Suppressing impulse noise via cloud filtering

Olivier Strauss[1], Frederic Comby[1], and Sebastien Destercke[2]

[1] LIRMM-UM2, 161, rue Ada, 34095, Montpellier, France
{olivier.strauss,frederic.comby}@lirmm.fr
[2] Université de Technologie de Compiègne, CNRS, Heudiasyc,
sebastien.destercke@hds.utc.fr

Abstract. In this paper, we propose a new approach for detecting and removing impulse noise. The method uses the new framework of cloud filtering for detecting noise locations. This new framework use efficient mathematical tools to filter with sets of filters rather than with a single one. Once a (very) noisy pixel is detected, its illumination is estimated by an extension of the median filtering applied on the neighborhood defined by the cloud. Experiments on various images demonstrate the capacity of the algorithm to identify noisy pixels (especially at low noise rate) while well preserving image edges.

Keywords: Filtering · Multi-dimensional· Imprecise · Impulse noise · Generalised p-boxes

1 Introduction

Removing impulse noise is a classical issue in image processing. This kind of noise occurs in image acquisition when a pixel of the camera is defective, when an error occurs during the analog to digital conversion or during image transmission through a noisy channel. It only corrupts isolated pixels by altering their values. Let $I_{i,j}$ be the illumination value at location (i,j). This image is said to be corrupted by an impulse with probability p if $I_{i,j} = s_{i,j}$ with probability p and $I_{i,j} = e_{i,j}$ with probability $(1-p)$, $s_{i,j}$ being the noise free illumination at location (i,j) and $e_{i,j}$ being a random value. p is called the level of contamination. If $e_{i,j}$ can only take extreme values (usually $\{0,255\}$), the impulse noise is called *salt and pepper noise*.

Usual linear filtering solutions tend to spread the corrupted values all over the image and are inefficient to remove impulse noise. Hence, many non-linear solutions have been proposed to remove impulse noise. A large number of these solutions are based on median filtering and its extensions: standard median filter (SMF) [1]; centered weighted median filter (CWMF) [2]; tri-state median filter (TSMF) [3] that combines median filter and center-weighted median filter; adaptive filters such as adaptive center-weighted median filter (ACWMF) or rank-order adaptive median filter (RAMF) [4]. Median based methods provide very good results at low noise rate. A strong limitation of median approaches is that even uncorrupted pixels are processed, leading to smooth edged images. The so-called switching strategy tries to overcome this limitation. It consists in detecting and modifying only corrupted pixels. For example, in [5], an Anderson-Darling test is used for detecting corrupted pixels whose noise free value is estimated via a neural network. [6] proposes a fuzzy rule-based detection with a classical median-based estimation of the noise-free value. A more parameter-free approach is proposed in [7]: corrupted pixels are detected using a ROAD[3] statistics while their noise free values are estimated by using an adaptive median filter. [8] proposes a switching strategy called tri-linear filtering combining spatial, radiometric and ROAD statistics. Switching strategies are more efficient for higher noise rates. However, as a major drawback, the efficiency of most of those methods is lowered by the difficulty of setting their different parameters (training of neural networks [5] and of fuzzy rules base [6] or thresholds in [7] and [8]). For all methods, edges are not well preserved.

In this paper we propose a switching strategy based on cloudy filtering for the detection of corrupted pixels and on weighted median filter for the restoration of corrupted values. This approach is easy to set and needs very few parameters. Section 2 introduces the tools used to detect and filter impulse noise, while Section 3 describes the method itself. Finally, Section 4 provides some experimentation that demonstrates the good performances of the method and compares it to other approaches.

2 Cloud filtering

This section presents the tools used in the noise removal method. We first introduce the filtering technique based on non-additive (cloudy) kernels used to detect noisy pixels and then present the weighted median technique used to remove noise.

2.1 Non additive kernels

In digital signal processing, kernels are instrumental for defining a weighted neighborhood around any sampled location. A (discrete) kernel is a function κ_i ($i \in \mathbb{Z}$) usually bounded unimodal centered and symmetric, which translates into the formal properties

$$\forall i \in \mathbb{N}, \kappa_i \geq \kappa_{i+1},$$

$$\kappa_i = \kappa_{-i}$$

and

$$\exists p \text{ such that for any } i > p \implies \kappa_i = 0.$$

Usually, discrete kernels are obtained by discretizing a continuous kernel. In traditional signal processing, kernel-based filtering is achieved by convolution, which can be seen as an expectation operation based on a Lebesgue integral.

