.

shown in Tables 1–3 help to sketch a (partial) assessment on the following points: (i) if our algorithms are good enough, (ii) which one to choose among 1, 1(a) and 1(b), (iii) which value for α is better.

1. Algorithm 1 gives smaller mean errors in Table 1 for almost all cases, and in all cases it is satisfactory enough. Moreover, it is faster than Algorithms 1(a) and 1(b).
2. The values for e_0^d are always small which justifies the prior model we adopted on the second-order vertical differences on natural images. It also shows that $\hat{d} - d$ is at least piecewise constant which is a good point.
3. All tables show that $\alpha = 0.5$ gives better results for textured images (Lena, Barbara), or images involving lots of regular curvatures (Peppers), while $\alpha = 1$ is better for Boat which has a simpler geometrical structure. In all cases, $\alpha = 1$ leads to acceptable results, $\alpha = 0.5$ leads to a higher quality in most of the cases.
4. From Table 2 we see that dejittering of Lenna and Boat gives rise to high percentages for $(e_\infty \leqslant 0.4\%, e_0^d \leqslant 0.8\%)$ —between half and 3/4—which corresponds to guaranteed high-quality restorations. These percentages are smaller for Barbara (10-20%) and Peppers (10% or less) but according to Remark 6, in such a case it is better to check the other error measures which are quite encouraging.
5. The variances of MAE and e_1 in Table 3 are small, especially for $\alpha = 0.5$.
6. Except for Barbara under uniform jitter in Table 1, Algorithm 1(b) seems to perform less well than the others. The simplest form—Algorithm 1—seems being the most "universal".

Remark 7. Whenever we fix α, our algorithms are parameter-free. For noise-free jittered images, it is safer to fix $\alpha = 0.5$. If a slight compromise in terms of quality can be tolerated, algorithms are faster for $\alpha = 1$ while the results range between very good and quite acceptable (especially when compared with the state of the art).

6. Color images

In this section we extend Algorithm 1 in order to deal with RGB color images where *all color channels incur the same jitter*. Let us remind that RGB discrete images are represented by vector-valued matrices f where each pixel $f_i(j)$ has three components, say $f_i(j;\kappa)$ for $1 \leqslant \kappa \leqslant 3$. More precisely, the jittering model (1) reads

$$\forall j \in \{1, \ldots, c\}, \forall i \in \{1, \ldots, r\},$$
$$g_i(j;\kappa) = \begin{cases} f_i(j + d_i; \kappa), & \text{if } 1 \leqslant j + d_i \leqslant c, \\ \text{any, e.g.} = 0, & \text{otherwize,} \end{cases} \quad 1 \leqslant \kappa \leqslant 3,$$

where d_i is as in (1).

6.1. Algorithms

The central part of the algorithm proposed below is based again on $\mathscr{J}_{2,\bullet}$ as given in (5). Since the jitter is the same for all color channels, we obtain from g a *gray-value image* γ and estimate the row displacements $\hat{d}_i = \hat{p}_i - C$ on γ using the ideas of Algorithm 1.

The equivalent counterpart of the n-length zero-valued row vector defined in (14) is denoted by $\theta_{n\times 3}$: each one of its components is a 3D zero-valued vector, that is

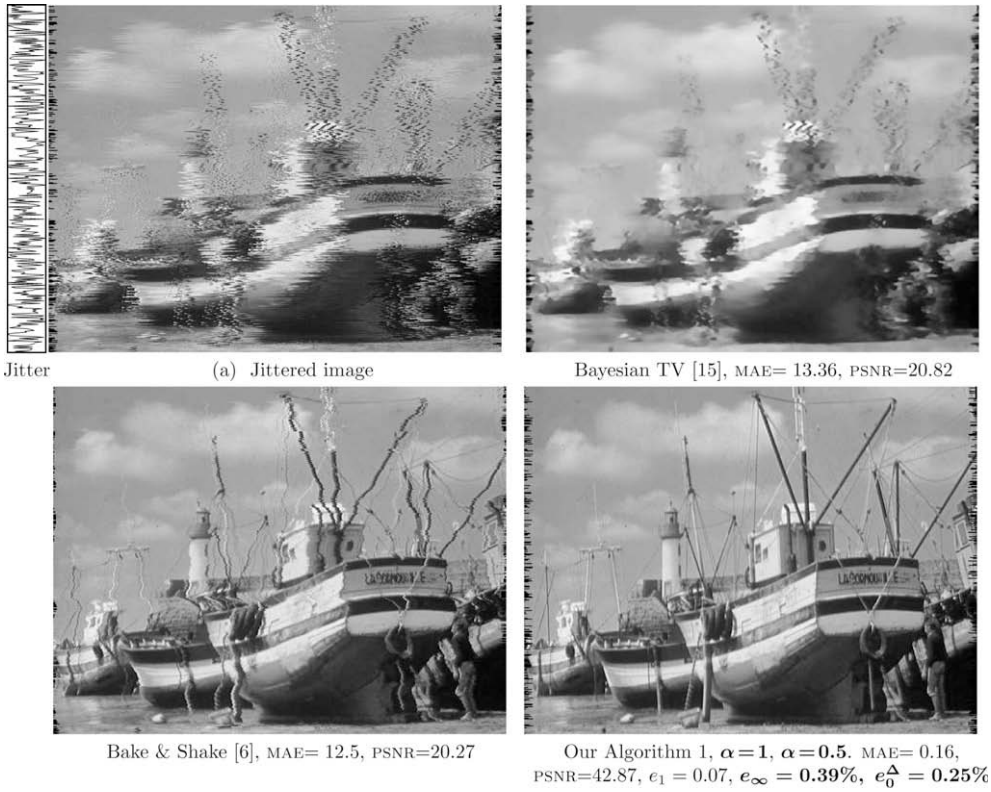


Fig. 10. Boat 400×512. The jitter is uniform on $\{-10, \ldots, 10\}$. Different dejittering methods. Our Algorithm 1, for $\alpha \in \{0.5, 1\}$, yields a nearly perfect result since $e_\infty = 0.39\%$ and $e_0^\Delta = 0.25\%$.

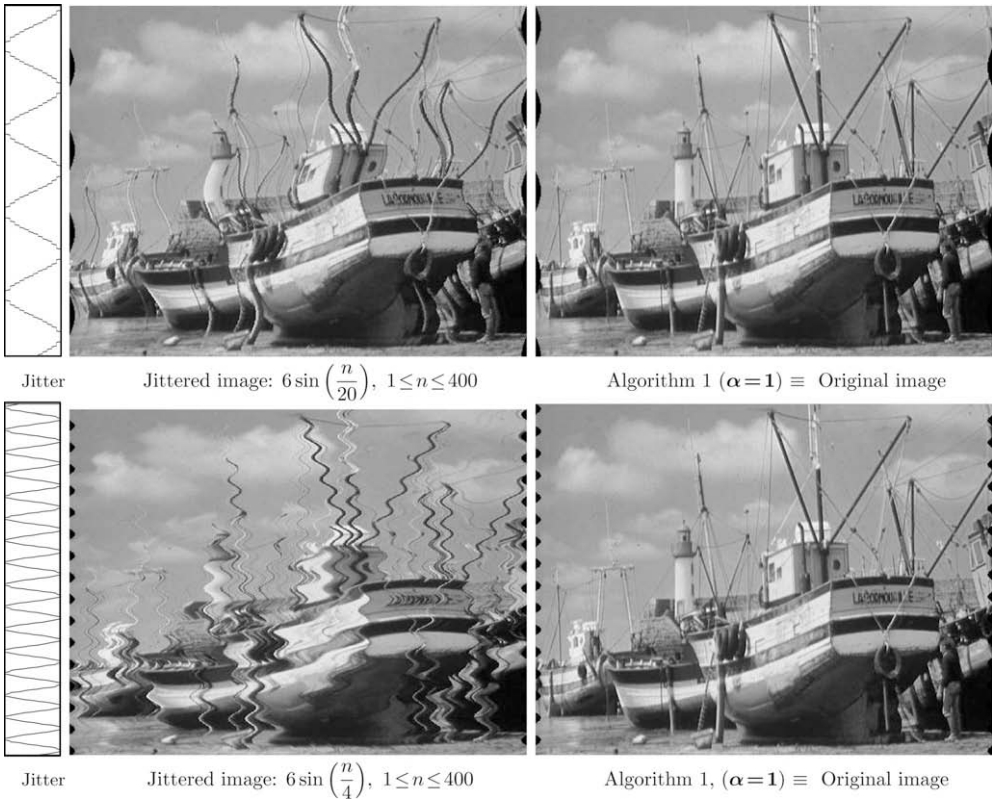


Fig. 11. The sinusoidal jitter is *quantized* on $\{-6, \ldots, 6\}$. The dejittered image is perfect—all errors are null.

$$\theta_{n\times 3}(.;\kappa) = \theta_n, \text{ for } 1 \leqslant \kappa \leqslant 3. \tag{16}$$

Before to start, we fix the following values:

- $N > M$, e.g., $N = M + 1$.
- $\alpha \in (0, 1]$, e.g. $\alpha = 1$ or $\alpha = 0.5$—if the image involves regions with complex structure.

