Scale Space and Variational Methods in Computer Vision, Eds. X.-C. Tai, K. Morken, M. Lysaker, K.-A. Lie, LNCS 5567, Springer, pp. 439-451, 2009.

Fast dejittering for digital video images

Mila Nikolova

CMLA, ENS Cachan, CNRS, PRES UniverSud, France,
nikolova@cmla.ens-cachan.fr,
WWW home page: http://www.cmla.ens-cachan.fr/∼nikolova/

Abstract. We propose several very fast algorithms to restore jittered digital video frames (their rows are shifted) in one iteration. The restored row shifts minimize non-smooth and possibly non-convex local criteria applied on the second-order differences between consecutive rows. We introduce specific error measures to assess the quality of dejittering. Our algorithms are designed for gray-value, color and noisy images. Some of them can be considered as parameter-free. They outperform by far the existing algorithms both in quality and in speed. They are a crucial step towards real-time dejittering of digital video.

1 Intrinsic Dejittering

Image jitter consists in a random horizontal shift of each row of a video frame. It occurs when the synchronization row pulses are corrupted e.g. by noise or degradation of the storage medium, or in wireless transmission. The visual effect is disturbing since all shapes are jagged, *cf.* e.g. Fig. 4. Structured jitter can be provoked by acoustic or electrical interferences [7], *cf.* e.g. Fig. 8. *Time base corrector* machines recover with some success the row synchronization pulses. This operation is often unsuccessful or impossible [6]. An alternative—restoring the video frames directly from the jittered data, called *intrinsic dejittering* [5]—is much more flexible and widely applicable.

State of the Art. Intrinsic dejittering was invented in [5]. The method is based on a 2D auto-regressive (AR) image model. The unknown AR coefficients and row starts are estimated iteratively, jointly by blocs; a drift compensation is applied afterwards [6]. In [7], the ℓ_1 norm of the differences between 2 or 3 consecutive shifted rows is compared in the framework of dynamic programming. A fully Bayesian iterative method using a TV-based prior for joint dejittering and denoising is derived in [12]. The *Bake and Shake* method in [3] uses a good PDE image model (e.g. Perona-Malik) to recover the row positions. In [4], the same authors analyze the vertical slicing moments of images of bounded variation and derive a variational method (faster than [3] but less effective for difficult data).

Our Approach. We exhibit a pertinent model enabling to discriminate natural images from their jittered versions. Each row is restored based on the previously restored rows using a simple non-smooth and possibly non-convex local criterion. We thus construct *one-iteration* effective and fast dejittering algorithms. Noisy jittered images are restored in two stages: (a) dejittering of the raw data; (b) denoising of the obtained dejittered image.

2 The main points of our approach

Notations. For any positive integers m and n, the rows of a matrix $h \in \mathbb{R}^{m\times n}$ are denoted by h_i, $1 \le i \le m$, and the components of a row h_i by $h_i(j)$, $1 \le j \le n$. The components of any n-length vector u are denoted by u_i, $1 \le i \le n$.

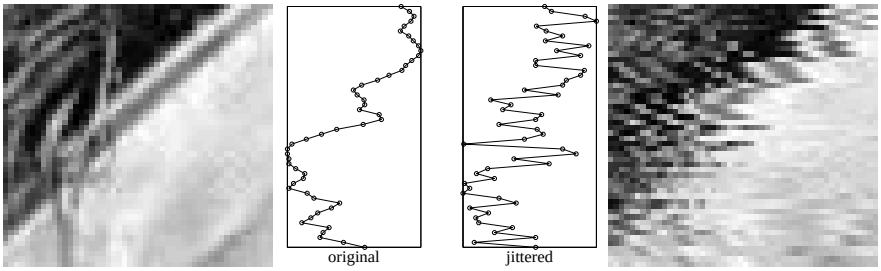
Given an original image $f \in \mathbb{R}^{r\times c}$, a jittered image g is produced according to:

$$1 \le i \le r,\ d_i \in \mathbb{Z},\ 1 \le j \le c,\quad g_i(j) = \begin{cases} f_i(j+d_i) & \text{if } 1 \le j+d_i \le c, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

In practice, the row shifts d_i are bounded, $|d_i| \le M$, for $M \le 6$ or more [6]. The restored image and row shifts are denoted by $\hat{f}$ and $\hat{d}$, respectively.

2.1 Choice of a Local Criterion on Consecutive Rows

We wish to estimate the shift $\hat{d}_i$ of each row i based on the previously restored rows. So we need a good model for the columns of natural images.

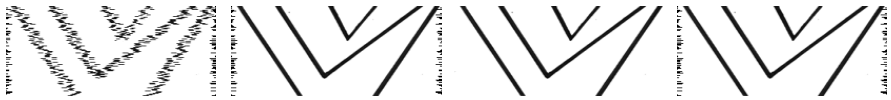


(a) Original (b) One column (c) Jittered

Fig. 1. 50×50 zoom of Lena. (b) Gray value of column 15 in (a) and in (c).