In previous papers, we have introduced a new way of defining a weighted neighborhood by mean of non-additive (discrete) kernels. Roughly speaking, a non-additive kernel offers a convenient way to define a convex set of summative kernels, i.e. kernels that are positive functions with an integral equal to one, which in the discrete case translates into the additional properties

$$\kappa_i \geq 0 \text{ for any } i \text{ and } \sum_{i=1}^{\infty} \kappa_i = 1.$$

[3] Rank-Ordered Absolute Differences

In practice, a non-additive kernel uses the fact that summative kernels are formally equivalent to probability distributions, hence that working with a set $\mathscr{M}$ of summative kernels is equivalent to work with sets of probabilities. As filtering with a summative kernel κ amounts to compute expectations with respect to a kernel, filtering with a set $\mathscr{M}$ of kernels then amounts to compute lower and upper expectations according to $\mathscr{M}$.

Using this formal equivalence, we can denote by $P(A) = \sum_{i \in A} \kappa_i$ the additive measure (i.e., for any A, B with $A \cap B \neq \emptyset$, $P(A \cup B) = P(A) + P(B)$) induced by κ. Assuming that we have a set $I_0, \dots, I_p$ of p values and a function κ defined over $\{0, \dots, p\}$, then the "expected" value of I with respect to p can be written

$$\mathbb{E}(I) = \sum_{i=0}^{p} \left(I_{(i)} - I_{(i-1)}\right) P(A_{(i)}) \tag{1}$$

with $0 := I_{(-1)} \leq I_{(0)} \leq \dots \leq I_{(p)}$ and $A_{(i)} = \{i, \dots, p\}$. Similarly, given a set $\mathscr{M}$, we can denote by

$$\underline{P}(A) = \inf_{\kappa \in \mathscr{M}} P_\kappa(A) \text{ and } \overline{P}(A) = \sup_{\kappa \in \mathscr{M}} P_\kappa(A)$$

the upper and lower probabilities induced by $\mathscr{M}$. In the same way, we can defined lower/upper expectations of a function I as taking the infinimum/supermum of Equation (1) over $\mathscr{M}$.

When $\mathscr{M}$ has some particular properties such as being induced by 2-monotone lower probabilities[4], that is when $\underline{P}_{\mathscr{M}}$ satisfies for any pair of events A, B the inequality

$$\underline{P}(A \cup B) + \underline{P}(A \cap B) \geq \underline{P}(A) + \underline{P}(B),$$

computing lower/upper expectations, i.e., filtering a signal with a non additive kernel leads to replacing the Lebesgue integral by a Choquet integral [10]. One then simply has to replace $P(A_{(i)})$ by the lower probability $\underline{P}(A_{(i)})$ (resp. upper probability $\overline{P}(A_{(i)})$) in Equation (1)) to obtain the lower expectation (resp. upper expectation). More formally, this amounts to compute

$$\underline{\mathbb{E}}(I) = \sum_{i=0}^{p} \left(I_{(i)} - I_{(i-1)}\right) \underline{P}(A_{(i)}), \tag{2}$$

$$\overline{\mathbb{E}}(I) = \sum_{i=0}^{p} \left(I_{(i)} - I_{(i-1)}\right) \overline{P}(A_{(i)}). \tag{3}$$

The output of such a filtering is an interval-valued image, each interval containing all the values that would have been obtained by filtering the input signal with each summative kernel belonging to the set represented by the non-additive kernel. Among the different non-additive kernels, cloudy kernels (the ones we are interested in for this paper) aim a representing a family of summative kernels whose bandwidth is both lower and upper bounded. It is composed of two comonotonic maxitive kernels (see [11] for a short introduction on maxitive kernels), and are instrumental when interpreting interval-valued fuzzy sets as uncertainty models [12]. The upper maxitive kernel π has a bandwidth

[4] We refer to [9] for details.

