Chapter 69
Video Self-Adaptive Noise Reduction Algorithm

Jing Li

Abstract In digital video processing systems, studying an efficient noise reduction algorithm has become an important issue. As the noise is uncertain, so I hope to develop an algorithm that can adapt to different levels of noise. The temporal filter has the advantage of retaining the edge and details of a picture while the spatial filter has better abilities of reducing noise. In order to take into account the noise filtering and the protection of details and edges of video and pictures, I approached the joint temporal and spatial filtering algorithm. As different methods have various effects to noises of different levels, I adopted the self-adaptive system to achieve the best result.

Keywords Joint temporal and spatial filtering algorithm • Adaptive • Images and video • Recursive • Mean filter • Median filter

69.1 Introduction

The algorithm in this paper is targeted at YUV video streaming. *Y* component is brightness, so it is necessary to process the *Y* component and convert it with *U*, *V* components to RGB format to be displayed together [1]. Research has shown that the result from processing the three channels of RGB, respectively, will encounter distortion in color. Therefore, to avoid such a distortion, it is necessary to directly process the *Y*.

J. Li (✉)
Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310000, Zhejiang, China
e-mail: lijing@cssci.info

Z. Zhong (ed.), *Proceedings of the International Conference on Information Engineering and Applications (IEA) 2012*, Lecture Notes in Electrical Engineering 220,
DOI: 10.1007/978-1-4471-4844-9_69,

69.1.1 Pre-filtering

The gray value of most salt and pepper noise points is concentrated at 255 or 0, while most signal points are not concentrated at this area [2]. Hence, the author proposes this algorithm: (1) assume the pixel value is judged as the salt and pepper noise point when the gray range is between [0, 16] or [235, 255] and other areas are regarded as signal points; (2) the noise point (Z_{ij}) can be judged from the above and the peripheral pixels of the 3*3 window centered at Z_{ij} are marked as (Z_1, Z_2, …, Z_8), from which the points whose gray is not in the salt and pepper noise gray range can be selected and also marked as $\{X_1, X_2, \ldots, X_l\}$, $0 \leq 1 \leq 8$; Z_{ij} is the gray value of the noise point ij, and Z_1, Z_2, …, Z_8 and X_1, X_2, …, X_l are the gray value of the point around the pixel ij; (3) when there is $l = 0$, there are signal points around the pixel ij, and S_{ij} will not be processed and its value is still equal to itself; (4) when there is $l \neq 0$, there are some signal points around the pixel ij, and now the gray mid-value of the surrounding signal points can be used as the

Table 69.1 The noise reduction effect of pre-filtering on the forman sequence added with salt and pepper or Gaussian noise

The *n*th PSNR (dB)	1	2	3	4	5	6	7	8	9
Gaussian noise after pre-filtering	30.353	30.087	29.960	30.164	30.399	30.139	30.263	30.087	30.218
Salt and pepper noise after pre-filtering and processing	30.888	30.496	30.727	30.572	30.496	30.970	30.496	30.806	30.572

Fig. 69.1 The seventh frame image of forman: **a** image after salt and pepper noise experienced pre-filtering; **b** image with 24 dB salt and pepper noise; **c** image after Gaussian noise experienced pre-filtering; **d** image with 32 dB Gaussian noise; **e** original image

gray value of this point; order *Zij* = median $\{X_1, X_2, \ldots, X_l\}$, in which median $\{X_1, X_2, \ldots, X_l\}$ is the mid-value operator to solve the set $\{X_1, X_2, \ldots, X_l\}$, and this is called as the noise mid-value algorithm as $\{X_1, X_2, \ldots, X_l\}$ is possible signal points. For the Forman sequence, when the Gaussian noise (variance: 36; PSNR: 32.6 dB) or the salt and pepper noise (PSNR: 32.6 dB) is added into it, the result in Table 69.1 can be gained after the pre-filtering [3].

From the above, it can be known that pre-filtering has a large elimination effect on the salt and pepper noise, but it is not obvious for the sequence disturbed by Gaussian noises. After an objective evaluation by PSNR is made, the subjective evaluation can be seen as shown in Fig. 69.1.

69.1.2 Time Domain Filtering

The time domain filtering method in this paper is based on the weighted recursive filtering algorithm of front and back frames. In this algorithm, when the front frame and the frame before the front frame are the images after experiencing pre-filtering, their salt and pepper noise have been removed, but have do not proceed by the spatial domain smoothing method. In selecting the threshold, the video sequence Football within movement can be used for detection for the sake of solving the critical threshold. When it is larger than this threshold ($T1$), it is unnecessary to be processed. Gaussian noise (variance: 36; PSNR: 32.6 dB) can be used for detection to test the sequence input result when the threshold is between 5 and 100. The details can be seen in Table 69.2.