Remark 1. The gray-value of the columns of natural images can be seen as pieces of 2^{nd} *or* 3^{rd} *order polynomials—see Fig. 1(b) left or Fig. 3 in [9]. Such a claim is clearly false for a jittered column—see Fig. 1(b) right. This observation provides a sound basis to discriminate a natural image from its jittered versions.*

Suppose that $\hat{f}_1, \dots \hat{f}_{i-1}$ are already dejittered. By Remark 1, we will estimate the next $\hat{d}_i$ using a criterion that compares $\hat{f}_{i-1}$, $\hat{f}_{i-2}, \dots$ with all possible shifts of the ith data row, $g_i(j-d_i)$, $d_i \in \{-N, \dots, N\}$ for $N \ge M$.



Uniform jitter, $M=6$ $\arg\min \mathcal{J}$, $\alpha = 1$ Original (116×200) $\arg\min \mathcal{J}$, $\alpha = 0.5$

Fig. 2. Uniform jitter on $\{-M, \dots, M\}$. Restorations using (2)-(3) and (4).

Remark 2. Each row of g *has no more than* N *zero-valued pixels at both extremities because of the jitter, see e.g. Fig. 2. Involving them in our criterion can seriously distort its meaning. So for any row* i*, we will use only data samples* $g_i(j)$ *for* $j \in \{N+1, \dots, c-N\}$ *which certainly belong to the original image.*

Guided by Remarks 1 and 2, as well as by a series of preliminary experiments (see e.g. Fig. 3), our main focus is on

$$\hat{d}_i = \arg\min \big\{ \mathcal{J}(d_i) \ : \ d_i \in \{-N, .., N\} \big\}, \ N \geq M, \tag{2}$$

$$\mathcal{J}(d_i) = \sum_{j=N+1}^{c-N} \left| g_i(j - d_i) - 2\hat{f}_{i-1}(j) + \hat{f}_{i-2}(j) \right|^{\alpha}, \quad \alpha \in \{0.5,\, 1\}\,. \tag{3}$$

$\hat{d}_i$ is easily found by exhaustive search since it belongs to a small finite set. Then:

$$\forall j \in \{1, \cdots, c\}, \quad \hat{f}_i(j) = g_i(j - \hat{d}_i) \text{ if } 1 \leq j - \hat{d}_j \leq c \text{ and } \hat{f}_i(j) = 0 \text{ else }. \tag{4}$$

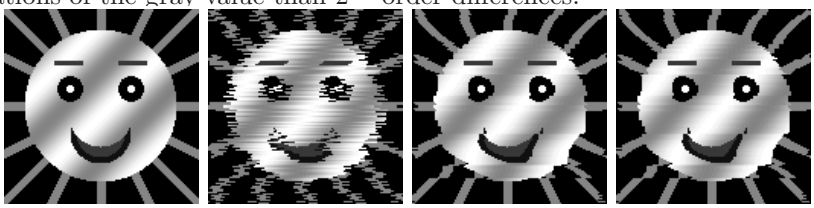
Criterion $\mathcal{J}$ for $\alpha \in (0, 1]$ is minimized by a $\hat{d}_i$ such that for a maximum number of components j we have $\hat{f}_i(j) \approx 2\hat{f}_{i-1}(j) - \hat{f}_{i-2}(j)$—i.e. $\hat{f}_i(j)$, $\hat{f}_{i-1}(j)$ and $\hat{f}_{i-2}(j)$ form a nearly linear segment—while breakpoints are preserved; for a mathematical flavor, see [10, 11]. Then each column of $\hat{f}$ has a nearly piecewise linearly varying gray value. More details are given in [9]. When $\alpha < 1$, (2)-(3) is a special form of a *Modes*-estimator [13].

Remark 3. Dejittering a single frame yields a *translated* estimate $\hat{p}$ of the row shifts, say $\hat{p} = \hat{d} + C$. Given the original d, the integer C is such that

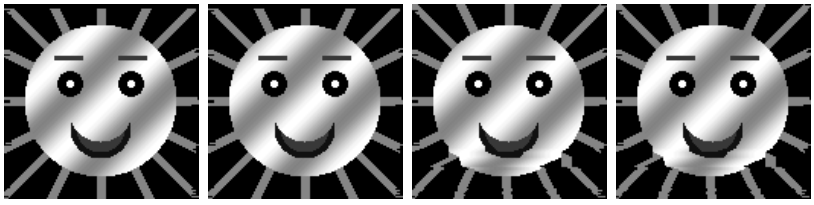
$$C = \arg\max_{n \in \mathbb{Z}} \ \#\big\{ i \in \{1, \cdots, r\} : \hat{p}_i - n = d_i \big\}, \tag{5}$$

where # means cardinality.

Alternative criteria. The jitter in Fig. 3(b) is considerable. Minimizing $J_1(d_i) = \sum_{j=N+1}^{c-N} \left| g_i(j - d_i) - \hat{f}_{i-1}(j) \right|^{\alpha}$ for $\alpha \in \{\frac{1}{2}, 1\}$ yield (c)-(d). Criteria J_1 work poorly—they tend to recover constant gray-value vertical pieces. Solving (2)-(3) yields the original image in (e)-(f). Criteria $J_3(d_i) = \sum_{j=N+1}^{c-N} \left| g_i(j-d_i) - 3\hat{f}_{i-1}(j) + 3\hat{f}_{i-2}(j) - \hat{f}_{i-3}(j) \right|^{\alpha}$ cannot discriminate well enough a natural image from its slightly shifted versions, see (g)-(h). 3rd order differences enable larger variations of the gray value than 2nd order differences.