equal to Δ_{sup} and the lower maxitive kernel η has a bandwidth equal to Δ_{inf} (see Figure 2.1). As shown in [13], such a kernel defines a family of summative kernel denoted $\mathscr{M}(\eta,\pi)$ whose bandwidth is included in $[\Delta_{inf},\Delta_{sup}]$. The choice of the shape of the kernel depends on the application, but a sub-optimal choice of shape has usually less impact on the final result than a sub-optimal choice of bandwidth. In this paper we consider symmetric unimodal centered clouds, and use them in a multivariate setting. Like for additive kernels, discrete clouds are obtained by discretizing a continuous cloud. The obtained discrete cloud should verify the following properties:

$$\sup_i \pi_i = \pi_0 = 1;$$

$$\forall i,j \in \mathbb{N}, \pi_i = \pi_{-i}, \eta_i = \eta_{-i};$$

$$\pi_i < \pi_j \implies \eta_i \leq \eta_j \text{ (comonotonicity)}\ .$$

Discrete clouds must also satisfy an additional condition to induce a non-empty set $\mathscr{M}$ [14] :

$$\forall i \in \mathbb{N}, \eta_i \leq \pi_{i+1}.$$

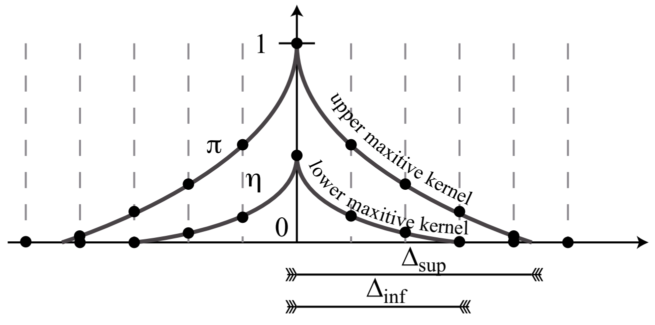


Fig. 1. Discretized cloudy kernel

2.2 Filtering an image with a cloudy kernel

Most $2D$ kernels $\phi_{i,j}$ $(i,j \in \mathbb{Z})$ used in image processing are isotropic and separable, i.e. they are the product of two unidimensional kernels defined by the same generic function: $\phi_{i,j} = \kappa_i \kappa_j$. In this paper we mainly consider symmetric unimodal centered kernels which are extensively used in images processing. Within this context, extending cloudy kernels to two dimensions is relatively straightforward. We propose such an extension based on the construction proposed in [15] with an efficient algorithm to perform the filtering. Let I be the $N \times M$ input image, with $I_{i,j}$ being the intensity value of the pixel located at the i^{th} row and the j^{th} column. Filtering an image I with a cloudy kernel results in an interval-valued image $[\underline{I},\overline{I}]$, i.e., interval valued intensities $[\underline{I}_{i,j},\overline{I}_{i,j}]$.

The $2D$ cloudy kernel we propose is built from a $1D$ cloudy kernel $[\eta,\pi]$. It defines the set of all the $2D$ summative kernels $\phi_{i,j} = \kappa_i \kappa'_j$, with $\kappa,\kappa' \in \mathscr{M}(\eta,\pi)$. Let p be the smallest integer such that $\forall i \in \mathbb{N}$, $i > p \implies \pi_i = 0$. Then, the $2D$ cloudy kernel based on $[\eta,\pi]$ is composed of $(p+1)$ nested sets of pixels $(R_0,\ldots,R_p)$ that are at same city-block distance from the central pixel, and corresponds to equivalence regions (see Figure 3). Within each set, the pixels are considered to have an equivalent role in the aggregation process. The central pixel form the set R_0. It is the pixel whose interval valued intensity has to be estimated. Each set R_k $(k = 0,...,p)$ has a cloudy weight of $[\eta_{R_k},\pi_{R_k}] = [(\eta_k)^2,(\pi_k)^2]$ and is associated to an interval-valued intensity $[\underline{I}_{R_k},\overline{I}_{R_k}]$ defined by

$$\underline{I}_{R_k} = \inf_{(u,v)\in R_k} I_{u,v} \text{ and } \overline{I}_{R_k} = \sup_{(u,v)\in R_k} I_{u,v}.$$

As a more numerical and formal example, we can consider a matrix M of points around a central pixel x_0,y_0, and denote by $R_k = \{x_i,y_j \,|\, \max(|i|,|j|) = k\}$ the rectangle of points whose mannathan distance is k. $I_{i,j}$ will then be the intensity of point x_i,y_j of the matrix. Figure 2 describes such a matrix (up to $k = 2$) for which $\underline{I}_{R_1} = 49$ and $\overline{I}_{R_1} = 218$.

