
Politechnika Poznańska
Wydział Elektryczny
Instytut Elektroniki i Telekomunikacji
ul. Piotrowo 3A, 60-965 Poznań

Jamal K. Abbas

# Decision-Based Nonlinear Filters for Image and Video Restoration

Doctoral Dissertation
Advisor: Prof. Marek Domański

Poznań, 2000

# Symbols and notations

| | |
|---|---|
| $\lvert\cdot\rvert$ | absolute value. |
| $\lVert\cdot\rVert$ | Euclidean norm value. |
| $\pi$ | pi. |
| $\beta$ | adjusted threshold. |
| $\Theta$ | closing. |
| $*$ | convolution. |
| $\mu$ | mean value of samples. |
| $\oplus$ | opening. |
| $\Sigma$ | summation. |
| $\sigma$ | the standard deviation of samples. |
| $\alpha$ | threshold value. |
| $\Diamond$ | times, symbol of duplication ($2\Diamond u,3\Diamond y = u, u, y, y, y$). |
| CWMF | center weighted median filter. |
| CWMPF | center weighted median filter with prediction error processing. |
| $\underline{d}(\tau;u,t)$ | vector displacement of spatial location of $u$. It has an expression of $(\Lambda_d)_t$ |
| $d(n_1,n_2)$ | prediction error for a pixel located at point $(n_1,n_2)$. |
| dB | decibel. |
| $e(n_1,n_2)$ | the output of prediction error processing unit for a pixel located at point $(n_1,n_2)$. |
| $g$ -1$(\cdot)$ | inverse function of $g$. |
| $g(\cdot), f(\cdot)$ | functions. |
| $K$ | number of samples inside a processed window $M\times N$, where $K = M\times N$. |

| | |
|---|---|
| $M \times N$ | window size. |
| max | maximum value. |
| MCF | median filter with motion-compensation. |
| MCPF | median filter with motion-compensation and prediction error processing. |
| med | median value. |
| MF | median filter. |
| min | minimum value. |
| MPF | median filter with prediction error processing. |
| *MSE* | mean square error. |
| $n(n_1,n_2)$ | noise sample located at point $(n_1,n_2)$. |
| $N_1 \times N_2$ | image size. |
| $p$ | noise probability |
| $P$(n) | probability distribution function (*pdf*) of n. |
| *PSNR* | peak signal to noise ratio. |
| RCWMF | recursive center weighted median filter. |
| RCWMPF | recursive center weighted median filter with prediction error processing. |
| RVCM | recursive median filter with motion-compensation. |
| RVCPM | recursive median filter with motion-compensation and prediction error processing. |
| RVM | recursive median filter. |
| RVMCF | recursive vector median filter with motion-compensation. |
| RVMCPF | recursive vector median filter with motion-compensation and prediction error processing. |
| RVMF | recursive vector median filter. |
| RVMPF | recursive vector median filter with prediction error processing. |
| RVPM | recursive median filter with prediction error processing. |
| $s(n_1,n_2)$ | the original value of a pixel located at point $(n_1,n_2)$. |
| *Segm* | segmentation. |
| *SNR* | signal to noise ratio. |
| $T$ | estimated threshold. |
| $\underline{u}$ | vector representation of $u$. |
| $u(n_1,n_2)$ | the input value of a pixel located at point $(n_1,n_2)$. |

| | |
|---|---|
| $v(n_1,n_2)$ | the prediction value of a pixel located at point $(n_1,n_2)$. |
| V3-MPF | variant 3 of median filter with prediction error processing. |
| V3-VMPF | variant 3 of vector median filter with prediction error processing. |
| VMCF | vector median filter with motion-compensation. |
| VMCPF | vector median filter with motion-compensation and prediction error processing. |
| VMF | vector median filter. |
| VMPF | vector median filter with prediction error processing. |
| $y(n_1,n_2)$ | the output value of the filter for a pixel located at point $(n_1,n_2)$. |

# Contents

# Abstract

This dissertation deals with denoising of color images and video sequences using decision-based filters. In the dissertation, it is proven that application of prediction error processing (which is a sub-class of decision-based filters) results in improved efficiency of image and video restoration.

The basic idea of prediction error processing filter is to predict a pixel value by using a nonlinear filter, and then to compare this value to that in the input (corrupted) image. Usually these two values are different and a decision has to be made about the pixel value at the filter output. This value can be considered as a sum of the prediction and the prediction error processed in some way. For example, large values of prediction errors are set to zero because they are can be classified as caused by impulsive noise samples. Soft decisions on classification of prediction errors lead to good results for test images.

The class of filters with prediction error processing proposed in the dissertation is very suitable for impulse noise removal from color images and video sequences. Preservation of textures and fine details are advantages of these filters. The experimental data prove that prediction error processing filters outperform classic nonlinear median-based filters like vector median, recursive median and weighted median.

# 1. Introduction

## 1.1. Scope of the work

This dissertation deals with digital image and video processing which become recently a large, important and rapidly expending discipline of science and technology [BER93a, GON92a, JAI89a, TEK95a]. The modern advancement in this area is due to availability of digital hardware for processing of large data files and fast transmission over communication networks. The research has been stimulated by new emerging applications related to image communication, computer networking, medical imaging, video surveillance systems, electronic publishing, computer-aided education, home cinema, digital broadcasting, etc. An important area within image and video processing is restoration [AND77a, JAI89a]. Restoration is a process by which an image suffering some forms of distortion or degradation can be recovered to its original form. Among various techniques developed for image and video restoration, digital filters have proved to be important and very useful tools.

A digital image is an inherently two-dimensional discrete signal while a digital video sequence is a three-dimensional signal [DUD84a]. Therefore multi-dimensional filters are efficiently used in order to process image and video. Among multi-dimensional digital systems, linear filters have gained most attention hitherto [LU92a]. They are easy to design and control but they often fail remove noise effectively. Nonlinear digital filters form a large class of systems used nowadays for this task [PIT90a].

This dissertation has been conceived to extend already available methods of nonlinear filtering with particular emphasis on fast techniques applicable in real-time processing of

video. The dissertation provides a comprehensive background to the methods available for the realization of both recursive and nonrecursive nonlinear digital filters, and gives an insight into the more recent implementation procedures.

Impulse noise is a model of quite many types of image and video degradation produced both during acquisition and transmission. Therefore rejection of impulse noise is a vital task of image restoration. Experience gained during many years of research proved that linear methods do not lead to satisfactory results for this application [JAI89a, KIN89a, PIT90a]. Nonlinear methods have shown to be more efficient. Many types of nonlinear filters have been already proposed and examined. Most of them were nonrecursive. Such filters are inherently stable and easy to design, but recursive structures are more efficient from the point of view of computational complexity related to their implementations. The concept of passive digital systems [DOM92a, DOM94a] showed its usefulness in solving stability problems for recursive systems.

The most popular nonlinear filter is a median filter. The median filter is an efficient robust impulse noise attenuation tool, with a property of edge preservation. Vector median filters are considered as a generalization of the median filters to vector-valued signals. Vector median filtering is a powerful tool for processing color images. A very nice advantage is that they output only colors present in input images.

Median is not a perfect mean of filtering because it may cause edge jitters, streaking and may remove important image details. In response to these difficulties, decision-based processing has been already proposed by [ABR96a, GAR98a, MAC94a, SAW96a, SUC94a, WAN99a]. The assumption is that processing does not change those pixels, which are presumably uncorrupted by noise. The idea is to use a nonlinear filter as a predictor and to use the prediction error in order to control the output of the system. An output value is either the input pixel itself or the nonlinear (median) filter output or a function of both. The output image quality is significantly improved as compared to the output of a median filter because small details and fine textures are much bettered reproduced.

## 1.2. The goals and the thesis

The main thesis of this dissertation is the following:

- The efficiency of median denoising may be significantly improved by application of decision-based processing. Decision-based processing can be successfully applied for rejection of impulse noise from monochrome and color images as well as from video.
- Application of decision-based processing improves denoising performance by means of numerous variants of median filters.

The main goal of the research is to prove the above thesis. Decision-based processing and its special case prediction error processing are core techniques, which allow achieving efficient rejection of impulse noise from images and video sequences.

The main goal is related to the following tasks:

- Systematic experimental comparisons of individual variants of median filters employed directly and in the decision-based structures.
- Experimental comparisons for color static images as well as for color video sequences.
- Experimental examination of video processing for two- and three-dimensional schemes including those with motion compensation.
- Development of simple and efficient adaptation schemes.
- Analysis of special applications of decision-based filters, e.g. for removal a comet-like impulses.

## 1.3. Dissertation structure

The dissertation is organized as follows:

- In chapter 2, the basics of image and video restoration problems are described. Some sources of image degradation are also considered.
- The main nonlinear restoration techniques in two and three-dimensions for images and video sequences are described in Chapter 3.
- Decision-based and prediction error processing filters are defined and explained in Chapter 4. In addition, prediction error processing filters are explained as a special case of decision-based filters.

- Experimental results are presented in Chapter 5. These results illustrate the performance of the new two-dimensional filters for still-images. This chapter includes also a comparison to the other median-based filters.
- Chapter 6 shows the applications of prediction error processing filters for video restoration. Images and tables support the subject for a demonstration purpose.
- Finally, some conclusions and final remarks are presented in Chapter 7.

# 2. Basic problems of image and video restoration

## 2.1. Digital image and video degradation

Degradation of digital image and video is a common problem in many areas of digital communication. Images are often degraded during recording and transmission due to imperfections of sensors and communication channels [BLU99a, LAB99a]. A channel acts some times as a filter that attenuates signals and distorts their waveforms, or causes a temporary lose of signal. As a result of these, a transmitted image is degraded. Also, the degradation is caused by undesirable signals along the communication path. Degradation is often described as *noise*, which means random and unpredictable signals [LAT89a].

### 2.1.1. Noise sources

There exist various sources of noise, which cause degradation of digital image and video. Some of them will be mentioned below.

a) Degradation due to propagation

Unreliable transmission poses severe problems for the transmission of video. Some existing communication networks cannot provide a high guaranteed quality of service, because high bit error rates cannot be avoided during fading periods [AST99a, BAR99a]. This short-term fading is mainly caused by multi-path reflections of a transmitted radio signal by local scatters such as building walls or other obstacles [DUB94a, GIR96a]. The received signal strength depends on the spatial position and strong variations may result for a local receiver situation [BLU99a]. Multipath propagation is also common in cellular systems. Several replicas of signal will, in general, be incident on a receiving array. As a

result of this, the signal can be seen as emitted from a disturbed source [AST99a]. Within or near building walls or other reflectors the echoes in the receiver channels arrive at various delays causing multi-path spread. In each of these cases, the transmission channel can be represented as several channels in parallel, each with a different relative attenuation and different time delay. Echoes with short delay may cancel the received signal and echoes with longer delays degrade the signal [DUB94a]. In a simple case, only two paths are considered. One with a unity gain and random delay $t_d$, and the other with a gain $a$ and a delay $t_d + \Delta t$.

The transfer function of the two paths is given by $e^{-j\omega t_d}$ and $ae^{-j\omega(t_d+\Delta t)}$, respectively. The overall transfer function $H(\omega)$ of such a channel is given by

$$\begin{aligned} H(\omega) &= e^{-j\omega t_d} + ae^{-j\omega(t_d+\Delta t)} \\ &= e^{-j\omega t_d}(1 + ae^{-j\omega\Delta t}) \\ &= e^{-j\omega t_d}(1 + a\cos(\omega\Delta t) - ja\sin(\omega\Delta t)) \\ &= \sqrt{1 + a\cos(\omega\Delta t) - j\,a\sin(\omega\Delta t)} \cdot e^{-j\left[\omega t_d + \tan^{-1}\left(\frac{a\sin(\omega\Delta t)}{1+a\cos(\omega\Delta t)}\right)\right]} \end{aligned} \tag{2.1}$$

Both the magnitude $\sqrt{1 + a\cos(\omega\Delta t) - ja\sin(\omega\Delta t)}$ and the phase $\left[\omega t_d + \tan^{-1}\left(\frac{a\sin(\omega\Delta t)}{1+a\cos(\omega\Delta t)}\right)\right]$ characteristics of $H(\omega)$ are periodic in $\omega$ with a period $2\pi/\Delta t$. From (2.1) we can conclude that, multi-path transmission causes a change in both the magnitude and the phase of the signal which is called *pulse dispersion* [LAT89a].

When the transmitted signal is an image, and the value of $t_d$ is randomly changed from pixel to pixel, this situation is referred to a time variant. On the receiver side, where a degraded image by this type of degradation is received, the degradation will be considered as one of two states,

1. All the levels of the received image have a random change in the amplitude; in this case the noise is regarded to an additive noise.
2. A sudden change has been happen for certain random pixels in the image; in this case the noise is regarded to an impulse noise.

b) Degradation due to acquisition errors

In many data acquisition systems noise is introduced into the data. Data recorded in this case are completely outlying due to error in the acquisition device. In image

processing applications, such errors produce the so-called salt and pepper noise (positive and negative impulses) [PIT90a].

**c) Degradation due to coding error**

Coding error degradation happens in several cases [GIR96a, PAN94a], for example:

- When decoder and encoder do not use the same reference information for motion compensation, in this situation the data is misinterpreted.
- Improper placement of quantizer levels causes a degradation to be seen.

This may cause image artifacts. Image artifacts can appear as additive noise or impulse noise.

d) Degradation due to image filtering

In situations where a signal is corrupted by noise, noise reduction is often a necessary pre-processing step. In many image processing systems (for instance, image coding or image filtering), the output image contains a residual operation error which depends on a function of the image. This kind of operation error could be considered as an additive noise [BAR97a, PIT90a, RAM97a].

e) Degradation due to the conversion

Degradation is also a common problem associated with television standards conversion (e.g., a composite to component conversion, interlaced to progressive conversion, compression and decompression, etc.). This degradation manifests itself in three different forms [DUB94a, LUT99a]:

- Waveform degradation of video signal, and the deterioration of the frequency characteristics, is due to the luminance and chrominance separation, and demodulation and its reverse composition.
- Image degradation due to the conversion of the number of lines
- Discontinuity of moving images due to the conversion of the number of fields.

### 2.1.2. Noise models

Image denoising is a process by which an image suffering some forms of degradation can be recovered to its original form [KIN89a]. From this point of view, it is very necessary to study noise models. Very common noise models are additive noise and impulse noise [LAB99a, STE92a].

a) Additive noise

The most familiar kind of additive noise is the white noise. Additive white noise *n* is spatially non-correlated, it is modeled as a realization of a random vector *n*. The components of *n* are complex valued and independently distributed, each component is modeled by a probability density function *pdf*. Common noise models are:

- **Gaussian noise,** provides a good model of noise in many imaging systems. Its probability density function is:

$$p(n) = \frac{1}{\sqrt{\pi\sigma^2}} e^{-\frac{n^2}{\sigma^2}} \tag{2.2}$$

where, $\sigma$ is the noise variance. The Gaussian distribution has an important property to estimate the mean of a stationary Gaussian random variable, one can't do any better than linear average. This makes Gaussian noise the worst case scenario for nonlinear image restoration filters, in the scene that the improvement over linear filters is least for Gaussian noise. To improve linear filtering results, nonlinear filters can exploit only the non-Gaussianity of the signal distribution [LAB99a, KIN89a].

- **Bi-exponential noise,** is called Laplacian noise in some literature which has this *pdf*:

$$p(n) = \frac{1}{\sqrt{2\sigma}} e^{-\frac{\sqrt{2}|n|}{\sigma}} \tag{2.3}$$

Nonlinear estimators can provide a much more accurate estimate of the mean of a stationary Laplacian random variable than the linear average [BOV83a, STE92a].

- **Uniform noise,** is not often uncounted in real world imaging systems, but provides a useful comparison with Gaussian noise. The linear average is a comparatively poor estimator for the mean of a uniform distribution. This implies that nonlinear filters should be better at removing uniform noise than Gaussian noise. The uniform *pdf* is given by [AND77A, JAI89a]:

$$p(n) = \begin{cases} \frac{1}{2\sqrt{3}\,\sigma} & for\ |n| \le \sigma\sqrt{3} \\ 0 & else \end{cases} \tag{2.4}$$

Even that linear filters are proposed for such a kind of filtering, many literatures suffer from the results of these filters in image processing. That is why, they attend to use some

modifications for these filters to solve the problem of the nonlinearity [AUR95a, SMI97a]. As an example, Fig. 2.1 shows a degraded image by Gaussian noise of $\sigma = 20$.

Figure 2.1. Degraded *Lena* image by Gaussian noise of $\sigma = 20$.

b) Impulse noise

This noise model was chosen to model the severe degradation that can occur in the transmission of analog and digital video sequences.

Impulse noise is highly dependent on the physical environment and may be relatively infrequently occurring and non-stationary, which often renders it impossible to obtain an

accurate statistical description. In a variety of impulse noise models for images, corrupted pixels are often replaced with a certain value different from the value of the neighbor pixels. The main purpose of noise suppression is to remove the impulses while preserving signal details and edges. For this purpose, a large number of nonlinear techniques has been developed [ABR96a, MAD97, MIT93a]. Unfortunately, most of them suffer from filtering degradation such as:

- Impulses are not well removed, due to the detection failure.
- Residual error is introduced when replacing a pixel with an estimated value.

For a human viewer a pixel is considered as an impulse noise only if its difference from the neighboring samples exceeds a certain threshold [LIN88a]. Below this threshold, the impulses are either not visible or they affect the image like noise with a continuous distribution if the probability is high. And above it, the visual error is rather independent of the exact height of the impulses. From this point of view, it is necessary to study the noise behavior to design the proper detector and estimator [KIM95a, SUC96a]. As a result of this, the filtering operation could be done only to the corrupted pixels and keeping the uncorrupted pixel without filtering. This topic is well discussed in prediction error processing section.

Pixels corrupted by impulse noise are often replaced with values equal to or near the maximum or minimum of the allowable dynamic range. For 8-bit images, this typically corresponds to fixed values near 0 or 255. A more general noise model also can be considered in which a noisy pixel can take on arbitrary values in the dynamic range according to some underlying probability distribution. Let $s(n_1,n_2)$ and $u(n_1,n_2)$ denote the luminance values of the original and the noisy images respectively. Then, for an impulse noise model with error probability $p$,

$$u(n_1,n_2) = \begin{Bmatrix} s(n_1,n_2) & \textit{with probability} & 1-p \\ \eta(n_1,n_2) & \textit{with probability} & p \end{Bmatrix} \tag{2.5}$$

where, $\eta(n_1,n_2)$ is an identically distributed independent random process with an arbitrary underlying probability density function. In color images, $\underline{u}(n_1,n_2)$, $\underline{s}(n_1,n_2)$, and $\underline{\eta}(n_1,n_2)$ are vectors of three components and $p$ in (2.5) could be considered as,

- the probability that, all $\underline{u}(n_1,n_2)$ components are corrupted at the same time,
- or, the probability of each color component of $\underline{u}(n_1,n_2)$ to be corrupted independently.

a)

b)

Figure 2.2. Spot-like impulse noise examples, a) degraded *Lena* image by salt and paper impulse noise, b) degraded *Claire* image by irregular impulse noise.

Common noise models are:

- **Salt and pepper spots noise**

Impulse noise appears as single spots with a value either the maximum or the minimum of the gray scale level of the pixels. In such a kind of noise (Figure 2.2a), we can

see that we are able to determine these spots due to the fact that the respective pixel value is very different from the majority of the values of the surrounding pixels [KUN84a].

- **Irregular spots noise**

This kind of noise is similar as salt and pepper but the corrupted pixel has random amplitude 0-255 (see Figure 2.2b) [ABB98a, BAR97a].

a)

b)

Figure 2.3. Comet-like impulse noise examples.

- **Comet-like scratches**

Scratches are a kind of impulse noise that often degrades badly quality of video. A very common example is related to a satellite receiver who produces scratches in the video signal when the antenna is not directed exactly to the satellite. These scratches remain in

video after digitization and constitute a distortion, which is very annoying for a viewer. The noise appears here as bright lines or a tailed-spot like a comet (see Fig. 2.3) [ABB99b, KOK96a].

## 2.2. Quality assessments

Image quality criteria are useful for measuring and rating the performance of a processing technique or a vision system. For example, when a certain restoration technique is used as a noise removal, the result will include a certain amount of degradation (which is called an operation error). Comparing restoration results with another present techniques requires a measure of image quality. There are two types of criteria that are used for evaluation of image quality, subjective and quantitative (objective).

### 2.2.1. Objective quality measure

A common quantitative measure of the performance of a de-noising algorithm is the mean square error *MSE* between the uncorrupted signal $s(n_1,n_2)$ and the restored signal $y(n_1,n_2)$ [JAI89a]. The *MSE* for $N_1 \times N_2$ image size, is defined as,

$$MSE = \frac{1}{N_1 \cdot N_2} \sum_{n1=0}^{N_1-1} \sum_{n2=0}^{N_2-1} \left( s(n_1,n_2) - y(n_1,n_2) \right)^2 \tag{2.6}$$

In many applications the *MSE* is expressed in terms of a signal to noise ratio *SNR* (the ratio between the original image variance $S^2$ and the *MSE* value), that is defined in decibels dB as,

$$SNR = 10 \cdot \log_{10} \frac{S^2}{MSE} \tag{2.7}$$

$$S^2 = \frac{1}{N_1 \cdot N_2} \sum_{n1=0}^{N_1-1} \sum_{n2=0}^{N_2-1} \left( s(n_1,n_2) \right)^2 \tag{2.8}$$

Another definition of *SNR*, commonly used in image coding applications, is peak signal to noise ratio *PSNR* (in decibel dB). It is also used for quantitative measure of image performance. *PSNR* is calculated by scaling the *MSE* according to the image range $S_{\max}$ (the maximum pixel value). The *PSNR* is defined as,

$$PSNR\,[dB] = 10 \cdot \log_{10} \frac{(S_{\max})^2}{MSE} \tag{2.9}$$

To extend this approach for color images, pixel value is considered as a triple of the values of the color components related to a given pixel. Such as in vector image processing of color images, simultaneous processing of all the three signal components is done. There exist a variety of distance measures for calculation of vector spaces. The Euclidean norm is considered here because it is shown to be better in vector processing [BAR95a]. The choice of the color coordinate system should be considered to take the maximum advantage of the vector approach. On the other hand, the maximum concordance with human perception rules should be kept [BAR95a, LUK82a]. In this case *MSE* could be calculated in different color spaces. The input and output images are then transformed to a uniform color space and the distortion measure is calculated. This method is implemented because it is mathematically treatable and it provides initial numerical assessment of the quality measure. For *RGB* color space *MSE* will take a form of,

$$\begin{aligned} MSE = \frac{1}{N_1 \cdot N_2} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} \left(s_R(n_1,n_2) - y_R(n_1,n_2)\right)^2 + \\ \left(s_G(n_1,n_2) - y_G(n_1,n_2)\right)^2 + \left(s_B(n_1,n_2) - y_B(n_1,n_2)\right)^2, \end{aligned} \tag{2.10}$$

*PSNR* is calculated by scaling the *MSE* according to the image range $3 \cdot S_{\max}$, where $S_{\max}$ here is the maximum component value.

$$PSNR = 10 \log_{10} \frac{3 \cdot (S_{\max})^2}{MSE}, \tag{2.11}$$

where, $s_R$ and $y_R$ are the red components of the original and the output pixels, respectively, and so on [BAR95a, SAN98a, VAN91a]. The same way will be applied in case of *L*a*b** color space, and the *MSE* is calculated as,

$$\begin{aligned} MSE = \frac{1}{N_1 \cdot N_2} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} \left(s_L(n_1,n_2) - y_L(n_1,n_2)\right)^2 + \\ \left(s_a(n_1,n_2) - y_a(n_1,n_2)\right)^2 + \left(s_b(n_1,n_2) - y_b(n_1,n_2)\right)^2, \end{aligned} \tag{2.12}$$

where, $s_L$ and $y_L$ are the *L* components of the original and the output pixels, respectively, and so on. The *PSNR* calculated in the *L* a* b** color space is more reliable as the color

difference calculated in this space is a better measure of the color distance as perceived by a human being [CON97a].

For video sequence processing, calculation of the *PSNR* for the whole image of video sequence is related to a problem that arises due to different resolution of individual components [HAS97a]. Usually, the input in digital video data format comprises chrominance components, which are decimated in respect to the luminance component. For this purpose, the chrominance components must be interpolated. In this theme, image quality has been assessed using *PSNR* calculated individually for each $Y$, $C_B$, and $C_R$ component to avoid the interpolation error. Such a quality measure application is explained in [ABB98a, ABB98e].

### 2.2.2. Subjective quality measure

The *PSNR* measurement is also not ideal method for quality measure, but is in common use. Its main disadvantage is that the maximum signal strength is estimated as $(S)^2$, rather than the actual signal strength for the image. *PSNR* is a good measure for comparing of degradation of the same kind. Therefore, the *PSNR* does not give a complete status of the image restoration result. Moreover, it does not distinguish between visually important and unimportant features. For this reason, it is important to look at the images by us for a subjective estimation of image quality.

Many of the models of the human visual system assume that there is an initial nonlinearity, followed by a linear system that is frequency and orientation dependent. Subjects were asked to judge the quality of the images and subjective rating was then applied to the images. Subjective rating was then compared to the distortion measure if the original image and the processed images are first transformed to a perceptually uniform color space and the distortion measure is calculated in this space [LUK82a, VAN91a].

Subjective quality depends on the human evaluation of the image quality. The test is performed by collecting data from the scores given by human subjects to each image or video clip. The subjective test is essential because the existing human visual models do not correlate well with the subjective test. The subjective test is classified as *blind* test or *supervised* test [BEL97a, OLI97a].

- *Blind* test, for on-line receiving signals (such as, television signals received by a satellite receiver), it is too hard to calculate the *PSNR* because original signal is not available. Person performing test without the presence of the original one.
- *Supervised* test, the subject evaluates a degraded image while comparing it against the original.

Both tests result in a distribution of values for each pair of conditions. The method of analysis depends on the nature of the judgment and the information required. Table 2.1 shows such a kind of comparison [ITU94a, JAI89a].

Another scaling method used for subjective quality judgment, is the single-stimulus method. In this method, observers assign an image or image sequence to one of the set of categories that, typically, are defined in semantic terms. The categories may reflect judgments of whether or not an attribute is detected. Categorical scale that assess image quality and image impairment has been used most often, and the ITU-R scales are given in Table 2.2.