(a) Original (b) Jittered (c) $J_1,\ \alpha = 1$ (d) $J_1,\ \alpha = 0.5$



(e) $\mathcal{J},\ \alpha = 1$ (f) $\mathcal{J},\ \alpha = 0.5$ (g) $J_3,\ \alpha = 1$ (h) $J_3,\ \alpha = 0.5$

Fig. 3. (b) Independent uniform jitter. Next: restorations for $N = M + 1$.

2.2 Error Measures for Dejittering

Remind that $\hat{f}$ is translated with respect to (w.r.t.) f and that the extremities of its rows are null because of the jitter. In order to apply standard error measures for the restored $\hat{f}$, we shrink $\hat{f}$ to $\hat{f}^s \in \mathbb{R}^{r\times c-2N}$ according to

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

so that $\hat{f}^s$ contains only *proper* image information. Then we select an $r\times(c-2N)$ inner submatrix f^s of the original f that matches $\hat{f}^s$ the best. Note that any error measure on $\hat{f}^s - f^s$ is sensitive to the choice of f^s. We select f^s using the ℓ_1 norm: $\|f^s - \hat{f}^s\|_1 = \min_{0\le k\le 2N} \sum_{i=1}^{r} \sum_{j=1}^{c-2N} \left|f_i(j+k) - \hat{f}^s(j)\right|$. Then we consider the *mean absolute error* $\text{MAE}(\hat{f}, f) = \|f^s - \hat{f}^s\|_1 / \big(r(c-2N)\big)$ and the *peak signal to noise ratio*, $\text{PSNR}(\hat{f}, f) = 10\log_{10}\left(\delta^2 r(c-2N)/\|f^s - \hat{f}^s\|_2^2\right)$, where $\|.\|_2$ is the ℓ_2-norm and δ is the dynamic range of $(\hat{f}^s, f^s)$.

The quality of dejittering can also be evaluated using $d - \hat{d}$. The error measure e_1 below gives the *average displacement of the pixels along any column:*

$$e_1(\hat{d}, d) \stackrel{\text{def}}{=} (1/r)\|d - \hat{d}\|_1 \,. \tag{6}$$

The following two measures are quite interesting:

$$e_\infty(\hat{d}, d) \stackrel{\text{def}}{=} \frac{100}{c}\|d - \hat{d}\|_\infty \% \ ; \tag{7}$$

$$e_0^\Delta(\hat{d}, d) \stackrel{\text{def}}{=} \frac{100}{r-1}\#\big\{(\hat{d}_i - d_i) - (\hat{d}_{i+1} - d_{i+1}) \neq 0,\ 1 \le i \le r-1\big\}\% \ . \tag{8}$$

e_∞ measures the maximum horizontal error w.r.t. the width c of the image while e_0^Δ measures the number of changes in $d - \hat{d}$ w.r.t. the height r of the image.

Remark 4. When both e_∞ and e_0^Δ are small (e.g. $e_\infty \le 0.4\%$ and $e_0^\Delta \le 0.8\%$), we are guaranteed that dejittering is nearly perfect, independently of any other error measure (see Figs. 6, 7, 10 and 12). 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, no conclusion can be done—*cf.* Fig. 9 and the relevant comments.

3 Algorithms for Gray-Value Natural Images

We construct an $r\times(c+2N)$-size matrix f^* for $N > M$. The middle of its first row f_1^* is g_1, so $\hat{p}_1 = N+1$. Then we restore the relative row shifts $\hat{p}_i \in \{1,\dots,2N+1\}$, $\forall i \in \{2,\cdots,r\}$ based on (2)-(3) and (4). The $\hat{f}$ is an inner sub-matrix of f^*.

Notations. $[a \vdots b \vdots c]$ means that a, b and c are concatenated horizontally; $a \leftarrow b$ means that we replace a by b. For any $n \in \mathbb{N}$, we denote by $\theta(n)$ the n-length zero-valued row-vector:

$$\theta(n) = \big[0 \vdots \cdots \vdots 0\big], \quad \#\theta(n) = n. \tag{9}$$

Algorithm 1 (Gray value images)

– Fix $N > M$, e.g., $N = M + 1$.
– Choose $\alpha = 1$ or $\alpha = 0.5$.