$$M = \begin{pmatrix} 153 & 108 & 26 & 33 & 73 \\ 178 & 124 & 49 & 69 & 92 \\ 220 & 174 & 80 & 130 & 147 \\ 188 & 218 & 90 & 120 & 76 \\ 130 & 142 & 108 & 95 & 60 \end{pmatrix}$$

Fig. 2. Intensity matrix and region R_1 (dashed rectangle)

In practice, these values $[(\eta_k)^2,(\pi_k)^2]$ can be associated to the membership values of an interval-valued fuzzy set interpreted as a probability set, regions R_k of the image correspond to the focal elements of the induced credal set (as this latter is induced by a belief function, see [14] for more details), and the values $\underline{I}_{R_k}$ and $\overline{I}_{R_k}$ are simply the infinimum and supremum values on these focal sets, that are needed to compute lower and upper expectations through the Choquet integral.

Algorithm 1 describes the computation of $[\underline{I}_{i,j},\overline{I}_{i,j}]$ at each pixel (i,j) by a Choquet integral. Within this algorithm, we define two fictive weights: $[\eta_{R_{-1}},\pi_{R_{-1}}] = [1,1]$ and $[\eta_{R_{p+1}},\pi_{R_{p+1}}] = [0,0]$. The multivariate $[\eta,\pi]$ on is built on Lines 2-3 (Note that $\pi(R_k) = \pi^2(x_k)$ if marginal clouds are identical). Lines 4-5 just store minimal and maximal values in rectangles that will be used to compute filtered bounds. Lines 10 to 15 simply corresponds to the application of Equation (2) to $\underline{I}_k$, while Lines 16 to 22 simply corresponds to the application of Equation (3) to $\overline{I}_k$. Note that the orderings $()$ used for the two operations may be different.

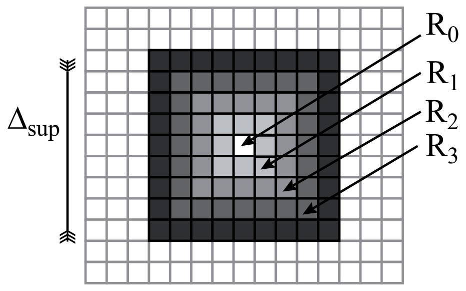


Fig. 3. Nested sets of a 2D cloudy kernel with $p = 4$

Algorithm 1 Computing $[\underline{I}_{i,j}, \overline{I}_{i,j}]$ with the $2D$ cloudy kernel defined by the $1D$ cloudy kernel $[\eta, \pi]$.

```
1: **for** $k = 0$ to $p$ **do**
2:     $\pi(R_k) \leftarrow \pi(x_k)\pi(y_k)$
3:     $\delta(R_k) \leftarrow \delta(x_k)\delta(y_k)$
4:     $\underline{I}_k \leftarrow \min\{I_{u,v} | (u,v) \in R_k\}$
5:     $\overline{I}_k \leftarrow \max\{I_{u,v} | (u,v) \in R_k\}$
6: **end for**
7: $\delta(R_{-1}), \pi(R_{-1}) \leftarrow 1$
8: $\delta(R_{n+1}), \pi(R_{n+1}) \leftarrow 0$
9: Order $\underline{I}_k$ such that $\underline{I}_{(0)} \leq \ldots \leq \underline{I}_{(p)}$
10: $\underline{I}_{i,j} \leftarrow 0, A_{(0)} \leftarrow \cup_{k=0}^{p} R_k$
11: **for** $k = 0$ to $p$ **do**
12:     $A_{(k+1)} \leftarrow A_{(k)} \setminus R_{(k)}$ $\{A_{(p+1)} = \emptyset\}$
13:     Compute $\underline{P}(A_{(k+1)})$ and $\underline{P}(A_{(k)})$ $\{\underline{P}(A_{(p+1)}) = 0\}$
14:     $\underline{I}_{i,j} \leftarrow \underline{I}_{i,j} + \underline{I}_{(k)}(\underline{P}(A_{(k)}) - \underline{P}(A_{(k+1)}))$
15: **end for**
16: Order $\overline{I}_k$ such that $\overline{I}_{(0)} \leq \ldots \leq \overline{I}_{(p)}$
17: $\overline{I}_{i,j} \leftarrow 0, A_{(0)} \leftarrow \cup_{k=0}^{p} R_k$
18: **for** $k = 0$ to $p$ **do**
19:     $A_{(k+1)} \leftarrow A_{(k)} \setminus R_{(k)}$ $\{A_{(p+1)} = \emptyset\}$
20:     Compute $\overline{P}(A_{(k+1)})$ and $\overline{P}(A_{(k)})$ $\{\overline{P}(A_{(p+1)}) = 0\}$
21:     $\overline{I}_{i,j} \leftarrow \overline{I}_{i,j} + \overline{I}_{(k)}(\overline{P}(A_{(k)}) - \overline{P}(A_{(k+1)}))$
22: **end for**
```