Algorithm 2. (*Color images*)

1. Put $\tilde{f}_1 = [\theta_{N\times 3} \vdots g_1, \vdots \theta_{N\times 3}]$.
2. Split g into 3 vector-valued sub-matrices $g = \left[g^L \vdots \overline{g} \vdots g^R\right]$ where $g^L, g^R \in \mathbb{R}^{r\times N}$ and $\overline{g} \in \mathbb{R}^{r\times(c-2N)}$.[1]
3. Calculate $\gamma_1(j) = |\overline{g}_i(j;1)| + |\overline{g}_i(j;2)| + |\overline{g}_i(j;3)|, \quad 1 \leqslant j \leqslant c - 2N.$
4. Put $\hat{p}_0 = \hat{p}_1 = N + 1$ and $\phi_1 = \phi_2 = [\theta_N, \vdots \gamma_1 \vdots \theta_N]$.
5. For any $i = 2, \ldots, r$, do the following:
   (a) for any $k = 1, \ldots, 2N + 1$
      i. Calculate the scalar-valued row vector γ_i by $\gamma_i(j) = \sum_{\kappa=1}^{3} |\overline{g}_i(j;\kappa)|$ $1 \leqslant j \leqslant c - 2N$;
      ii. Set $h^k = [\theta_{k-1}, \gamma_i, \theta_{2N-k+1}]$;
      iii. Find $m = \max\{k, \hat{p}_{i-1}, \hat{p}_{i-2}\}$ and $n = \min\{k, \hat{p}_{i-1}, \hat{p}_{i-2}\} + c - 1$;
      iv. Calculate $\mathscr{J}(k) = \frac{1}{n-m+1}\sum_{j=m}^{n} |h^k(j) - 2\phi_1(j) + \phi_2(j)|^{\alpha}$;
   (b) Find $\hat{p}_i = \arg\min_{1\leqslant k\leqslant 2N+1} \mathscr{J}(k)$;
   (c) Set $\phi_2 \leftarrow \phi_1$ and $\phi_1 \leftarrow h^{\hat{p}_i} = \left[\theta_{\hat{p}_i-1} \vdots \gamma_i \vdots \theta_{2N-\hat{p}_i+1}\right]$;
   (d) $\tilde{f}_i = [\theta_{(\hat{p}_i-1)\times 3} \vdots g_i \vdots \theta_{(2N-\hat{p}_i+1)\times 3}]$;
6. Extract $\hat{f} \in \mathbb{R}^{r\times c}$ from $\tilde{f} \in \mathbb{R}^{r\times(c+2N)}$ by eliminating $2N$ columns at the extreme left and right ends that contain the largest number of zeros (jitter).

Notice that step 5(a)i can be replaced by any operation that transforms a color image into a gray-value image. A popular way to do this transform is

$$\gamma_i(j) = \sqrt{\overline{g}_i(j;1)^2 + \overline{g}_i(j;2)^2 + \overline{g}_i(j;3)^2}, \quad 1 \leqslant j \leqslant c - 2N.$$

However, the computational cost of the latter is much higher than the one used in step 5(a)i. Let us mention that it is important that the obtained gray-value image has a good contrast. In all experiments we realized, the calculation proposed in step 5(a)i gave rise to satisfying results. Some refinements by the weighting the different color channels in this step could improve the results in some cases; we did not explore such issues.

Computation times. The computational time for a color image is naturally higher than for a gray-value image. However, in our Algorithm 3, most of the calculation is done using a gray-value version of the color image. We did some comparisons under the same conditions described at the end of Section 4. For a $512 \times 512 \times 3$ image and $N = 7$ we got the solution in 1 second for $p = 1$ and in 1.4 second for $p = 0.5$.

[1] More precisely, $\forall i \in \{1, \ldots, r\}$ we have

$$\begin{aligned} &g_i^L(j;\kappa) = g_i(j;\kappa), \quad \forall j \in \{1, \ldots, N\}, && 1 \leqslant \kappa \leqslant 3;\\ &\overline{g}_i(j;\kappa) = g_i(j;\kappa), \quad \forall j \in \{N+1, c-N\}, && 1 \leqslant \kappa \leqslant 3;\\ &g_i^R(j;\kappa) = g_i(j;\kappa), \quad \forall j \in \{r-N+1, r\}, && 1 \leqslant \kappa \leqslant 3. \end{aligned}$$

6.2. Refinements

The refinements relevant to the first row, namely Algorithms 1(a) and 1(b), are straightforward to extend to Algorithm 2, by using the gray-value transform γ. The modification for images involving important vertical features, formalized in Algorithm 1(c), is easily extended to color images:

Algorithm 2(c)

Do Algorithm 2 (or 2(a), or 2(b)); the only change is to replace step 5(a) iv by Eq. (15):

$$5(a)i \quad \mathscr{J}(k) = \frac{1}{n-m+1} \times \sum_{j=m}^{n} \left(|h^k(j) - 2\phi_1(j) + \phi_2(j)| + \beta|h^k(j) - \phi_1(j)|\right)^{\alpha}.$$

6.3. Illustrations

In all cases, we apply Algorithm 2 or its modifications (e.g. Algorithm 2(c)) with $N = M + 1$.

The Man image in Fig. 12 incurs a uniform (quantized) jitter on $\{-8, \ldots, 8\}$ (shown in the left upper image). Dejittering is realized using Algorithm 2 for $\alpha = 1$. The displacement error $\hat{d} - d$, shown on the left lower image, ranges on $[-17, 3]$ and has quite a lot of constant pieces: we have $e_\infty = 3.76\%$ and $e_0^A = 4.19\%$ which seem too large. A more careful analysis shows that its most important part is corresponds to the sky area which is very homogeneous. Part of it reaches the level of the boat. For this reason, we present a zoom of the boat in the original and the dejittered image bit it is difficult to see the wrong displacement. There are errors also in the bottom part of the image; it corresponds to the ground area which is quite homogeneous as well. Overall, the displacement error remains invisible to the naked eye and PSNR = 33.82.

The image of a cheetah (707×579) in Fig. 13 has a very textured appearance. It is degraded with independent uniform jitter on $\{-20, \ldots, 20\}$. The restoration is the same for $\alpha = 1$ and $\alpha = 0.5$. It is nearly-perfect since $e_\infty = 0.35\%$ and $e_0^A = 0.28\%$.

Baboon (512×512) in Fig. 14 is degraded with strong jitter corresponding to $\mathscr{N}(0, \sigma^2 \mathbb{1}_r)$ for $\sigma = 6$, truncated and quantized on $\{-12, \ldots, 12\}$. The restoration using Algorithm 2 for $\alpha = 1$ is acceptable. The restoration for $\alpha = 0.5$ is visually better. We apply also Algorithm 2(c) for $\alpha = 0.5$ and $\beta = 2$ or $\beta = 3$. Now the visual result is really satisfying even though the error measures are worse than for Algorithm 2, $\alpha = 1$.

In Fig. 15 we consider the same image contaminated with low-frequency and high-frequency sinusoidal jitter, quantized on the set $\{-6, \ldots, 6\}$. The results obtained using Algorithm 2 are acceptable (not displayed). Better results are obtained using Algorithm 2(c) for $\alpha = 0.5$ and $\beta = 3$. Let us mention that these are not sensitive to the exact value of β.

Fig. 16, size 542×410, shows a store window composed of numerous golden jewelries which are very finely engraved. It was corrupted with independent jitter which is uniform on $\{-8, \ldots, 8\}$. Both restorations using Algorithm 2 for $\alpha = 1$ and $\alpha = 0.5$ are visually very satisfying and the PSNR is high. The restoration for $\alpha = 0.5$ is nearly perfect since $e_\infty = 0.24\%$ and $e_0^A = 0.37\%$ (one pixel maximal horizontal error on two rows).