1. Define $f^* \in \mathbb{R}^{r\times(c+2N)}$ and set $f_1^* = \big[\theta(N) \,\vdots\, g_1 \,\vdots\, \theta(N)\big]$.
2. Split $g = \big[\ g^L \,\vdots\, \boldsymbol{\gamma} \,\vdots\, g^R\ \big]$ where $g^L \in \mathbb{R}^{r\times N}$, $\boldsymbol{\gamma} \in \mathbb{R}^{r\times(c-2N)}$ and $g^R \in \mathbb{R}^{r\times N}$.
3. Put $\hat{p}_0 = \hat{p}_1 = N + 1$ and $u = v = \big[\theta(N) \,\vdots\, \boldsymbol{\gamma}_1 \,\vdots\, \theta(N)\big]$.
4. For any $i = 2, \ldots, r$, do:
   (a) $\forall\, k = 1, \ldots, 2N{+}1$, do
   (i) Put $h^k = \big[\theta(k-1) \,\vdots\, \boldsymbol{\gamma}_i \,\vdots\, \theta(2N-k+1)\big]$;
   (ii) Find $m = \max\big\{k, \hat{p}_{i-1}, \hat{p}_{i-2}\big\}$ and $n = \min\big\{k, \hat{p}_{i-1}, \hat{p}_{i-2}\big\}+c-1$;
   (iii) $\mathcal{J}(k) = \dfrac{1}{n-m+1}\displaystyle\sum_{j=m}^{n}\Big|h_j^k - 2u_j + v_j\Big|^{\alpha}$;
   (b) Find $\hat{p}_i = \arg\min\{\mathcal{J}(k) : 1 \le k \le 2N+1\}$;
   (c) Replace $v \leftarrow u$ and $u \leftarrow h^{\hat{p}_i} = \big[\theta(\hat{p}_i - 1) \,\vdots\, \boldsymbol{\gamma}_i \,\vdots\, \theta(2N+1-\hat{p}_i)\big]$;
   (d) Set $f_i^* = \big[\theta(\hat{p}_i - 1) \,\vdots\, g_i \,\vdots\, \theta(2N-\hat{p}_i+1)\big]$.
5. Extract $\hat{f} \in \mathbb{R}^{r\times c}$ from $f^* \in \mathbb{R}^{r\times(c+2N)}$: cancel $2N$ columns at the left and right ends of f^* that have the largest number of zeros.

Explanations. u, v and h^k are c-length rows such that at step i, u and v correspond to the restored rows $i-1$ and $i-2$, respectively, while h^k in 4a(i) realizes all possible shifts for row i. In 4a(ii), m and n help to satisfy Remark 2. In 4b, $\hat{p}_i$ is the estimate for relative shift of row i.

Computation time. We used Matlab 7.2 on a PC with Pentium 4 CPU 2.8GHz and 1GB RAM, under 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 $\alpha = 1$ and in 1 second for $\alpha = 0.5$. (Note that our Matlab code is not fully optimized.)

Translation Recovery. In order to compute the the errors defined in § 2.2, we need the translation constant C given in (5). Note that $1 - N \le C \le 3N + 1$.

*Algorithm (*Translation Recovery)

1. Define $I = \{-N+1, \ldots, 3N+1\}$.
2. Compute the histogram $\mathcal{H}(n) = \#\big\{j \in I : \hat{p}(j) - d(j) = n\big\}, \forall n \in I$.
3. Obtain $C = \arg\max_{n\in I}\mathcal{H}(n)$. Then $\hat{d}_i = \hat{p}_i - C,\ 1 \le i \le r$.

Compound models. If the gray-values of the columns of an image are nearly constant on large pieces, we should involve in $\mathcal{J}$ a 1st-order differences term.

Algorithm 1(a)

In Algorithm 1, 4a(iii), use $\mathcal{J}$ below where β is a weight for 1st-order differences:
$\mathcal{J}(k) = \frac{1}{n-m+1}\sum_{j=m}^{n}\big(|h_j^k - 2u_j + v_j| + \beta|h_j^k - u_j|\big)^{\alpha},\ \beta \ge 0.$

Illustrations. In all experiments, Algorithm 1, is applied with $N = M + 1$.

The jitter in Fig. 4 is significant. We kept this first trial since our method found the original for $\alpha \in \{0.5, 1\}$. In Fig. 5 (Peppers), the dejittered image is hard to distinguish from the original. However, the error image $f^s - \hat{f}^s$ shows a slight displacement of several pixels. The dejittered image in Fig. 6 is nearly perfect since e_0^{Δ}=0.6% and e_∞=0.39%. We observe that $\hat{d} - d$ has a 1-pixel error at rows 83, 84 and 401. The first two are within the zooms in the same figure. The restored Boat in Fig. 7 is quasi-perfect since e_0^{Δ}= 0.25% and e_∞= 0.39%. The original Boat can be seen in Fig. 8 where the restorations are exact (all errors are null). For the results concerning [12] and [3], *cf.* section 7, p. 12.

Note that for our algorithms, $\hat{d} - d$ is piecewise constant, which is a good point. This cannot be claimed for the concurrent methods.

Uniform jitter, M=6 | Bayesian TV [12] MAE=11.7, PSNR=22 | Bake & Shake [3] MAE=7.4, PSNR=23 | Algorithm 1≡Original MAE=0, PSNR=∞

Fig. 4. Algorithm 1 for $\alpha = 1$ and $\alpha = 0.5$ yields the original image.

Uniform jitter,M=10 | Original (512×512) | Algorithm 1, α=0.5 | Error: $f^s - \hat{f}^s$

Fig. 5. Algorithm 1 with $\alpha = 0.5$ yields MAE= 1.35, PSNR=31.51 and $e_1 = 0.4$.