Algorithm 2 Computing $\underline{P}(A)$.

```
1: $\underline{P}(A) \leftarrow 0$
2: **while** $A \neq \emptyset$ **do**
3:     $\underline{k} \leftarrow \min\{k | R_k \in A\}$, $i \leftarrow \underline{k}$ , $A \leftarrow A \setminus R_{\underline{k}}$
4:     **while** $R_{\underline{k}+1} \in A$ **do**
5:         $\underline{k} \leftarrow \underline{k} + 1, A \leftarrow A \setminus R_{\underline{k}}$
6:     **end while**
7:     $j \leftarrow \underline{k}$
8:     $\underline{P}(A) \leftarrow \underline{P}(A) + \max\{0, \eta(R_{i-1}) - \pi(R_{j+1})\}$
9: **end while**
```

Algorithm 2 describes the computation of the lower probability $\underline{P}(A)$ of a subset $A \subseteq \{R_0,...,R_p\}$. The upper probability can be obtained by the dual relation $\overline{P}(A) = 1 - \underline{P}(A^c)$, where A^c is the complement of A in $\{R_0,...,R_p\}$. For example, if $A = \{R_1,R_2,R_4,R_5,R_6\}$ then $\underline{P}(A) = \max\{0,\delta(R_0)-\pi(R_3)\} + \max\{0,\delta(R_3)-\pi(R_7)\}$ (note that the ordering used is the original one, not the re-ordering used in Lines 9 or 16 of Algorithm 1).

Example 1. Let us consider the case $n = 4$ (e.g., we take regions of 4 pixels around the central one). Fictitious unordered and ordered values $\underline{I}$ are summarised in the table below:

$\underline{I}_0$	$\underline{I}_1$	$\underline{I}_2$	$\underline{I}_3$	$\underline{I}_4$
80	31	73	47	63
$\underline{I}_{(4)}$	$\underline{I}_{(0)}$	$\underline{I}_{(3)}$	$\underline{I}_{(1)}$	$\underline{I}_{(2)}$

and we have

- $A_{(0)} = \cup_{i=0}^{4} R_i \to \underline{P}(A_{(0)}) = 1$
- $A_{(1)} = \{R_0,R_2,R_3,R_4\} \to \underline{P}(A_{(1)}) = \max\{0,\delta(R_{-1})-\pi(R_1)\} + \max\{0,\delta(R_1) - \pi(R_5)\}$
- $A_{(2)} = \{R_0,R_2,R_4\} \to \underline{P}(A_{(2)}) = \max\{0,\delta(R_{-1})-\pi(R_1)\} + \max\{0,\delta(R_1)-\pi(R_3)\} + \max\{0,\delta(R_3)-\pi(R_5)\}$
- $A_{(3)} = \{R_0,R_2\} \to \underline{P}(A_{(3)}) = \max\{0,\delta(R_{-1})-\pi(R_1)\} + \max\{0,\delta(R_1)-\pi(R_3)\}$
- $A_{(4)} = \{R_0\} \to \underline{P}(A_{(4)}) = \max\{0,\delta(R_{-1})-\pi(R_1)\}$

2.3 Weighted median

The final element we need to formalise our noise removal procedure is the one of median extended to the fuzzy case.