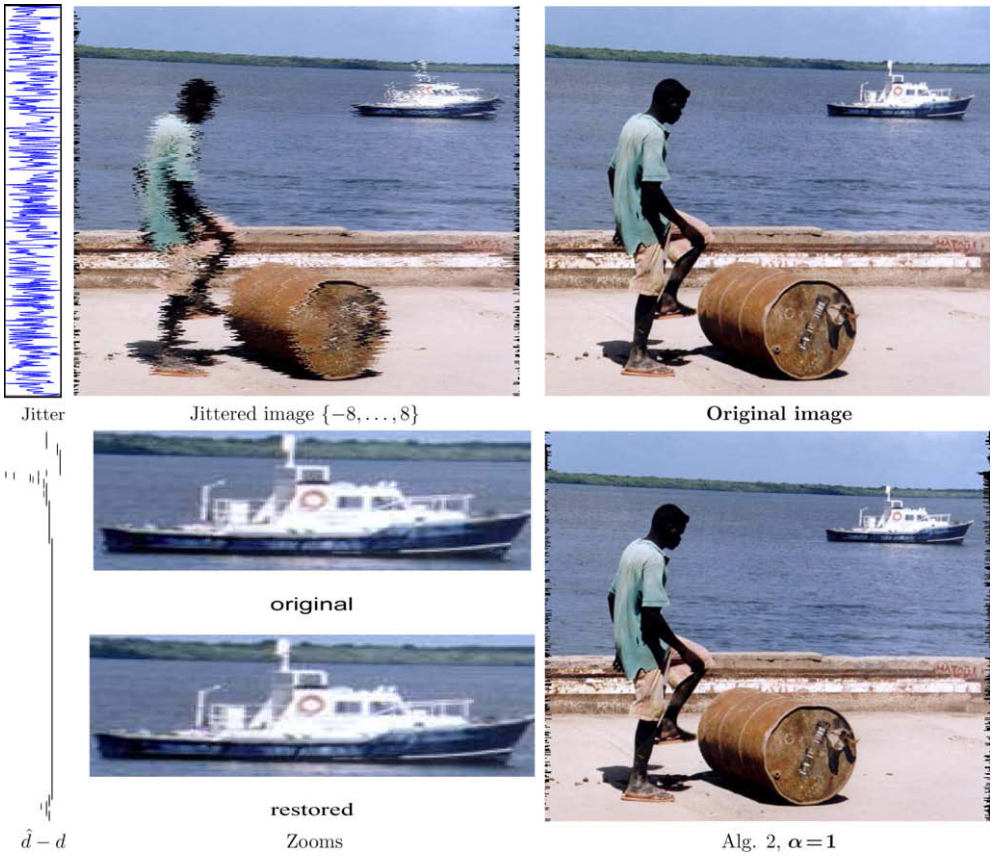


Fig. 12. Man image, 478 × 532. The jitter is quantized uniform on $\{-8,\dots,8\}$. The dejittered image is obtained using Algorithm 2 for $\alpha = 1$. The errors read MAE = 1.45, PSNR = 33.82, $e_1 = 0.76$, $e_\infty = 3.76\%$ and $e_0^4 = 4.19\%$.

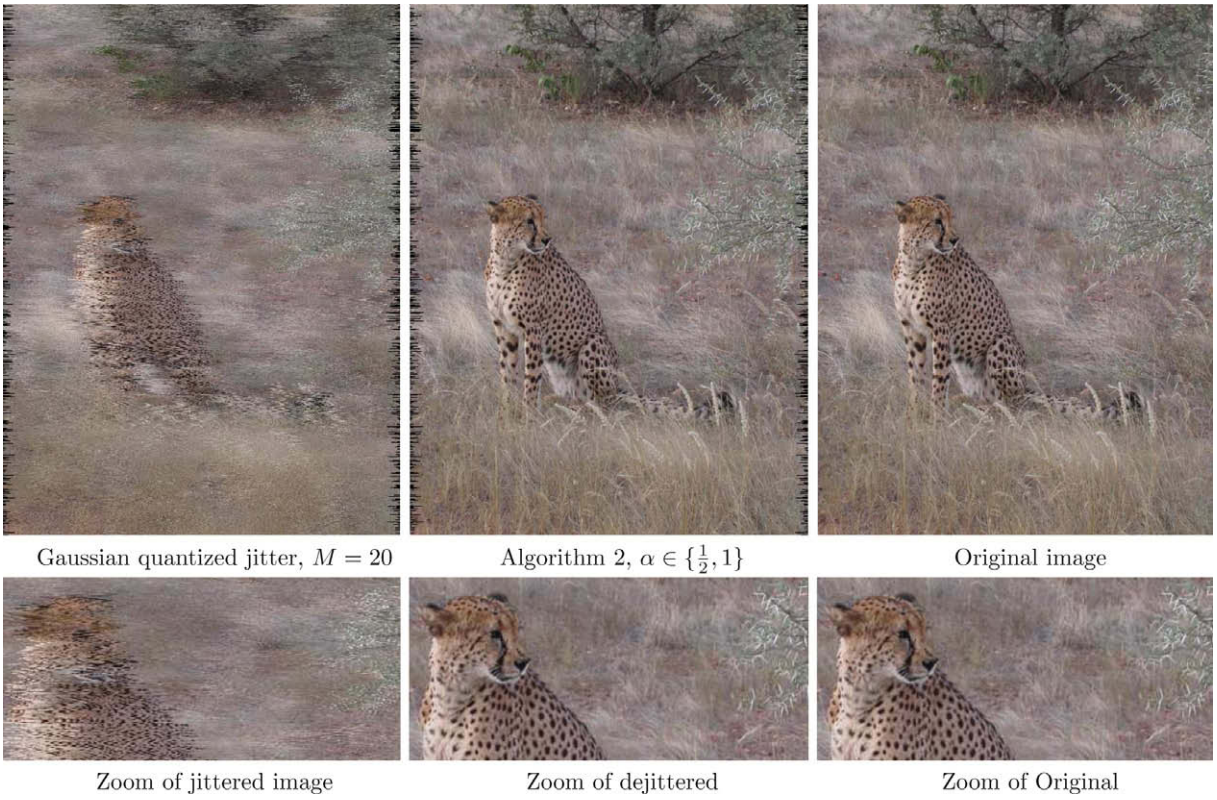


Fig. 13. Cheetah, 707 × 579. The jitter is independent and uniform on $\{-20,\dots,20\}$. The dejittered image is obtained using Algorithm 2 for $\alpha = 0.5$ or $\alpha = 0.5$. The errors read MAE = 0.52, PSNR = 37.27, $e_1 = 0.06$, $e_\infty = 0.35\%$ and $e_0^4 = 0.28\%$.

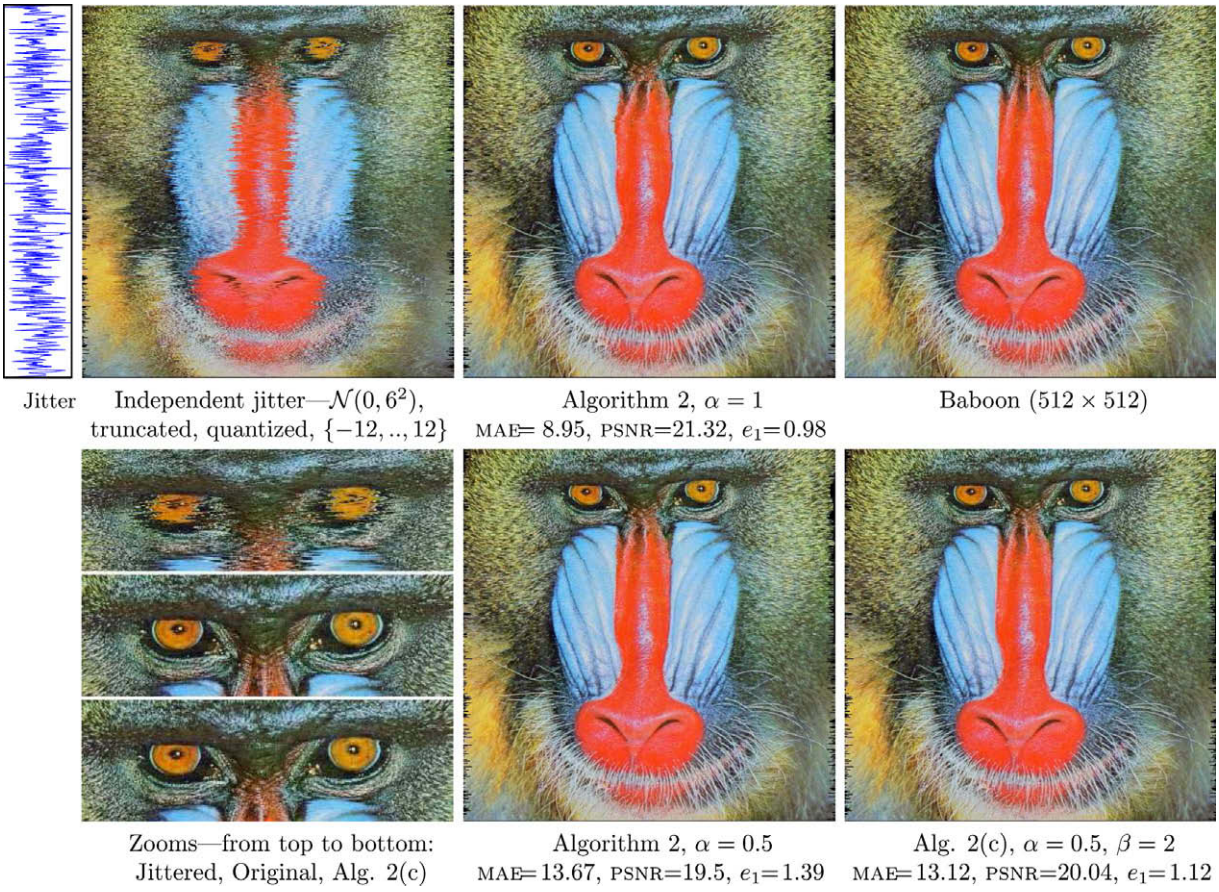


Fig. 14. Baboon image restored using different dejittering schemes.