From the above, PSNR value decreases when threshold is too small; the difference between two frames is larger than the set thresholds most until the result of the front frame is output; PSNR value also decreases when threshold is too big, but results of the front frame and the frame before the front frame are still output by the recursive filtering within the threshold range; when the difference between two frames is larger than 18, W is 1 and is weighed with the front frame.

Table 69.2 The time domain filtering effect under different threshold ($T1$)

PSNR after noise reduction	PSNR (dB) after the *n*th frame noise reduction								Average of 8 frames
	9	10	11	12	13	14	15	16	
$T1 = 5$	31.496	31.349	31.277	31.572	31.970	31.422	31.349	31.206	31.455
$T1 = 10$	32.648	32.422	32.422	32.648	32.648	32.648	32.572	32.137	32.518
$T1 = 15$	33.349	33.572	33.349	33.572	33.727	33.496	33.422	33.277	33.470
$T1 = 18$	34.572	34.349	34.277	34.727	34.807	34.496	34.277	34.206	34.464
$T1 = 20$	33.496	33.277	33.496	33.572	33.888	33.496	33.422	33.206	33.482
$T1 = 100$	27.496	27.277	27.349	27.572	27.727	27.572	27.422	27.349	27.470

Table 69.3 The filtering effect of diffident threshold (*T*2) on gaussian noise

PSNR (dB) after the *n*th frame noise reduction									Average of 8 frames
	1	2	3	4	5	6	7	8	
$T2 = 1$	32.786	32.737	32.993	32.7856	32.736	32.837	32.837	32.736	32.806
$T2 = 5$	34.637	34.736	34.888	34.686	34.786	34.686	34.786	34.786	34.749
$T2 = 10$	32.686	32.837	32.686	32.737	32.589	32.637	32.540	32.498	32.650
$52 = 15$	31.637	31.492	31.637	31.637	31.637	31.540	31.540	31.493	31.577

Table 69.4 The filtering effect of diffident weighted value (*W*1) on gaussian noise

PSNR (dB) after the *n*th frame noise reduction								Average of 8 frames
1	2	3	4	5	6	7	8	
$W1 = 0.632.588$	32.540	32.786	32.588	32.686	32.686	32.588	32.588	32.631
$W1 = 0.733.736$	33.940	33.686	33.786	33.67	33.686	33.686	33.837	33.749
$W1 = 0.834.637$	34.736	34.888	34.686	34.786	34.686	34.786	34.786	34.749
$W1 = 0.933.736$	33.888	33.888	33.786	33.837	33.736	33.736	33.837	33.805

Table 69.5 The filtering effect of diffident weighted value (*W*1) on Gaussian noise

PSNR (dB) after the *n*th frame noise reduction								Average of 8 frames
1	2	3	4	5	6	7	8	
$W1 = 0.632.488$	32.540	32.786	32.588	32.686	32.686	32.588	32.588	32.631
$W1 = 0.532.437$	32.366	32.978	32.856	32.646	32.786	32.826	32.689	32.749
$W1 = 0.433.645$	33.823	33.543	33.886	33.332	33.324	33.243	33.543	33.529
$W1 = 0.331.736$	31.888	31.888	31.786	31.837	31.736	31.736	31.837	33.805

The movement intensity is large when the difference between two frames is larger than threshold, but is small when the difference is smaller than threshold. Test the second threshold from 1 to 17, and assume *W* is 0.8 (larger than *T*2 but smaller than *T*1) or 0.5 (smaller than *T*2). The domain filtering has a good effect on the stationary sequence, so the stationary sequence News is selected to be added with Gaussian noise (variance 36, PSNR: 32.6 dB) for testing *T*2 so as to solve the best *T*2 value, as shown in Table 69.3.

From the above, the threshold (T2) is 5. Determine the value of *W*1 first, namely, the weighted value when the difference in the pixel brightness values related to two frames is between [5, 15]. To test *W*1, the value range [0.6, 0.9] can be considered. The details are shown in Table 69.4.

The value of *W*2 can be determined from Table 69.5.