In general, the subjective quality scales can be classified as:

a) ***Quality scale***, a quality scale may be a global (Table 2.1) or group scale (Table 2.2). A training set of images is used to calibrate such a scale. The group quality scale is based on comparison within a set of images.

b) ***Impairment scale***, the impairment scale rates an image on the basis of the level of degradation presence in image when compared with an ideal image. It is useful in applications such as image coding, where the encoding process introduces degradation of the output image. If several observers are used in the evaluation process, then the mean rating is given by,

$$R = \frac{\sum_{k=1}^{n} s_k n_k}{\sum_{k=1}^{n} n_k}, \tag{2.13}$$

where, $s_k$ is the score associated with the *k*-th rating, $n_k$ is the number of grades in the scale [JAI89a].

Table 2.1. Comparison scales for subjective categorical judgment methods.

| Supervised test | | Blind test | |
|---|---|---|---|
| Much better | ( +3 ) | Best | (7) |
| Better | ( +2 ) | Well above average | (6) |
| Slightly better | ( +1 ) | Slightly below average | (5) |
| The same | ( 0 ) | Average | (4) |
| Slightly worse | ( -1 ) | Slightly below average | (3) |
| Worse | ( -2 ) | Well below average | (2) |
| Much worse | ( -3 ) | Worst | (1) |

Table 2.2. Subjective quality scales.

| Quality scale | | Impairment scale | |
|---|---|---|---|
| Excellent | (5) | Imperceptible | (5) |
| Good | (4) | Perceptible, but not annoying | (4) |
| Fair | (3) | Slightly annoying | (3) |
| Poor | (2) | Annoying | (2) |
| Unsatisfactory | (1) | Very annoying | (1) |

## 2.3. Basic restoration techniques

One of the most important tasks in image restoration is noise filtering. Noise removal in images is particularly difficult due to various reasons:

- Images are non-stationary two-dimensional processes.
- They are often corrupted by multiplicative, additive, and signal dependent noise.
- The exact characteristics of our visual system are not well understood. However experimental results indicate that the first processing levels of our visual system possess non-linear characteristics.
- Our visual perception is heavily based on edge information. Thus noise filtering must preserve edges.

The first technique used for noise removal was linear one. Experience gained during many years of research proved that linear methods do not lead to satisfactory results for this application. Nonlinear methods have shown to be more efficient [PIT90a].

Nonlinear digital signal or image processing and analysis cover a very broad and non-homogeneous area. It is related to nonlinear systems theory, statistics and stochastic processes, communications theory, pattern recognition, algebraic theory of lattices, etc. A source of criticism of nonlinear signal and image processing area is the lack of a unifying theory. This is the major theoretical and practical drawback when it is compared to linear digital signal processing [PIT90a].

Linear filters have been the dominating filter class throughout the history of signal processing. They are easy to implement and analyze and linear filters minimizing the mean square error MSE can usually be found in closed form. Furthermore, linear filters minimizing the MSE are optimal among the class of all filtering operations when the noise is additive and Gaussian. It is indeed unfortunate that, despite the elegant linear system theory, many signal processing problems cannot be satisfactorily addressed through the use of linear filters. In image processing applications, linear filters tend to blur the edges, fail to remove heavy tailed noise effectively, and do not perform well in the presence of signal dependent noise.

In order to overcome the shortcoming of linear filters, several new classes of nonlinear digital filters have been developed [KIN89a, PIT90a]. Perhaps the most popular classes of nonlinear filters are rank order-based filters. These filters perform well in many situations where linear filters fail. Various types of rank order-based filters have been proposed during the last decade and these filters usually fall into one of two categories,

- the first category, is the set of hybrid order statistic filters which combine rank order filters with linear filters to exploit the desirable properties of both filter classes such as *L*-filters [DOM86a, DOM89a].
- the second category, is the set of generalized order statistics property, such as median, weighted median, weighted order statistics, morphological, multi-level median, stack, and generalized stack filters [COM92a, COY91a, IMA96a, SUN91a, YIN96a].

In nonlinear case, a unifying theory is much more difficult to develop and to generalize than in the linear systems case and it might be of no use if it does not help us to solve practical problems. Efforts to formulate unifying concepts have already been made with promising results [COY91a, IMA96a, KIN89a]. The theory of stack filters establishes a link between order statistics and morphological filters and even more, the threshold decomposition gives the tool for their analysis and synthesis. Robust statistics tools have

been successful in analysis and merging classes of nonlinear filters. Homomorphic filters theory provides an elegant way to link certain classes of nonlinear systems to linear ones. Order statistics and robust estimation theory are basis for a broad class of nonlinear filters called order statistics filters. The most prominent representative of this class is the median filter, very attractive for its computational simplicity as well as its useful properties, namely edge preservation and robustness against impulse noise. Since the median filter has been proposed, the class of order statistics filters has been extensively developed [ABB98b, KIN89a, PIT90a]. Vector median filters are considered as a generalization of the median filters to vector-valued signals. Vector median filtering is shown to be a powerful tool for processing color images compared with scalar median filtering [AST90a].

# 3. Survey of nonlinear filters for image and video restoration

## 3.1. Importance of nonlinear processing of image and video

Nonlinear techniques in digital processing are traced back in the last decades [PIT90a]. And nowadays, the growth is still explosive and many topics seem to be of a considerable concern for the future. Most of image and video processing systems are highly nonlinear; therefore nonlinear processing is required, for example:

- Analog to digital conversion is a nonlinear process.
- Transmission through nonlinear channels introduces nonlinear degradation.
- Segmentation, feature extraction, analysis and classification tasks are highly nonlinear by their nature.
- The human visual perception mechanism is proven to have nonlinear characteristics and this has to be taken into account in image enhancements.
- High frequency information carries very important information for visual perception demanding for filters with good edge and image detail preservation properties. Since linear filters have generally low-pass characteristics, nonlinear filters have to be used.

For such reasons, nonlinear filtering techniques for signal, image, and video processing were considered as early as 1958 [PIT90a].

## 3.2. Classes of nonlinear filters

At present, we can identify several classes of well-defined nonlinear techniques based on solid mathematical theories. Also, there are many *ad-hoc* nonlinear techniques of practical importance. A specific tool of selected nonlinear filters will be presented in the dissertation, such as median-based filters as a generation of order statistics filters, mathematical morphology, polynomial and homomorphic filters.

Many types of nonlinear filters have been already proposed and examined [IMA96a, JAI89a, KIN89a]. It is the opinion of the authors that median-based filters today give one sound approach to nonlinear filtering. The success of median filters is based on two intrinsic properties: edge preservation and efficient noise attenuation with robustness against impulse type noise. The main drawback of median filter is that, it uses only rank-order information of the input data within the filter window, and discards its original temporal-rank information. In order to utilize both rank- and temporal-information of input data [YIN96a], several classes of temporal-order based filters have been developed in recent years, such as FIR-median hybrid filters, *L*-filters, weighted median-based filters, weighted order statistics filters, stack filters and Boolean filters. FIR median hybrid and *L*-filters make use of the desirable properties of both linear and median filters [DOM86a, PIT88a, YIN96a]. Several surveys have been written to trace the development from median to stack filtering. Connections and comparisons among the presented topics for image and video restoration in two and three-dimension techniques will be made in this chapter.

## 3.3. Nonlinear filters in two-dimensions

Image formation is a classical example of a nonlinear process, which involves signal-dependent noise. It is described by the following equation [PIT88a],

$$u = t(h * s) + r(h * s)n_1 + n_2 \tag{3.1}$$

where, $s$ is the object illumination, $u$ is the recorded image, $h$ is the point-spread function of the imaging system, $t(\cdot)$ is the nonlinear function and $n_1$, $n_2$ are white zero mean Gaussian noises independent of each other and independent of the signals. The symbol $*$ denotes convolution. The term $r(h * s)n_1$ is the signal-dependent noise. The aim of image

filtering is to recover $s$ from $u$. When only signal-dependent noise is considered, a simpler image formation model could be considered as,

$$u = t(h * s) + r(h * s)n \tag{3.2}$$

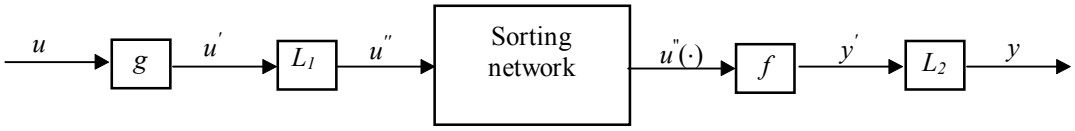



Figure 3.1. General structure of nonlinear filter.

The general nonlinear structure is shown in Figure 3.1. The nonlinear functions $g(\cdot)$ and f(·) are used for the decoupling the noise from the image. The function $f(\cdot)$ and the first derivative $g^{(1)}(\cdot)$ can be specified as follows:

$$g^{(1)}\left[t(s)\right] = \frac{1}{r(s)} \tag{3.3}$$

$$f(s) = g^{-1}\left[t(s)\right] \tag{3.4}$$

The nonlinear function $g(\cdot)$ can be found by integrating $g^{(1)}(\cdot)$. The linear filters $L_1$, $L_2$ and the sorting network are used for additive noise removal. Specifically, the linear filter $L_1$ is used to incorporate neighborhood (the number of the samples in the window) information in the filter structure, i.e., to use weight for the image pixels in an image neighborhood. Such neighborhood information is very important for the performance of the nonlinear filter, because it is shown that the ordering process in the sorting network destroys it. This fact deteriorates from $L_1$ can be represented by the weight factors $b_i$, $i$ =1,..., $K$ which operate on the samples $u_i$ = g($u_i$), $i$ = 1, .., $K$ inside the filter window of dimension $K$=$M$×$N$:

$$u_i'' = b_i u_i' = b_i g(u_i), \qquad i = 1,\ldots,K. \tag{3.5}$$

Until now, no method has been proposed for the optimal choice of $b_i$. However, techniques similar to the ones proposed in [DOM94a] can be used. The sorting network and the linear filter $L_2$ can be used for the additive noise filtering after the incorporation of the neighborhood information by (3.5). The output of $y$ of $L_2$ is a linear combination of the ordered data $u''_{(i)}$, where $u''_{(K)} \geq \ldots \geq u''_{(1)}$. The best known application of such a kind of filters is $L$-filter or order statistic filter. These filters have a form of,

$$y' = \sum_{i=1}^{K} a_i u''_{(i)} \tag{3.6}$$

The filter coefficients $a_i$ are given by the following formula,

$$a = \frac{\underline{H}_r^{-1}\underline{e}}{\underline{e}^T \underline{H}_r^{-1}\underline{e}}, \qquad \underline{a}^T = \left[a_1, \ldots, a_K\right] \tag{3.7}$$

where, $\underline{e}$ is the unit vector and $\underline{H}_r$ is the known correlation matrix of the ordered noise variates. The choice of $L_1$ and $L_2$ according to (3.5) and (3.6) leads to the filter structure described by the following formula,

$$y = f\left(\sum_{i=1}^{K} a_i \left(b_i g\left(u_i\right)\right)_{(i)}\right) \tag{3.8}$$

Three classes of nonlinear filters are special cases of the nonlinear module in (3.8): order-statistics, nonlinear statistical mean and homomorphic filters, and morphological filters [PIT88a, PIT90a]. Order statistics and robust estimation theory are the basis for a broad class of nonlinear filters called order statistics filters [LEE98a]. The most prominent representative of this class is median filter, which is a very attractive tool for its computational simplicity as well as for its useful properties, namely edge preservation and robustness against impulse noise. Since the median filter has been proposed, the class of order statistics filters has been extensively developed [COY91a].

The median filter is a nonlinear filter, which has the useful property of removing (reducing) impulse noise without (severely) smoothing the edges of the signal. In the past 20 years, median filters have been generalized and modified in many ways, for example, weighted median filters, center weighted median filters, vector median filters, and their generalizations. Good results can be obtained by using these generalized filters, even for non-impulse noise [SUN95a]. Numerous papers, monographs and book chapters have explored the median operation, analyzing its behavior and proposing new and viable extension [PIT90a, KIN89a].

### 3.3.1. Median filter

***a) Definition:***

The median filter performs a nonlinear filtering operation where a window moves over a signal, and at each point the median value of the data within the window is taken as the output. It is a nonlinear filter, thus for two sequences $\underline{u}$ and $\underline{y}$, $median\left\{\underline{u} + \underline{y}\right\} \neq median\left\{\underline{u}\right\} + median\left\{\underline{y}\right\}$ [PIT90a]. Median filtering has some desirable properties that can not be achieved with linear algorithm. The impulse response of the median filter is zero. This property makes its use attractive in suppressing impulse noise. Median filters are robust and are well suited for data smoothing when the noise characteristics are unknown. A stepwise change in a signal passes the median filter unaltered. This property is used in applications such as image filtering, where data needs to be smoothed but blurring of the signal edges is not acceptable. The median of $K$ samples of $u(k)$ , $k = (1, ....., K)$, can be defined as the value $u_{med}$ such for all $y$,

$$\sum_{i=1}^{K} \left| u_{med} - u(i) \right| \leq \sum_{i=1}^{K} \left| y - u(i) \right| \tag{3.9}$$

The output of a median filter is by definition always one of the input samples within the data window. Thus, it is possible that the filtered signal is identical to the original signal. To compute the output of a median filter, an odd number of sample value is used as the filter output. To simplify the median calculation of two-dimensions, $k$ number of samples inside a $M \times N$ widow size is assumed, $K = M \times N$. For each input pixel $u$ at certain point $n_1=1,...,N_1$, $n_2=1,...,N_2$ on ($N_1 \times N_2$ size) image, the filtering procedure is denoted as:

$$y(n_1, n_2) = med\,[u(1), ....., u(K)] \tag{3.10}$$

To be able to filter also the out-most input sample, where a part of the filter window fall outside the input signals, the latter is appended to the required size. The appending of the input signal is commonly performed by replicating the most out-most input samples as many times as needed. This appended strategy is referred to as the first and last values carry on appending strategy [AST90a, SUN91a, SUN95a].

***b) Statistical properties:***

The noise suppression characteristics of median filter are investigated when a constant signal is embedded in additive white noise. The edge and detail preservation properties are

examined by considering two-dimensional inputs with step edges and lines that are also corrupted by additive white noise. For independent identically distributed white noise n with $\mu$ mean and $\sigma^2$ variance, the corresponding distribution is denoted by $F(n)$ with density function $f(n)$ [SCH86a, SPI72a]. The distribution $P_{med}(n)$ and the density $p_{med}(n)$ of median filter output of $K$= 2$k$-1 number of samples provide the basis for quantitative analysis of the noise attenuation [YIN96a].

$$P_{med}(n) = \sum_{i=k+1}^{K} \binom{K}{i} F(n)^i (1 - F(n))^{K-i} \tag{3.11}$$

$$p_{med}(n) = \frac{K!}{k!k!} f(n) F(n)^k (1 - F(n))^k \tag{3.12}$$

When $n_{0.5}$ is considered as $F(n)$=0.5 then, median filter output will be with mean $\mu_{med} = n_{0.5}$, and variance $\sigma_{med} = 1/(4K[f(n_{0.5})]^2)$. This result shows the robust noise smoothing properties of the median filter. In case of heavy-tailed or impulse noise distributions, the variance of the distribution grows with the amplitude of the sample; the mean does not posses this property.

**c) *Deterministic properties***

In median filtering, the output is always one of the input samples. Therefore, it is conceivable that certain signals could pass through the median filter unaltered. This has shown to hold for median filter and many median-based filters [PIT90a].

In noise filtering, a problem is often how to preserve some desired signal features while attenuating noise. An optimal situation would arise if the filter could be designed so that the desired features were invariant to the filtering operation and only noise would be affected [GAR98a].

Since the median filter is a nonlinear filter, and the superposition principle does not apply, this of course can never be fully obtained. However, when a signal consists of constant areas and stepwise between these areas, a similar effect is achieved. Noise will be attenuated, but stepwise changes will remain.

In image processing, a common approach is to design a median filter such that certain image details (e.g. lines) are root signals and thus not disturbed by filtering operation. With these concepts, [GAR98a, YIN96a] prove that,

- An arbitrary finite length signal is a median root if it consists of constant neighborhoods and edges only.
- A repeated median filter is very significant and is called the *convergence property*.

Recursive median filters do also posses the convergence property. Furthermore, they are *idempotent* i.e. they produce root signals after a single pass. A recursive median filter has exactly the same set of root signals as non-recursive median filter of the same widow width, but given input signal may be filtered to different roots by two filters [YIN96a].

### 3.3.2. Weighted median filter

***a) Definition:***

The weighted median filter is an extension of the median filter, which gives more weight to some values within the window [SUN95a]. This filter is a promising image enhancement technique. For the discrete-time continuos-valued $K$ input samples, $\underline{u}$ =[$u$(1),....,$u$($K$)], the output y of weighted median filter of span $K$ associated with the integer weights $\underline{w}$=[$w$(1),...,$w$($K$)] is given by,

$$y = med[w(1) \lozenge u(1), \ldots, w(K) \lozenge u(K)] \tag{3.13}$$

where, *med* [•] denotes the median operation and $\lozenge$ denotes duplication, $w \lozenge u = w$ times of $u$ [YIN96a]. The filtering procedure can be stated as:

- Sort the samples inside the filter window.
- Duplicate each sample $u(k)$ to the number of the corresponding weight $w(k)$.
- Choose the median value from the new sequence.

***b) Statistical properties:***

Let the inputs of weighted median filter be white noise, with a number of samples $K=2k+1$, identically distributed with a common distribution function $F(n)$. And the weighted median filter has a different weight vector combinations $M_i$ with the same window width, The output distribution of weighted median filter $P_{wm}(n)$ has the following form, in which there are $i$ weights and the sum of these $i$ weights is not less than a certain threshold, for $i$=1,...,$K$.

$$P_{wm}(n) = P_{med}(n) + \sum_{i=1}^{k} M_i \left\{ F(n)^i (1 - F(n))^{K-i} - F(n)^{K-i} (1 - F(n))^i \right\} \tag{3.14}$$

This theorem demonstrates that the output central moment of the weighted median filter consists of two parts, the first corresponds to that of the standard median filter and the second quantifies the distribution of the weights. If all weights are equal, the second term is vanished [YIN96a]. When the input distribution is symmetric with respect to its mean, the output distribution is also symmetric with respect to its mean. That is, weighted median filter is unbiased estimator of the mean. In fact, it has been shown that all stack filters are statistically unbiased. In practice, the unbiasedness of stack filters is a typically constraint [RUI96a].

***c) Center Weighted Median***

When we give more weight only to the central value of the window a special case of weighted median filters called the Center Weighted Median filter will be produced, and thus it is easier to design and implement than general weighted median filters [SUN95a]. For the discrete-time continuos-valued of $K$ input samples in $M \times N$ window $W$ at point $(n_1,n_2)$, $n_1=1, ..., N_1$, $n_2=1, ..., N_2$, $u(n_1,n_2)=[u(1), ..., u(K)]$, the output $y$ of center weighted median filter of span $K$ samples is given by,

$$y(n_1,n_2) = MED\ [u(1), ..., u(K), 2\ l\ copies\ of\ u(1)\ \ K \in W] \qquad (3.15)$$

where, $l$ is a non-negative integer. When $l=0$, the CWM filter becomes the median filter, and when $2l+1$ is greater than or equal to the window size, it becomes the identity filter (no filtering).

Many applications of center weighted median filters in signal processing have been reported in the literature due to their useful properties known as detail preserving and noise suppressing, particularly heavy-tailed noise. These applications can be found in both one-dimensional and multi-dimensional cases [SUN91a].

Because of the nonlinear nature of median based filters, the analysis of center weighted median filters is mainly based on statistical and deterministic properties of the filters. The statistical properties of center weighted median filters have been studied to evaluate the noise suppression, edge and detail, e.g. fine lines, preservation characteristics, while the study of the deterministic properties includes root sets and convergence behavior of the filters in time domain. For identically and independently distributed inputs $F(n)$, the output distribution function $P_{cwm}\ (n)$ of the center weighted median with $K$ number of samples, $K = 2k+1$, and center weight $L = 2l+1$ is given,

$$P_{wm}(n) = \sum_{i=k+1-l}^{2k} \binom{2k}{i} F(n)^{i+1} (1 - F(n))^{2k-i} + \sum_{i=k+1+l}^{2k} \binom{2k}{i} F(n)^{i} (1 - F(n))^{2k+1-i} \quad (3.16)$$

Obviously, a center weighted median filter with a larger central weight performs better in detail preservation but worse in noise suppression than one with a smaller central weight [SUN95a, YAN94a]. However, there exists a clear trade-off between detail preservation and noise suppression property of this filter. The central weight should be carefully selected depending on the characteristics of the input image and its noise. In an attempt to improve center weighted median filters, an adaptive center weighted median filter having a variable central weight has been proposed in [SUN91a]. In summary, the center weighted median filter can preserve edges and details while reducing noise.

### 3.3.3. Vector median filter

***a) Definition:***

Median filtering of monochrome images selects the pixel with median intensity from the neighborhood. This filter has the useful property of removing impulse noise without blurring edges. The vector median filter may be regarded as a generalization of the median filter to vector valued signals as color images where pixel values are vector quantities rather than separate scalar values [AST90a, BAR95a]. The advantages in color images are that a vector median filter will remove impulse noise because the filter replaces the noisy pixel with one of its neighbors, thus preserving local color properties. On the other hand, median filtering of the *RGB* image components independently might replace one color component of a pixel, leaving the others unchanged; achieving only a change of color of the noise corrupted pixel rather than its removal. Vector median filtering is a powerful tool for processing of color images. Vector median of a set of vectors $\{\underline{u}(i)\}$ is $\underline{y} \in \{\underline{u}(i)\}$ such that $\sum_i \left\| \underline{u}(i) - \underline{y} \right\|$ is minimum.

The vector norm can be of various type, $l_1$ or $l_2$. The norm can be calculated in various color spaces. The vector median filter is evaluated as follows for an RGB image. For each pixel in the chosen neighborhood the following sum must be computed:

$$s(i) = \sum_{j=1}^{K} \left\| \underline{u}(i) - \underline{u}(j) \right\|, \quad i = 1, \ldots K \quad (3.17)$$

where, $\underline{u}(i)$ is the $i$-th pixel in the neighborhood, and $K$ is the number of neighborhood inside the $M \times N$ window size. The pixel, which yields the minimum value of s, is written to the filtered image. In rectangular *RGB* space the two pixels thus:

$$\left\| \underline{u}(i) - \underline{u}(j) \right\| = \sqrt{\left(u_r(i) - u_r(j)\right)^2 + \left(u_g(i) - u_g(j)\right)^2 + \left(u_b(i) - u_b(j)\right)^2} \tag{3.18}$$

where, $u_r(i)$ is the red component of the pixel $\underline{u}(i)$ and so on. Thus $s$ is simply the sum of the distances in *RGB* space from the pixel to all other pixels in the neighborhood. The selection of the pixel with minimum $s$ may be visualized as finding the pixel nearest the center of the pixels within the neighborhood viewed as a cluster in $R$, $G$, and $B$ color space.

The value of $\left\| \underline{u}(i) - \underline{u}(j) \right\|$ could be calculated by choosing other color spaces such as $L^*$, $a^*$, and $b^*$ according to the requirement. For digital video processing, calculation of $\left\| \underline{u}(i) - \underline{u}(j) \right\|$ using $Y$, $C_B$, *and* $C_R$ color space is recommended. The vector median filter can be seen as a simple generalization of the standard monochrome median filter. It is well known that applying a center median filter to a color image will remove more noise than simply one pass over the corrupted image [ABB99e, BAR95a, BAR97a].

***b) Statistical properties:***

When extending the median operation to vector-valued signal, some requirements is replaced for the resulting vector median operation,

- the operations have to have properties similar to those of median operation in the scalar case, that is, zero impulse response and good robust data smoothing ability while retaining sharp edges in a signal,
- the vector median reduces to the scalar median when the vector dimension is one.

One of the basic properties of the median filter is that it does not introduce any sample values that are not present in the filter input; this should be the case also with the vector median filters. Therefore, the vector median filter output must be one of the input vectors [AST90a].

Now, consider a vector median filter of length $K = 2k+1$ and a length $k$ vector impulse,

$$\underline{u}(n) = \begin{cases} 0 & \text{if } n < 0 \quad \text{or} \quad n \geq k \\ \underline{c}(n) & \text{if } n \geq 0 \quad \text{and} \quad n < k \end{cases} \tag{3.19}$$

When the filter is centered at $\underline{u}(k)$, all the non-zero signal values fall inside the filter window. To find the filter output $\underline{y}(k)$ we compute the sums s(0),...,s(2$k$) as in (3.17), for the $\underline{u}(2k)$ inside the window. Obviously

$$s(k) = s(k+1) = \ldots = s(2k) = \sum_{i=0}^{k-1} \left\| \underline{c}(i) \right\| \tag{3.20}$$

$$s(j) = (k+1) \left\| \underline{c}(j) \right\| + \sum_{i=0}^{k-1} \left\| \underline{c}(j) - \underline{c}(i) \right\| \qquad j = 0, \ldots, k-1 \tag{3.21}$$

From (3.20) and (3.21), a combination is yield as,

$$s(j) - s(k) = \sum_{i=0}^{k-1} \left( \left\| \underline{c}(j) - \underline{c}(i) \right\| + \left\| \underline{c}(j) \right\| - \left\| \underline{c}(i) \right\| \right) + \left\| \underline{c}(j) \right\| \geq 0 \tag{3.22}$$

As $s(k) \leq s(j)$ for all $j$ = 0, ..., $k$-1, the filter output is $\underline{x}(k) = \underline{0}$. The vector median thus removes impulses in a fashion similar to that of the scalar median filter.

***c) Extensions of vector median filters***

Vector median *VM* filters suffer from the same limitations as scalar median filters, since no design parameter can be adjusted but the window size. Therefore, a generalization of weighted median filters to the vector median filter with fixed weights, center weight, or adaptively varying weights also can be considered [ALP96a]. The vector median filters are invariant to scale and bias. This means that,

$$VM\left[ a\underline{u}(1) + \underline{c}, \ldots\ldots, a\underline{u}(k) + \underline{c} \right] = a\, VM\left[ \underline{u}(1), \ldots\ldots, \underline{u}(k) \right] + \underline{c} \tag{3.23}$$

Here, $a$ is a scalar and $\underline{c}$ is a vector constant. As a rotation of the coordinate system does not affect the $L_2$ norm of a vector, the vector median filter based on this norm is also rotation invariant. The above mentioned property is similar to the property of scalar median, and if the vector dimension is one, the vector median definition reduce to the scalar median definition [AST90a].