Uniform jitter,M=6 | Alg. 1, $\alpha = 0.5$ | Zoom dejittered | Zoom original

Fig. 6. (512×512). Algorithm 1: MAE= 4.16, PSNR=25.53, $e_0^{\Delta} = 0.6\%$ and $e_\infty = 0.39\%$.

Uniform jitter,M=10 | Bayesian TV [12] MAE=13.4, PSNR=20.8 | Bake & Shake [3] MAE=12.5, PSNR=20.3 | Alg. 1, $\alpha \in \{0.5, 1\}$ MAE=0.6, PSNR=42.9

Fig. 7. Boat (400×512). Algorithms 1 is nearly perfect: $e_0^{\Delta} = 0.25\%$ and $e_\infty = 0.39\%$.

$d = \lfloor 6 \sin\left(\frac{n}{20}\right) \rfloor$ | Alg.1,α=1 ≡Original | $d = \lfloor 6 \sin\left(\frac{n}{4}\right) \rfloor$ | Alg.1,α=1 ≡Original

Fig. 8. Boat (400×512). Here $\lfloor . \rfloor$ denotes approximation to the nearest integer.

Large-Scale Experiment. We tested all proposed algorithms using 1000 independent experiments where 4 images were degraded with 2 different types of random jitter and restorations were done for $\alpha = 1$ and $\alpha = 0.5$. The main conclusion is that $\alpha = 0.5$ is better for images with texture or curvatures (Lena, Barbara, Peppers); $\alpha = 1$ is better for images with many straight lines (Boat). In all cases $\alpha = 1$ yields good results, usually $\alpha = 0.5$ works better. The details are reported in [9]. Globally, the obtained mean results are very encouraging.

4 Algorithms Color Natural Images

We extend Algorithm 1 to RGB color images where all channels incur the same jitter. RGB images are represented by vector-valued matrices f where each pixel $f_i(j)$ has 3 components, $f_i(j;\kappa),\ 1 \le \kappa \le 3$. The jittering model now reads:

$$g_i(j;\kappa) = \begin{cases} f_i(j+d_i;\kappa), & \text{if } 1 \le j+d_i \le c, \\ 0, & \text{otherwise,} \end{cases} \quad |d_i| \le M, \quad \begin{cases} 1 \le i \le r, \\ 1 \le j \le c, \end{cases} \quad 1 \le \kappa \le 3.$$

The main algorithm is based on (2)-(3) and (4), yet again. Since the jitter is the same for all color channels, we obtain from g a *gray-value image* $\boldsymbol{\gamma}$ and estimate the relative row shifts $\hat{p}_i$ using $\boldsymbol{\gamma}$ as in Algorithm 1. The dejittered color image $\hat{f}$ is obtained by inserting $\hat{p}$ into g.

Similarly to (9), for any positive integer n we denote by $\theta(n \times 3)$ the n-length vector-valued row whose components are $(0, 0, 0)$ for all $i = 1, \cdots, n$.

Algorithm 2 (Color images)

– Fix $N > M$, e.g., $N = M + 1$.
– Choose $\alpha = 1$ or $\alpha = 0.5$.

---

1. Define $f^* \in \mathbb{R}^{r\times(c+2N)\times 3}$ and set $f_1^* = \big[\theta(N\times 3) \,\vdots\, g_1 \,\vdots\, \theta(N\times 3)\big]$.
2. Split $g = \big[\, g^L \,\vdots\, \overline{g} \,\vdots\, g^R \,\big]$, 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. Calculate $\boldsymbol{\gamma}_1(j) = |\overline{g}_1(j;1)| + |\overline{g}_1(j;2)| + |\overline{g}_1(j;3)|$ for $1 \le j \le c - 2N$.
4. Put $\hat{p}_0 = \hat{p}_1 = N + 1$ and $u = v = \big[\theta(N),\ \boldsymbol{\gamma}_1,\ \theta(N)\big]$.
5. For any $i = 2, \dots, r$ do:
   (a) $\forall\ k = 1, \dots, 2N + 1$ do: $\begin{cases} \text{i. } \boldsymbol{\gamma}_i(j) = \big|\overline{g}_i(j;1)\big| + \big|\overline{g}_i(j;2)\big| + \big|\overline{g}_i(j;3)\big|; \\ \text{ii. do step 4a as in Algorithm 1;} \end{cases}$
   (b) Do steps 4b and 4c as in Algorithm 1 ;
   (c) Set $f_i^* = \big[\theta\big((\hat{p}_i - 1)\times 3\big) \,\vdots\, g_i \,\vdots\, \theta\big((2N - \hat{p}_i + 1)\times 3\big)\big]$.
6. Find $\hat{f} \in \mathbb{R}^{r\times c}$ as in step 5, Algorithm 1.

Computation time. In the conditions of Remark 3, p. 5, for a 512×512 RGB image and $N=7$ we got the solution in 1 s. for $\alpha=1$ and in 1.4 s. for $\alpha=0.5$.