Thus, the best value for *W*1 is 0.4. Based on Tables 69.2, 69.3, 69.4, 69.5, the thresholds *T*1, *T*2, *W*1 and *W*2 are determined, and the equation of the first movement detection of the domain weighed recursive filtering can be gained as follows:

$$
\begin{aligned}
W &= 1, \quad |F_1(i,j) - F_2(i,j)| > 18 \\
W &= 0.8, \quad 5 < |F_1(i,j) - F_2(i,j)| < 18 \\
W &= 0.4, \quad |F_1(i,j) - F_2(i,j)| > 5
\end{aligned}
\tag{69.1}
$$

In Eq. (69.1), $F_1(i, j)$ is the pixel value of the last frame, and $F_2(i, j)$ is the pixel value of the current frame. Next, the second weighed recursive filtering is on, and the equation is as follows:

$$F(i,j) = (1 - W) * F_1(i,j) + W * F_2(i,j) \tag{69.2}$$

In Eq. (69.2), $F\ (i, j)$ is the result of the domain filtering, namely, the gray value of this point.

69.1.3 Spatial Domain Filtering

69.1.3.1 Edge Detection

First, edge detection can be made. Gauss-Laplacian is also called LOG operator. This operator will carry out a smoothing processing on the original image with Gaussian algorithm before solving edge, to realize the maximum suppression on noise. Two-dimension Gaussian function is used to smooth the image as follows:

$$G(x,y) = e^{-\frac{x^2+y^2}{2\delta^2}} \Big/ \left(2\pi\delta^2\right) \tag{69.3}$$

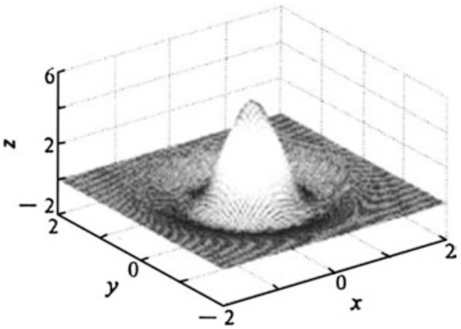
Its shape is like the clock inverted in the two-dimension space. For this function, Laplace transform can be used, namely, the second-order directional derivative can be taken for Gaussian function, so the function equation of the 2-D LOG operator can be gained as follows:

$$\nabla^2 G(x,y) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2} = \frac{1}{2\pi\delta^4}\left(\frac{x^2+y^2}{\delta^2} - 2\right) e^{-(x^2+y^2)/(2\delta^2)} \tag{69.4}$$

Use this LOG operator to do convolution operation on input image $I(x, y)$ and the output image is as follows:

$$
\begin{aligned}
F(x,y) &= \int_{\alpha\equiv-\infty}^{+\infty}\int_{\beta\equiv-\infty}^{+\infty} \nabla^2 G(\alpha,\beta) I(x-\alpha, y-\beta) d\alpha d\beta \\
&= \frac{1}{2\pi\delta^4}\int_{\alpha\equiv-\infty}^{+\infty}\int_{\beta\equiv-\infty}^{+\infty} \left(\frac{x^2+y^2}{\delta^2} e^{-\frac{x^2+y^2}{2\delta^2}}\right) * \\
&\quad I(x-\alpha, y-\beta) - 2e^{-\frac{x^2+y^2}{2\delta^2}} I(x-\alpha, y-\beta) d\alpha d\beta
\end{aligned}
\tag{69.5}
$$

Fig. 69.2 LOG operator



LOG operator is a low pass filtering process, which is used to eliminate the image intensity change when the spatial size is much less than Gaussian space coefficient to reduce noise. Then, use Laplace (∇^2) to gain the second-order directional derivative image $F(x, y)$ of smooth image $I(x, y)$. From the shape of LOG operator in Fig. 69.2, LOG operator is about symmetry of round, and is the same in sizes of edge detection in all directions.

The 5*5 template of the LOG edge detection operator in this algorithm is as follows:

$$\begin{array}{ccccc} -2 & -4 & -4 & -4 & -2 \\ -4 & 0 & 8 & 0 & -4 \\ -4 & 8 & 24 & 9 & -4 \\ -4 & 0 & 8 & 0 & -4 \\ -2 & -4 & -4 & -4 & -2 \end{array} \tag{69.6}$$

69.1.3.2 Noise Level Detection and Self-Adaptive Filtering

The noise level detection method is based on statistical method. The image without edges should be smooth under no noises. Under no noises, Forman, Football video sequences are detected, and noise detection is increased gradually, gaining cumulative probabilities of gray differences between [20, 256]. PSNR values are in Table 69.6 after sequences experience straight passing, 3*3 center-weighted filtering, 3*3 mean filtering, 5*5 mean filtering, and 7*7 mean filtering.