### 3.3.4. Rank order-based filters

***a) Definition***

Ranked-order filter is defined as, an $r$-th ranked order filter of the $u(i)$ is the $r$-th order statistics of the signal data within the filter window. Now, consider the real-valued two-

dimensional sequence $\{\underline{u}(n_1,n_2)\}$, and define $\underline{w}(n_1,n_2)\in W$ as a $K$-element observation vector that contains the elements of an $M\times N$ window $W$ centered on $u(n_1,n_2)$ such that, $\underline{w}(n_1,n_2)=\left[u(1),....,u(K)\right]$. In this case, the $u(k)$'s correspond to a left to right, top to bottom mapping from $M\times N$ window to one dimension vector $K = M\times N$. The observation samples can be also ordered by rank, which defines the vector $\underline{r}(n_1,n_2)=\left[r(1),....,r(K)\right]$, where $r(1),....,r(K)$ are the elements of $\underline{w}(n_1,n_2)$ arranged in ascending order such that $r(K) \geq r(K\text{-}1).....\geq r(1)$. The median filter, previously discussed, is a particular case of ranked-order filters [BOV83a], which is defined as

$$y(n_1,n_2) = r((K+1)/2), \tag{3.24}$$

where, $y(n_1,n_2)$ is the output of rank-order filter. Rank order mean filter [ABR96a] is used without including the central pixel in the filter operation, and the output of the filter is defined as,

$$y(n_1,n_2)=(r((K\text{-}1)/2)+r((K+1)/2))/2 \tag{3.25}$$

Rank order mean filter is shown to be more robust than the simple median filter because the window on which it operates does not include the corrupted pixel. Maximum filter also can be defined as $y(n_1,n_2) = r(K)$. Rank order filters are closely related to morphological *dilation* and *erosion*. Such filters are proven to be consistent and biased estimators of the received signal distributions. Threshold decomposition uses such a type of these filters [AMI92a].

***b) α-trimmed mean filters***

The moving average filter suppresses additive white Gaussian noise better than rank order filter, whereas the second is better at preserving edges and rejection of impulses. A good compromise between the two is achieved using the α-trimmed mean filters [KIN89a, PIT90a]. This filter is defined as,

$$y(n_1,n_2) = \frac{1}{(2\alpha+1)} \sum_{j=((K+1)/2)-\alpha}^{((K+1)/2)+\alpha} u(j), \qquad \alpha = 0,1,2,\ldots,\frac{K+1}{2}-1 \tag{3.26}$$

The α-trimmed means filter rejects the smaller and larger samples. Data rejection depends on the coefficient $\alpha$. At $\alpha = 0$, the filter output is the median value.

***c) L-filters***

These filters are based on the theory of robust *L*-estimators. They are linear combinations of order statistics:

$$y(n_1, n_2) = \sum_{j=1}^{K} a(j)\, u(j) \tag{3.27}$$

The moving average, median, *r*-th ranked-order, α-trimmed mean filters are special cases of *L*-filters when appropriate sets of coefficients $a(j)$ , $j$=1, ..., $K$ are used. Also, these coefficients can be chosen such as for the filter to satisfy an optimization criterion for the probability distribution of the input noise [DOM94a, PIT90a]. The filter output could be a median value when,

$$a(j) = \begin{cases} 1 & when \quad j = (K-1)/2 \\ 0 & otherwise \end{cases} \tag{3.28}$$

***d) Statistical mean filter***

The class of the nonlinear statistical mean filters [JAI89a, PIT88a] is obtained from (3.8) by choosing,

$$f(u) = g^{-1}(u) \quad \text{and} \quad a(k) = a(1), \ldots, a(K) \tag{3.29}$$

If the weight $b(k)$ are constant, these filters are equivalent to the homomorphic filters. The following nonlinear mean filters are obtained for different choices of the nonlinear functions $g(\cdot), f(\cdot)$, that

$$g(u) = \begin{cases} 1/u & \text{harmonic mean filter} \\ \log u & \text{geometric mean filter} \\ u^p & L_p \text{ mean filter} \end{cases} \tag{3.30}$$

The contra-harmonic statistical mean filter is a special case of (3.29) for the following choice of $b(i)$,

$$b(i) = \frac{u_i^p}{\sum_{i=1}^{K} u_i^p} \tag{3.31}$$

***e) R-filters***

R-filters are based on another large class of robust estimators, the so-called R-estimators. The most important R-filter is defined as [KIN89a, PIT90a]:

$$y(n_1, n_2) = med\left\{\frac{u(j) + u(k)}{2}, \qquad 1 \le j \le k \le n, \quad k - j < D\right\} \tag{3.32}$$

where, $D$ is ranged from 1 to $k$.

### 3.3.5. Threshold decomposition based-filters

The fundamental difference between threshold decomposition filter and other nonlinear filter classes is that the signal space is decomposed before filtering operations. This is called threshold decomposition and it maps the $M$-level valued into $M$-1 binary signals by thresholding the original signal at each of the allowable levels. The set $M$-1 signals is then filtered by $M$-1 Boolean operators that are constrained to have stacking property attributes. The multi-level output is finally obtained as the sum of the $M$-1 binary output signals [COY91a, YOO99a].

The threshold decomposition of an $M$-valued signal $U(n)$, where the samples are non-negative integers, $0 \le U(n) < M$, means decomposing $U(n)$ into $M$-1 binary signals, $u^1(n)$,.....,$u^{M-1}(n)$, according to the following rule:

$$u^m(n) = T^m(U(n)) = \begin{cases} 1 & if\ U(n) \ge m \\ 0 & otherwise \end{cases} \tag{3.33}$$

It is important to note that this threshold scheme can also be applied to non-integer signals or, in general, to all signals that are quantified to a finite number of arbitrary levels. The original multi-valued signal can be reconstructed from its binary signals

$$U(n) = \sum_{m=1}^{M-1} u^m(n) \tag{3.34}$$

#### *a) Stack filters*

The concept of threshold decomposition and binary filtering has led to a general class of lowpass type filters, called stack filters [COY91a]:

$$U(n) = \sum_{m=1}^{M-1} f(u^m(n)) \tag{3.35}$$

where, $f(\cdot)$ is the Boolean function at each level of the architecture. Only Boolean function possessing the stacking property can be used in stack filtering.

The stacking property ensures that the output is one of the input samples. The necessary and sufficient condition for a binary function to possess the stacking property is that it can be expressed as a Boolean expression that contains no complements of input variables. Stack filters can be implemented either using the threshold decomposition architecture in the binary domain or using a real domain architecture, which is based on MAX/MIN operations [COY91a, YOO99a]. The real domain stack filter can be expressed by replacing Boolean AND and OR operations by MIN and MAX operations, respectively. For example, the three points median over integer variables $U_1$, $U_2$, and $U_3$, is a stack filter by equation (3.34) corresponding to equation (3.35).

$$f(U_1, U_2, U_3) = U_1U_2 + U_1U_3 + U_2U_3, \tag{3.36}$$

$$MED[U_1, U_2, U_3] = MAX[MIN[U_1, U_3], MIN[U_2, U_3]] \tag{3.37}$$

***b) Mathematical morphology filters***

Mathematical morphology was developed at the beginning for binary images. One approach is by using threshold decomposition. Gray-level operators are computed by using the binary ones. Morphological operators are essentially special cases of stack filters. Two well-known classes of morphological filters are opening and closing. The morphological opening is the cascade of erosion and dilation and similarly the morphological closing is the cascade of dilation and erosion [COM92a, NIE98a]. Opening and closing smooth in a nonlinear way. An image f(u) is a function defined on $D \subset R^2$ with values in R. In the digital setting, D is a subset of the Cartesian grid and the gray values lie in the integer range [0,n]. The structuring function g(n) is also an image defined on $G \subset D$. The symmetric structuring function with respect to the origin is given by $g^s(u) = g(-u)$. The gray-scale dilation and erosion of signal f(u) by the structuring function g(u) are in (3.36) and (3.37). Furthermore, (3.38) and (3.39) give grayscale opening and closing.

$$[f \oplus g^s](u) = \max_{z \in D, z-x \in G} \{f(z) + g(z-u)\} \tag{3.38}$$

$$\left[f \Theta g^{s}\right](u) = \min_{z \in D, z-x \in G} \{f(u) - g(z-u)\} \tag{3.39}$$

$$f_g(u) = [(f \Theta g^{s}) \oplus g](u) \tag{3.40}$$

$$f_g(u) = [(f \oplus g^{s}) \Theta g](u) \tag{3.41}$$

***b) Median filters and threshold decomposition***

A very important property of median filters that applying a median filter to an *M*-valued signal. It is equivalent to decomposing the signal to *M*-1 binary components, then filtering these components separately with the corresponding binary signals together using equation (3.10). That is

$$Y(n) = MED[U(1), \ldots\ldots, U(n)] = \sum_{m=1}^{M-1} MED[u^m(1), \ldots\ldots, u^m(n)] \tag{3.42}$$

The importance of this property arises from the fact that binary signals are much easier to analyze than multi-valued signals. The median operation on binary samples reduces to a simple Boolean operation [COY91a].

### 3.3.6. Other examples of nonlinear filters

***a) Polynomial filters***

The filters based on Volterra and Wiener representation are frequently called polynomial filters [RAM88a]. Their input-output relation is given by:

$$y(n_1, n_2) = h_0 + \overline{h}_1[u(n_1, n_2)] + \overline{h}_2[u(n_1, n_2)] \tag{3.43}$$

where, the linear part of the nonlinear filter is,

$$\overline{h}_1[u(n_1, n_2)] = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} h_1(i, j) \cdot u(n_1 - i, n_2 - j) \tag{3.44}$$

and the quadratic Volterra operator is

$$\bar{h}_2[u(n_1,n_2)] = \sum_{i_1=0}^{M-1}\sum_{i_2=0}^{N-1}\sum_{j_1=0}^{M-1}\sum_{j_2=0}^{N-1} h_2(i_1,i_2,j_1,j_2) \cdot u(n_1-i_1,n_2-i_2)\cdot u(n_1-j_1,n_2-j_2) \tag{3.45}$$

This model is widely used in image processing application. In a recursive term this formula is represented by the inclusion of the linear y terms to the model of the first filter in equation (3.43),

$$y(n_1,n_2) = h_0 + \bar{h}_1[u(n_1,n_2)] + \bar{h}_2[u(n_1,n_2)] + \bar{h}_3[y'(n_1,n_2)] \tag{3.46}$$

where,

$$\bar{h}_3[y'(n_1,n_2)] = \sum_{i=1}^{M'-1}\sum_{j=1}^{N'-1} h_1'(i,j)y(n_1-i,n_2-j) \tag{3.47}$$

In this models $h_0$, $h_1(i,j)$, $h_2(i_1,i_2,j_1,j_2)$, and $h'_1(i,j)$ are the Volterra kernels, $M\times N$ and $M'\times N'$ are the input and output filter support regions respectively. The input image size is $N_1\times N_2$, $n_1$=0, ..., $N_1$-1, $n_2$=0, ..., $N_2$-1.

However, the design of two and multi-dimensional recursive filters is difficult due to the fact that polynomial in two or more variables may not be factored into lower order polynomials. This difficulty can be partly circumvented by designing two and multi-dimensional systems as one-dimensional system. Several applications are presented to simplify the technique either in the calculation of the kernels [CHA77a, RAM88a, SUN94a] or to achieve a good representation of the filter stability [DOM92a, DOM96a, TSA88a].

Polynomial filters are of interest in many cases where linear filters are known to be sub-optimal. They have been used in nonlinear system modeling, detection and estimation problems, in the equalization of nonlinear channels, for compensating typical nonlinearities that appear in the echo path and effect adaptive echo cancellers, prediction. Polynomial filters have also many applications in image and image sequences processing like: image enhancements, edge extraction, texture analysis, nonlinear image and image sequence prediction [RAM88a].

***b) Homomorphic filters***

Homomorphic filters are a well-known class of nonlinear filters that copes with non-additive combined signals [KIN89a]. They form one of the oldest nonlinear digital filter

classes. Their basic idea is to map the signals from the input space into space where they are additives combined, to apply a linear filter and to map the result back to the original space. Let us denote by $*$ a non-additive operation and let us consider two signals $u_1$, $u_2$. The signal x to be processed is $u(n)=u_1(n) * u_2(n)$. A nonlinear operator $D$ has to found such as:

$$D(u_1(n) * u_2(n)) = D(u_1(n)) + D(u_2(n)) \tag{3.48}$$

Using a linear filter $L$, (3.49) is obtained. The output of the linear filter is transferred afterwards by the inverse nonlinearity using (3.50).

$$L[\, D(u_1(n)) + D(u_2(n))\,] = y_1(n) + y_2(n) \tag{3.49}$$

$$D^{-1}[y_1(n) + y_2(n)] = D(u_1(n)) + D(u_2(n)) \tag{3.50}$$

Thus, the homomorphic systems satisfy the so-called generalized superposition. Homomorphic filters are of great interest for multiplicative signals. Nonlinear mean filters can also be considered to be special cases of homomorphic systems [KIN89a, PIT90a].

### 3.3.7. Recursive nonlinear filters

Another simple extension of the filter is the recursive filters. The recursive filter of 3×3 window is defined by replacing some of the input samples in the window by previously derived output samples as follow:

$$\begin{aligned} y(n_1,n_2) = MED[&y(n_1 - 1,n_2 - 1),\ y(n_1 - 1,n_2),\ y(n_1 - 1,n_2 + 1),\ y(n_1,n_2 - 1),\\ &u(n_1,n_2),\ u(n_1,n_2 + 1),\ u(n_1 + 1,n_2 - 1),\ u(n_1 + 1,n_2),\ u(n_1 + 1,n_2 + 1)] \end{aligned} \tag{3.51}$$

In recursive filtering, the filtering operation is performed "in place" so that the output of the filter replaces the old input value before the filter window is moved to the next position. With the same amount of operations, the recursive filter usually provides better smoothing than the non-recursive filter, at the expense of increased distortion [SUN91a, SUN95b].

In our proposed recursive approach the filter passes over the image only once. Each time an output value is calculated it is written back to the image. This means that, as the filter passes over the input image it is simultaneously operating on old and new information [BAR97a].

Many types of nonlinear filters have been already proposed and examined. Most of them were non-recursive. Such filters are inherently stable and easy to design, but recursive structures are more efficient from the point of view of computational complexity related to their implementations [BAU91a, MAC92a, TSA88a]. The concept of passive digital systems showed its usefulness in solving stability problems for recursive systems.

The concept of passivity of multidimensional digital systems has been generalized by formulating the theory of generalized passivity. The important property is that $l_1$-passivity implies stability for two-dimensional systems.

The previous papers on $l_1$-passive filters described some basic building blocks of such linear filters processing real-valued two-dimensional signals. The filter built of these wide classes of blocks is stable.

We demonstrate how to use this theory in order to simplify stability checking. The theory gives sufficient but not necessary conditions for stability.

*DEFINITION*: Consider the vectors $\underline{a}(\underline{n})$ and $\underline{b}(\underline{n})$ of the inputs and outputs of the shift-free network. We define their measures as,

$$M\,[\underline{a}(\underline{n})] = g_1|a_1(\underline{n})| + g_2|a_2(\underline{n})| + \ldots + g_K|a_K(\underline{n})| \;,$$

$$M\,[\underline{b}(\underline{n})] = g_1|b_1(\underline{n})| + g_2|b_2(\underline{n})| + \ldots + g_K|b_K(\underline{n})| \;. \tag{3.52}$$

where, $g_i$ (i =1,..., $K$) are positive weighting coefficients.

$l_p$-power absorbed in a digital shift-free multi-port at a point $\underline{n}$ is

$$w\,(\underline{n}) = M\,[\underline{a}(\underline{n})] - M\,[\underline{b}(\underline{n})]. \tag{3.53}$$

It is straightforward that classic median, dilation and erosion filters are non-recursive nonlinear $l_1$-passive digital filters while recursive median filters are $l_1$-passive recursive filters. The considerations of linear $l_1$-passive digital filter have shown that the class of systems processing non-negative-valued signals is of particular interest. Only in this class $l_1$-losslessness can be defined. Fortunately, it is the class of signals which includes all image and video signals being inherently non-negative-valued [DOM92a, DOM94a, DOM96a].

## 3.4. Nonlinear three-dimensional filters

Many applications in image processing require the processing of three-dimensional signals; namely image sequences [VIE94a]. Television applications, target tracking, robot navigation, dynamic monitoring of industrial process, study of cell motion by microcinematography, highway traffic monitoring, and video transmission are only a few examples where the signal to be processed is three-dimensional, the third dimension being time. It has been shown in many cases that one-dimensional algorithms do not produce optimum results in image processing. In other words, while processing images, their two-dimensional nature has to be taken into account. Likewise, in processing image sequences, one- and two-dimensional algorithms do not yield optimum results. Although similar in some respects, the extension of two-dimensional to three-dimensional signals is not straightforward. The motion content of the image sequence requires the time dimension to be approached in a different manner [ALP91a].

Temporal filters have been developed to make use of the information in the time dimension in many image processing problems. However, temporal filters usually blur the moving parts of the image sequence, resulting in poor image quality. It is known that two-dimensional spatial processing gives better results in still parts of the image sequence. This observation leads to the development of adaptive algorithms that require motion-estimation or motion-compensation to obtain acceptable image quality. However, motion-estimation and motion-compensation are critical processes. Therefore, it is highly desirable to have three-dimensional filters, which would be insensitive to motion in image sequence filters [DUB93b].

In image processing, median filters preserve edges and high frequency details in the image, resulting in improved image quality. In this dissertation, three-dimensional algorithm is presented, that is insensitive to the motion content of the image sequences. Color image processing presents good performance of median filters in presence of impulse noise. The growing number of their applications in image enhancement, reconstruction, and restoration has stimulated intensive sites on vector median filters [DUB93a, DUB94a].

The vector image processing of color images consists in simultaneous processing of all the three signal components. There exist a variety of distance measures for calculation of vector median. First of all various norms in the three-dimensional vector spaces can be used. The Euclidean norm is of the greatest interest for it is easy to calculate. The Euclidean norm can be calculated in various color spaces. Considered is the problem of the

best choice of the color space for vector median calculations. The choice of the color coordinate system should be considered to take the maximum advantage of the vector approach. On the other hand, in order to obtain minimum visible distortion, the color distance measure should well agree with the human perception rules [BAR95a, AST90a].

### 3.4.1. Fundamentals of three-dimensional filters

Image sequence filtering finds applications in picture phones, videoconference, robot vision, and various parts of the television transmission chain. Even though the trend is towards digital processing, at least the imaging device remains analog in nature and thus prone to noise. In fact, the trend to higher definition corresponding to smaller feature sizes in image sensors tends to increase the noise content. This calls for image sequence filtering methods that can be used to reduce the noise level without causing visible artifacts [ABB99c, DUB93a].

Color image sequences are multi-spectral three-dimensional signals, which typically consist of three color components. The high dimensional multi-spectral image sequence offers great design flexibility in the design of filters for noise reduction. However, one and two-dimensional filters have traditionally been used in image sequence enhancement due to limited computational and memory resources. Advances in integrated circuit technology decreased these limitations [BUB93a, VIE94a]. From this point of view, image filtering and noise reduction can be divided into two main categories:

- **Temporal filters** remove noise without impairing the spatial resolution in stationary areas. However, temporal filtering distorts moving objects and therefore the filtering action should be inhibited in moving regions.
- **Motion-compensated filters** estimate the direction of motion at each pixel and perform the filtering along the motion trajectory. Motion vectors indicate motion direction. The performance of this filter class is highly dependent on the accuracy of the motion estimation and computationally high, but it leads to good results.

### 3.4.2. Motion vectors

Sampling structure conversion and noise are two examples of image sequence processing that can benefit greatly from the knowledge of motion. In the case of conversion, two scenarios are possible. In one situation a missing image must be

recovered. Again, due to the high correlation along motion trajectories, motion-compensated interpolation is the most effective tool. In the other situation, only part of an image must be reconstructed. Since some spatial information is available and since motion estimates are occasionally unreliable, methods can be devised which combine *smart* spatial interpolation with motion compensated interpolation and filtering. Noise reduction also benefits from high correlation along the motion trajectories [DUB93a, DUB93b].

One remark is in order at this point. Optimal motion fields, in the sense of final image quality, are not necessary identical for motion-compensated predictive coding and for motion-compensated interpolation and filtering. This is due to the fact that the sole task of a predictor is to find the best match of a pixel value given previous images. To find this match any motion trajectory providing minimum prediction error is appreciated. This trajectory does not have to correspond to the real 2-D motion of objects in the image. On the contrary, the task of a temporal filter is to calculate a pixel value for which the continuity and smoothness of object motion are preserved. Thus, an estimate of the real 2-D motion is needed. It is a different situation, however, when motion field needs to be transmitted along with the prediction error. Optimal motion field for prediction requires relatively high bit rates. To reduce the bit rate in this case, joint optimization of prediction error and of motion correlation can be carried out. Interestingly, since real 2-D motion is usually quite smooth (except for motion discontinuities), it is often estimated by combining a prediction-like error with smoothness constraint imposed on the estimated motion field. Thus, methods used to estimate 2-D motion for the purpose of temporal filtering are also applicable to predictive coding with simultaneous transmission of motion data [DUB93a, LEU97a].

#### *3.4.2.1. Motion vectors representation*

Motion can be represented in different ways. The most common approach is to specify the displacements as scene points between image frames [TEK95a]. Thus the displacement field $\underline{d}(\tau;u,t)$ specifies that an image point $u$ located at spatial position $\underline{n}$ at time $t$ was located at position $\underline{n}-\underline{d}(\tau;u,t)$ at time $\tau < \mathrm{t}$. In general, $\underline{d}(\tau;u,t)$ is specified on a dense set of points $u$ at time $t$, denoted $\left(\Lambda_{\mathrm{d}}\right)_{\mathrm{t}}$. This can be achieved by explicitly determining $\underline{d}(\tau;u,t)$ at each point $u$, or implicitly through models such as constant or affinity functions on block [DUB93b].

The motion vector field obtained from the *MSE* selection can be improved by smoothing and segmenting in order to reduce block artifacts in the interpolated picture [ALP96a, DES93a].

### *3.4.2.2. Frame interpolation*

Vector filtering considers that the pixel is a vector of three components. In video processing, a different resolution between the luminance and the chrominance components has to be considered. Interpolation is used to the chrominance components to be able to be used in vector processing. Many conventional interpolation techniques have been used to increase the spatial resolution of an image. Some of these include pixel duplication, linear and bilinear interpolation. These techniques perform poorly in a subjective sense and tend to cause blurring or artifacts. On the other hand, when the degraded image has to be interpolated, the noise is also duplicated. Order statistic filters perform well in solving of this problem; median and vector median filters being good examples [HER95a]. These kinds of filters are highly desirable for color image interpolation due to their retention of edge information. In this case, the interpolated area will be with less degradation in presence of noise. The interpolation procedure is as follows,

$Z(1) = VM[U(1), U(2), U(3), U(4)]$,
$Y(1) = VM[U(1), U(2), Z(1), Z(2)]$,
$Y(2) = VM[U(1), U(3), Z(1), Z(3)]$,

$$\begin{array}{ccccc} & & Z(2) & & \\ & U(1) & Y(1) & U(2) & \\ Z(3) & Y(2) & Z(1) & Y(3) & \\ & U(3) & Y(4) & U(4) & \end{array}$$

### *3.4.2.3. Vector field smoothing*

The adopted strategy is already used to smooth the vector field and to try at the same time to eliminate incorrect decisions [DES93a]. Although, median filtering is the most commonly used approach [ALP96a], it cannot be applied simultaneously to the two components of the motion vectors. In order to obtain a smooth vector field, both vector

components are processed in the same time by vector filtering. Smoothing is applied to the motion vectors to get results with less image degradation and artifacts. Here recursive vector median filter is applied to the motion vectors. The vector median operation has been found well suited for the motion vector processing. The vector median *VM* of the set $v_1$, $v_2$, ..., $v_k$ is defined as $v_{VM}$, such that

$$\sum_{i=1}^{K} \left\| v_{VM} - v_i \right\| \le \sum_{i=1}^{K} \left\| v_j - v_i \right\|, \tag{3.54}$$

where, $v_{VM} \in \left\{ v_i \mid i = 1,2,.....,K \right\}$ *for all* $j = 1,2,\ldots,K$, and *K* is the number of samples in $M \times N$ ($K = M \times N$) window.

With Euclidean distance $\left\| \cdot \right\|$ in (4.54), *VM* can be defined using the $L_1$ or $L_2$ norms. The first of them was selected for its better performance and feasible implementation [ALP96a, DUB94a, DES93a].

#### *3.4.2.4. Motion vector refinement*

The vector field smoothing operation is not sufficient by itself, as it introduces occasionally new artifacts to the interpolated picture. As a countermeasure, block segmentation has to be used. In area of connecting motion, the vector field is further spatially segmented to obtain better resolution. Without any a priori knowledge of the picture content, a fairly threshold is adopted. There are two major procedures used for motion vector segmentation, as shown below:

- The motion estimation procedure minimizes the *MSE* among all the blocks in the searching area. The advantage could be used that the *MSE* should not exceed a certain threshold $\alpha$ otherwise this estimation is considered a false one and filtering operation has to be held to these vectors:

$$Segm(n_1, n_2) = \begin{cases} true & if \quad \min(MSE) < \alpha \\ false & otherwise \end{cases} \tag{3.55}$$

- Segmentation is only performed when any of the vector components of the estimated vector deviates somewhat from the component of the neighboring vectors. A spatial $M \times N$ square window is used:

$$Segm(n_1, n_2) = \begin{cases} true & if \quad |v_u - v_{um}| < i \\ & or \quad |v_y - v_{yn}| < i \\ false & otherwise \end{cases} \qquad (3.56)$$

where, m and n are the block indices. ($v_{um}$ , $v_{yn}$) are the displacement vector of the center block in the processing window, and ($v_u$ , $v_y$) are the displacement vectors of the surrounding neighboring blocks [ABB99d, ALP96a, DUB94a].

### 3.4.3. Motion estimation

Motion estimation plays an important role in video compression. Therefore, very many results have been already reported for motion estimation. The most popular way for motion estimation is block-matching method. The basic idea is to divide an image into blocks and to search for the best matching block in a certain area of the last previous or the next future frame. When the searching area is located in the past frame we refer to it as *backward motion estimation*, and when the searching area is located in the future frame the technique is called *forward motion estimation*. Estimated motion vectors are used to compensate motion in individual frames. It is very important to calculate motion vectors in order to estimate the right pixel locations in the past and future frames [ABB99c, ABB99d, TEK95a].