The weighted empirical median of a fuzzy subset of an image I is a very simple and straightforward generalization of the crisp empirical median. Let E be a fuzzy subset of the pixels defined by the membership function $\mu^E_{i,j}$. Computing the weighted median intensity value of the subset E consists in finding a value γ such that $\forall \varepsilon > 0$:

$$\sum_{i,j/I_{i,j}<(\gamma-\varepsilon)} \mu^E_{i,j} \leq \frac{|E|}{2} \text{ and } \sum_{i,j/I_{i,j}<(\gamma+\varepsilon)} \mu^E_{i,j} \geq \frac{|E|}{2},$$

with $|E| = \sum_{i,j} \mu^E_{i,j}$. It can be easily computed by sorting all the intensity values of the pixels (i,j) such that $\mu^E_{i,j} > 0$ and computing the cumulative function $F(\gamma) = \sum_{i,j/I_{i,j}<\gamma} \mu^E_{i,j}$ for every sorted intensity values.

3 Cloud kernel based impulse noise detection and removal

The method we propose is achieved in three steps:

- the first step consists in filtering the noisy image with a $2D$ cloudy kernel constructed from a triangular $1D$ cloudy kernel;

– the second step consists in detecting the corrupted pixels: a pixel is considered corrupted if its intensity value $I_{i,j}$ in the original image is either above the upper filtered value $\overline{I}_{i,j}$ or below the lower filtered value $\underline{I}_{i,j}$. This procedure is illustrated in Figure 4 on a 1D signal: the corrupted signal is plotted in black while the upper (resp. lower) filtered signal is plotted in blue (resp. red). Impulse noise is detected when the black curve gets out of the envelope created by the upper and lower filtered signals. Note that the filtering has a very interesting property: the upper and lower filtered values tends to get closer at the impulse noise location, making the detection easier;

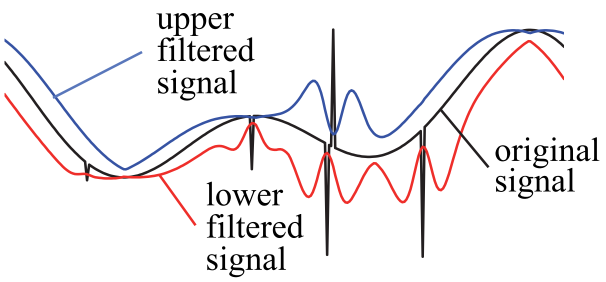


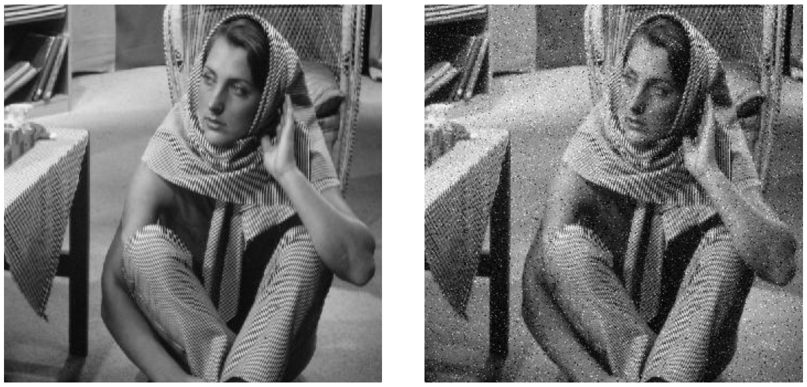
Fig. 4. 1D illustration of noise detection with cloud filter.

– the third step consists in replacing the noisy pixel by computing the weighted median by considering the fuzzy set E obtained by translating the membership function $\mu_{u,v}$ $(u,v \in \mathbb{Z})$ defined by $\mu_{u,v} = \min(\pi_u, \pi_v)$ to the location (i,j). Repeating these steps iteratively can improve the filtering.

4 Experimentations

In this section we compare our method with different well known algorithms: SMF, CWMF, trilateral filter, and a ROAD-based detector presented in [7]. Tests have been carried out on 5 classical images: ”Lena”, ”boat”, ”peppers”, ”cameraman” and ”Barbara”. Each image has been corrupted 20 times by an impulse noise with different level of contamination (p varies from 0 to 0.9). These tests have been extensively repeated to provide significant statistics. The settings of state-of-art methods were defined as the authors suggested in their publications. The settings for our method are: $\Delta_{inf} = 1.8$ and $\Delta_{sup} = 3$. The filtering process is iterated 3 times to improve its efficiency. We propose a quantitative and qualitative comparison of the methods.