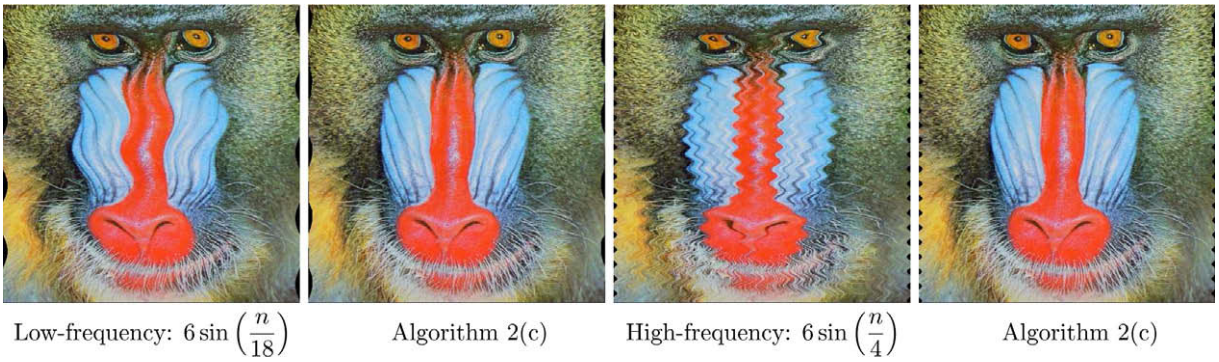


Fig. 15. The sinusoidal jitter is quantized on $\{-6,\ldots,6\}$. Dejittering uses Algorithm 2(c) for $\alpha = 0.5,\ \beta = 3$.

7. Noisy jittered images: dejitter then denoise

Our approach is to first dejitter images using the ideas of Algorithms 1 or 2, and then—at a second stage—to apply a fast, standard denoising method such as shrinkage estimation—see e.g. [2,10]. A variety of fast denoising methods could be envisaged at the second stage.

7.1. Moderate noise

In the presence moderate of noise (SNR equal to 15–20 dB or more), taking $\alpha < 1$ may be harmful since it favors too strongly locally polynomial constraint along the columns of the image; the latter is less characteristic in the presence of noise or other impairments. Whenever Algorithms 1 and 2 can be applied, we should choose

$$\alpha = 1.$$

In particular Algorithm 1(c) (resp. 2(c)) has provided better results in several cases.

The Boat image (256×256) in Fig. 17 is corrupted with zero-mean white Gaussian noise with 20 dB SNR and with independent uniform jitter on $\{-6,\ldots,6\}$. The dejittering step is realized using Algorithm 1 for $\alpha = 1$. The results are close to the noisy non-jittered images, shown on the third row. Denoising of the latter images is performed using hard-thresholding of the coefficients of the 2D Daubechies wavelet transform with 2 vanishing moments with threshold $T = 15$. The whole restoration is fast and

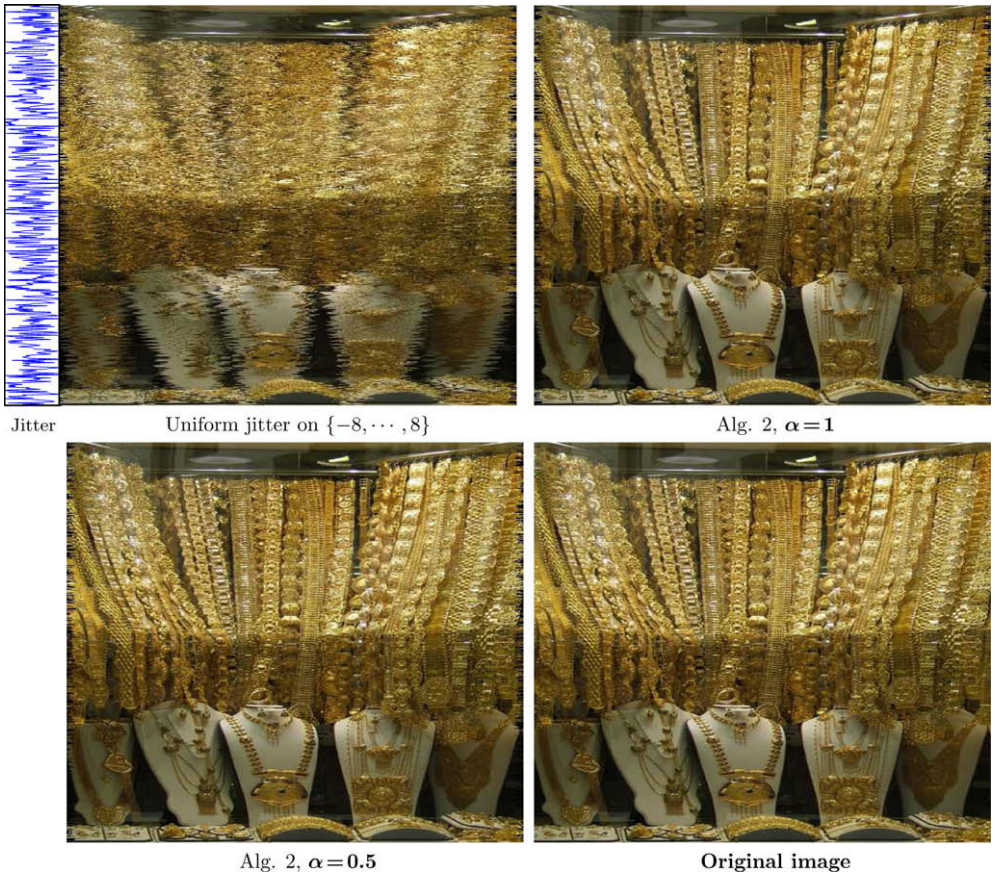


Fig. 16. Jewelry image, 542 × 410, with independent uniform jitter on $\{-8,\ldots,8\}$. The restoration using Algorithm 2 for $\alpha = 1$ yields errors MAE = 0.2, PSNR = 40.34, $e_1 = 0.06$ and $e_\infty = 7$. The restoration for $\alpha = 0.5$ is nearly perfect, MAE = 0.14, PSNR = 45.15, $e_1 = 0.03, e_\infty = 0.24\%$ and $e_0^4 = 0.37\%$.

easy since the shrinkage restoration we apply is almost instantaneous. The thresholds are chosen in order to obtain a good denoised image. The restored image is pretty clean. The obtained PSNR with respect to the original image is 25.90.

The picture of Lena in Fig. 18 (512 × 512) is corrupted with white zero-mean Gaussian noise (15 dB SNR) and independent uniform jitter on $\{-6,\ldots,6\}$. This image is more sophisticated than Boat and involves an important vertical column in the background. Algorithm 1 was not efficient enough for dejittering and the result is not displayed. Instead, Algorithm 1(c) yields better results for a large set of values for β; the dejittered image in Fig. 18 corresponds to $\alpha = 1$ and $\beta = 3$. The denoising step is performed by hard thresholding of the 2D Daubechies wavelet transform with 4 vanishing moments for $T = 30$. Compared with the original (see Fig. 6), for the restored image we have PSNR = 28.79.

The peppers in Fig. 19 are corrupted in the same way as the previous two images. Taking into account the presence of important nearly-vertical features, we dejitter the noisy image using Algorithm 1(c) for $\beta = 3$. Here again, the robustness with respect to the choice of β is good. The denoising step is realized in the same way, for $T = 30$ again. The result can be compared with the original image in Fig. 7.

7.2. Strong noise

When the noise is strong (e.g. with a SNR less than 15 dB), we suggest a sightly different scheme which keeps a comparable computational coast. The idea is to slightly denoise each rows using a fast shrinkage estimator and to replace in the dejittering function $|.|^\alpha$ by a better adapted edge-preserving function φ.

For $q \in \mathbb{N}_+$, let $W : \mathbb{R}^{1\times q} \to \mathbb{R}^{1\times q}$ denote a 1D wavelet transform and W^* its inverse. Let us also introduce the hard thresholding operator $\tau : \mathbb{R}^{1\times q} \to \mathbb{R}^{1\times q}$ by

$$\tau_T(\gamma)(j) = \begin{cases} 0 & \text{if} |\gamma(j)| \leqslant T \\ \gamma(j) & \text{otherwise} \end{cases} \quad 1 \leqslant j \leqslant q, \tag{17}$$

where T is a threshold. It is well known that hard thresholding is asymptotically optimal in the minimax sense if γ is contaminated with white Gaussian noise of standard deviation σ and $T = \sigma\sqrt{2\ln n}$. In practice, we are far from these asymptotical conditions and this optimal thresholding is known to oversmooth edges. The latter can be harmful for the dejittering goal. For this reason we prefer an under-optimal threshold T.