Algorithm 2(a) (Compound models)

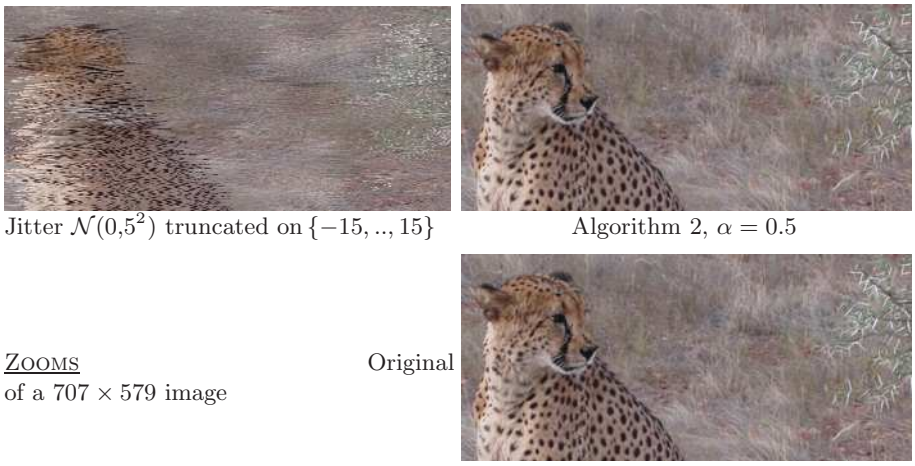
In step 5a, Algorithm 2, replace $\mathcal{J}$ as done in Algorithm 1(a).

Illustrations. In all examples, Algorithms 2 and 2(a) are used with $N = M + 1$.

In Fig. 9, the main part of the error in $\hat{d}$ corresponds to the sky and to the ground which are quite homogeneous, so the error is invisible to the naked eye. Part of it reaches the the boat, so we display a zoom of the latter. Fig. 10 shows a zoom of a 707×579 image. The dejittering of the full image is nearly perfect since $e_\infty = 0.17\%$ and $e_0^\Delta = 0.28\%$. The jitter in Fig. 11 is a centered Gaussian with standard deviation $\sigma = 6$, truncated and quantized on $\{-12, \dots, 12\}$. Algorithm 2(a) for $\alpha = 0.5$ and $\beta \in \{2, 3\}$ gives better visual results than Algorithm 2. Fig. 12 shows a nearly perfect restoration since $e_\infty = 0.18\%$ and $e_0^\Delta = 0.37\%$.

Uniform jitter, $M=8$ Man (478×532) Algorithm 2 $\alpha=1$ Zooms.

Fig. 9. Dejittering yields MAE$= 1.45$, PSNR$=33.82$, $e_1 = 0.76$ and $e_\infty = 3.76\%$.

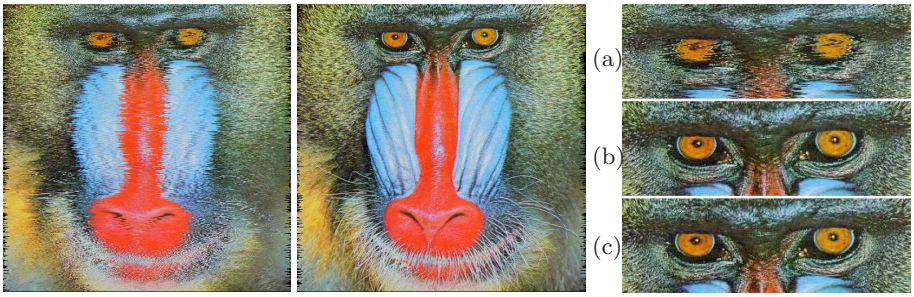


Jitter $\mathcal{N}(0,5^2)$ truncated on $\{-15,..,15\}$ Algorithm 2, $\alpha = 0.5$

ZOOMS
of a 707×579 image

Original

Fig. 10. The restoration of the whole image quasi-perfect: e_∞=0.17% and e_0^Δ=0.28%.



Gaussian jitter, $M = 12$ Algorithms 2(a) Zooms

Fig. 11. Zooms: (a) Jittered, (b) Original, (c) Dejittered

d Uniform jitter M=8 Original (542×410) Algorithm 2, α=0.5

Fig. 12. The result is quasi-perfect, MAE=0.14, PSNR=45.15, e_0^Δ=0.37% and e_∞=0.18%.

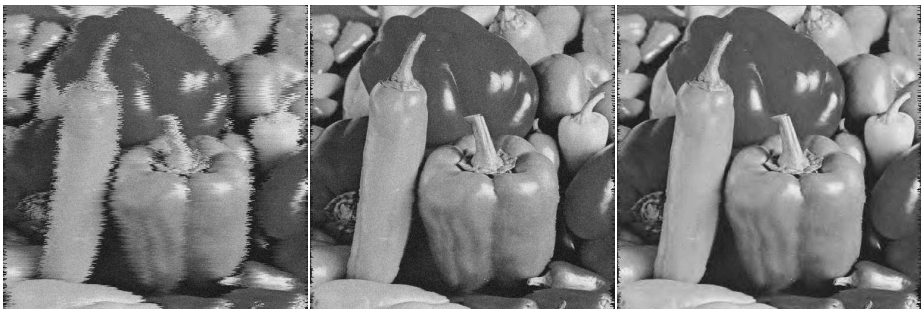
5 Restoration of Noisy Jittered Images

Our approach is to first dejitter the raw data using the ideas of Algorithms 1-2 and then to denoise the dejittered image. In the second stage, we use fast shrinkage estimators, see e.g. [8]. Better methods would improve the final result.