We first start with the more qualitative comparison of the methods. Figure 5 shows two images of Lena, the first one without any noise, the second one with a noise where $p = 0.2$ (so 20% of noised pixels). A qualitative comparison is proposed in Figure 6. It can be seen on the two zoomed parts that our method better preserve high frequency details than other approaches, while efficiently removing noisy pixels.



(a) (b)

Fig. 5. Barbara's (a) Original image, (b) corrupted with 20% of pixels contaminated with random noise.

We propose to quantitatively compare the methods by using three criteria: the PSNR defined as

$$PSNR = 10 * log(255/\sum_{i=0}^{m}\sum_{j=0}^{n}(I(i,j) - \widehat{I}(i,j))^2,$$

the MAE defined as

$$MAE\sum_{i=0}^{m}\sum_{j=0}^{n}|I(i,j) - \widehat{I}(i,j)|$$

and SSIM (see [16]) measures. Table 1 presents the mean PSNR, MAE and SSIM for each algorithm over a large amount of experimentations on various images. According to those criteria, our method provides the overall best results for a contamination level under 50%. Over 70%, SMF algorithm works better.

	ROAD			Cloud			SMF			CWMF			Trilateral		
p	PSNR	MAE	SSIM	PSNR	MAE	SSIM	PSNR	MAE	SSIM	PSNR	MAE	SSIM	PSNR	MAE	SSIM
0.1	79.52	2.66	0.98	**82.61**	**1.39**	**0.99**	71.11	3.88	0.95	75.52	2.63	0.97	79.70	2.38	**0.99**
0.2	77.23	3.01	0.97	**78.51**	**1.82**	**0.98**	69.89	4.13	0.95	73.45	2.94	0.96	75.98	2.78	**0.98**
0.3	74.70	3.44	0.96	**74.94**	**2.33**	**0.97**	68.84	4.37	0.94	71.62	3.29	0.96	72.17	3.34	0.96
0.4	**71.35**	4.06	0.94	70.63	**3.08**	**0.95**	67.50	4.66	0.93	69.64	3.70	**0.95**	67.88	4.19	0.93
0.5	**67.79**	4.94	0.89	66.43	**4.18**	0.90	66.36	5.02	0.92	67.71	4.27	**0.94**	63.55	5.43	0.87
0.6	63.65	6.28	0.82	62.36	5.74	0.83	65.14	5.48	0.91	**65.48**	**5.08**	**0.91**	59.01	7.32	0.79
0.7	59.82	8.06	0.74	58.29	7.95	0.75	**63.67**	**6.16**	**0.88**	62.98	6.28	0.86	54.99	9.76	0.71
0.8	56.07	10.43	0.65	54.47	10.80	0.66	**61.29**	**7.34**	**0.82**	59.63	8.13	0.78	51.12	13.09	0.62
0.9	52.63	13.43	0.56	51.03	14.30	0.57	**58.76**	**9.15**	**0.73**	56.36	10.70	0.69	47.72	17.16	0.54

Table 1. Comparison of mean PSNR, MAE and SSIM measures for different contamination levels