Before to run the algorithm, several details must be fixed in advance.

- Fix $N > M$, e.g., $N = M + 1$.
- Choose a 1D wavelet transform W (e.g. Daubechies wavelets).
- Fix an *under-optimal* threshold T and the coarsest level of the decomposition L (e.g. 1 or 2).
- Choose an edge-preserving potential function $\varphi : \mathbb{R}_+ \to \mathbb{R}_+$ and $\alpha > 0$, e.g.

$$\varphi(t) = |t|^\alpha \text{ or } \varphi(t) = \begin{cases} t^2/2 & \text{if} \quad |t| \leqslant \alpha \\ \alpha|t| - \alpha^2/2 & \text{if} \quad |t| > \alpha. \end{cases} \tag{18}$$

- According to the nature of the image (see Section 4.2.2), choose $\beta \geqslant 0$.

Algorithm 3. (*Quite noisy images*)

1. Put $\begin{cases}\text{RGB}: & \tilde{f}_1=[\theta_{N\times 3}, g_1, \theta_{N\times 3}].\\ \text{Gray}: & \tilde{f}_1=[\theta_N, g_1, \theta_N].\end{cases}$
2. Split g into 3 sub-matrices $g=\left[g^L \vdots \overline{g} \vdots g^R\right]$ where $g^L\in\mathbb{R}^{r\times N}$, $\overline{g}\in\mathbb{R}^{r\times(c-2N)}$ and $g^R\in\mathbb{R}^{r\times N}$.
3. $\begin{cases}\text{RGB}: & \gamma_1(j)=|\overline{g}_1(j;1)|+|\overline{g}_1(j;2)|+|\overline{g}_1(j;3)|,\ 1\leqslant j\leqslant c-2N.\\ \text{Gray}: & \gamma_1=\overline{g}_1.\end{cases}$
4. Compute $\tilde{\gamma}_1=W^*(\tau_T(W\gamma_1))$.
5. Set $\hat{p}_0=\hat{p}_1=N+1$ and $\phi_1=\phi_2=[\theta_N,\tilde{\gamma}_1,\theta_N]$.
6. For any $i=2,\ldots,r$ do the following:
   (a) for any $k=1,\ldots,2N+1$
      0. $\begin{cases}\text{RGB}: & \gamma_i(j)=|\overline{g}_i(j;1)|+|\overline{g}_i(j;2)|+|\overline{g}_i(j;3)|,\ 1\leqslant j\leqslant c-2N;\\ \text{Gray}: & \gamma_i=\overline{g}_i;\end{cases}$
      i. $\tilde{\gamma}_i=W^*(\tau_T(W\gamma_i))$;
      ii. $h^k=[\theta_{k-1},\tilde{\gamma}_i,\theta_{2N-k+1}]$;
      iii. $m=\max\{k,\hat{p}_{i-1},\hat{p}_{i-2}\}$ and $n=\min\{k,\hat{p}_{i-1},\hat{p}_{i-2}\}+c-1$;
      iv. $\mathscr{J}(k)=\frac{1}{n-m+1}\sum_{j=m}^{n}\varphi\left(|h^k(j)-2\phi_1(j)+\phi_2(j)|+\beta|h^k(j)-\phi_1(j)|\right)$;
   (b) find $\hat{p}_i=\arg\min_{1\leqslant k\leqslant 2N+1}\mathscr{J}(k)$;
   (c) Set $\phi_2\leftarrow\phi_1$ and $\phi_1\leftarrow h^{\hat{p}_i}=\left[\theta_{\hat{p}_i-1},\tilde{\gamma}_i,\theta_{2N+1-\hat{p}_i}\right]$;
   (d) $\begin{cases}\text{RGB}: & \tilde{f}_i=\left[\theta_{(\hat{p}_i-1)\times 3}, g_i, \theta_{(2N-\hat{p}_i+1)\times 3}\right];\\ \text{Gray}: & \tilde{f}_i=[\theta_{\hat{p}_i-1}, g_i, \theta_{2N-\hat{p}_i+1}].\end{cases}$
7. Find $\hat{f}\in\mathbb{R}^{r\times c}$ from $\tilde{f}\in\mathbb{R}^{r\times(c+2N)}$ by eliminating $2N$ columns at the extreme left and right ends that contain the largest number of zeros.

Hard-thresholding is better than other shrinkage functions since it keeps unchanged the important coefficients bearing the most important information. *We do not recommend a preliminary denoising of the entire image before the dejittering step* since this can destroy important information in the vertical direction. In our implementation, we used Daubechies 1D wavelets. This stage is of critical importance and additional research is needed to exhibit more accurate and fast row denoising methods.

Remark 8. Step 2 of Algorithm 3 supposes that the length of γ_i, $1\leqslant i\leqslant r$ is a power of 2, in order to apply a wavelet transform. Whenever this is not the case, some arrangements are necessary. If the image width is a power of 2 we apply the row under-denoising on the entire row and split each row into three pieces of length $N, c-2N$ and N, so that vertical matching using $\mathscr{J}$ is done only on rows that are not affected by the jitter (Remark 2). Otherwise, we split $g=\left[g^L \vdots \overline{g} \vdots g^R\right]$ so that the number of columns of $\overline{g}$ is a power of 2, no larger than $c-2N$.

The function φ involved in $\mathscr{J}$ (step 6(a)iv) must be edge-preserving. If the row-denoising in steps 4 and 6(a) is efficient enough, φ can be non-smooth at zero (e.g. $\varphi(t)=|t|^\alpha$ for $\alpha=1$ or $\alpha=0.5$). Otherwise, it is safer to use a smooth-at-zero function φ in order to relax the polynomial constraint in the vertical direction. In the latter case, several choices can be done, giving quite similar results, e.g. $\varphi(t)=\sqrt{\alpha+t^2}$, $\alpha>0$, or $\varphi(t)=\log(\cosh(\alpha t)),\alpha>0$, or Huber's function as given in the right side of (18) which is fast to compute and quadratic near the origin.

7.3. Experiments

The peppers image in Fig. 20 is degraded with white, zero-mean Gaussian noise with 10 dB SNR and uniform jitter on the set $\{-6,\ldots,6\}$. Dejittering is realized using Algorithm 3 for $N=M+1$, $\beta=5$, Daubechies 1D wavelet thresholding for $T=50$ and $\varphi(t)=|t|$. The obtained dejittered image is denoised using 2D Daubechies wavelets with 4 vanishing moments, whose coefficients are hard-thresholded using $T=60$.

Boat in Fig. 21 is corrupted with 10 dB white zero-mean Gaussian noise and strong jitter, uniform on $\{-8,\ldots,8\}$ (i.e. $M=8$). Its restoration using B-TV [14] is unsatisfactory. The result obtained using B & S [5] is better. The underlying image contains a lot of sloping (non-vertical) fine lines. This suggests to run our Algorithm 3 for $\beta=0$ and $\varphi(t)=|t|^\alpha$ for $\alpha=0.5$ in step 6(a)iv; furthermore, in step 6(a)i we use hard-thresholding of the Daubechies wavelet coefficients with 2 vanishing moments for $T=30$. The obtained dejittered image is very satisfying. Denoising is performed by hard thresholding of the curvelet transform of the dejittered image using the *enhanced-denoising* program in the *CurveLab 2.1.2 toolbox*.

In Fig. 22 we focus on restoring the same noisy version of the boat image (zero-mean Gaussian noise with 10 dB SNR), deformed by sinusoidal jitter. We consider both low-frequency and high-frequency jitter whose details are given in the caption. Dejittering is realized using Daubechies wavelets with 2 vanishing moments and threshold $T=30$. In both cases we observe that the dejittering step yields a very correct usual noisy image. Let us mention that the choice of the number of vanishing moments in the Daubechies wavelet basis plays an important role. The ultimate denoising step is performed again using the *enhanced-denoising* program in the *CurveLab 2.1.2 toolbox*. The whole restoration has quite a natural appearance and is close to the original—visible on the right side of Fig. 11.

The experiments in Figs. 23 and 24 illustrate closely the B-TV method of Shen described in [14]. Data are contaminated with white zero-mean Gaussian noise with 40 dB SNR. The jittering vector is a realization of a binomial distribution, which in practice led us to random displacements within the range $\{-8,\cdots,+9\}$. We used the codes of Dr. J. Shen to generate the noisy data and the B-TV restorations, according to [14]; they are roughly the same as in [14]. Then we processed the same data sets using our algorithms. In Fig. 23 we present our full method, first dejitter then denoise, while in Fig. 24 we present just the dejittered noisy image.

7.4. Comments on dejittering of noisy images

We would like to notice several points concerning the restoration of noisy jittered images.