#### *3.4.3.1. Estimation method*

The most popular way for motion estimation is the block-matching method. The block matching algorithm is also used in the MPEG-1, MPEG-2 and H.261-263 video compression standards. The basic idea is to divide an image into blocks and to search for the best matching of each block in a certain area of the last previous or the next future frame. Block matching methods estimate motion vectors in certain points only otherwise the task would be too much time-consuming. So we deal with a grid where motion vectors are estimated. As the density of the grid grows, motion field is estimated more accurately but the estimator comprises more operations. For filters, density of this grid is a tradeoff between motion estimation accuracy and implementation complexity. So, the conclusion is that motion field should be estimated on a grid whose density is limited only by implementation factors. This situation is different in video compression where also the

number of motion vectors must be kept small otherwise much data is needed to encode motion vectors.

Another important factor, which determines motion estimation accuracy, is search area. Again, large search areas allow to find motion vectors even by very fast motion. Unfortunately, the number of block pairs that must be checked grows as the search area is enlarged [ABB99c, ABB99d, DUB93a, TEK95a].

#### *3.4.3.2. Matching criteria*

Block matching is the most time consuming of the filter process. During block matching each target block of the current frame is compared with past and future frames in order to find a matching block. When the receiver reconstructs the current frame this matching block is used as a substitute for the block from the current frame.

Block matching takes place only on the luminance component of frames. The color components of the block are included when coding the frame but they are not usually used when evaluating the appropriateness of potential substitutes or candidate blocks. Motion vectors are used to indicate how much the estimated block differs from the current block.

The motion vector $[h,v]$ is the one that minimizes certain optimization index, for example the mean absolute difference in the searching area $A$. For the case of forward motion estimation, this mean absolute difference $MAD$ is defined as

$$MAD(h,v) = \frac{1}{M \cdot N} \sum_{m,n \in B} \left| u(m,n,k) - u(m+h,n+v,k+1) \right| \quad h,v \in A, \tag{3.57}$$

where, B is the current block, e.g. of size 8 × 8. Adoption of the above definition to the backward estimation is straightforward (see Figure 3.2).

Sometimes, for more accurate calculation the mean square difference $MSD$ criterion is used. The $MSD$ function is similar to the $MAD$ function, except that the difference is squared before summation. For forward motion estimation, $MSD$ is defined by

$$MSD(h,v) = \frac{1}{M \cdot N} \sum_{m,n \in B} \left( u(m,n,k) - u(m+h,n+v,k+1) \right)^2 \quad h,v \in A \tag{3.58}$$

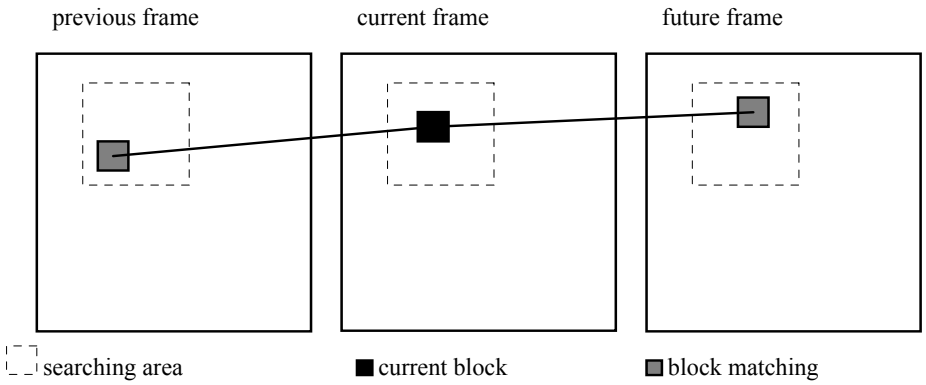



Figure 3.2. Principle of motion estimation.

The *MSD* is more commonly called the mean square error *MSE*. The matching criteria depend on the minimum of certain calculated *MSE* values. This criterion is said to result in better matches than the *MAD* criterion although it is too slight to be perceived.

In some cases, the Pel-Difference Classification *PDC* distortion function is used. The *PDC* function compares each pixel of the target block with its counterpart in the candidate block and classifies each pixel pair as either matching or not. Pixels are matching if the difference between their values is less than some threshold. This function is not considered in the dissertation because it needs to determine the threshold value, which increase the algorithms of the filter structure. Nevertheless, the filtering operation takes most of the hints of the dissertation.

Full search consists in calculation of *MAD* or *MSD* at all pixels inside the searching area *A*. Such a strategy leads to an optimal solution but is extremely time-consuming. Therefore a commonly employed practice to lower the computational load is the fastest approach such as three-step search, cross search, etc. The experiments that reported in this work used the *MSD* criterion [ABB99c, TEK95a].

Unfortunately, motion estimation is a task that needs extremely much computational effort. Therefore filters without motion compensation are much simpler to implement but they are not able to reach so high performance as motion-compensated filters. First of all, rapid motion of objects in processed image results in worse denoising efficiency if it is not compensated [ABB99c]. Therefore only motion-compensated three-dimensional filters will be considered in the further parts of this work.

### 3.4.4. Motion-compensated filters

Two-dimensional filters are appropriate for processing of still images. They can be also used for processing of video sequences but they do not exploit temporal dependencies between consecutive images in a sequence. Therefore, we are going to consider three-dimensional filters that will adopt the ideas just presented for two-dimensional filters.

Three-dimensional filters calculate an output pixel value as a function of the neighboring pixels in the current frame and the pixels from the last previous frame (or frames) and the next future frame (or frames). Two kinds of such filters have been presented: with motion compensation and without motion compensation.

Motion-compensated filters need a motion estimator to be implemented. Estimated motion vectors are used to compensate motion in individual frames. Unfortunately, motion estimation is a task that needs extremely much computational effort. Filters without motion compensation are much simpler to implement but they are not able to reach so high performance as motion-compensated filters. First of all, rapid motion of objects in processed image results in worse denoising efficiency if it is not compensated [ABB99c, TEK95a]. Therefore only motion-compensated filters will be considered in the further parts of the paper. The basic idea of motion compensation is to use pixels shifted according to motion vectors (Figure 3.3).

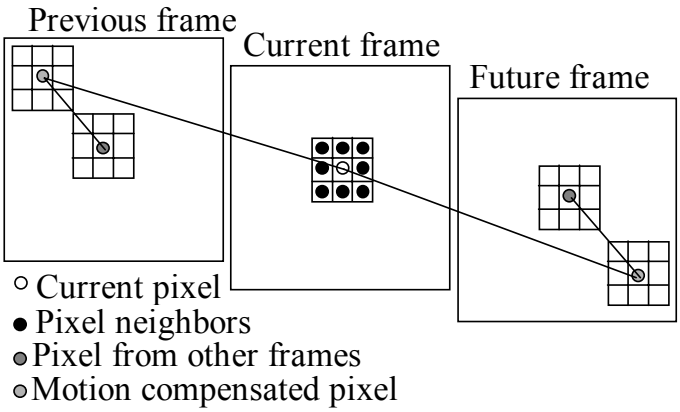



Figure 3.3. Support of a motion-compensated three-dimensional filter.

## 3.5. Recent developments in nonlinear filters for image and video restoration

Nonlinear filters have been proven to be exceptionally useful in many signal and image restoration applications. In particular, rank order based filters are well known for their ability to successfully treat heavy tailed noise and non-stationary signals. The common occurrence of such signals, and the poor performance of linear filters operating on them, have motivated the development of rank order filters. The first, and most well known, of these rank order based filters is the median filter [SUN91a]. Since its introduction, the median filter has been extensively studied. Building on the success of the median filter, many more sophisticated rank order filters have been proposed. These include multi-stage median filters [YAN94a], center weighted median [SUN91a, YAN94a], general weighted median [SUN92a], stack filters, weighted order statistics filters [AMI92a, RAM97a], Fuzzy filters, morphological filters [GAS98a], and conditional rank selection generation of filters [HAR95a].

All the above filters can be formulated as rank selection filters, since their output is constrained to be one of the order statistics from the observation set. However, they differ in the information that they use to perform the selection operation. Fuzzy filter and the conditional rank selection filter use the ranks of the input samples as the bases for output rank selection. These filters are highly effective as smoothers. However, they are not suited to perform edge enhancement. Thus, they do not help to identify on which side of an edge lies the observation. Consequently, different rank selection cannot be made on each side of an edge to yield gradient enhancement.

For these reasons, the application of rank order based filters to edge enhancement has received limited attention. The key point is to prove that such filters are able to remove impulse noise while preserving edges, small details, and fine textures. However, some edge preserving rank order filters have been proposed. These include the comparison and selection *CS* filter [BAR97a, SAW96a], the lower-upper-middle *LUM* filter [HAR95A], and the weighted majority of samples with minimum rank *WMMR* filter [AMI92A, RAM97A, SUN92a]. The *CS* filters decrease the filtering operation to be applied just to the corrupted pixels by applying a certain type of noise detector, and *LUM* filters utilize a mean estimate to aid in rank selection. In particular, the observation sample mean is compared to a specified sample within the observation window to determine on which side of an edge lies the filter window. In a somewhat similar fashion, *WMMR* filters use rank

ranges to delineate different regions of an edge. Having partitioned the edge into different regions, an appropriate output sample is chosen in each region to increase the edge gradient.

The most important type of the above mentioned filters is the *CS* filters. The *CS* is a simple expression for wide range applications for the *decision-based filters.* The assumption of decision-based filter is that processing does not change those pixels, which are presumably uncorrupted by noise. The idea is to use a nonlinear filter as a predictor and to use the prediction error in order to control the output of the system. An output value is either the input pixel itself or the nonlinear filter output or a linear combination of both. Decision-based filter depends on the following principles:

- ***Prediction***; a predictor plays a very important role as in filter processing. Some authors use the same principles of decision-based filters as in *CS* filters but a linear filter is used as a predictor [KIM95a, MAC92a, SAW97a, SAW96a, TSA88a, TSA88b]. In such approaches, image details are preserved but impulses are not well removed due to the detection failure or introduction of residual error when replacing a pixel with an estimated value. Referring to chapter 2 (section 2.3), nonlinear filter as a predictor has good specifications in the final output. In [ABR96a, ABB99d, LEE98a], a median-based filters are used as predictors.
- ***Decision and Estimation***; another important principle is the decision-making to estimate the filter output. The simple role of decision-based filter is shown in [SUC94a] with hard-decision. The hard-decision based roles suffer from a hard detection of noise when the value of impulse is very near to the pixel value in texture area. In [KIM95a, SAW96a, SAW97a], a linear combination of soft-decision is proposed. In the relaxed-median filter [GAR98a], the number of samples inside the widow is controlled by the decision roles. The decision role depends on the variance of the sample data in [PAR99a] as a modification to the filter structure to be applied to remove a Gaussian type of noise.
- ***Threshold***; the decision-based roles depend on a certain threshold value. In literature different techniques to determine the threshold value are presented. The threshold value is either a fixed value [MAC92a, SAW97a, TSA88b], or an adaptive selected from local statistics of the data sample [DUB94a, LEE98a, PAR99a, PLA98a]. The fixed value is not able to deal with all variations of the image details. Adaptive structures can give a

flexible treatment for each image area, flat area and texture area, etc. A small number of samples inside the window which depend on the local statistics can failed in a complex texture areas due to not sufficient information to recognize the noise between all the pixel neighbors.

Prediction error processing filter as a sub-class of decision-based filter is proposed by [ABB00a, ABB00b, ABB00c]. This technique uses the histogram of the prediction errors for the whole image structure [ABB99c, ABB99d]. This approach is more flexible and can examine all changes in the image information. The output image quality has significantly improved as compared to the output of a nonlinear filter without prediction error processing because small details and fine textures are much better reproduced.

In video processing, all the above mentioned methods are modified to be capable to deal with the sequence structure and get the advantage of the available information from the other image frames [ABB99d, DUB93b, TEK95a]. The presence of multi-frames could cause the loss of information concerning the processed corrupted pixel from other frames. The information from other frames are motion-compensated [ABB99c, DUB93b, SAI99a] or without motion-compensation [ALP91a, VIE94a]. Motion-compensated data need a high computational cost but lead to better results.

# 4. Decision-based filters

## 4.1. Main idea of decision-based filters

An important drawback of many filter structures considered in Chapter 3 was related to some degradation of images caused by restoration process itself. It means that a filter, which processes an original uncorrupted image, introduces some distortion. Therefore, it is advantageous to classify individual picture elements as uncorrupted or corrupted. Only latter pixels are processed by the filter, which is aimed at rejection of impulse noise. Therefore, no distortion due to restoration is introduced for the pixels classified as uncorrupted. If all corrupted pixels have been classified properly, all disturbing impulses have a chance to be rejected by the nonlinear filter. Therefore we reduce distortions caused by the filter and simultaneously we do not decrease the efficiency of the restoration process. The conditions for this is that classification of corrupted pixels is made properly, that is all disturbing impulses are related to the pixels classified as corrupted. On the other hand, pixels not related to the disturbing impulses should be classified as uncorrupted. Otherwise they will be processed by the nonlinear filter, which would introduce unnecessary distortions.

The idea described above is not used in classic nonlinear filters that process whole images in the same way. Therefore classic filter process even uncorrupted portions of the images causing unnecessary distortion introduced to this image portion.

In the following chapters, we will show that application of decision-based restoration yields improved efficiency.

A decision-based filter consists of two main parts:

1. Classifier that makes the decision about correctness of individual pixels.
2. Actual restoration filter.

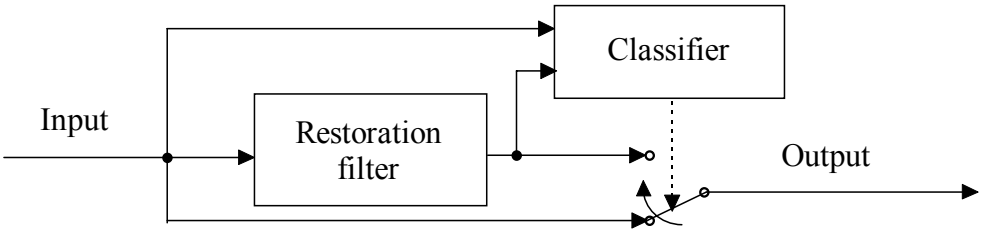



Figure 4.1. Basic structure of decision-based filter.

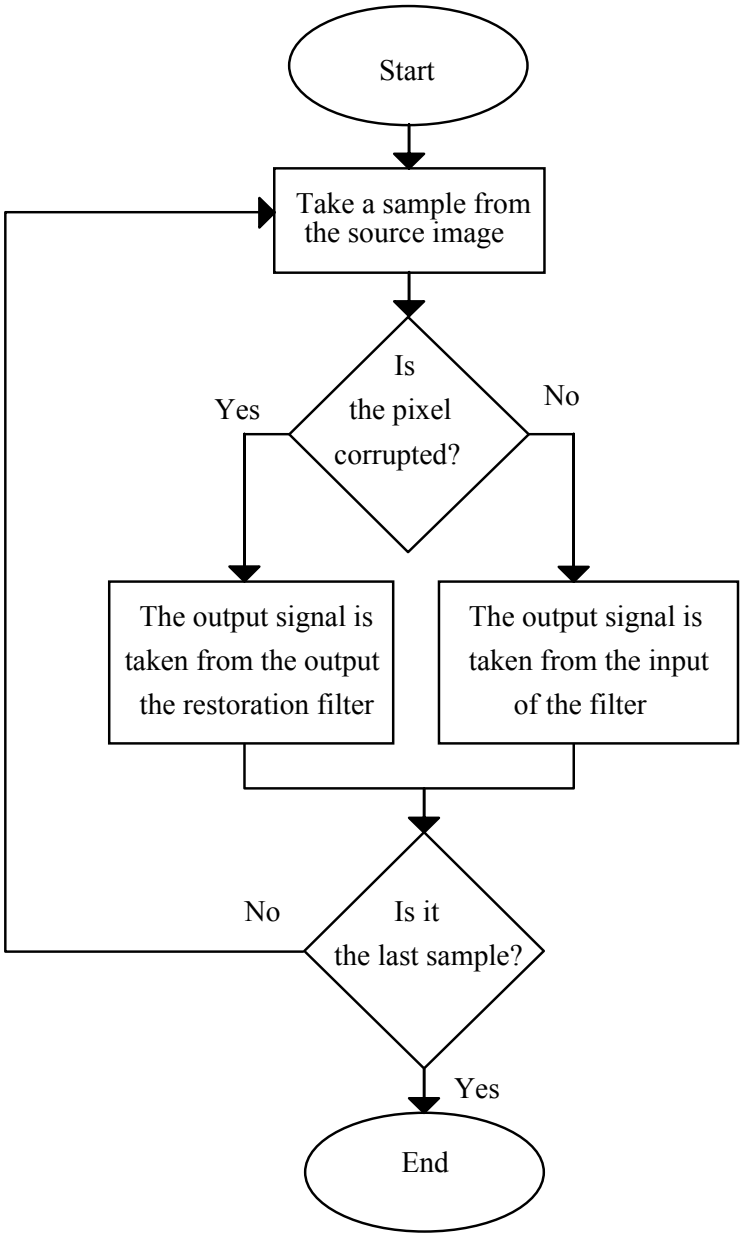



Figure 4.2. The flow-chart of decision-based filter.

The latter is mostly a median-related filter as we consider here impulse noise rejection. The above structure is shown in Fig. 4.1 while Fig. 4.2 describes the flow chart of a decision-based filter.

The classifier decides if the pixel is corrupted or not. This decision is also a decision about whether to replace the input pixel by the result of restoration or not.

The above proposed general model includes some filter types already proposed in [ABR96a, GAR98a, SAW96a, SUC94a, WAN99a] (see Section 3.4).

The classification includes estimation of the component values related to an individual pixel in the original (uncorrupted) image. Estimation of the signal values is possible because images are highly correlated signals and they can be relatively well modeled as Markov processes. In particular, even simple model of a line or column as a first order Markov chain with the correlation coefficient $\rho$ = 0.90 - 0.99 is astonishingly relevant [JAY84a]. On the other hand, location and amplitude of disturbing impulses are mostly white stochastic processes. Therefore signal values can be relatively well estimated even from noisy data.

The estimator predicts the correct value of a pixel using the values of the neighboring pixels in the corrupted image. Therefore this estimator will be called as predictor in order to be compliant with previous papers on one-dimensional signal processing, e.g. [MAC94a, SAW96a, SAW97a].

Classifier categorizes pixels according to an algorithm that exploits:

- the estimated (predicted) uncorrupted value of the current pixel,
- previous estimations (predictions) made for some neighboring pixels,
- input data representing neighboring pixels.

Therefore the classifier consists of:

- predictor (estimator),
- decision unit.

Premises for the decision on categorizing an individual pixel are often uncertain. Therefore soft decisions are applied here.

The predictor can be both linear and nonlinear but the considerations of this work are restricted to applications of nonlinear predictors only. Similarly we assume an application of a nonlinear restoration filter.

## 4.2. Filters with prediction error processing

A special subclass of decision-based filters is filter with prediction error processing. The latter use the same predictor in the classifier and as the restoration filter (Fig. 4.3). Nonlinear filters of various types are used here as predictors. Prediction error $d$ is calculated as a difference between the nonlinear filter output $v$ and the input signal $u$ to the whole structure (Fig. 4.4). The output is a sum of the prediction and the prediction error processed in some way. In the simplest case, large values of prediction errors are set to zero because they are classified as caused by impulse noise samples. Small prediction errors are added to the prediction $v$ in order to obtain the actual output sample $y$.

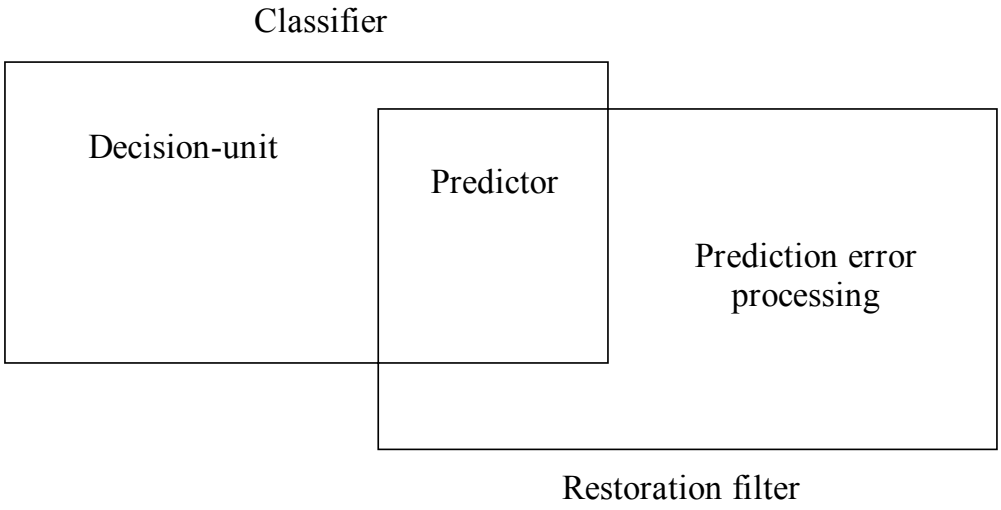



Figure 4.3. The major parts of a filter with prediction error processing.

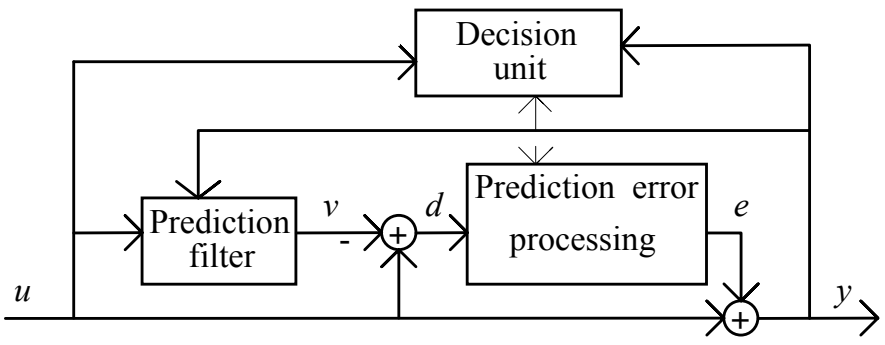



Figure 4.4. Basic structure of filter with prediction error processing.

Nonlinear filter employed as a predictor is either a non-recursive or a recursive. A recursive filter uses pixels from the output of the whole structure.

The fundamental advantage of these filters is good preservation of small details and fine textures. Prediction error processing is performed by use of a multiplier or even a nonlinear filter.

In the simplest case, the decision-unit is a memory-less nonlinear processing unit. The memory-less nonlinear processing of the prediction error is shown in Figure 4.5. The pixels are classified as correct or according to a value of the prediction error $d$. Large prediction errors are related to erroneous pixels while small values of $d$ are related to correct pixels. A soft-decision algorithm implements this role.

The correction $e$ is calculated as $e = k \cdot d$, where the coefficient $k$ is calculated according to a function $f_k(d)$ (see Fig. 4.5). The prediction error is controlled *by* a threshold value $\alpha$. The threshold value could be fixed as a value or even better, it can be auto-adapted by use of a certain technique. In Fig. 4.5, the width of the soft-decision band is set as $\alpha$.

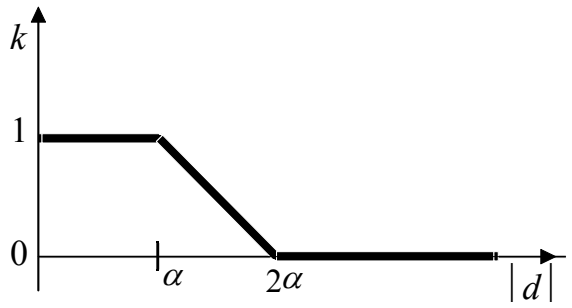



Figure 4.5. Function $f_k(d)$.

## 4.3. Color image processing

### 4.3.1. Scalar and vector prediction filters

In color images, pixels are represented by triples of integers, i.e. a color image is a vector-valued signal. Therefore, prediction for a decision-based filter is done by means of scalar or vector filters.

- ***Scalar prediction filter***

In scalar prediction, each color component of a color image is filtered separately in the same manner as for a gray-scale image. This type of filtering is easy to design, with lower time consumption compared to vector filters. On the other hand, independent scalar filtering of the image components might replace one color component of a pixel, leaving the others unchanged. The result is only a change of color of the noise corrupted pixel rather than its removal.

- ***Vector prediction filter***

Such filters replace a corrupted pixel by one of its neighbors. Therefore, a very nice advantage of these is that they output only colors present in the input images, thus preserving local color properties. But, we have to consider that - even only one component is corrupted - all the components are replaced by components of another pixel from the pixels which are located inside the processing window. Also, the performance of a vector filter depends on the vector norm calculation, which could be carried out in various color spaces. Moreover, we have to choose the proper color space.

### 4.3.2. Calculation of the prediction error

The prediction error $d$ is calculated as a difference between the input value $u(n_1,n_2)$ and the estimated value by the prediction filter $v(n_1,n_2)$. When a gray-scale image is considered, the prediction error exhibits scalar values $d(n_1,n_2) = u(n_1,n_2) - v(n_1,n_2)$, and the factor $k$ is calculated according to the value of $|d|$ (Fig. 4.5). For a color image, prediction error is calculated in scalar or vector form.

In the scalar form, the prediction error is calculated for each color component as shown in (4.1).

$$\begin{aligned} d_r(n_1,n_2) &= u_r(n_1,n_2) - v_r(n_1,n_2), \\ d_g(n_1,n_2) &= u_g(n_1,n_2) - v_g(n_1,n_2), \\ d_b(n_1,n_2) &= u_b(n_1,n_2) - v_b(n_1,n_2), \end{aligned} \tag{4.1}$$

where, $u_r$, $u_g$, and $u_b$ are the input pixel components, $v_r$, $v_g$, and $v_b$ are the estimated values by the predictor, and $r$, $g$, and $b$ are related to the red, blue, and green color components respectively, when *RGB* color space is used.

In the vector form, a norm of the prediction error is calculated as,

$$\left\| d(n_1, n_2) \right\| = \left\| \underline{u}(n_1, n_2) - \underline{v}(n_1, n_2) \right\| \tag{4.2}$$

There is an important issue related to the choice of the color space used in order to calculate $\|d\|$.

Processing of images represented in the *RGB* color space leads to a straightforward formula,

$$\|d\| = \sqrt{d_r^{\,2} + d_g^{\,2} + d_b^{\,2}} \tag{4.3}$$

The value of $\|d\|$ could be calculated using different types of color spaces, *RGB* as well as *L*a*b**, $YC_RC_B$ which are related to human color vision system [CON97a, FAI97a, SAN98a].