5.1 Moderate Noise

For a noise with 15-20 dB SNR or more, Algorithms 1, 2 perform well.

Experiment. Fig. 13(a) is corrupted with white zero-mean normal noise, 15 dB SNR, and independent uniform jitter on $\{-6, \ldots, 6\}$. Taking into account that the columns of the image are nearly constant on large segments, dejittering in (b) is done using Algorithm 1(a) for $\beta = 3$. Denoising of (b) is done in (c) by hard thresholding the 2D Daubechies wavelet transform with 4 vanishing moments for $T = 30$. The restoration is fast and the result is clean, compared to Fig. 5.



(a) 15 dB SNR+Jitter (b) Dejittered, Alg.1 (c) Denoised

Fig. 13. Pepers (512×512). For the restored image in (c), PSNR=29.34.

5.2 Strong Noise

When the noise is strong, we propose a sightly different scheme having a comparable computational cost. The idea is to partially denoise each row of the image using hard thresholding and to replace the function $|.|^\alpha$ in step 4a(iii) of Algorithm 1 by a better adapted edge-preserving function ψ. Let $W : \mathbb{R}^{1\times n} \to \mathbb{R}^{1\times n}$ denote a 1D wavelet transform and W^* its inverse. Given a threshold $T > 0$, let us introduce the hard thresholding operator $H_T : \mathbb{R}^{1\times n} \to \mathbb{R}^{1\times n}$ by

$$H_T(w)(j) = \begin{cases} 0 & \text{if } \big|w(j)\big| \le T \\ w(j) & \text{otherwise} \end{cases} \quad 1 \le j \le n, \quad \forall w \in \mathbb{R}^{1\times n}. \tag{10}$$

Knowing that the asymptotically optimal T, *cf.* [2], oversmooths rows, we use an under-optimal T. In order to simplify the presentation, we give the algorithm for gray-value images. The extension to color images is straightforward, *cf.* [9].

Algorithm 3 (Quite noisy images)

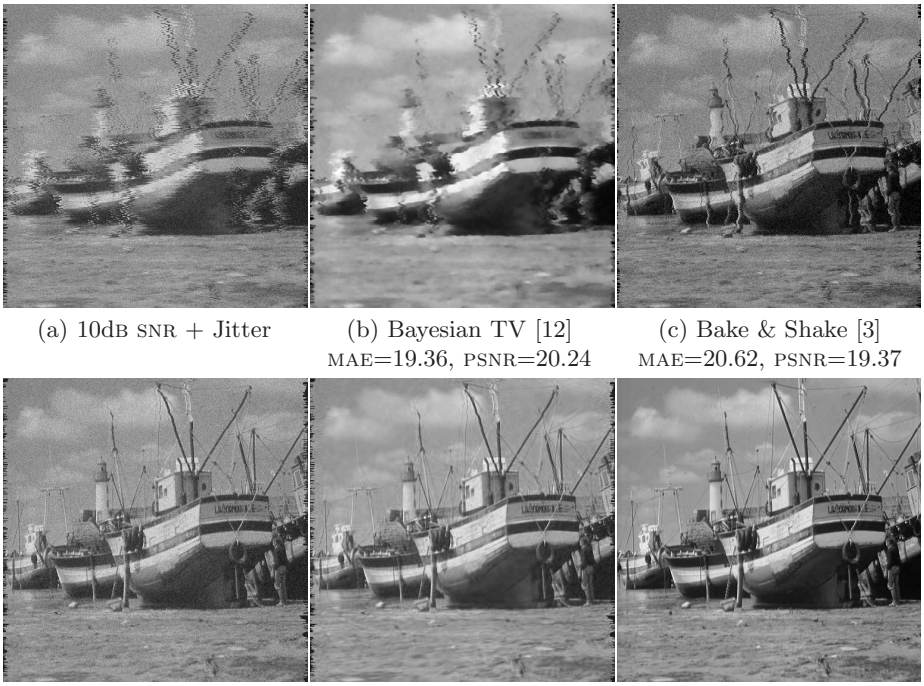
- Fix $N > M$, e.g., $N = M + 1$.
- Choose a 1D wavelet transform W (e.g. Daubechies).
- Fix an *under-optimal* threshold T.
- Choose $\psi : \mathbb{R}\times\mathbb{R} \to \mathbb{R}_+$, e.g. $\psi(s,t) = (|s| + \beta|t|)^\alpha$, and fix $\alpha > 0$ and $\beta \ge 0$.

1. Define $f^* \in \mathbb{R}^{r\times(c+2N)}$ and set $f_1^* = \big[\theta(N) \,\vdots\, g_1 \,\vdots\, \theta(N)\big]$.
2. Split $g = \big[\, g^L \,\vdots\, \overline{g} \,\vdots\, g^R \,\big]$ 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. Compute $\boldsymbol{\gamma_1} = W^*\big(H_T(W\overline{g}_1)\big)$.