From the above, the divided four thresholds are $T_1 = 0.7$, $T_2 = 0.5$, $T_3 = 0.2$ and $T_4 = 0.1$. Order P is the cumulative probability of adjacent point brightness difference between [20, 256]. If there is $P > T_1$, 7*7 mean filtering is used; if $T_1 > P > T_2$, 5*5 mean filtering is used; if $T_2 > P > T_3$, 3*3 mean filtering is used; if $T_3 > P > T_4$, 3*3 center-weighted filtering is used; If $P < T_4$, straight output is used, which is based on Weber's Law, and now the noise is in the acceptable range. The algorithms of mean filtering, mid-value filtering and center-weighted filtering have been introduced above. Template of 3*3 center-weighted is as follows:

$$\begin{array}{ccc} 0.08 & 0.08 & 0.08 \\ 0.08 & 0.36 & 0.08 \\ 0.08 & 0.08 & 0.08 \end{array} \tag{69.7}$$

Table 69.6 PSNR values after forman and football sequences cumulative probabilities reduce noise when entering different filters or straight under different Gaussian noise

	Gaussian noise variance/PSNR (dB)	Cumulative probabilities	Straight output (dB)
Forman	9/38	0.071	38
	37/32	0.11	32
	81/29	0.21	29
	144/26.5	0.32	26.5
	324/23	0.48	23
	900/19	0.65	19
Football	9/38	0.07	38
	36/32	0.11	32
	81/29	0.22	29
	144/26.5	0.32	26.5
	324/23	0.48	23
	900/19	0.65	19

3*3 canter weighted filtering (dB)	3*3 mean filtering (dB)	5*5 mean filtering (dB)	7*7 mean filtering (dB)
29.4	29	27.6	27
30	31	29.6	28.5
26.3	29.8	28.5	27.3
25.6	27.9	27.5	25.8
25.3	25.8	26.2	25.9
29.4	29	26.3	25.6
29	30.8	29.5	27.8
27	29	28.8	28.5
24.2	25.8	26.2	24
21	22.9	23.3	23.2

69.1.4 Weighted Output

The space-time combined filter is used in this algorithm, and its results are shown in Table 69.7; the Wz is the weighted value of the spatial domain.

From the above, it can be known: $Wz = 0.9$, if $P > T1$; $Wz = 0.8$, if $T1 > P > T2$; $Wz = 0.55$, if $T2 > P > T3$; $Wz = 0$, if $P < T3 > P > T4$; $Wz = 0$, if $P <$ T4. The weighted equation is as follows:

$$Y(x, y) = Y_1(x, y) * (1 - Wz) + Y_2(x, y) * Wz \tag{69.8}$$

In Eq. (69.8), $Y(x, y)$ is the brightness value of space-time outputs; $Y1(x, y)$ is the brightness value of domain filtering; the $Y2(x, y)$ is the brightness value of spatial domain filtering. The final result is outputted with the form of RGB converted by Y component after processed and the U and V component. The conversion method is introduced above.

Table 69.7 Filtering effect of different space–time weighted value Wz on different level Gaussian noise

Movement detection in domain filtering(Wz/PSNR): $\|F_1(i,j)-F_2(i,j)\| < 18$ (dB)					$\|F_1(i,j)-F_2(i,j)\| > 18$
$P > T_1$ /17	$T_1 > P > T_2$ /21.7	$T_2 > P > T_3$ /28	$T_2 > P > T_4$/ 31	$P < T_4$ /33	
$Wz = 0.9$ 23.6	$Wz = 0.8$ 26.2	$Wz = 0.7$ 32	$Wz = 0.5$ 31.4	$Wz = 0.1$ 33.8	$Wz = 1$
$Wz = 0.8$ 23.5	$Wz = 0.7$ 26.1	$Wz = 0.6$ 32.2	$Wz = 0.4$ 31.8	$Wz = 0$ 33.8	
$Wz = 0.6$ 22.4	$Wz = 0.6$ 25.8	$Wz = 0.5$ 32.2	$Wz = 0.3$ 32.2		
–	–	$Wz = 0.4$ 31.9	$Wz = 0.2$ 32.5		
–	–		$Wz = 0$ 32.7		

69.2 Conclusion

The time domain in this algorithm can keep the edges and details of each frame, and the spatial domain has a good effect on smooth noise reduction. For different speed and sequences with different characteristics, the noise reduction effect is excellent. Subjective evaluation on it is good; the PSNR objective evaluation also suggests it has a good noise reduction effect.

References

1. Xie Z, Bao Z, Xu C, Zhang G, Zhang S, Yang Y (2009) Video pre-processing algorithm based on motion-compensation. Comput Eng 4(19):521–528
2. Li Y, Qiao Y, Gao F, Gao Y, Sun Z (2007) Adaptive temporal filter based on motion compensation for video noise reduction. Chin J Electron Devices 3(05):288–297
3. Song H, Xun Y (2009) Study on the whole variation image noise reduction. J Chengdu Electromech Coll 34(04):379–386