1. *RGB* color space has an approximately flat and uniform distribution of each component *R, G*, and *B*. Digital images are normally represented in *RGB* space. But *RGB* color space has poor coincidence between $\|d\|$ and impression of color difference.
2. *L*a*b** color space is a perceptual uniform color space, in a sense that the distance between points is directly proportional to perceived color difference. It is used here in some experimental parts to calculate the filter quality. The *PSNR* in this color space seems to be very near to the subjective quality from the psychological point of view. The prediction error $\|d\|$ that calculated in this color space is more reliable and achieves better measure of the color difference as perceived by a human being.
3. $YC_RC_B$ color space represents one component of the luminance and two components of the color. The advantage of this color space is that luminance is separated from chrominance, an aspect which is useful in compression and image processing applications. The color components in such a kind of color space are not uniform. It is used here in the experimental parts to calculate the filter quality and the prediction error for video sequence processing.

In the experimental part, $\|d\|$ is calculated in *RGB* color space for still images and $YC_RC_B$ color space for video sequences, because the test images were presented in these color spaces. Therefore, the output results did not include the conversion error from one color space to another, where accurate filter performance has to be measured.

### 4.3.3. Processing of the prediction error

Each form of the prediction error needs different form of processing.

- ***Processing of scalar prediction error***

Scalar prediction error of each pixel in a color image is calculated separately for each color component as shown in (4.1). The prediction filter could be scalar or vector filter. Even vector filter is used; the scalar prediction error processing is applicable. Prediction error processing (Fig. 4.5) is separately applied to the prediction error of each color component. In this case, three factors are produced for each pixel ($k_r$, $k_g$, and $k_b$). Filter output for each color component ($y_r$, $y_g$, and $y_b$) is defined for each pixel by,

$$\begin{aligned} y_r(n_1,n_2) &= v_r(n_1,n_2) + k_r(n_1,n_2)\cdot(u_r(n_1,n_2) - v_r(n_1,n_2)), \\ y_g(n_1,n_2) &= v_g(n_1,n_2) + k_g(n_1,n_2)\cdot(u_g(n_1,n_2) - v_g(n_1,n_2)), \\ y_b(n_1,n_2) &= v_b(n_1,n_2) + k_b(n_1,n_2)\cdot(u_b(n_1,n_2) - v_b(n_1,n_2)), \end{aligned} \tag{4.4}$$

This kind of processing has two advantages:

1. The advantage of a vector filter mentioned above in noise suppression and preservation of local color properties is achieved.
2. The prediction error processing will keep the uncorrupted component unchanged and modify only the noisy component. The advantage here is that we do not need to modify all the pixel components when only one component is corrupted.

- ***Vector prediction error processing***

In vector processing, the prediction error is calculated as shown in (4.2). We have one prediction error for each pixel in this case. The decision of the prediction error processing unit depends on the value of this prediction error. The prediction error is used to produce factor $k$ as shown in Fig. 4.5. This factor is applied to all color components of the processed pixel. Filter output at each pixel for each color component is defined by,

$$\underline{y}(n_1,n_2) = \underline{v}(n_1,n_2) + k(n_1,n_2)\cdot(\underline{u}(n_1,n_2) - \underline{v}(n_1,n_2)) \tag{4.5}$$

Scalar processing works better than vector processing when a noisy image is directly processed. But, when the noisy image is converted to another color space, the noisy component will be distributed to all other components of the noisy pixel. In this case, vector processing is preferred, and the modification of all the pixel components is necessary.

## 4.4. Recursive decision-based filters

A general nonlinear system can be modeled using the general nonlinear function $g(\cdot)$ in (3.8) and (3.29) as explained in Section 3.3. In case of a nonrecursive system, the final output of the system $y(n_1,n_2)$ depends on the output of the nonlinear function $g$ which includes input samples of $u(n_1,n_2)$ as shown below,

$$y(n_1,n_2) = g\left[u(n_1+k_1,n_2+k_2)\right] \tag{4.6}$$

where,

$k_1$= -($M_1$-1)/2, ..., ($M_1$-1)/2,

$k_2$= -($M_2$-1)/2, ..., ($M_2$-1)/2,

$u(n_1,n_2)$ is the input signal at a position that depends on $(n_1,n_2)$ in the system,

$k_1$ and $k_2$ point the input samples in the array within the support region $M_1 \times M_2$, which is used to determine the shape of the input, mask.

In case of a recursive system, the nonlinear function includes also other previously calculated samples of $y(n_1,n_2)$ that,

$$y(n_1,n_2) = g\left[u(n_1+k_1,n_2+k_2),\, y(n_1-l_1,n_2-l_2)\right] \tag{4.7}$$

where,

$l_1$= 0, ..., $N_1$, $l_2$ = 0, ..., $N_2$, $(l_1,l_2) \neq (0,0)$,

$y(n_1,n_2)$ is the output signal of the system at a position $(n_1,n_2)$,

$l_1$ and $l_2$ point the output samples in the array within the support region $N_1 \times N_2$ that is used to determine the shape of the output mask.

Here, we are going to generalize the considerations already given for linear systems [DUD84a]

An example for a recursive system operation is shown in Fig. 4.6. The big circle in Fig. 4.6 indicates the currently calculated sample. The simplest two-dimensional output mask that allows the output to be computed recursively is the first-quadrant or causal filter. In this case, the coefficient of the output array is non-zero only in the finite region $\left\{0 \le l_1 \le N_{1,}\ 0 \le l_2 \le N_{2,}\right\}$. The shape of the output mask is given in Fig. 4.7 (middle). Another shape of mask could be used such as the left and right shapes in Fig. 4.7, but the output mask must cover sample values that are known (excluding, of course, the sample located under the processed signal). This means that certain samples must be computed

before others. If there is no possible ordering that allows the outputs to be computed sequentially given a set of initial conditions, the system in not implementable as recursive system [DUD84a].

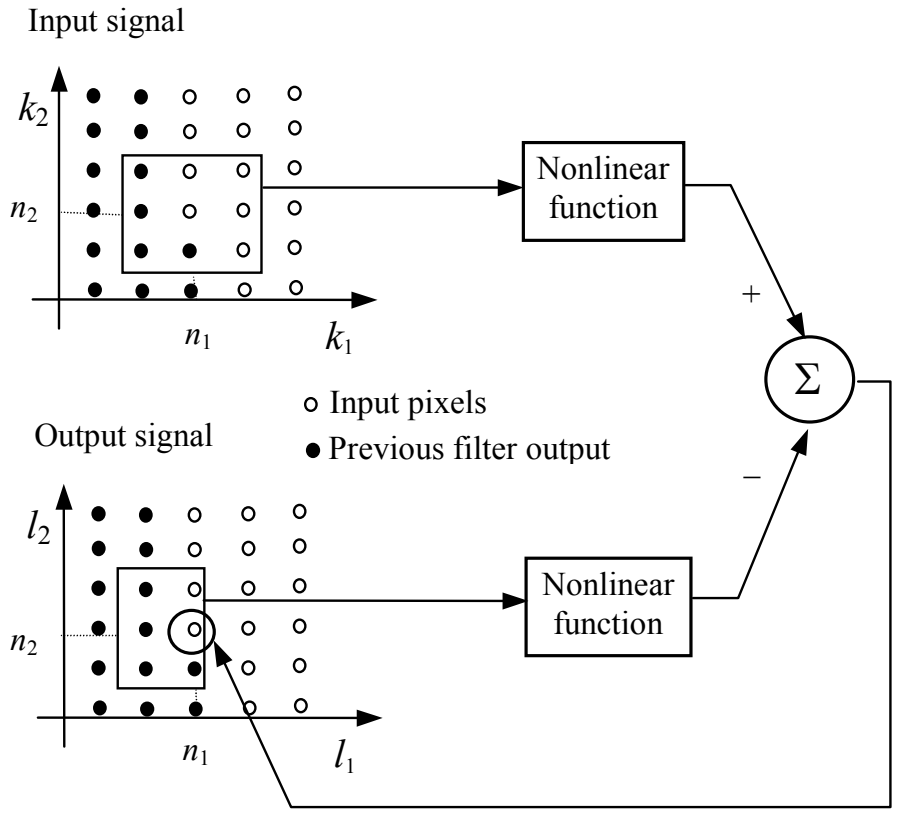



Figure 4.6. Example of the input and output masks for recursive filter.

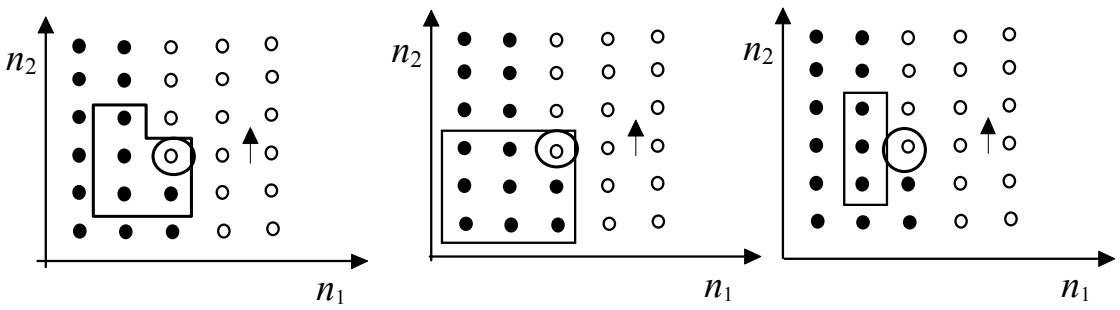



Figure 4.7. Different shapes of the output mask of recursive filter.

In a general case, a certain angle (r) as shown in Fig. 4.8 determines the shape of the mask in Fig. 4.7 and the allowable recursion direction. This mask can be generalized further by a reflection, a rotation of the output mask, or a combination of the two operations [DUD84a].

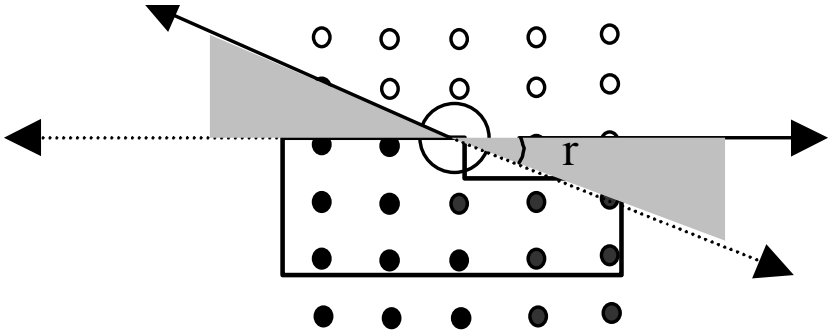



Figure 4.8. The rotation angle (r) of the allowable direction of recursion.

When the mask is on the boundary of the processed two-dimensional signal, a part of the mask will be out of the boundary of the signal. In this case, two possible ways are applicable. The first is setting all the samples of the mask, which are located out of the boundary to zero. The second is using the mirror method, i.e. the values inside the boundary is reflected outside the boundary as same as the mirror reflection.

For decision-based filter, a recursive structure is used as a predictor that $y(n_1,n_2)$ calculated by using (4.8) is expressed as $v(n_1,n_2)$. The final output in this case will be,

$$y(n_1,n_2) = v(n_1,n_2) + e(n_1,n_2) \tag{4.8}$$

and,

$$e(n_1,n_2) = k \cdot (u(n_1,n_2) - v(n_1,n_2)) \tag{4.9}$$

When a median filter is considered, the output $v(n_1,n_2)$ is calculated as one of the input samples. In case of a nonrecursive median filter, the samples are taken from a processed image $u(n_1,n_2)$. In case of a recursive median filter, the samples are taken from

the previously estimated output $y(n_1,n_2)$ by the decision-based filter as well as the samples of the processed image. In prediction error processing, the value of $e(n_1,n_2)$ is calculated as,

$$e(n_1,n_2) = k \cdot d(n_1,n_2) \tag{4.10}$$

The prediction error $d(n_1,n_2)$ is multiplied by a factor $k$ (as explained in Section 4.3), where $0 \leq k \leq 1$. Therefore even in a recursive structure, the output value does not grow over the maximum input value; i.e. the filter is bounded in bounded out "BIBO". Therefore, the filter is always stable.

# 5. Applications of decision-based filters to image restoration

## 5.1. Reference filter variants and noise model

In this chapter, decision-based filters for still image restoration are considered. Various versions of decision-based filters will be presented. The filters are obtained by augmenting median-based filters, which are called reference filters.

The following are the reference filters considered:

- median filter (MF),
- recursive median filter (RMF),
- vector median filter (VMF),
- recursive vector median filter (RVMF),
- centered weighted median filter (CWMF),
- and recursive centered weighted median filter (RCWMF).

In this chapter, an experimental comparison of filters without and with decision-based structures is presented. Standard 8-bit representation of component samples is assumed. This assumption does not influence the considerations essentially.

Artificial noise is used to corrupt a known test image by noise with known type and probability. This will give us an ability to calculate all the *PSNR* values and examine filter performance. Then we are going to use these filters for natural images, which are corrupted by real noise, even when the original image is not presented.

*The types of artificial noise*

- Type A - this type of noise has a form of random disturbance of components. The components are independently corrupted with impulse noise of amplitude (0-255). For the sake of simplicity, in the experiments, the distribution of impulse values as well as that of pixel choice is uniform. The probability that a given pixel in a given component is corrupted is $p$. This noise model is often introduced into the video sequence due to acquisition errors when the three components of the frame are separately processed.
- Type B - this type of noise is a random simultaneous change of all the components of a pixel by applying impulse noise of amplitude (0-255). The distribution of impulse values as well as of pixel choice is uniform. The probability that a given pixel is corrupted is $p$. This model of noise is also a common problem associated with television standards conversion, for example a composite to components conversion.
- Type C - this type of noise is a simultaneous change in the gain of all components of a pixel. The simulation is done by multiplying the respective components of a pixel by a certain value $g$. The range of $g$ is between 0 to 1 (0.5 was chosen in the experimental part as an example). The probability that a given pixel is corrupted is $p$. This model shows degradation due to unreliable transmission, which cause a sudden change in the data gain.

## 5.2. Basic structure with prediction error processing

### 5.2.1. Filter structure

A special subclass of the decision-based filter structure, which is called a prediction error processing filter, is shown in Fig. 5.1. It is a special case of the structure shown in Fig. 4.4. The basic structure and functions are explained in Section 4.2.

The output is a sum of the predicted value $v$ and the prediction error $e$ processed as explained in the previous chapter. In the simplest case, large values of prediction errors are set to zero because they are classified as caused by impulse noise samples. Small prediction errors are added to the prediction $v$ in order to obtain the actual output sample $y$.

Nonlinear filter employed as a predictor is either a nonrecursive median-based filter (including vector median, centered weighted median etc.) or a recursive median-based

filter (including recursive vector median and recursive centered weighted median). A recursive filter uses pixels from the output of the whole structure.

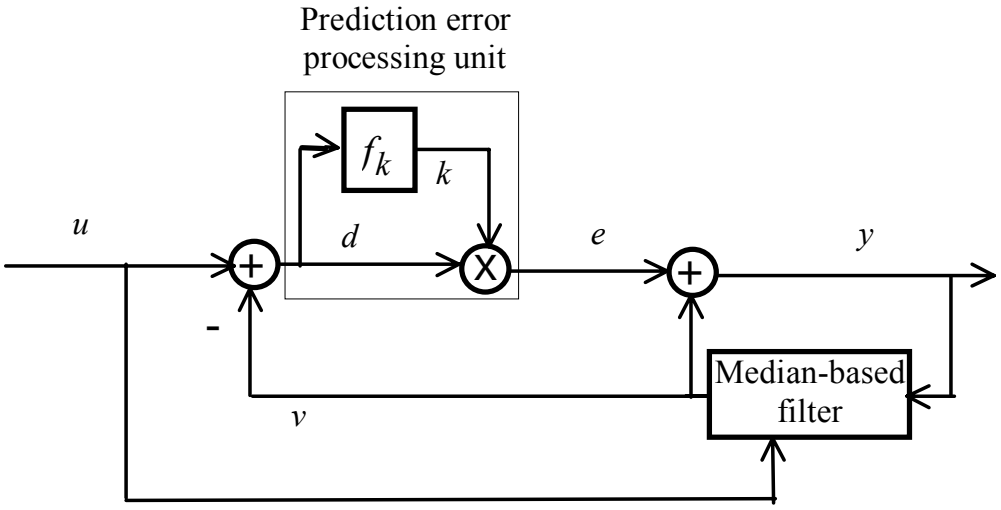



Figure 5.1. A structure of a filter with prediction error processing.

The test images are corrupted by artificial types of impulse noise, which are presented in section 5.1. We are going to prove that the fundamental advantages of these filters consist of better preservation of small details and fine textures. The estimation of the parameters of the nonlinear function, which is used in the prediction error processing unit, will be explained in the next paragraph of this section.

### 5.2.2. Performance of a nonadaptive filter

Prediction error processing is performed by the use of the decision unit (section 4.2). A threshold $T$ controls the output of the decision unit. When the decision is hard (Fig.5.2a), the unit is linear in the central region and zero on the tail, and it is called blanker nonlinearity [MAC94a]. It is bounded, but the amount of rounding error is large due to discontinuity outside the linear region.

Another choice is the soft decision [KIM95a, SAW97a], the unit is linear in the central region and falls on the tail as the input exceeds the threshold (Fig. 5.2b). Soft decision should be bounded and continuous [KIM95a, SAW97a]. Boundness insures that no impulse component has a large influence on the reconstructed data or the output of the predictor coefficients. Continuity reduces quantization error, which may be introduced by

the decision unit. In case of the soft decision, the switching zone is extended to be within the region between $\alpha$ and $2T$-$\alpha$. As a starting point, the decision zone assumed to be within $\alpha$ as specified by [MAC94a, SAW97a], and $\alpha$ is assumed to be $0.667T$ in order to have the threshold $T$ in the center of the switching zone for soft decisions. For further consideration see Section 5.2.3.

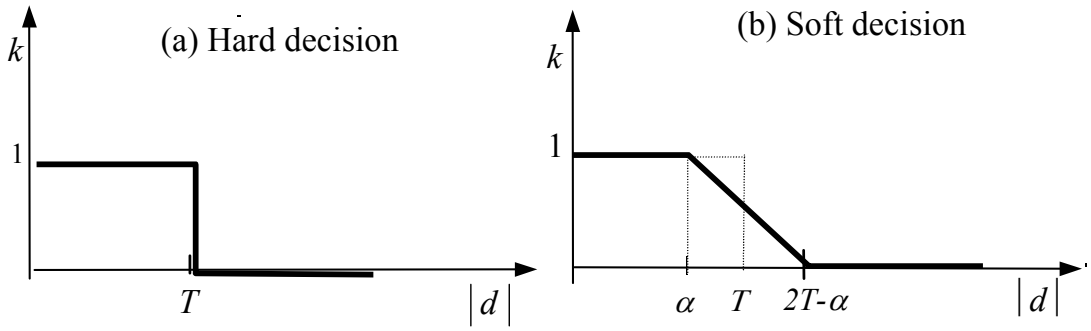



Figure 5.2. A decision choice inside the decision unit.

In order to examine the performance of the filters with prediction error processing, a series of experiments is done using the 512×512 test images shown in Fig.5.3. The comparison is done by testing all the above reference filters as a predictor in filters with prediction error processing. The following are the filters with prediction error processing:

- median filter with prediction error processing MPF,
- recursive median filter with prediction error processing RMPF,
- vector median filter with prediction error processing VMPF,
- recursive vector median filter with prediction error processing RVMPF,
- centered weighted median filter with prediction error processing,
- recursive centered weighted median filter with prediction error processing RCWMPF.
- and without prediction error processing, MF, RMF, VMF, RVMF, CWMF, RCWMF, respectively.

For center weighted median filters, the weight of the central pixel in the window is equal to $2l$+1, where $l$ is an integer number.

*Lena*

*Penguin*

*Boats*

*Clown*

Figure 5.3 Selected test images.

The test images shown in Fig. 5.3 have been corrupted by artificial noise Type A, B and C in order to use it as an input to the filters used in the experiments. As an example, Fig. 5.4 shows the corrupted test images by Type A noise with noise probability of $p$=5%.

Image quality will be assessed objectively using *PSNR* (*peak signal-to-noise ratio*) calculated in both *RGB* and *L*a*b** color spaces as explained in Section 2.2. The *PSNR* calculation depends on equation (2.10) and (2.12). The comparison will be done by means of the *PSNR* values calculated for the output images of filters with prediction error processing and without prediction error processing.

Scalar and vector reference filters will be examined as predictors in filters with prediction error processing. For prediction errors, scalar prediction error processing will be considered.

A window size of 3×3 is used in the experiments, and a fixed value is used to set the threshold. Other compendium will be discussed in the end of this section.

In order to examine the influence of the threshold value, at first the threshold is set to $\alpha$=25. The same value is applied for all the test images.

Figure 5.4 Images corrupted by artificial Type A noise of $p$ = 5%.

Table 5.1 shows the output of the examined filters in *PSNR* calculated in *RGB* color space. We got an increase of *PSNR* at outputs of all nonrecursive and recursive filters with prediction error processing as compared to filters without prediction error processing. The

results in the table show the filters improvement when Type A noise is applied to *Lena* image with various probabilities $p$=1, 5, and 10% with $\alpha$=25.

Table 5.1. *PSNR* at the output of filters with prediction error processing and without prediction error processing for various noise levels of Type A. The input image is the corrupted image of *Lena*. *PSNR* is calculated in *RGB*. The threshold is $\alpha$=25.

| | *PSNR* [dB] | | |
|---|---|---|---|
| Filter type | $p$=1% | $p$=5% | $p$=10% |
| *Corrupted input image* | 28.1 | 21.1 | 18.0 |
| MF | 39.6 | 39.1 | 38.1 |
| MPF | 48.3 | 38.0 | 37.3 |
| RMF | 38.4 | 38.0 | 37.5 |
| RMPF | 48.3 | 40.0 | 37.3 |
| CWMF | 42.4 | 40.9 | 37.4 |
| CWMPF | 48.0 | 40.5 | 38.0 |
| RCWMF | 41.5 | 40.5 | 38.9 |
| RCWMPF | 48.0 | 40.5 | 40.5 |
| VMF | 39.2 | 38.4 | 36.6 |
| VMPF | 48.2 | 39.9 | 39.4 |
| RVMF | 37.5 | 37.1 | 35.9 |
| RVMPF | 48.2 | 40.0 | 39.8 |

Table 5.2 shows the output of examined filters using corrupted *Lena*, *Boats*, *Clown*, and *Penguin* images as input. The noise Type A is chosen as an artificial noise to corrupt these images with probability $p$=5%. The output results are shown at the fist part of this table. The output is calculated using *PSNR* in *RGB* and *L*a*b** color spaces. An increase of *PSNR* at outputs of all the examined filters with prediction error processing is observed. The same conclusion is obtained when the images are corrupted by Type B noise with probability of $p$=5% as shown in the second part of the table.

Table 5.3 summarizes the experimental results for $\alpha$=25. The input of the examined filters is the corrupted images of *Lena*, *Boats*, *Clown*, and *Penguin*. The noise type used to corrupt the images are Type A and B with noise probability of $p$=5%. The results shown in this table are taken from Table 5.3.

Table 5.2. *PSNR* at the of a prediction error processing filter. The input images are corrupted by Type A and B noise with noise probability of $p$ =5%. The threshold value is $\alpha$=25.

| *PSNR* [dB] | calculated in *RGB* color space | | | | calculated in *L*a*b** color space | | | |
|---|---|---|---|---|---|---|---|---|
| Test Images | *Boats* | *Lena* | *Clown* | *Penguin* | *Boats* | *Lena* | *Clown* | *Penguin* |
| Filter output for Type A noise | | | | | | | | |
| Corrupted Image | 21.0 | 21.0 | 21.6 | 19.3 | 17.7 | 16.2 | 16.4 | 15.7 |
| MF | 27.5 | 39.1 | 31.0 | 26.2 | 25.8 | 35.6 | 28.2 | 27.4 |
| MPF | 30.3 | 38.0 | 35.2 | 28.8 | 28.5 | 35.3 | 32.0 | 28.1 |
| RMF | 26.6 | 38.0 | 29.5 | 25.3 | 25.8 | 35.3 | 27.6 | 27.1 |
| RMPF | 30.3 | 40.0 | 34.9 | 29.0 | 28.5 | 35.3 | 31.7 | 28.2 |
| CWMF | 29.7 | 40.9 | 33.0 | 28.7 | 27.3 | 36.7 | 29.6 | 28.2 |
| CWMPF | 31.8 | 40.5 | 34.6 | 30.7 | 28.6 | 36.7 | 30.6 | 28.0 |
| RCWMF | 29.0 | 40.5 | 32.3 | 28.1 | 27.1 | 36.8 | 29.3 | 28.3 |
| RCWMPF | 31.7 | 40.5 | 34.7 | 30.5 | 28.6 | 36.8 | 29.5 | 28.1 |
| VMF | 27.1 | 38.4 | 30.4 | 26.2 | 25.6 | 35.5 | 27.9 | 29.1 |
| VMPF | 28.4 | 39.9 | 32.1 | 30.1 | 29.9 | 34.8 | 39.8 | 28.9 |
| RVMF | 25.9 | 37.1 | 28.6 | 25.6 | 25.5 | 34.8 | 26.8 | 28.0 |
| RVMPF | 28.5 | 40.0 | 32.0 | 31.4 | 30.3 | 34.9 | 35.1 | 30.7 |
| Filter output for Type B noise | | | | | | | | |
| Corrupted image | 20.7 | 30.2 | 25.1 | 23.4 | 18.5 | 32.4 | 24.0 | 21.7 |
| MF | 27.4 | 39.3 | 31.1 | 26.3 | 26.1 | 36.3 | 28.7 | 29.1 |
| MPF | 29.2 | 35.7 | 32.9 | 31.6 | 30.5 | 38.1 | 34.5 | 29.6 |
| RMF | 26.5 | 38.2 | 29.5 | 25.4 | 26.1 | 35.8 | 27.8 | 28.1 |
| RMPF | 29.1 | 38.1 | 32.7 | 31.1 | 30.1 | 37.9 | 34.1 | 28.8 |
| CWMF | 29.4 | 41.5 | 33.2 | 29.1 | 27.9 | 38.6 | 31.0 | 31.3 |
| CWMPF | 30.3 | 40.8 | 32.9 | 33.0 | 31.0 | 38.0 | 34.1 | 32.0 |
| RCWMF | 28.9 | 40.9 | 32.5 | 28.4 | 27.6 | 38.0 | 30.5 | 30.8 |
| RCWMPF | 30.1 | 40.8 | 32.9 | 33.0 | 31.3 | 38.0 | 34.1 | 31.9 |
| VMF | 27.1 | 38.9 | 30.7 | 26.3 | 25. 8 | 36.0 | 28.4 | 29.4 |
| VMPF | 29.3 | 35.9 | 33.0 | 32.0 | 30.0 | 34.6 | 34.6 | 29.6 |
| RVMF | 26.1 | 37. 7 | 28.9 | 25.3 | 25.7 | 35.2 | 27.3 | 28.2 |
| RVMPF | 29.3 | 35.9 | 32.8 | 31.4 | 29.9 | 34.6 | 34.1 | 28.8 |

Table 5.3. An average improvement (for four test still images, *Lena*, *Boats*, *Clown*, and *Penguin*) of *PSNR* [dB] in the output images caused by application of the prediction error processing, noise probability $p = 5\%$, $\alpha = 25$.

| Reference filter | *PSNR* calculated in the *RGB* color space | | *PSNR* calculated in the *L*a*b** color space | |
|---|---|---|---|---|
| | Type A noise | Type B noise | Type A noise | Type B noise |
| MF | 2.1 | 1.3 | 1.7 | 3.1 |
| RMF | 3.7 | 2.9 | 2.0 | 3.3 |
| CWMF | 1.3 | 0.9 | 0.5 | 1.6 |
| RCWMF | 1.8 | 1.5 | 0.4 | 2.1 |
| VMF | 2.1 | 1.8 | 3.8 | 2.3 |
| RVMF | 3.8 | 2.9 | 4.0 | 2.8 |

Performance of the filters depends on the value of parameter $\alpha$. In a simple case, the parameter $\alpha$ is assumed to be constant. Fig. 5.5c shows that, when the prediction error processing is not applied, all the impulses are eliminated but most of the fine image details are lost. In case of prediction error processing filter, when low threshold value is applied, the filter decreases the number of the destroyed pixels within the fine detailed area (Fig. 5.5d). In Fig. 5.5e and f, we can see that when the threshold value is increased the filter tends to preserve the fine texture and details while it permits low level impulses to appear. In this case, it is important to set the threshold on a proper value to keep a tradeoff between the preservation of the image details and removing the impulse noise.