- The 1D row under-denoising in step 6(a)i of Algorithm 3 is of critical importance; further improvements need to optimize the choice of a frame for shrinkage estimation or choose a different approach.
- Denoising of the dejittered image can be done by various methods. The approach of [5] suggests that PDE-based denoising might remove some remaining artifacts. Much better quality can be obtained using hybrid methods, such as [4,3] etc., but this kind of methods need a considerable computation time.
- Overall, the proposed methods work well but they involve a numerous parameters whose role need to be clarified in order to fix them in a more pragmatic way. Nevertheless, it is much easier to fix them compared to methods such as those proposed in [14,5,6].
- The presence of additional impairments (especially when they involve sets of pixels) are harmful for the proposed methods. Adapted ways to deal with such situations have to be envisaged.
- Considerable improvement and simplifications can be expected if we use the correlation between consecutive images in a video sequence.

8. Conclusion and perspectives

The proposed dejittering approach is very simple, the obtained results have a remarkable quality while the algorithms are very fast, nearly real-time. The crux of the approach is (1) to minimize

Fig. 17. Boat (256×256), corrupted with white centered Gaussian noise and independent uniform jitter on $\{-6,\ldots,6\}$. Dejittering is realized using Algorithm 1 for $\alpha = 1$. The errors for the dejittered images (with respect to the noisy jitter-free images) read: MAE = 4.73, PSNR = 25.63 and $e_1 = 1.03$. Denoising is realized by hard-thresholding of the coefficients of the 2D Daubechies wavelet transform.

Fig. 18. Lena image, 512×512, corrupted with 15 dB SNR white centered Gaussian noise and with independent, uniform jitter on $\{-6,\ldots,6\}$. Dejittering is done using Algorithm 1(c) for $\alpha = 1$ and $\beta = 3$. The errors with respect to the noisy non-jittered image read: MAE = 3.19, PSNR = 29.43 and $e_1 = 0.59$. Denoising: hard thresholding of the coefficients of the 2D Daubechies wavelet transform, $T = 30$.

the p-power, for $p \in [0.51]$ of the magnitude of the second-order differences in the vertical direction and (2) to exclude from the displacement estimation all pixels at the left-end and the right-end of the image that can be due to the jitter, and (3) to dejitter the rows successively. In the presence of additive noise, we proceed in two steps: we slightly denoise each row so that dejittering is done in the same way; then we use standard denoising tools.

The natural evolution of this work is to involve it in a full video sequence restoration and to take advantage of the correlation between consecutive frames. Much better results can be expected then.

The dejittering of images corrupted with a strong noise clearly needs further improvements. Alternative ways to deal with jitter in presence of various impairments should be envisaged.

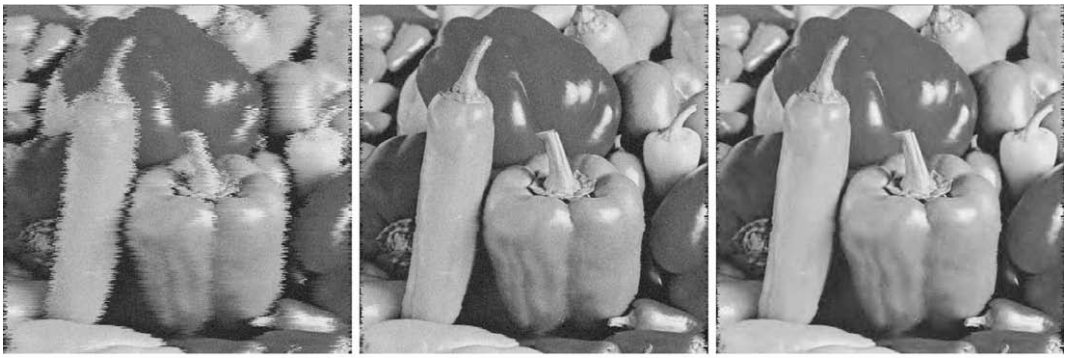


Fig. 19. Peppers (512 × 512) corrupted with 15 dB SNR white centered Gaussian noise and with independent, uniform jitter on $\{-6, \ldots, 6\}$. Dejittering is done using Algorithm 1(c) for $\beta = 3$. The errors with respect to the noisy non-jittered image read: MAE = 5.8, PSNR = 27.59 and $e_1 = 0.73$. Denoising is done by hard thresholding of the coefficients of the 2D Daubechies wavelet transform for $T = 30$, PSNR = 29.34.

Fig. 20. Peppers (512 × 512) contaminated with 10 dB white Gaussian noise and independent uniform jitter on $\{-6, \ldots, 6\}$. Dejittering is done using Algorithm 3 for $T = 50$, $\varphi(t) = |t|$ and $N = M + 1 = 7$. Denoising uses hard-thresholding of the coefficients of the 2D Daubechies wavelet transform for $T = 50$.

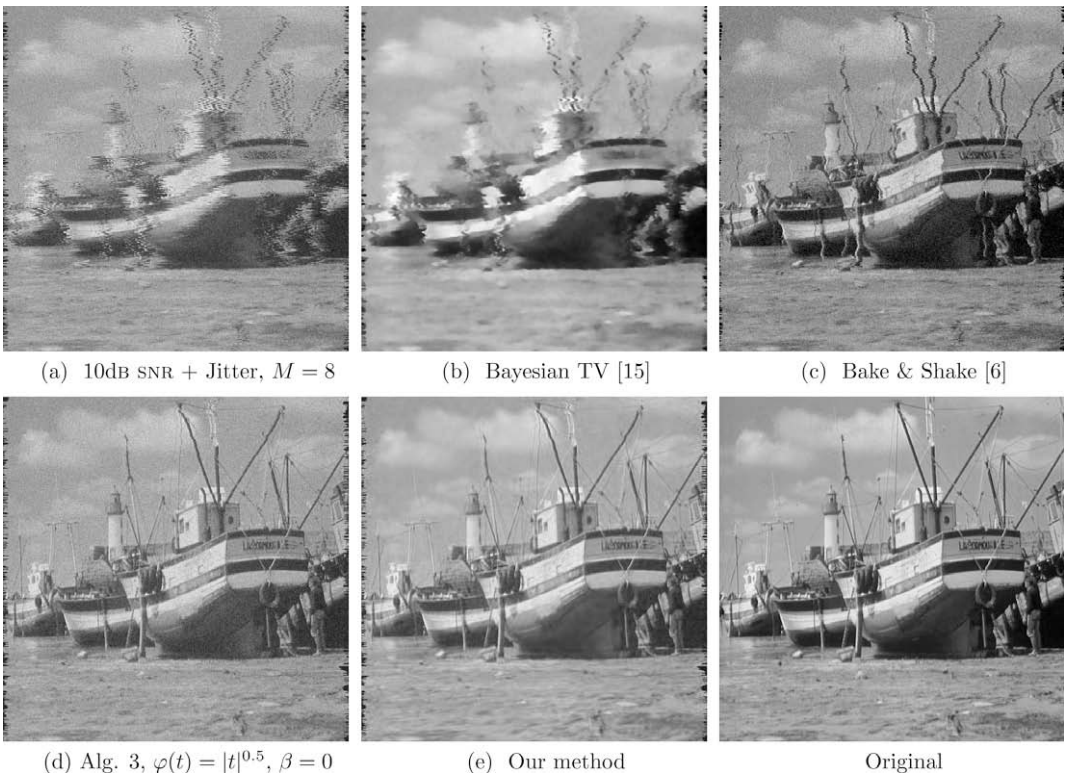


Fig. 21. (a)Boat image, 512 × 512 contaminated with 10 dB white, zero-mean Gaussian noise and independent jitter, uniform on $\{-M, \ldots, M\}$ for $M = 8$. The restoration in (b) is obtained by the method proposed in [14]. The result of the Bake and Shake method [5] is shown in (c). Dejittering in (d) is obtained using Algorithm 3 for $T = 50$, $\varphi(t) = |t|^{0.5}$, $\beta = 0$ and $N = M + 1$. Denoising in (e) is obtained by hard thresholding of the curvelet transform of the dejittered image in (b) using the *enhanced-denoising* program in the *CurveLab 2.1.2 toolbox*.