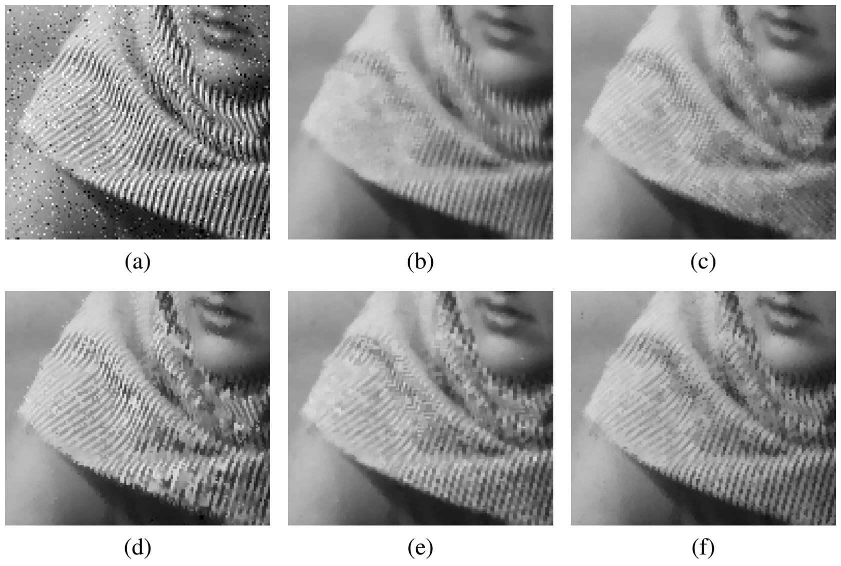


Fig. 6. (a) a zoomed part of Barbara's noisy picture. Zoomed part filtered with SMF (b), CWMF (c), Trilateral (d), ROAD-based (e), and Cloud-based filter (f).

5 Conclusion and discussion

In this paper, a new switching strategy based on cloud filtering is presented to remove impulse noise on images. This technique provides stable results with different kind of images without changing any settings. Experiments have shown that our algorithm can efficiently preserve image edges while removing impulse noise. However, since this method has been designed to detect isolated impulse noise, its performance decreases when the contamination rate is higher than 50%, as the amount of clustered noisy pixels inrease rapidly after reaching this threshold. Different improvements can be considered, as, for example, building a more specific $2D$ cloud by considering equivalence sets based on other distances (e.g., Euclidean) than the city block distance, that may possibly also make more sense for continuous images. A generalization to color images can also be considered.

References

1. I. Pitas and A. Venetsanopoulos, "Order statistics in digital image processing," *Proc. of the IEEE*, vol. 80, no. 12, pp. 1893 – 1921, December 1992.
2. L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: A tutorial," *IEEE Trans. on Circuits and Sys. II: A. and D. Sig. Proc.*, vol. 43, no. 3, pp. 157 – 192, March 1996.

3. T. Chen and K.-K. M. L.-H. Chen, "Tri-state median filter for image denoising," *IEEE Trans. on Image Proc.*, vol. 8, no. 12, pp. 1834 – 1838, December 1999.
4. H. Hwang and R. A. Haddad, "Adaptive median filters: New algorithms and results," *IEEE Trans. on Image Proc.*, vol. 4, no. 4, pp. 499 – 502, April 1995.
5. E. Besdok, "A new method for impulsive noise suppression from highly distrorted images using anfis," *Engineering Applications of Art. Int.*, vol. 17, pp. 519 – 527, 2004.
6. K. Arkawa, "Median filter based on fuzzy rules and its application to image restoration," *FSS*, vol. 77, pp. 3 – 13, 1996.
7. V. R. Vijaykumar and P. Jothibasu, "Decision based adaptive median filter to remove blotches, scratches, streaks, stripes and impulse noise in images," in *Proc. of IEEE ICIP 2010*, Hong Kong, September 26-29 2010, pp. 117 – 120.
8. R. Garnett, T. Huegerich, C. Chui, and W. He, "A universal noise removal algorithm with impulse detector," *IEEE Trans. on Image Proc.*, vol. 14, no. 11, pp. 1747 – 1754, November 2005.
9. S. Destercke and D. Dubois, "Special cases," *Introduction to Imprecise Probabilities*, pp. 79–92, 2014.
10. O. Strauss and K. Loquin, "On the granularity of summative kernels," *FSS*, vol. 159, no. 15, pp. 1952–1972, 2008.
11. ——, "Linear filtering and mathematical morphology on an image: a bridge," in *IEEE ICIP*, 2009, pp. 3965–3968.
12. D. Dubois and H. Prade, "Interval-valued fuzzy sets, possibility theory and imprecise probability." in *EUSFLAT Conf.* Citeseer, 2005, pp. 314–319.
13. S. Destercke and O. Strauss, "Filtering with clouds," *Soft Computing (in press)*, 2011.
14. S. Destercke, D. Dubois, and E. Chojnacki, "Unifying practical uncertainty representations: II. Clouds," *Int. J. of Approximate Reasoning*, vol. 49, no. 3, pp. 664–677, 2008.
15. M. C. M. Troffaes and S. Destercke, "Probability boxes on totally preordered spaces for multivariate modelling," *Int. J. of Approximate Reasoning*, vol. 52, pp. 767–791, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.ijar.2011.02.001
16. H. R. S. Z. Wang, A. C. Bovik and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. on Image Proc.*, vol. 13, no. 4, pp. 600 – 612, April 2004.