From Fig 5.5, we can conclude that the value of $\alpha$ should not be constant. Fig. 5.6 shows the output of median filter with prediction error processing when different values of threshold are applied. Type A and B noise are used to corrupt the input images.

The optimum threshold is different for each image according to the image details. At $\alpha = 0$, the filter output is the same as the output of filter without prediction error processing. The output is the corrupted image without filtering when $\alpha$ approaches to the maximum gray level value. The optimum value of the threshold $\alpha$ depends on the image details. The threshold value is increased when the image has a large texture area such as in *Boats* and *Penguin*. A small value of threshold is needed for *Lena* image because it contains a large

flat region. Also, we can see from this figure that the threshold value is different for each type of noise. Therefore, we need a technique to estimate an optimum value of $\alpha$.

The threshold value could be estimated manually according to experimental data shown in Fig. 5.6. More sophisticated filter version would employ the same idea to local adaptation of filter using a sliding window for histogram estimation.

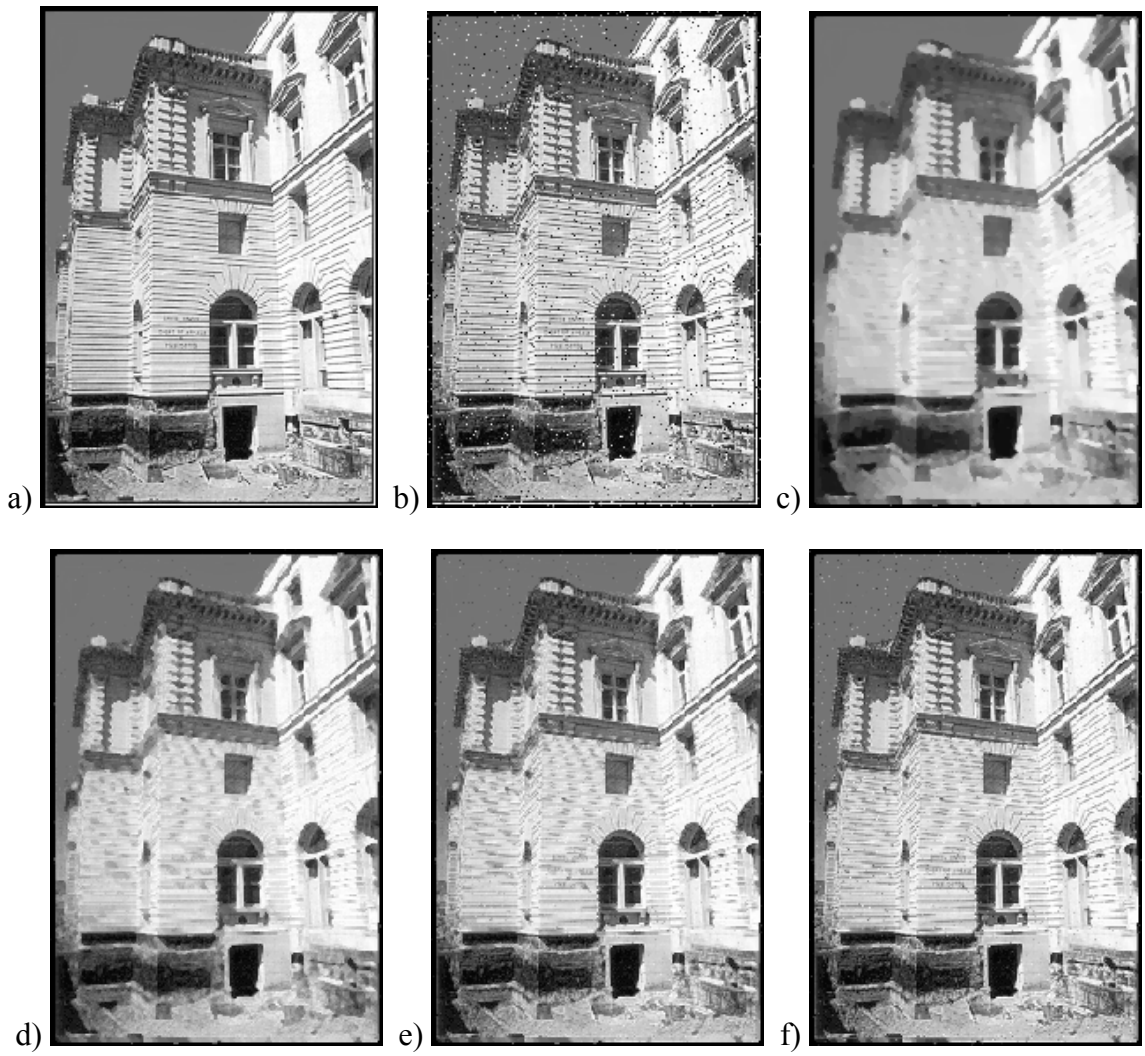



Figure 5.5. An example of using prediction error processing, a) original image, b) distorted image by Type A noise $p$ = 5%, c) an output of recursive median filter, d, e, f) the output of the proposed filter with $\alpha$ = 20, 40, 60 respectively.

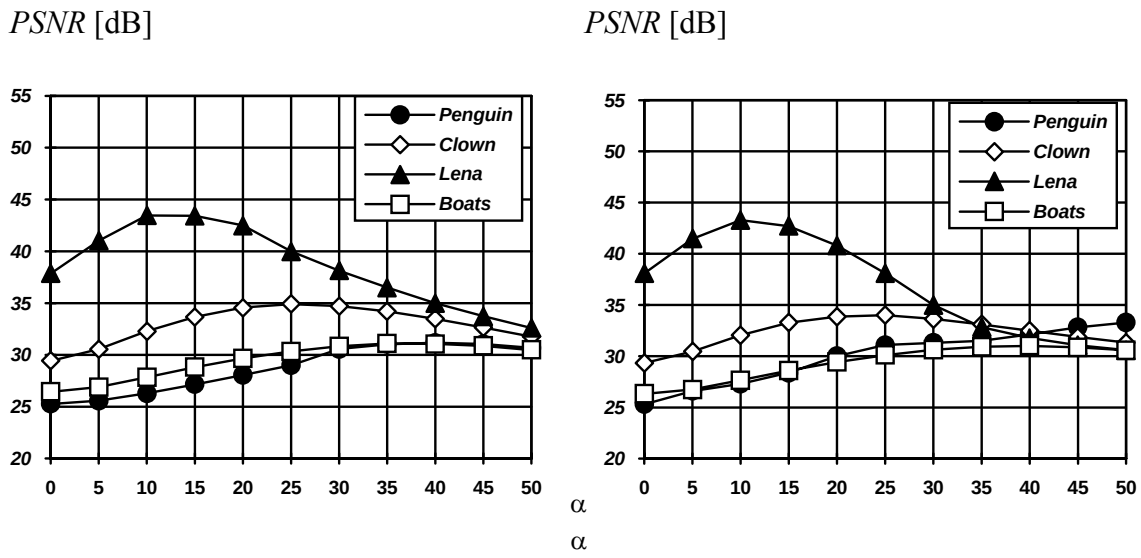



Figure 5.6. *PSNR* at the output of recursive median filter with prediction error processing (calculated in the *RGB* color space) versus α for images corrupted by noise of Type A (left) and B (right) of 5% noise probability.

### 5.2.3. Automatic estimation of thresholds

Referring to Fig 5.2, [KIM95a] suggests a fixed threshold value α and the slope between α and 2α decreases exponentially to zero as defined below,

$$f(d) = \alpha \cdot e^{\frac{-(d-\alpha)^2}{\sigma}} \tag{5.1}$$

where,

$d$ = the prediction error,

σ = a parameter used to shape the exponential curve.

In [SAW97a], the suggestion was also a fixed value of α but adaptively changing the angle of the slope. In actual case, Fig. 5.6 shows that a proper estimation of α leads to better result. In [MAC94a], a way to estimate α to be 3σ, where σ is the processed samples standard deviation, is proposed. In this dissertation a way to estimate α from the histogram of the prediction errors is proposed. In this case, the estimated value of α will be presented exactly or within the actual topography of the processed image. The suggested procedure to estimate this value is by extracting the optimum value of α from the histogram of

absolute values of $d$ (Fig. 5.7) for individual components of a color image in case of scalar-wise processing is used or from $\|d\|$ in case of vector-wise processing.

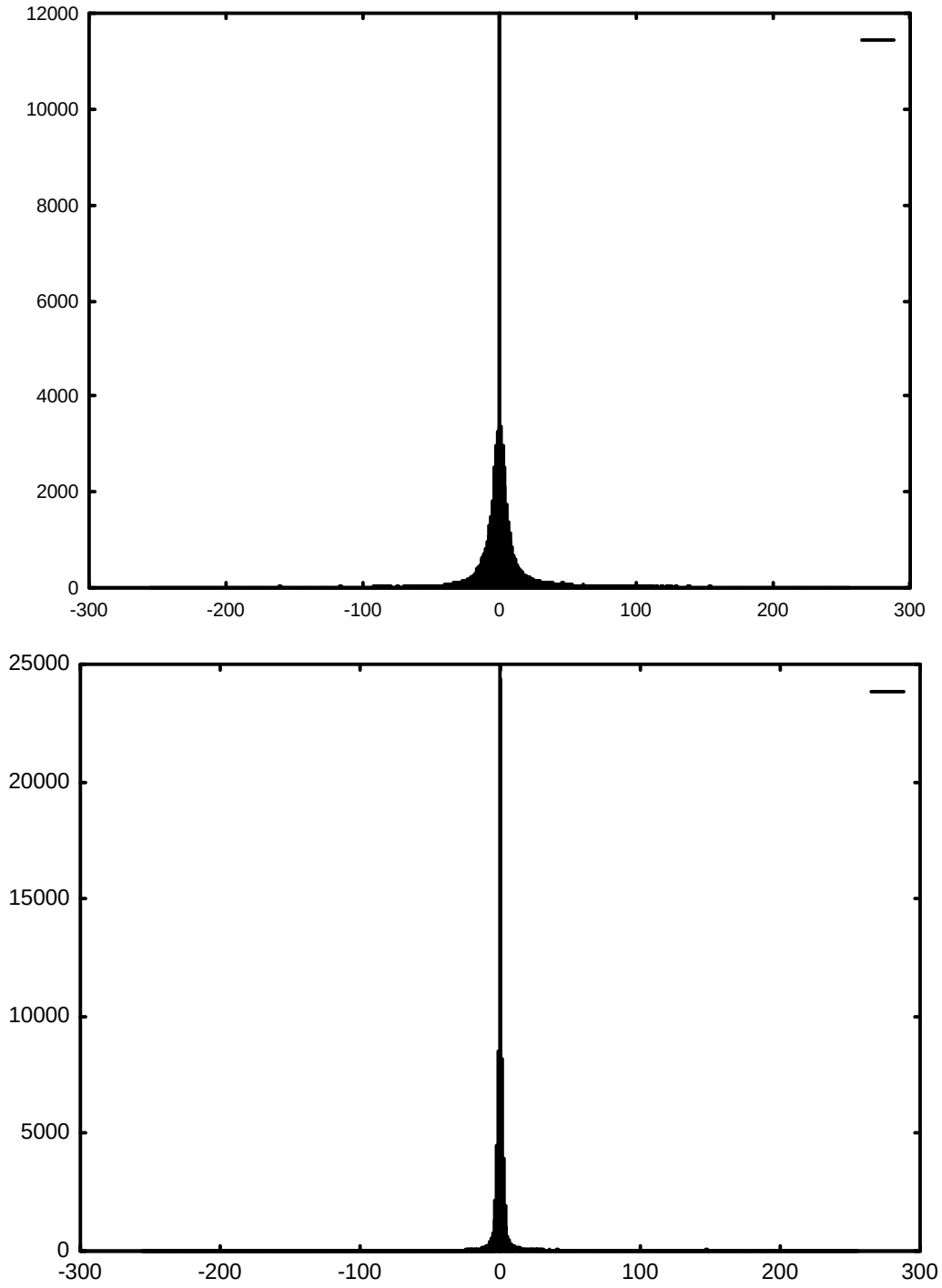



Figure 5.7. The histogram of the prediction errors for *Penguin* image (top) and *Lena* image (bottom).

Let us denote $T$ as such a value of $T$ that probability that $|d| \leq T$ is (1- $p$). Then the estimated value of $\alpha$ is chosen as $0.667T$ in order to have the threshold $T$ in the center of the switching zone for soft decisions as explained in Section 5.2.2 (see Fig. 5.2).

According to the practical results shown below (Fig 5.8), $\alpha = 0.667T$ gives a proper tradeoff between the impulse rejection and details preservation. For a value less than $0.667T$, the rounding error will be more noticeable while the filter has a good rejection for the impulses. When $\alpha$ is more than $0.667T$ value and approaches to the value of $T$, the low level impulses will be noticeable and annoy the viewer, even the filter output will be improved by high *PSNR*. The clear conclusion is seen in Table 5.4 and Fig. 5.8. The table shows high *PSNR* at the filter output at $\alpha=T$, which seems to be better result while the image is shown with less quality as in Fig.5.8.

Table 5.4. Example of the prediction error processing filter output in *PSNR* [dB] for rejection of noise with $p$ = 5% using different estimation ways of $\alpha$.

| Test image | $\alpha = 0.5\ T$ | $\alpha = 0.667\ T$ | $\alpha = T$ |
|---|---|---|---|
| *Penguin* | 24.7 | 27.3 | 28.2 |
| *Clown* | 26.6 | 27.8 | 28.0 |
| *Boats* | 39.0 | 39.4 | 38.7 |
| *Lena* | 26.6 | 30.1 | 31.6 |

Table 5.5 shows an example for the threshold estimation when $\alpha = 0.667\ T$. The value $\alpha$ obtained in this way is usually smaller than $\alpha'$ in Fig. 5.6 related to the highest *PSNR* but the best subjective quality is usually obtained for values of $\alpha$ smaller than $\alpha'$. The reason that the actual noise probability is $p$ and the detected outlayer of $|d|$ values has a probability $p$, and it is not a sure saturation that all the impulses are detected because there is a noise detection error. Then, $\alpha$ is chosen to be smaller than $\alpha'$ to decrease the detection error caused by the prediction filter. And, the smoothing area is set to be within $\alpha$ to $2\alpha$ to reduce the round-off errors.

This way of threshold value estimation is easy to simulate with less complexity when it will be extended to deal with video sequence, as it will be explained in the next chapter. Moreover, it will be shown that this way is very active to adapt the estimated threshold value in case of video processing.


$\alpha = T$


$\alpha = 0.667\ T$


$\alpha = 0.5\ T$

Figure 5.8. Example of the prediction error processing filter output for rejection of noise Type A with $p = 5\%$ using different values of $\alpha$.

Table 5.5. Estimation of the optimum value of $\alpha$ for noise with $p$ = 5% for still images.

| | Type A noise | | Type B noise | |
|---|---|---|---|---|
| Test image | $\alpha'$ | $\alpha$ | $\alpha'$ | $\alpha$ |
| *Penguin* | 45 | 28.9 | 55 | 24.1 |
| *Clown* | 25 | 16.5 | 25 | 13.8 |
| *Boats* | 35 | 23.4 | 40 | 26.8 |
| *Lena* | 10 | 8.2 | 10 | 8.0 |

### 5.2.4. Experimental results for impulse noise removal using adaptive filters

Filters with prediction error processing by using the proposed estimated value of $\alpha$ from the histogram of the prediction errors are examined here. Same situation considered in the applications of fixed value of $\alpha$ will be taken into account. Also, the reference filters will be considered. In this case, a new estimation of threshold values is used. The estimated values of the threshold are different for each image and for each reference filter calculated according to the histogram of prediction errors. The estimated threshold is adapted to the prediction error processing unit.

Inspection of Table 5.6 lead to a conclusion that, the filters with prediction error processing by means of a suitable threshold selection perform better than median filters of different kinds. The noise type used in the experiment is type A and B. The noise probability at the input is $p$=5%. Table 5.7 summarizes the results shown in Table 5.6. The numbers in Table 5.7 are the increases of *PSNR* for median filters with prediction error processing as compared to classic median filters without prediction error processing. We got an increase of *PSNR* at the outputs of all nonrecursive and recursive filters examined.

The increasing in the *PSNR* is in both *RGB* and *L*a*b** color spaces. The *PSNR* in *L*a*b* color space seems to be very near to subjective quality because it is a perceptual uniform color space, in a sense that the distance between points is directly proportional to perceived color difference. This conclusion is clear in Fig. 5.9, when the experiment shown in Fig. 5.6 is repeated in *L*a*b* color space. The result in Fig 5.9 shows that, the maximum *PSNR* at the filter output is at lower value of threshold compared with that in *RGB* color space. As explained for Fig. 5.5, we can conclude that the *PSNR* calculated in *L*a*b* color space is more acceptable to show the optimum situation of the output in subjective point of view.

Table 5.6. The output of prediction error processing filters calculated in *PSNR* for Type A noise rejection. Noise probability is $p$ = 5%, using estimated value of $\alpha$ from the image histogram.

| *PSNR* [dB] | calculated in *RGB* color space | | | | calculated in *L*a*b** color space | | | |
|---|---|---|---|---|---|---|---|---|
| Test Images | *Boats* | *Lena* | *Clown* | *Penguin* | *Boats* | *Lena* | *Clown* | *Penguin* |
| Filter output for Type A noise | | | | | | | | |
| Corrupted Image | 21.0 | 21.0 | 21.6 | 19.4 | 17.7 | 16.2 | 16.4 | 15.7 |
| MF | 27.5 | 39.1 | 31.0 | 26.2 | 25.8 | 35.6 | 28.2 | 27.4 |
| MPF | 31.8 | 43.5 | 35.2 | 32.3 | 29.1 | 38.8 | 32.0 | 31.5 |
| RMF | 26.6 | 38.0 | 29.5 | 25.3 | 25.8 | 35.3 | 27.6 | 27.1 |
| RMPF | 31.1 | 43.5 | 34.9 | 31.2 | 28.5 | 39.1 | 32.0 | 28.2 |
| CWMF | 29.7 | 40.9 | 33.0 | 28.7 | 27.3 | 36.7 | 29.6 | 28.2 |
| CWMPF | 31.8 | 42.1 | 34.7 | 31.4 | 28.8 | 36.9 | 30.6 | 28.6 |
| RCWMF | 29.0 | 40.5 | 32.3 | 28.1 | 27.1 | 36.8 | 29.3 | 28.3 |
| RCWMPF | 31.7 | 42.9 | 34.7 | 31.4 | 28.6 | 37.8 | 30.8 | 29.9 |
| VMF | 27.1 | 38.4 | 30.4 | 26.2 | 25.6 | 35.5 | 27.9 | 29.1 |
| VMPF | 31.1 | 43.4 | 35.2 | 31.5 | 28.6 | 39.8 | 32.1 | 30.1 |
| RVMF | 25.9 | 37.1 | 28.6 | 25.6 | 25.5 | 34.8 | 26.8 | 28.0 |
| RVMPF | 31.1 | 43.4 | 34.8 | 31.9 | 28.5 | 39.8 | 32.1 | 30.1 |
| Filter output for Type B noise | | | | | | | | |
| Corrupted image | 20.7 | 30.2 | 25.1 | 23.4 | 18.5 | 32.4 | 24.0 | 21.7 |
| MF | 27.4 | 39.3 | 31.1 | 26.3 | 26.1 | 36.3 | 28.7 | 29.1 |
| MPF | 31.1 | 43.5 | 34.0 | 33.6 | 31.0 | 40.4 | 33.0 | 35.3 |
| RMF | 26.5 | 38.2 | 29.5 | 25.4 | 26.1 | 35.8 | 27.8 | 28.1 |
| RMPF | 31.1 | 43.3 | 34.0 | 33. 6 | 31.0 | 40.1 | 32.9 | 35.3 |
| CWMF | 29.4 | 41.5 | 33.2 | 29.1 | 27.9 | 38.6 | 31.0 | 31.3 |
| CWMPF | 31.4 | 43.0 | 34.1 | 32.8 | 31.1 | 40.7 | 40.7 | 34. 9 |
| RCWMF | 28.9 | 40.9 | 32.5 | 28.4 | 27.6 | 38.0 | 30.5 | 30.8 |
| RCWMPF | 31.1 | 43.8 | 34.5 | 33.1 | 31.1 | 41.2 | 33.2 | 35.2 |
| VMF | 27.1 | 38.9 | 30.7 | 26.3 | 25. 8 | 36.0 | 28.4 | 29.4 |
| VMPF | 30.9 | 43.3 | 34.5 | 33.2 | 30.7 | 40.5 | 33.1 | 35.5 |
| RVMF | 26.1 | 37. 7 | 28.9 | 25.3 | 25.7 | 35.2 | 27.3 | 28.2 |
| RVMPF | 29.9 | 38.9 | 34.1 | 28.8 | 29.3 | 36.7 | 32.8 | 31.4 |

Table 5.7. An average improvement (for four test images, *Boats*, *Lena*, *Clown*, and *Penguin*) of *PSNR* [dB] in the output images caused by application of the prediction error processing, using estimated value of α from the images histogram, noise probability is $p$ = 5%.

| Improvement in *PSNR* [dB] | | | | |
|---|---|---|---|---|
| | calculated in the *RGB* color space | | calculated in the *L*a*b** color space | |
| | Type A noise | Type B noise | Type A noise | Type B noise |
| MF | 4.8 | 4.5 | 3.6 | 4.9 |
| RMF | 5.3 | 5.6 | 3.0 | 5.4 |
| CWMF | 1.9 | 2.0 | 0.8 | 4.7 |
| RCWMF | 2.8 | 3.0 | 1.4 | 3.5 |
| VMF | 4.8 | 4.7 | 3.1 | 5.1 |
| RVMF | 6.0 | 3.4 | 3.9 | 3.5 |

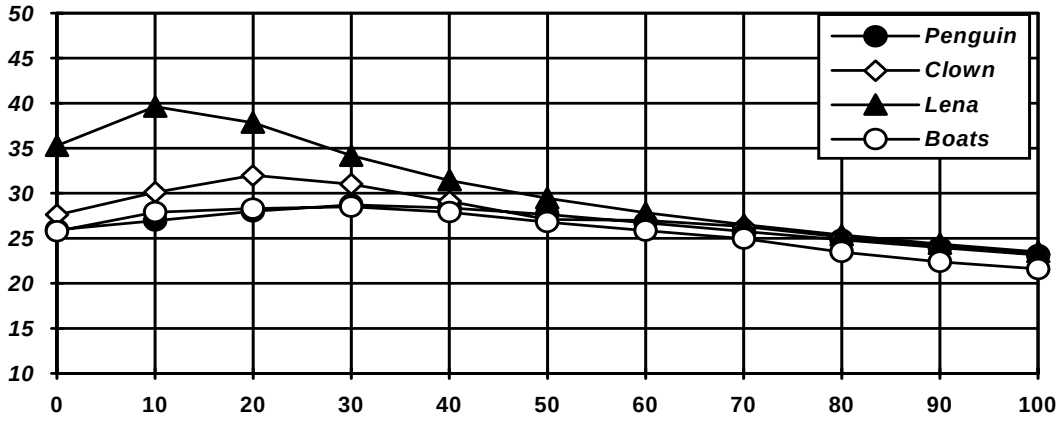



Figure 5.9. *PSNR* (calculated in the *L*a*b** color space) versus α for images corrupted by noise of Type A of 5% noise probability.

Another example is shown for Type C noise rejection in Fig. 5.10. The filter output is also improved. Fig 5.10 shows the output of the proposed filter with prediction error processing by use of recursive vector median as predictor compared with the results of classical recursive vector median filter without prediction error processing, the input is *Lena* image.

The results in Fig 5.11 show the comparison for other test images using vector and scalar reference filters, the input images are corrupted by noise of $p$=5%. Filters with

prediction error processing show better results as compared with filters without prediction error processing.