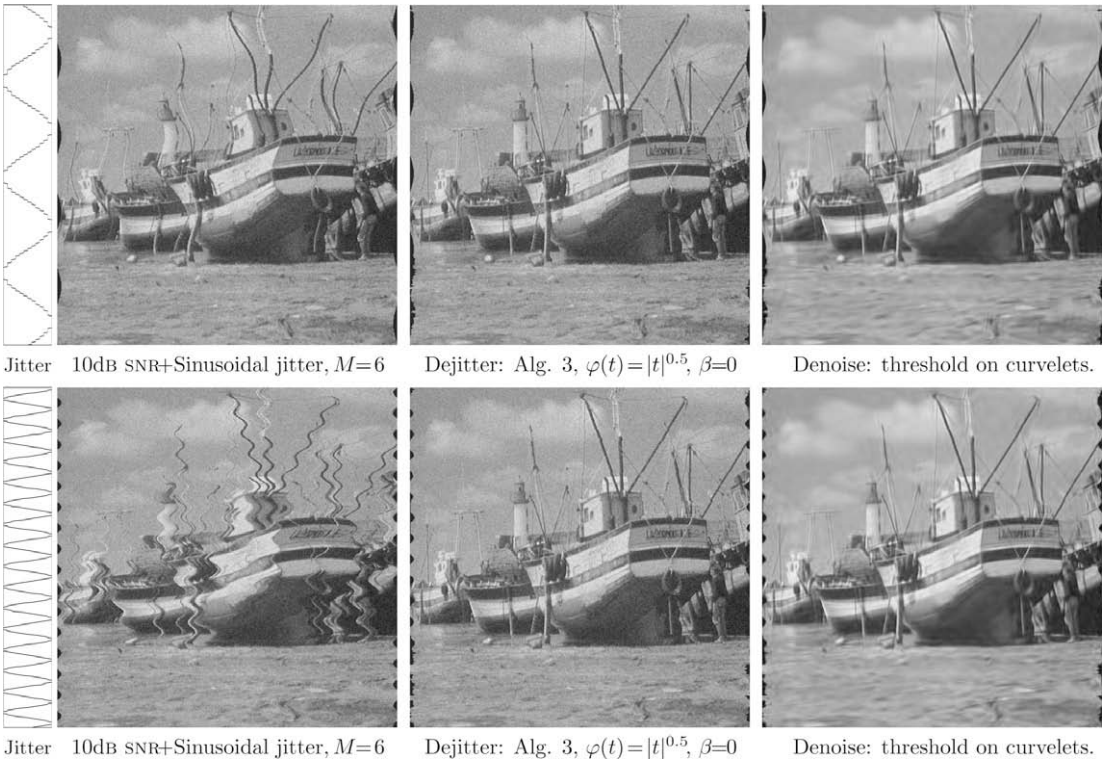


Fig. 22. Boat image, 512×512 contaminated with 10 dB white, zero-mean Gaussian noise and sinusoidal jitter—upper row: $6\sin(\frac{n}{18}),\ 1 \leqslant n \leqslant 512$, quantized on $\{-6,\ldots,6\}$; lower row $6\sin(\frac{n}{5}),\ 1 \leqslant n \leqslant 512$, quantized on $\{-6,\ldots,6\}$. In both cases dejittering is realized using Algorithm 3 for $T = 30,\ \varphi(t) = |t|^{0.5}$, $\beta = 0$ and $N = M + 1$. Denoising is obtained using the *enhanced-denoising* program in the *CurveLab 2.1.2 toolbox*.

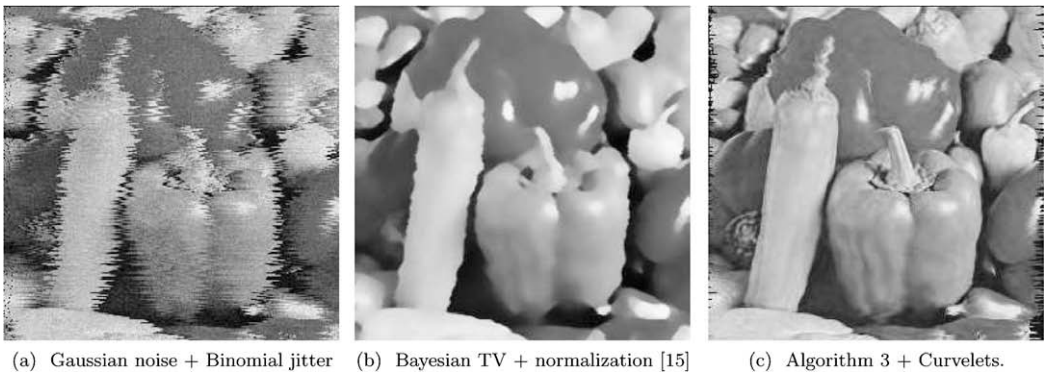


Fig. 23. (a) Peppers image, 256×256 contaminated with 40 dB white Gaussian noise and binomial jitter in $\{-8,\cdots,+9\}$. The result in (b) is obtained using the method of Shen [14]. (c) involves 2 steps: dejittering is done using Algorithm 3 using 1D Daubechies wavelets with 2 vanishing moments, $T = 20,\ L = 1\ \varphi(t) = |t|,\ \beta = 1$ and $N = 9$. Denoising is done by hard-thresholding of the coefficients of the curvelet transform of the dejittered image along with cycle-spinning [1].

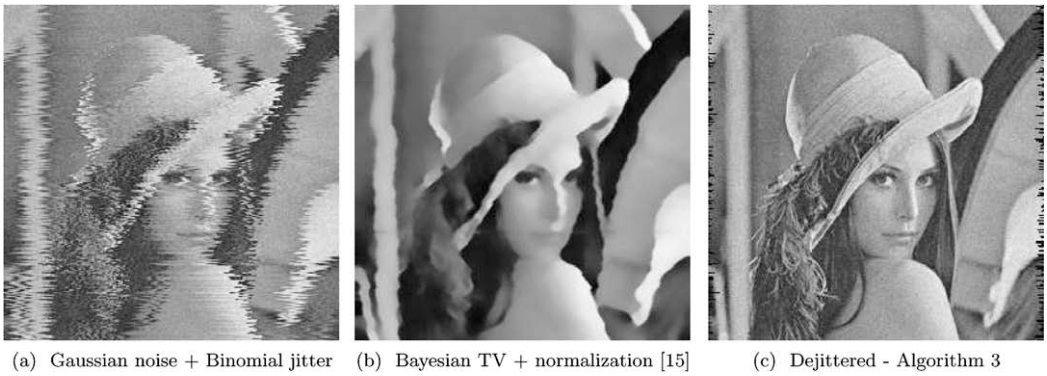


Fig. 24. Lena (256×256). Jittering follows a binomial distribution in $\{-8,\cdots,+9\}$.

Acknowledgments

- The author acknowledges the support of the French Agence Nationale de la Recherche (ANR), under grant FREEDOM (ANR07-JCJC-0048-01),"Films, REstauration Et DOnnes Manquantes".
- The author would like to thank Louis LABORELLI (*Institut National de l'Audiovisuel*—France) for his permanent and clever discussion on numerous practical questions concerning jittered video.
- The author thanks Dr. Jackie Shen (actually Manager at Barclays Capital, Wall Street) for providing me with his Matlab codes used for the experiments depicted in [14].
- The author is extremely thankful to Dr. Sung-Ha KANG, Assistant Professor (School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332-0160), who realized experiments both with the Bake and Shake method [6] and the Bayesian TV method of Shen [14] to restore images in Figs. 6, 7, 10 and 21.

Appendix A

Tables 1–3.

Table 1
Means of the errors (MAE, $e_1, e_\infty\%, e_0\%, e_0^A\%$) based on 1000 experiments (see Section 5). The letter "U" reminds that the jitter is uniform on $\{-6,\dots,6\}$ while "G"—that it is truncated Gaussian, quantized on the same set. The best performance in terms of mean(MAE), mean(e_1), mean($e_\infty\%$) and mean($e_0^A\%$) for each image and each case of jitter is in bold.

Alg.	α	Mean	LENA		PEPPERS		BARBARA		BOAT	
			U	G	U	G	U	G	U	G
1	1	MAE	0.0910	0.3326	0.571	1.0667	3.5171	5.5625	**0.0920**	**0.4299**
		e_1	0.0412	0.1115	0.1942	0.3430	0.3989	0.6948	**0.0319**	**0.0783**
		e_∞ %	0.4814	0.3416	1.9566	1.9914	1.1125	0.9953	0.5412	0.3023
		e_0 %	2.0596	6.4406	8.8574	14.712	28.730	44.334	1.5512	5.6631
		e_0^A%	0.3069	0.2971	1.9984	2.0714	2.0245	1.9348	0.2293	**0.199**
1	$\frac{1}{2}$	MAE	**0.0851**	**0.3253**	**0.5594**	**1.0035**	3.5034	**4.4382**	0.1353	0.461
		e_1	**0.0388**	**0.1068**	**0.1804**	**0.3027**	0.4301	**0.5650**	0.0454	0.0897
		e_∞ %	**0.4732**	**0.3285**	1.6725	1.5715	1.1109	0.9141	0.6734	0.401
		e_0 %	1.9656	6.4055	8.833	14.355	27.992	36.492	2.2102	6.0908
		e_0^A	**0.2941**	**0.2810**	1.672	1.6781	1.5665	1.4115	0.4143	0.3266
1(a)	1	MAE	0.0910	0.3326	0.5954	1.0947	3.5456	5.5638	0.0953	0.432
		e_1	0.0412	0.1115	0.2027	0.3516	0.4034	0.6949	0.0324	0.0785
		e_∞ %	0.4814	0.3416	1.9357	1.9756	1.1068	0.9906	**0.5301**	**0.3008**
		e_0 %	2.0596	6.4406	9.0459	14.985	29.031	44.351	1.6012	5.692
		e_0^A%	0.3069	0.2971	1.8597	1.9534	2.0049	1.9243	**0.2331**	0.2002
1(a)	$\frac{1}{2}$	MAE	0.0851	0.3253	0.5676	1.03	3.5157	4.4398	0.1392	0.4719
		e_1	0.0388	0.1068	0.1845	0.3107	0.4315	0.5654	0.0465	0.0915
		e_∞ %	0.4732	0.3285	**1.6285**	**1.5234**	**1.0908**	**0.9064**	0.6572	0.3961
		e_0 %	1.9656	6.4055	8.8369	14.756	28.231	36.539	2.2693	6.2256
		e_0^A%	0.2941	0.2810	**1.4965**	**1.4849**	**1.5014**	**1.3928**	0.4237	0.3301
1(b)	1	MAE	0.1206	0.4594	0.5987	1.0989	**3.1244**	5.6618	0.1467	0.6639
		e_1	0.0653	0.1742	0.2048	0.3529	**0.3384**	0.714	0.0662	0.1495
		e_∞ %	1.7887	1.5762	2.1447	2.0658	1.9543	1.7873	2.1092	1.7545
		e_0 %	2.3828	9.0568	9.0646	15.057	24.511	43.769	2.4264	8.6869
		e_0^A%	0.7812	0.9859	2.0129	2.1076	2.0746	2.3221	0.7411	0.8413
1(b)	$\frac{1}{2}$	MAE	0.1196	0.4819	0.5869	1.0987	3.3685	4.6749	0.1591	0.5879
		e_1	0.0644	0.1750	0.1926	0.3306	0.4223	0.6024	0.071	0.1416
		e_∞ %	1.7752	1.5504	2.0033	1.8443	1.952	1.743	1.965	1.6744
		e_0 %	2.416	9.609	9.0338	15.369	25.137	36.673	2.6246	7.7424
		e_0^A%	0.7575	0.9336	1.7411	1.8446	1.5933	1.7325	0.875	0.9239