4. Do steps 3 to 5 of Algorithm 1 with the following changes:
   (a) in step 4a(i), insert $\boldsymbol{\gamma_i} = W^*\big(H_T(W\overline{g}_i)\big)$;
   (b) in step 4a(iii), use $\mathcal{J}(k) = \frac{1}{n-m+1}\sum_{j=m}^{n} \psi\big(|h_j^k - 2u_j + v_j| + \beta|h_j^k - u_j|\big)$.

Comments. Hard-thresholding in steps 3 and 4a is better than other shrinkages since it keeps unchanged the important coefficients. The 1D row under-denoising (step 4a) helps to approach the model of Remark 1. Denoising of a dejittered image can be done by various methods. The proposed algorithm works pretty well even though it involves several parameters that need to be fixed cleverly.

Experiment. Boat in Fig. 14 is corrupted with 10 dB SNR white zero-mean normal noise and independent jitter, uniform on $\{-8, .., 8\}$. Its restoration using Bayesian TV [12] is unsatisfactory. The result using Bake and Shake [3] is better. For these results, *cf.* section 7, p. 12. The features of the image suggest to run our Algorithm 3 for $\beta = 0$ and $\psi(t) = |t|^\alpha$ for $\alpha = 0.5$ in step 4b. In steps 3 and 4a we use hard-thresholding of the Daubechies wavelet coefficients with 2 vanishing moments for $T = 30$. The dejittered image in (d) is denoised in (e) by hard thresholding of its curvelet transform using the *enhanced-denoising* program in the *CurveLab 2.1.2 toolbox* relevant to [1].



(a) 10dB SNR + Jitter

(b) Bayesian TV [12] MAE=19.36, PSNR=20.24

(c) Bake & Shake [3] MAE=20.62, PSNR=19.37

(d) Algorithm 3, dejittering

(e) Our full 2-stage method MAE=7, PSNR=28.31

Original

Fig. 14. Boat (512×512). Restoration of (a) using different methods.

6 Conclusions

The obtained results have a remarkable quality while the algorithms are nearly real-time. More details and examples are presented in [9]. The crux of our approach are (a) to minimize a nonsmooth and possibly nonconvex local criterion on the magnitude of the second-order differences between consecutive rows; (b) to exclude from $\mathcal{J}$ all pixels due to the jitter. In presence of strong noise, a critical step is to (under)-denoise the rows successively so that the prior mentioned in Remark 1 remains relevant, and to adapt the criterion $\mathcal{J}$ if necessary. The natural evolution of this work is to involve it in the restoration of video sequences and to take advantage of the correlation between consecutive frames.

7 Acknowledgements

This work has been supported by grant FREEDOM, ANR07-JCJC-0048-01.

The author thanks Louis LABORELLI, (*Institut National de l'Audiovisuel, France*), for his discussion on practical questions relevant to jittering. The author is thankful to Dr. Suhg-Ha KANG, Georgia Institute of Technology, Atlanta, who realized all experiments with the methods [3] and [12], as well as to Dr. Jackie SHEN (Barclays Capital, Wall Street) who provided his Matlab codes for [12].

References

1. E. J. Candès, L. Demanet, D. L. Donoho, and L. Ying. Fast discrete curvelet transforms. *SIAM J. on Multiscale Modeling and Simulation*, 5(3):861–899, 2006.
2. D. L. Donoho and I. M. Johnstone. Ideal Spatial Adaptation by Wavelet Shrinkage. *Biometrika*, 81(3): 425–455, 1994.
3. S.-H. Kang and J. Shen. Video dejittering by bake and shake. *Image and vision computing*, 24(2):143–152, 2006.
4. S.-H. Kang and J. Shen. *Image Dejittering Based on Slicing Moments*, p. 35–55. Springer Series on Mathematics and Visualization, 2007.
5. A. Kokaram, P. M. B. Roosmalen, P. Rayner, and J. Biemond. Line registration of jittered video. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, p. 2553–2556, 1997.
6. A. Kokaram. *Motion picture restoration*. Springer-Verlag, 1998.
7. L. Laborelli. Removal of video line jitter using a dynamic programming approach. In *Proc. of the IEEE ICASSP*, p. 331–334, 2003.
8. S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, London, 1999.
9. M. Nikolova. One-iteration dejittering of digital video images. Report CMLA n. 2008-20, http://www.cmla.ens-cachan.fr/fileadmin/Membres/nikolova/RT-DJ.pdf
10. M. Nikolova. Local strong homogeneity of a regularized estimator. *SIAM J. on Appl. Mathematics*, 61(2):633–658, 2000.
11. M. Nikolova. Analysis of the recovery of edges in images and signals by minimizing nonconvex regularized least-squares. *SIAM J. on Multiscale Modeling and Simulation*, 4(3):960–991, 2005.
12. J. Shen. Bayesian video dejittering by bv image model. *SIAM J. on Appl. Mathematics*, 64(5):1691–1708, 2004.
13. M. Welk, J. Weickert, F. Becker, C. Schnörr, C. Feddern, and B. Burgeth. Median and related local filters for tensor-valued images. *Signal Processing (special issue* Tensor Signal Processing*)*, 7:291–308, 2007.