*PSNR* [dB]

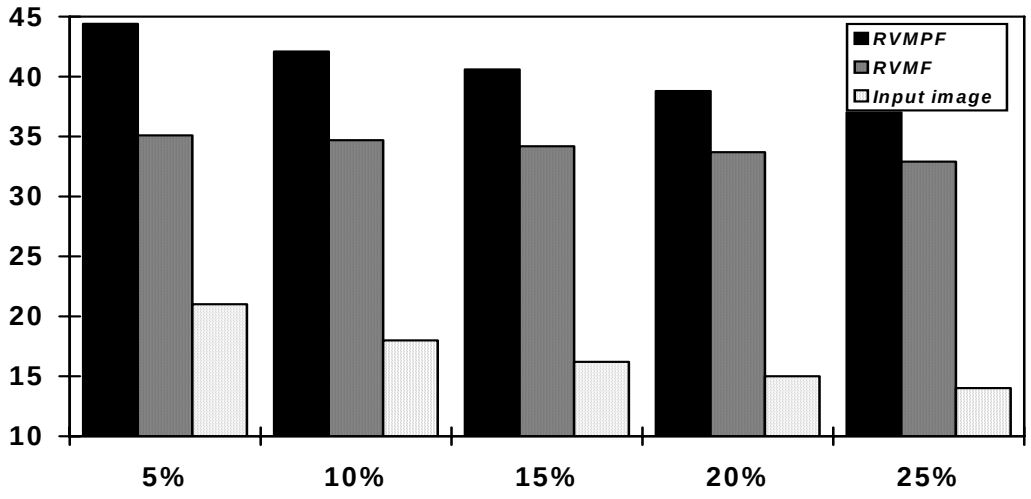



Figure 5. 10. A comparison between filters with prediction error processing and without prediction error processing using recursive median filter as predictor in *PSNR* calculated in *RGB* color space for corrupted *Lena* image with different noise probability of Type C.

*PSNR* [dB]

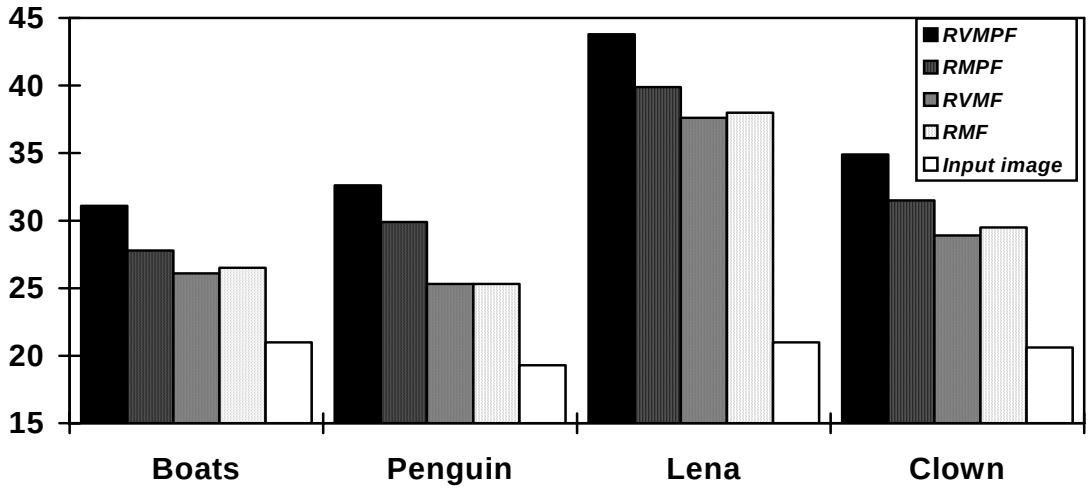



Figure 5. 11. A comparison between the vector and scalar filters with and without prediction error processing for removing type C noise of $p$=5% noise probability applied to various images. The *PSNR* is calculated in *RGB* color space.

Moreover, subjective quality was better because textures were not blurred (see Fig. 5.12). Fig. 5.13 shows the output of the filter for a highly corrupted image where *Penguin* image is corrupted by $p$ = 25% impulse noise. The subjective quality is evaluated according to the idea of the present stuff during the experiments time, and the present peoples during the local and international conferences when the results of these filters are presented. The results had a satisfied opinion of the peoples who had seen the output images of filters with prediction error processing.

The test procedure for subjective quality measure depends on the mean rating shown in (2.13). The results are compared using the grades in Table 2.2 for quality measure. The supervised test (as explained in Section 2.2.2) is used because the original images are available for a comparison purpose. All the test results had a grade of 5. The experimental results prove that this simple way of estimating near-optimum $\alpha$ is mostly quite efficient. The performance of various nonrecursive and recursive median-based filters with prediction error processing is superior to classic median-based filters.

Figure 5.12**.** Rejection of noise (type A) with $p$ = 5%: (left) recursive median filter without prediction error processing, (right) recursive median filter with prediction error processing.

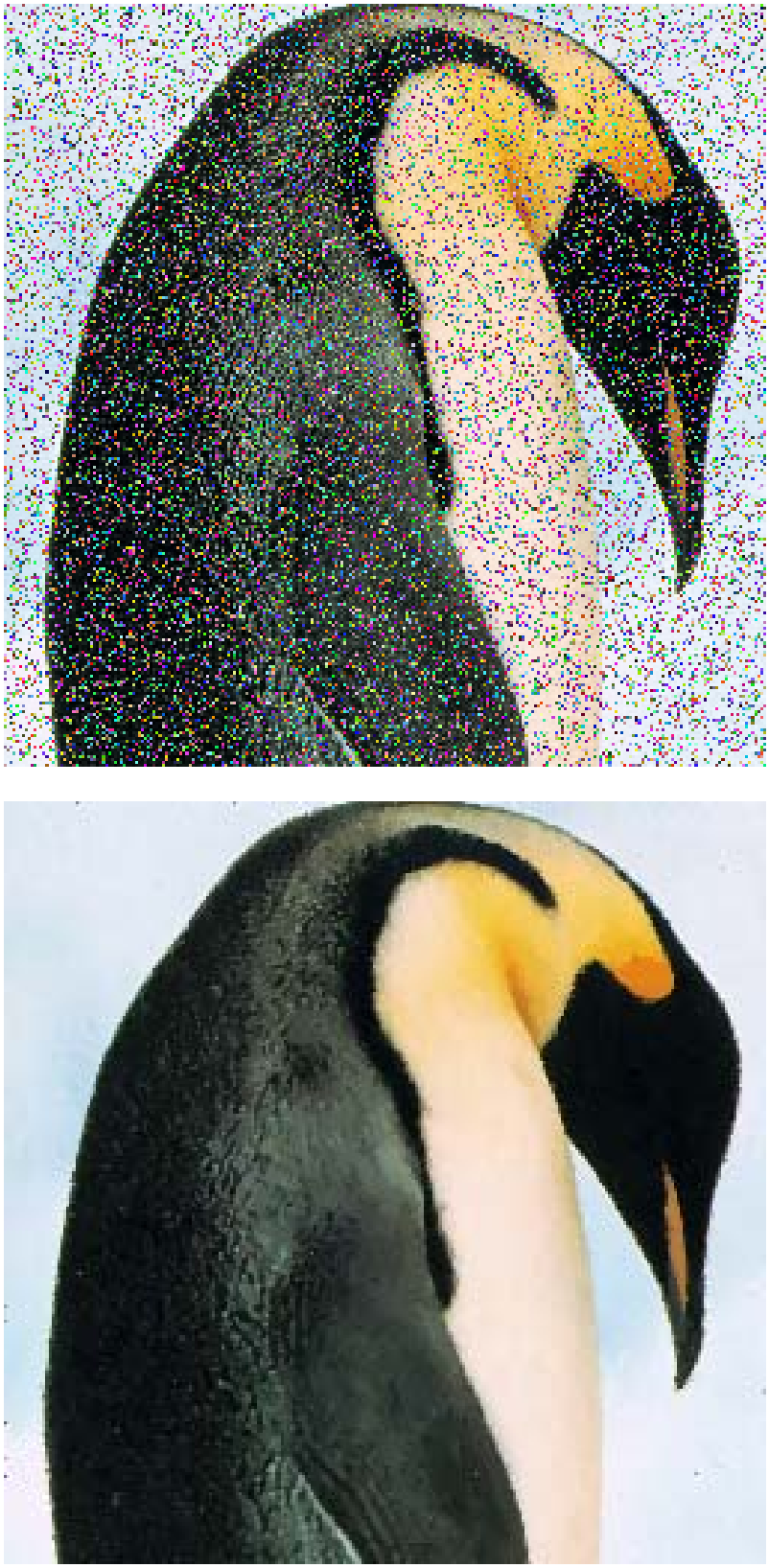


Figure 5.13a. Rejection of impulse noise Type A with noise probability of p=5%, corrupted image (top) recursive vector median filter output without prediction error processing (bottom).

Figure 5.13b. Rejection of impulse noise (Fig. 5.13a) using recursive vector median filter with prediction error processing, the threshold value is estimated from the histogram of the prediction errors of the image.

The prediction filter plays an important role in filters with prediction error processing as shown in the experimental results. When the prediction value is very near to the actual data it leads to an acceptable output. In the experimental results a 3×3 window size is used for the prediction filter because a large widow size may destroy the image fine details and make the detection operation too hard. The conclusion is the same that of increasing the window size for median filters.

The most important thing in the filter is to remove the noise and to keep the fine details undamaged. Fig. 5.14 shows an example of using 5×5 widow compared with 3×3 window size. The output results are worse when the window size is large.

*PSNR* [dB]

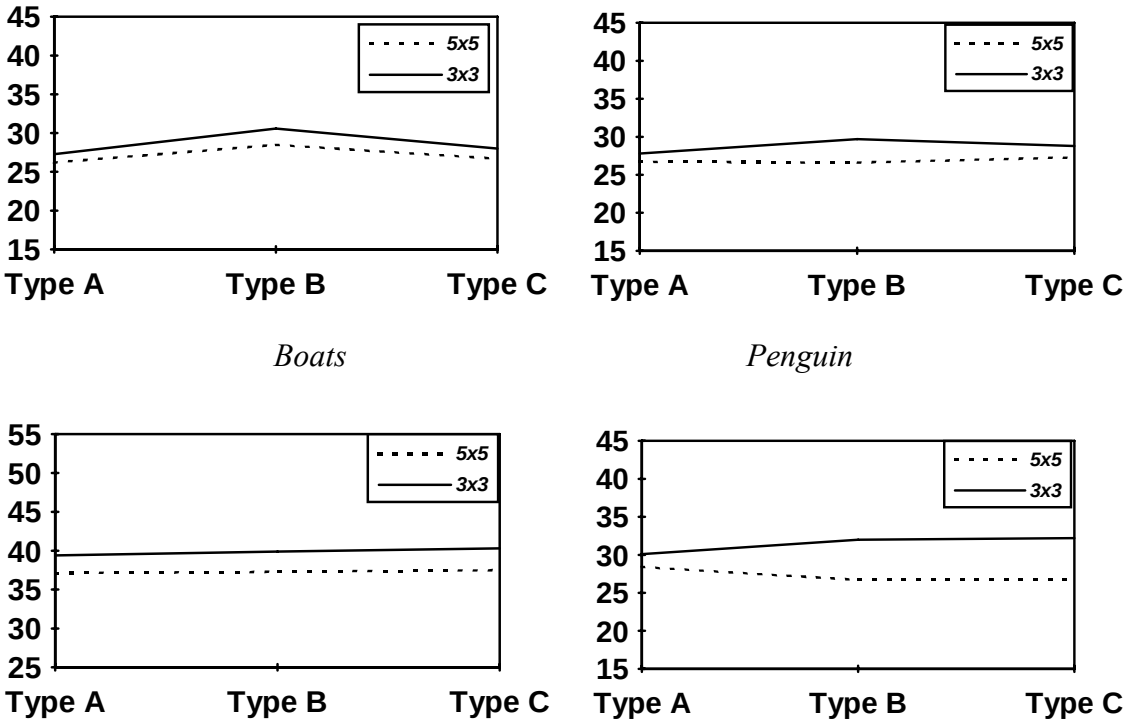



Figure 5.14. A comparison (of the filter output in *PSNR* in the *RGB* color space for various images) between using 3×3 and 5×5 widow size in prediction error processing when the reference filter is vector median filter.

## 5.3. Variant 2 of the filter with prediction error processing

Filters described in Section 5.2 process textures and small details much better than median filters of different kind. Nevertheless filters described above degrade thin lines exhibiting high contrast to background.

In order to avoid degradation of thin lines a modification of the filter structure is suggested (Fig. 5.15). It consists in processing of the prediction error by a nonlinear filter rather than by memoryless nonlinear element.

At first, parameter $s$ is calculated.

Step 1:

$if\ d(m,n)\ and\ d(m-1,n-1) > 2\alpha \quad then\ r = 1;$

$if\ d(m,n)\ and\ d(m\quad ,n-1) > 2\alpha \quad then\ r = 1;$

$if\ d(m,n)\ and\ d(m+1,n-1) > 2\alpha \quad then\ r = 1;$

$if\ d(m,n)\ and\ d(m-1,n\ \ \ ) > 2\alpha \quad then\ r = 1;$

$if\ d(m,n)\ and\ d(m-1,n-1) < -2\alpha \quad then\ r = -1;$

$if\ d(m,n)\ and\ d(m\ \ \ ,n-1) < -2\alpha \quad then\ r = -1;$

$if\ d(m,n)\ and\ d(m+1,n-1) < -2\alpha \quad then\ r = -1;$

$if\ d(m,n)\ and\ d(m-1,n\ \ \ ) < -2\alpha \quad then\ r = -1;$

else $r$ =0

Step 2:

$if\ r(m,n) = 1\ and\ r(m-1,n-1) = 1 \quad then\ s = 1;$

$if\ r(m,n) = 1\ and\ r(m\ \ \ ,n-1) = 1 \quad then\ s = 1;$

$if\ r(m,n) = 1\ and\ r(m+1,n-1) = 1 \quad then\ s = 1;$

$if\ r(m,n) = 1\ and\ r(m-1,n\ \ \ ) = 1 \quad then\ s = 1;$

$if\ r(m,n) = -1\ and\ r(m-1,n-1) = -1 \quad then\ s = 1;$

$if\ r(m,n) = -1\ and\ r(m\ \ \ ,n-1) = -1 \quad then\ s = 1;$

$if\ r(m,n) = -1\ and\ r(m+1,n-1) = -1 \quad then\ s = 1;$

$if\ r(m,n) = -1\ and\ r(m-1,n\ \ \ ) = -1 \quad then\ s = 1;$

else $s$ =0 (5.2)

The filter structure is augmented as shown in Figure 5.15. Prediction error $d$ is multiplied by the sum ($k$+$s$).

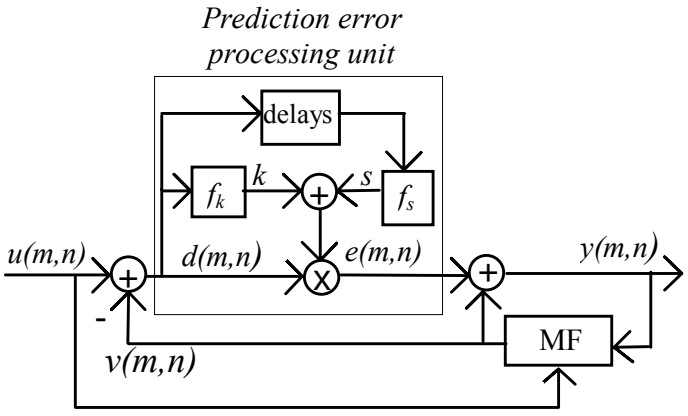



Figure 5.15. The modified filter structure. $f_s$ denotes calculations of s.

In the experiment, the test images corrupted by impulse noise Type A, and artificial lines are added in vertical, horizontal, and diagonal directions. The filter output of variant 2 has been tested in order to remove the impulse noise and preserve the lines.

Fig. 5.16 illustrates preservation of thin lines implied by application of the modified structure. The added lines have one pixel thickness. The idea is to test the filter ability in the critical conditions. The output of the filter proves the success of this operation. This advantageous property is obtained with small decrease of efficiency of rejection of impulse noise (Table 5.8). Table 5.8 shows a comparison between filters of variant 1 and 2 when they are applied to certain images corrupted by Type A noise of $p$=5%.

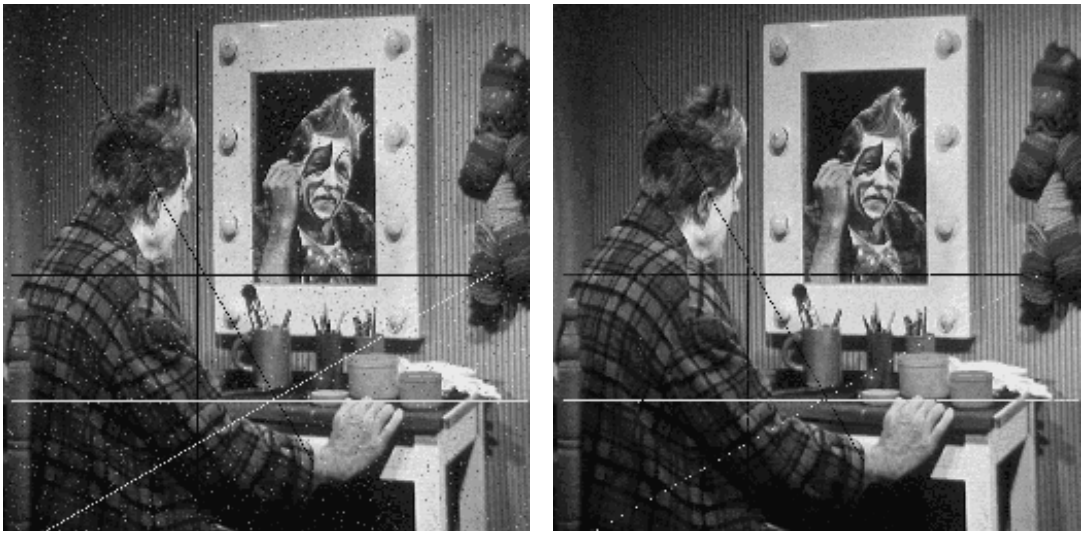


Figure 5.16. Corrupted image (left). Output of the filter of variant 2 (right).

Table 5.8. A comparison of the output of the tested filters of variant 1 and 2. The input images are corrupted by Type A noise of $p$=5%. The *PSNR* is calculated in *RGB* color space. The threshold is estimated from the histogram of the images.

| | *PSNR* [dB] | | | |
|---|---|---|---|---|
| Test images | *Boats* | *Clown* | *Boats* | *Clown* |
| Corrupted image | 21.0 | 21.6 | 21.0 | 21.6 |
| Reference filter | Variant 1 | | Variant 2 | |
| MF | 31.8 | 33.7 | 29.7 | 32.8 |
| RMF | 30.2 | 35.1 | 29.5 | 32.6 |
| VMF | 30.4 | 35.2 | 29.4 | 32.6 |
| RVMF | 29.9 | 34.7 | 29.2 | 32.4 |

## 5.4. Variant 3 of the filter structure

The variant 3 of the filter structure is a special case of decision-based filters. It exploits the idea that, the input pixel is not used in the median calculation when it is detected as corrupted pixel, and the median output is taken from the neighbors of the input pixel. In variant1, there is one reference filter used for making a decision, and applied for denoising when the input pixel is assumed as corrupted. In variant 3, two reference filters are used. The same decision filter of variant 1 is used to calculate the prediction error and decide that the pixel is corrupted or not. In this case, the input pixel is needed within the samples of the window to estimate the prediction value $\underline{v}_2$. The second filter is the denoising filter. The denoising filter is used to estimate the output value $\underline{v}_1$ when the input pixel is detected as corrupted. In this case, the input pixel is not involved within the input samples of the window. The filter structure is shown in Fig. 5.17.

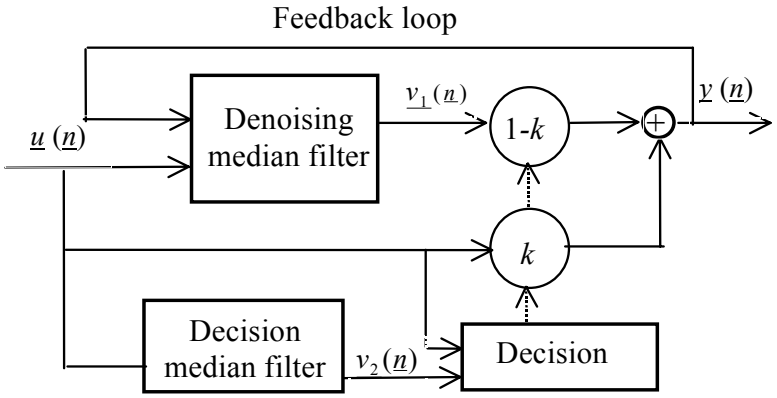



Figure 5.17. General structure of filter of variant 3.

The denoising median filter estimates $\underline{v}_1(\underline{n}_C)$ at the central pixel $\underline{n}_c$ of sliding window. The estimate $\underline{v}_1(\underline{n}_C)$ is used as output $\underline{y}(\underline{n}_c)$ only for corrupted pixels. The filter output $\underline{y}(\underline{n})$ at a pixel $\underline{n}=(n_1,n_2)$ is,

$$\underline{y}(\underline{n})=\begin{cases}\underline{u}(\underline{n}) & \textit{if } \underline{n} \textit{ is considered as "noncorrupted"},\\ \underline{v}_1(\underline{n}) & \textit{if } \underline{n} \textit{ is considered as "corrupted"},\end{cases} \tag{5.3}$$

where $\underline{v}_1(\underline{n})$ is an estimate of $\underline{u}(\underline{n})$ made on the bases of $u$ in the neighborhood of $\underline{n}$ defined as $C=\{\underline{n}_1,\ldots,\underline{n}_{N-1}\}$, $N$ is the number of the processed samples in the window. Calculation of the median value without using the central (processed) pixel results in increase of the efficiency when the processed pixel is corrupted by noise.

The decision upon pixel "corruptness" is made using the output of another median filter. This filter outputs median values of all pixels in a sliding window $C_t = \{\underline{n}_c, \underline{n}_1, ..., \underline{n}_{N-1}\}$. Then this value is used to determine that the center pixel is corrupted or not. The decision algorithm controls the filter output by producing an adaptively adjusted value $k$ from the range between 0 and 1. The value of $k$ is calculated by use of the decision median filter output $\underline{v}_2(\underline{n})$ for each sliding window. The filter output is calculated as

$$\underline{y}(\underline{n}) = k \cdot \underline{u}(\underline{n}) + (1-k) \cdot \underline{v}_1(\underline{n}) \tag{5.4}$$

Decision-based processing is exploited for classifying pixels as *correct* or *erroneous.* Large prediction errors $d = \|\underline{u} - \underline{v}_2\|$ are related to *erroneous* pixels while small values of $d$ are related to *correct* pixels. Nevertheless the decisions are soft as shown in Fig. 5.2.

As mentioned above, the median value $v_1$ is taken over the local neighbors where the center pixel is not included. This allows better reduction of impulse noise for highly corrupted images of noise probability $p$ = 10 ÷ 25% (Fig. 5.18 and 5.19). The figures show respective improvement filter performance as compared to the basic variant of the filter.

*PSNR* [dB]

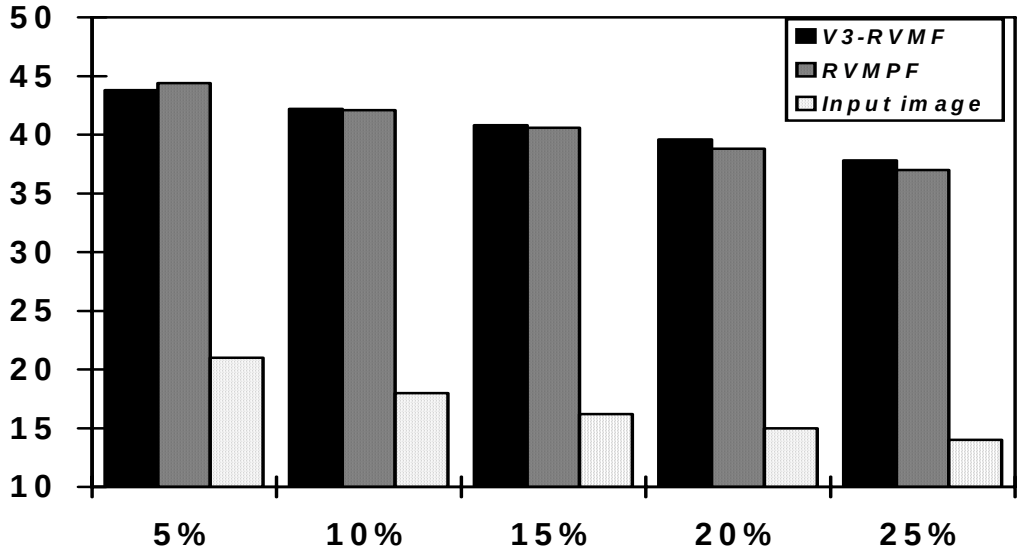



Figure 5.18. A comparison between V3-RVMF and RVMPF in *PSNR* for image *Lena* corrupted with noise of different probability $p$ of Type A.

*PSNR* [dB]

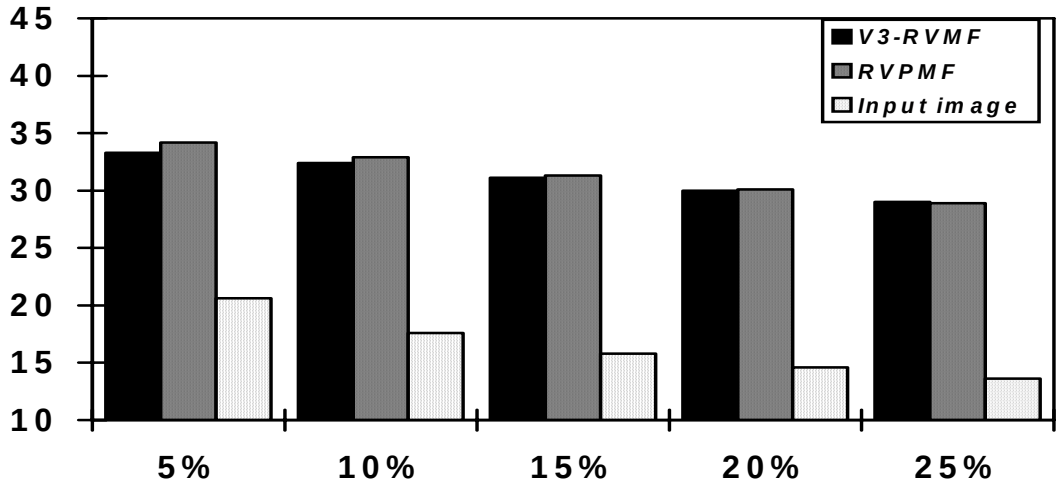



Figure 5.19. A comparison between V3-RVMF and RVMPF in *PSNR* for image *Clown* corrupted with different noise probability of Type A.