Table 2
Percentage of the cases when $e_0^A \leqslant 0.8\%$ jointly with $e_\infty \leqslant 0.4\%$ or $e_\infty \leqslant 0.6\%$ based on the same set of 1000 experiments (Table 1). The values for $e_\infty \leqslant 0.6\%$ are in italic to facilitate the reading. The best percentages for $(e_\infty \leqslant 0.4\%, e_0^A \leqslant 0.8\%)$ are in bold while the best ones for $(e_\infty \leqslant 0.6\%, e_0^A \leqslant 0.8\%)$ are in serif. For the interpretation of these result, see Remark 6.

Alg.	α	%	LENA		PEPPERS		BARBARA		BOAT	
			U	G	U	G	U	G	U	G
1	1	$e_\infty \leqslant 0.4$ %	52	71.4	1.7	0.0	8.9	11.8	48.7	74.6
		$e_\infty \leqslant 0.6$ %	*67.5*	*81.9*	*4.6*	*1.0*	*29.8*	*34.6*	55.8	*84*
1	$\frac{1}{2}$	$e_\infty \leqslant 0.4$ %	**52.8**	**73.5**	3.5	2	11.4	22.8	37.3	60.1
		$e_\infty \leqslant 0.6$ %	68.2	84.1	11.4	*8.3*	*28.2*	46.7	49.3	*78*
1(a)	1	$e_\infty \leqslant 0.4$ %	52	71.4	3.6	0.7	8.9	11.8	**49**	**74.8**
		$e_\infty \leqslant 0.6$ %	67.5	*81.9*	*9.3*	*3.2*	*30.4*	*34.9*	*56.7*	84.3
1(a)	$\frac{1}{2}$	$e_\infty \leqslant 0.4$ %	52.8	73.5	10	6.5	**11.5**	**22.9**	38.7	60.3
		$e_\infty \leqslant 0.6$ %	68.2	*84.1*	20.1	21.4	*29.1*	47.3	*51.7*	*78.7*
1(b)	1	$e_\infty \leqslant 0.4$ %	1.6	1.2	0.0	0.0	0	0	1.5	1.2
		$e_\infty \leqslant 0.6$ %	*4*	3.8	*0.3*	*0*	*1.5*	*0.0*	*4*	2.8
1(b)	$\frac{1}{2}$	$e_\infty \leqslant 0.4$ %	1.6	1.4	0.0	0	0.0	0.0	1.5	0.7
		$e_\infty \leqslant 0.6$ %	*4*	*4.1*	*0.5*	*0.0*	*1.3*	*0.3*	*3.4*	2.7

Table 3
Variances of the errors MAE and e_1, based on the same set of 1000 experiments of Table 1.

Alg.	α	Var	LENA		PEPPERS		BARBARA		BOAT	
			U	G	U	G	U	G	U	G
1	1	MAE	0.0148	0.3542	0.0573	0.4044	0.8986	4.4853	0.0206	0.5596
		e_0	0.0028	0.0296	0.0059	0.0256	0.0241	0.1262	0.0011	0.012
1	$\frac{1}{2}$	MAE	0.0108	0.3527	0.067	0.4625	0.3728	1.5198	0.0277	0.5502
		e_0	0.0022	0.0286	0.0062	0.0291	0.0079	0.0308	0.0017	0.012
1(a)	1	MAE	0.0148	0.3542	0.0537	0.3637	0.9146	4.4685	0.0225	0.5606
		e_0	0.0028	0.0296	0.0053	0.0229	0.0247	0.1258	0.0012	0.012
1(a)	$\frac{1}{2}$	MAE	0.0108	0.3527	0.0614	0.4041	0.3659	1.5345	0.0293	0.5688
		e_0	0.0022	0.0286	0.0057	0.0242	0.0077	0.0312	0.0017	0.0124
1(b)	1	MAE	0.0105	0.2832	0.0479	0.3519	0.3393	5.481	0.0257	0.739
		e_0	0.0025	0.0180	0.0047	0.0217	0.0081	0.157	0.0017	0.0147
1(b)	$\frac{1}{2}$	MAE	0.0092	0.31255	0.0504	0.4045	0.3723	1.9397	0.025	0.722
		e_0	0.0023	0.0191	0.0045	0.0236	0.0076	0.032	0.0017	0.0135

References

[1] E.J. Candès, L. Demanet, D.L. Donoho, L. Ying, Fast discrete curvelet transforms, SIAM Journal on Multiscale Modeling and Simulation 5 (3) (2006) 861–899.
[2] I. Daubechies, Ten Lectures on Wavelets, SIAM, Philadelphia, 1992.
[3] S. Durand, M. Nikolova, Denoising of frame coefficients using l1 data-fidelity term and edge-preserving regularization, SIAM Journal on Multiscale Modeling and Simulation 6 (2) (2007) 547–576.
[4] S. Durand, J. Froment, Reconstruction of wavelet coefficients using total variation minimization, SIAM Journal on Scientific Computing 24 (5) (2003) 1754–1767.
[5] S.-H. Kang, J. Shen, Video dejittering by Bake and Shake, Image and Vision Computing 24 (2) (2006) 143–152.
[6] S.-H. Kang, J. Shen, Image Dejittering Based on Slicing Moments, Springer Series on Mathematics and Visualization, Springer, 2007, pp. 35–55.
[7] A. Kokaram, P.M.B. Roosmalen, P. Rayner, J. Biemond, Line registration of jittered video, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 1997, pp. 2553–2556.
[8] A. Kokaram, Motion Picture Restoration, Springer-Verlag, 1998.
[9] L. Laborelli, Removal of video line jitter using a dynamic programming approach, in: Proceedings of the IEEE International Conference on Image Processing, 2003, pp. 331–334.
[10] S. Mallat, A Wavelet Tour of Signal Processing, Academic Press, London, 1999.
[11] M. Nikolova, Local strong homogeneity of a regularized estimator, SIAM Journal on Applied Mathematics 61 (2) (2000) 633–658.
[12] M. Nikolova, Minimizers of cost-functions involving nonsmooth data-fidelity terms. Application to the processing of outliers, SIAM Journal on Numerical Analysis 40 (3) (2002) 965–994.
[13] M. Nikolova, Analysis of the recovery of edges in images and signals by minimizing nonconvex regularized least-squares, SIAM Journal on Multiscale Modeling and Simulation 4 (3) (2005) 960–991.
[14] J. Shen, Bayesian video dejittering by BV image model, SIAM Journal on Applied Mathematics 64 (5) (2004) 1691–1708.
[15] P.L. Torroba, N.L. Cap, H.J. Rabal, W.D. Furlan, Fractional order mean in image processing, Optical Engineering 33 (2) (1994) 528–534.
[16] M. Welk, J. Weickert, F. Becker, C. Schnörr, C. Feddern, B. Burgeth, Median and related local filters for tensor-valued images, Signal Processing (special issue Tensor Signal Processing) (87) (2007) 291–308.
[17] M. Welk. Dynamic and geometric contributions to digital image processing, Habilitaion Thesis, University of Saarbruecken, 2008.
[18] G. Winkler, V. Aurich, K. Hahn, A. Martin, K. Rodenacker, Noise reduction in images: some recent edge-preserving methods, Pattern Analysis and Machine Intelligence 9 (4) (1999) 479–766.