*PSNR* [dB]

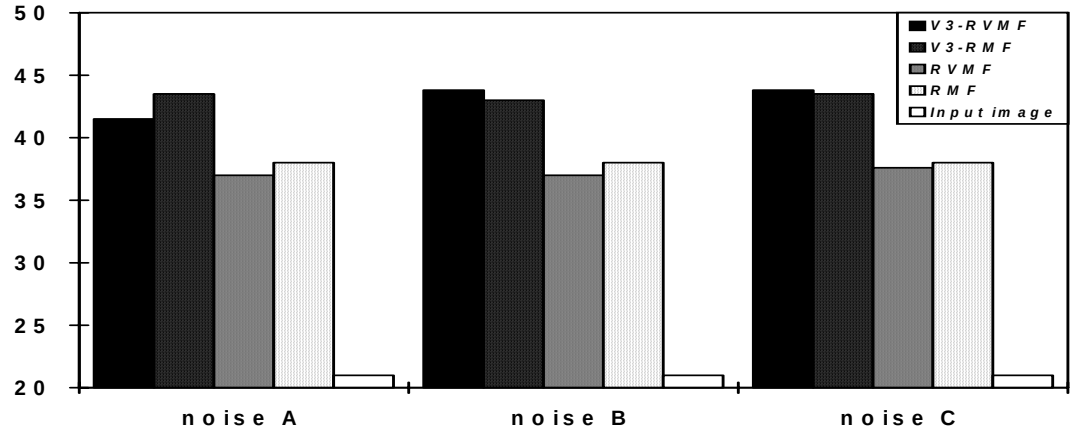



Figure 5.20. A comparison between vector and scalar variant-3 filters output for removing noise of type A, B, and with noise probability $p$=5% applied to *Lena* image.

Moreover, the results show that vector median filter is more efficient than scalar median filter for noise suppression when the noise is pixel dependent (Type C and B, for example), as shown Fig. 5.20 and 5.21. In this case, even the filter is failed in 5% type A noise rejection; it seems to be better for pixel dependent types of noise (Type B and C). This conclusion is clear from Fig. 5.22. Fig. 5.22 shows the output of the filter of variant 3

compared with variant 1 output at noise probability of p=15% when Type C noise is applied.

*PSNR* [dB]

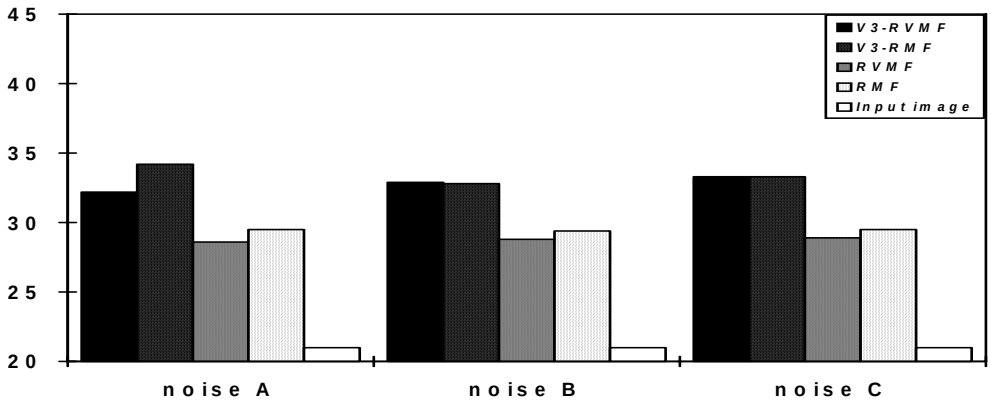



Figure 5.21. A comparison between vector and scalar variant-3 filters output for removing noise of type A, B, and C with noise probability *p*=5% applied to *Clown* image.

The subjective quality is assessed from Fig 5.23. The input image is the corrupted image of *Lena* by Type C noise of *p*=25%. The output image without prediction error processing shows that the impulse noise is rejected while most of the image details are lost.

The output of the filter with prediction error processing of variant3 shows that the impulse noise is rejected while the image fine details are preserved. Moreover, the output of the variant 3 of the filter is shown to be better than the output of variant 1 of the filter with higher complexity in the filter structure, because variant 3 needs two median values to be calculated.

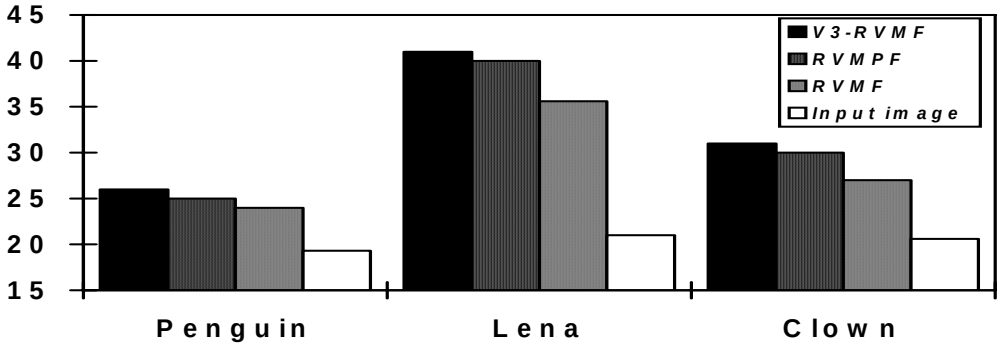



Figure 5.22. A comparison between variant-3 filter and variant 1 output for removing noise of type C with noise probability *p*=15%.

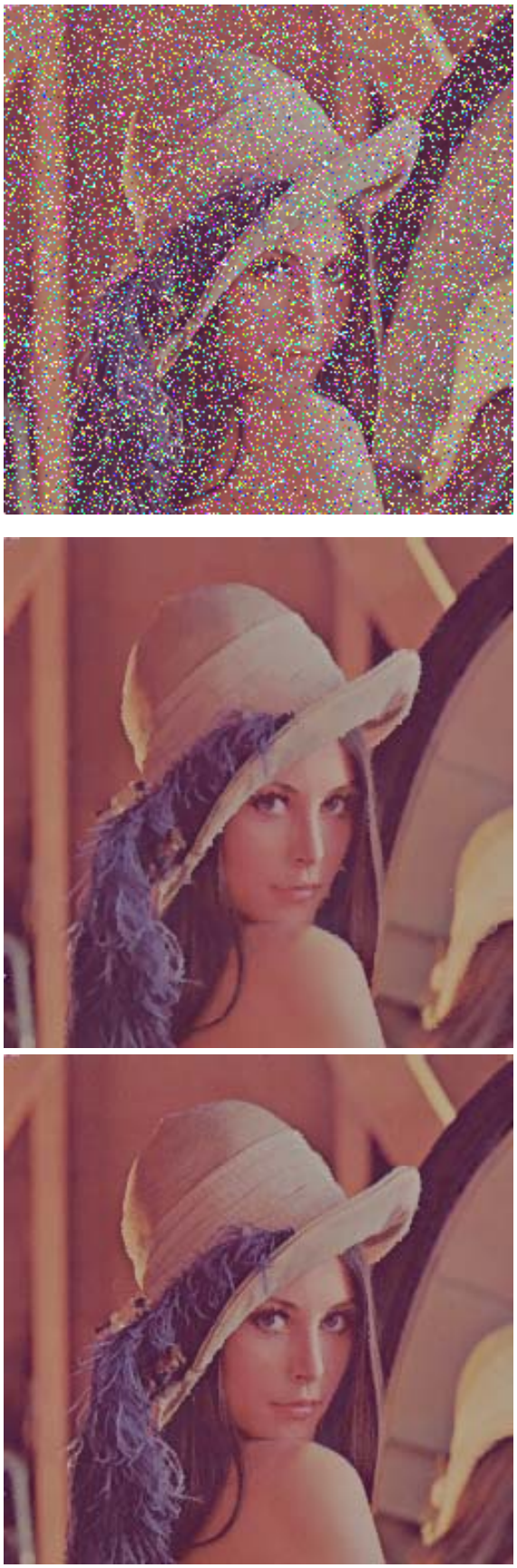


Figure 5.23. Rejection of noise (type C) with $p$ = 25%: (top) Corrupted *Lena* image, (middle) an output of RVMF, (bottom) an output of V3-VMF.

# 6. Video restoration using filters with prediction error processing

## 6.1. Three-dimensional filter structure for video restoration

In the previous chapter, the results related to two-dimensional nonlinear filters have been reported. For video restoration, three-dimensional filters are considered to exploit temporal dependencies between the frames in the sequence. Three-dimensional version of the filter structure shown in Fig. 6.1 is proposed. The filter support is explained in Section 3.4.4. The idea is the same of two-dimensional filters, but in this case other pixels from previous and future frames are also included in the samples of the processed window of the reference filter. The samples from previous and future frames are motion-compensated. Hence, the output of the reference filter as a median value of the input samples is defined as,

$$\begin{aligned} v(\underline{n}) = Median\{&u_1(\underline{n}_p, t-1),...,u_{NP}(\underline{n}_p, t-1), u_1(\underline{n}, t),...,u_{NC}(\underline{n}, t), \\ &u_1(\underline{n}_f, t+1),...,u_{NF}(\underline{n}_f, t+1)\} \end{aligned} \tag{6.1}$$

where,

$\underline{n}$ = the input pixel location at point $(n_1,n_2)$, $n_1$=1, ..., $N_1$, $n_2$=1, ..., $N_2$, $N_1 \times N_2$ is the image dimension.

$\underline{n}_p$ = the pixel location at point $(n_1+h_{p1}, n_2+h_{p2})$ in the previous frame, where $h_{p1}$ and $h_{p2}$ are the motion vectors.

$\underline{n}_f$ = the pixel location at point $(n_1+h_{f1}, n_2+h_{f2})$ in the future frame, where $h_{f1}$ and $h_{f2}$ are the motion vectors.

$NP$ = number of samples taken from the previous frame $t$-1.

*NC* = number of samples taken from the current frame *t*.

*NF* = number of samples taken from the future frame *t*+1.

The predictor is a motion-compensated median-based filter. In general case, the filter can be either recursive or nonrecursive. Both component-wise and vector-wise processing are possible.

The prediction error processing unit and the choice of its parameter α have been already described in Chapter 5. The same idea is adopted here. In the color video processing when the images are represented in $YC_RC_B$ color space, two problems have to be considered:

- the non-uniformity of the color components,
- different resolutions of the luminance and the chrominance components.

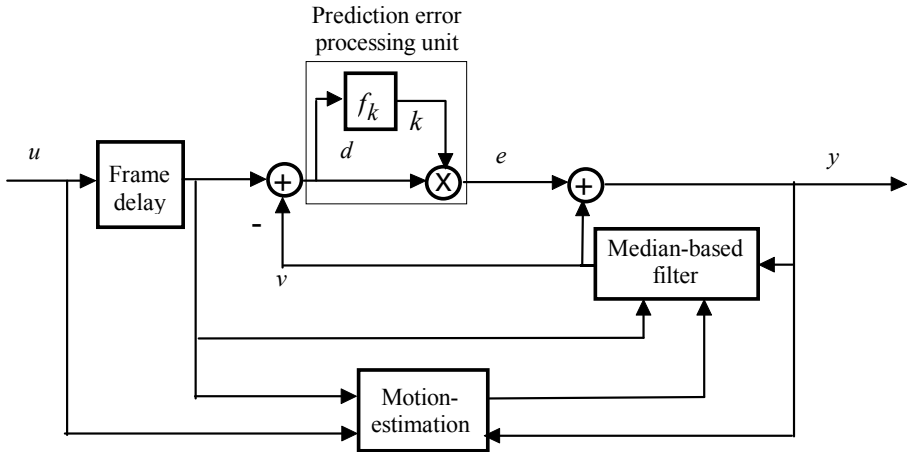



Figure 6.1. Basic structure of the three-dimensional nonlinear filters considered.

From this point of view, the threshold value is calculated in the experimental procedure separately for each color component. Examples of automatic choice of the parameter α for recursive filters are summarized in Tables 6.1 and 6.2 for five videophone color test sequences in the QCIF format. The results show the estimation of the threshold when the reference filter is scalar and vector recursive median filter, respectively. The values of the estimated threshold are different for each frame and color component, also for each test sequence according to the image details. Of course, for other reference filters the results will be also changed.

In the proposed video processing scheme, there is a frame delay to be used for motion estimation and compensation. This will give an advantage that we can calculate the

estimated threshold for a certain frame before the time of filter processing. Then the threshold will be adapted to the frame when it will be in processing time.

Table 6.1. Automatic estimation of the optimum value of $\alpha$ for Type A noise rejection with $p$ = 5% at three different frames by using recursive scalar-median filter as predictor.

| Test sequence | Frame no. 2 | | | Frame no. 3 | | | Frame no. 4 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $Y$ | $C_B$ | $C_R$ | $Y$ | $C_B$ | $C_R$ | $Y$ | $C_B$ | $C_R$ |
| *Carphone* | 20.4 | 6.5 | 6.5 | 18.9 | 9.6 | 8.2 | 20.0 | 8.5 | 7.1 |
| *Akiyo* | 20.6 | 11.4 | 4.9 | 18.2 | 14.4 | 6.7 | 20.2 | 13.4 | 4.6 |
| *Claire* | 13.3 | 7.7 | 5.2 | 12.3 | 11.9 | 8.4 | 12.8 | 8.5 | 5.9 |
| *Salesman* | 22.1 | 5.8 | 5.1 | 21.5 | 8.5 | 6.3 | 21.5 | 6.2 | 5.4 |
| *Susie* | 13.5 | 2.6 | 3.0 | 12.8 | 5.9 | 4.2 | 13.5 | 3.4 | 3.0 |

Table 6.2. Automatic estimation of the optimum value of $\alpha$ for Type A noise rejection with $p$ = 5% at three different frames by using recursive vector-median filter as predictor.

| Test sequence | Frame no. 2 | | | Frame no. 3 | | | Frame no. 4 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $Y$ | $C_B$ | $C_R$ | $Y$ | $C_B$ | $C_R$ | $Y$ | $C_B$ | $C_R$ |
| *Carphone* | 22.0 | 6.6 | 5.9 | 20.3 | 11.1 | 11.8 | 21.6 | 15.7 | 11.4 |
| *Akiyo* | 21.5 | 13.9 | 10.2 | 19.7 | 16.6 | 13.7 | 20. 9 | 16.8 | 11.7 |
| *Claire* | 13.6 | 6.7 | 5.0 | 13.2 | 14.7 | 14.2 | 13.8 | 8.8 | 5.6 |
| *Salesman* | 24.4 | 15.4 | 18.4 | 23.4 | 19.9 | 20.4 | 23.7 | 18.7 | 15.8 |
| *Susie* | 13.8 | 2.4 | 3.0 | 13.5 | 7.8 | 5.9 | 13.8 | 4.0 | 3.4 |

## 6.2. Artificial impulse noise rejection

In order to examine the properties of the filters proposed series of experiments with standard videophone test sequences have been performed.

The following are the assumptions for the experiments:

1. Video test sequences are in the QCIF and CIF format, i.e.
   a) In case of QCIF format, the luminance $Y$ has resolution of 176 × 144 and both chrominance components $C_R$, $C_B$ have resolution of 88 × 72.
   b) In case of CIF format, the luminance $Y$ has resolution of 352 × 288 and both chrominance components $C_R$, $C_B$ have resolution of 176 × 144.

2. Chrominance components are interpolated to the full size when vector median is performed.
3. The filters are recursive, i.e. they use output samples from previous frame and some neighboring pixels in the current frame.
4. Input images have been corrupted with noise of Type A with distortion probability *p*.

Same assumptions of two-dimensional reference filters in Chapter 5 are used here. In three-dimensional filtering, motion-compensation for samples taken from other frames is considered. Full-search block-matching method is used for motion estimation. As mentioned in Section 3, 8 × 8 matching blocks are used for motion estimation to calculate the motion vectors within a searching area of 27 × 27 pixels. Motion vector smoothing is also considered to reduce the probability of mismatching results in the calculation of motion vectors. The additional concepts used in this chapter are the followings:

- RMCF – motion-compensated three-dimensional component-wise recursive median filter (without prediction error processing),
- RMPCF – motion-compensated three-dimensional component-wise recursive median filter with prediction error processing,
- RVMCF – motion-compensated three-dimensional vector recursive median filter (without prediction error processing),
- RVMPCF – motion-compensated three-dimensional vector recursive median filter with prediction error processing.

The most important point considered in this chapter is that how to examine the filters proposed in the previous chapter to fit the video processing. Moreover, the advantage of three-dimensional motion-compensated processing will be focused. In this case, it is not important to repeat all the procedures of the previous two-dimensional filtering in video the processing because it is already examined. Improving the extended structure for selected points from the previous experiments, which is used for a comparison, is the main target. In the experiments, component-wise and vector-wise processing are compared. Moreover, filters with prediction error processing are compared to their classic counterparts.

Image quality has been assessed objectively using *PSNR* calculated in $YC_RC_B$ color space. The *PSNR* in $YC_RC_B$ is used because the image format is in this color space, and

using the same color space to evaluate the output results will eliminate the conversion error as explained in Section 2.2.

*PSNR*

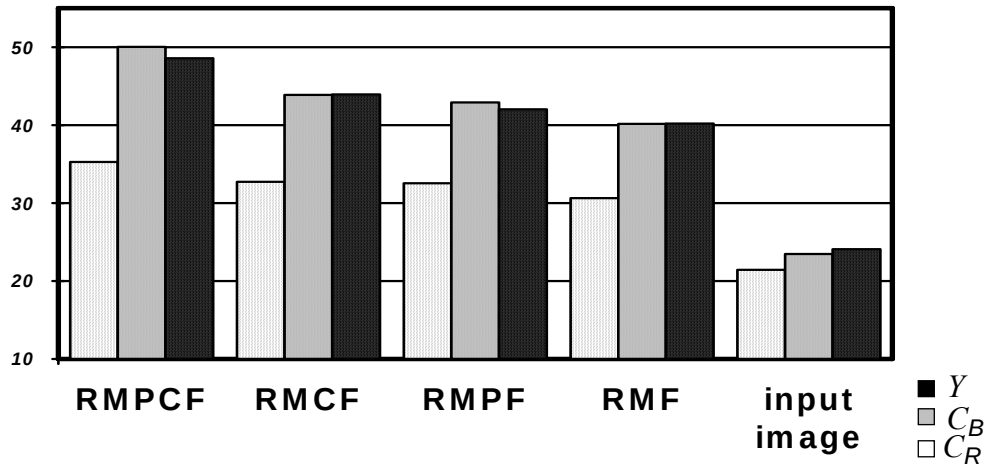



Figure 6.2. The output of component-wise recursive median filters in *PSNR* calculated for each color component *Y,* $C_B$*,* and $C_R$ for the fourth frame of the *Carphone* image sequences. Noise probability is $p$=5% of Type A, adaptive estimation of the parameter $\alpha$ is applied.

*PSNR*

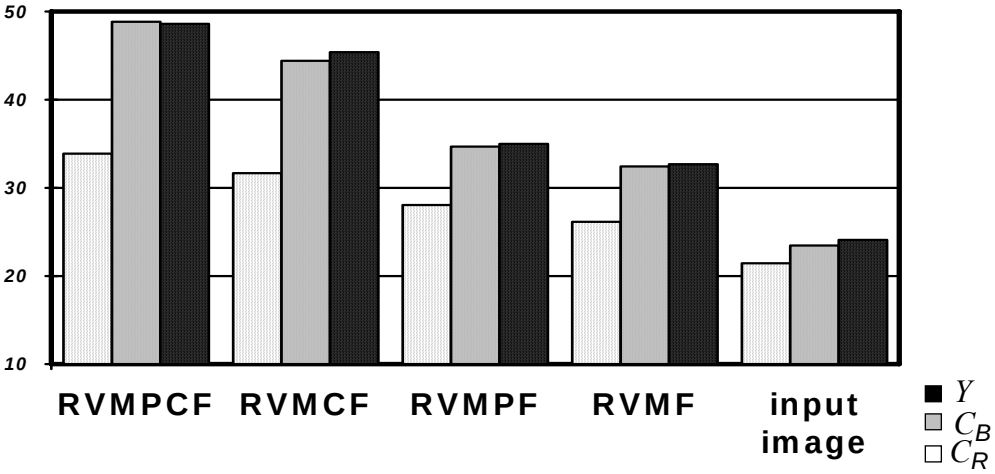



Figure 6.3. The output of vector recursive median filters in *PSNR* calculated for each color component *Y,* $C_B$, and $C_R$ for the fourth frame of the *Carphone* image sequences. Noise probability is $p$ = 5% of Type A, adaptive estimation of the parameter α is applied.

Table 6.3. *PSNR* calculated for each color component *Y*, $C_B$, and $C_R$ for output frames from the *Carphone* test sequences. Type A noise is applied with probability $p = 5\%$.

| Filter type | Frame no. | *PSNR* [dB] | | |
|---|---|---|---|---|
| | | *Y* | $C_B$ | $C_R$ |
| RVMCPF | 2 | 33.5 | 48.6 | 47.9 |
| | 4 | 33.9 | 48.8 | 48.6 |
| | 6 | 33.8 | 48.0 | 48.6 |
| | 8 | 35.2 | 49.2 | 48.5 |
| | 10 | 34.6 | 42.8 | 42.9 |
| RVMCF | 2 | 30.7 | 44.1 | 44.5 |
| | 4 | 31.3 | 44.4 | 45.1 |
| | 6 | 31.3 | 44.2 | 44.9 |
| | 8 | 32.0 | 44.1 | 44.4 |
| | 10 | 31.9 | 41.3 | 41.5 |
| RVMPF | 2 | 29.8 | 35.8 | 36.1 |
| | 4 | 29.8 | 35.2 | 35.2 |
| | 6 | 28.3 | 33.7 | 34.0 |
| | 8 | 31.9 | 35.5 | 35. 6 |
| | 10 | 31.2 | 34.6 | 34.3 |
| RVMF | 2 | 28.4 | 35.4 | 35.0 |
| | 4 | 26.2 | 32. 4 | 32.7 |
| | 6 | 27.9 | 34.1 | 35.1 |
| | 8 | 30.6 | 34. 9 | 36.3 |
| | 10 | 28.7 | 33.1 | 33. 9 |
| RMPCF | 2 | 34.8 | 48.7 | 48.9 |
| | 4 | 35.3 | 50.0 | 48.6 |
| | 6 | 35.9 | 46.6 | 46.8 |
| | 8 | 36.0 | 49.2 | 49.2 |
| | 10 | 35.7 | 46.7 | 49.6 |
| RMCF | 2 | 32.1 | 43.5 | 44.3 |
| | 4 | 32.7 | 43.9 | 43.9 |
| | 6 | 33.1 | 42.9 | 44.0 |
| | 8 | 33.3 | 43.4 | 43.7 |
| | 10 | 33.2 | 43.1 | 43.9 |
| RMPF | 2 | 32.2 | 41.5 | 41.6 |
| | 4 | 31.9 | 42.3 | 42.0 |
| | 6 | 32.2 | 41.9 | 37.9 |
| | 8 | 32.6 | 41.9 | 41.7 |
| | 10 | 32.9 | 42.3 | 39.1 |
| RMF | 2 | 30.4 | 39.8 | 39.9 |
| | 4 | 30.6 | 40.1 | 40.2 |
| | 6 | 30.8 | 39.8 | 37.2 |
| | 8 | 31.2 | 39.8 | 39.8 |
| | 10 | 31.5 | 39.9 | 38.1 |

The test results for *Carphone* test sequence are given in Fig. 6.2 and 6.3. The figures show an increase in the filter output both when prediction error processing and motion-compensation are used.

Combination of prediction error processing and motion-compensation leads to higher performance improvements as compared with classical filters operation in scalar processing (Fig. 6.2) and vector-wise processing (Fig. 6.3). The improvements could be seen in all the color components.

Filter output for other frames of this test sequence leads to the same conclusion (Table 6.3). The numbers in Table 6.3 are *PSNR* values at the output of median-based filters with prediction error processing and with motion-compensation as compared to classic median-based filters without prediction error processing and motion-compensation. The threshold value is automatically estimated for each frame and adapted to the prediction error processing unit continuously as explained previously. We got an increase of *PSNR* at outputs of all nonrecursive and recursive filters examined for both component-wise scalar and vector filtering. The results prove that the parameter $\alpha$ estimated for each frame in the sequence as frame statistics is strongly non-stationary.

Similar results can be obtained for other test sequences. The improvements are very clear from Tables 6.4 and 6.5 where average results for five test sequences (*Carphone, Akiyo, Claire, Salesman*, and *Susie*) are given. The results lead to the same conclusion for all the test images. The results show the filter improvements in each case of using prediction error processing or motion-compensation, even both together.

Table 6.4. An average improvement in *PSNR* [dB] caused by application prediction error processing for the fourth frame of (*Carphone, Akiyo, Claire, Salesman,* and *Susie*) sequences. Type A noise is applied to the input sequence, with probability of $p$ = 5%.

| Filter type | MF | MCF | VMF | VCF |
|---|---|---|---|---|
| $Y$ component | 2.3 | 2.3 | 0.5 | 2.4 |
| $C_B$ component | 1.7 | 5.3 | 0.8 | 2.5 |
| $C_R$ component | 1.2 | 5.4 | 0.4 | 1.8 |

Table 6.5. An average improvement in *PSNR* [dB] caused by application of motion-compensated three-dimensional filtering for the fourth frame of (*Carphone, Akiyo, Claire, Salesman,* and *Susie*) sequences. Type A noise is applied to the input sequence, with probability of $p = 5\%$.

| Filter type | MF | MPF | VMF | VPMF |
|---|---|---|---|---|
| *Y* component | 0.9 | 0. 9 | 1.1 | 3.0 |
| $C_B$ component | 5.1 | 8.7 | 7.2 | 8.9 |
| $C_R$ component | 5. 8 | 9.9 | 7.4 | 8.6 |


Figure 6.4. Selected frame of *Carphone* sequence: a) the original, b) corrupted with impulse noise ($p = 5\%$), c) two-dimensional vector median filter RVMF output, d) recursive vector median filter with prediction error processing and motion-compensation RVMPCF output.

Subjective quality can be assessed from Fig. 6.4 and 6.5. The figures show the preservation of the image details and fine textures while the impulse noise is rejected. The comparison is done between the output of filter with prediction error processing and

motion-compensation with classical filter without with prediction error processing and motion-compensation. The classical filter rejects the noise while it destroys some fine details of the image.

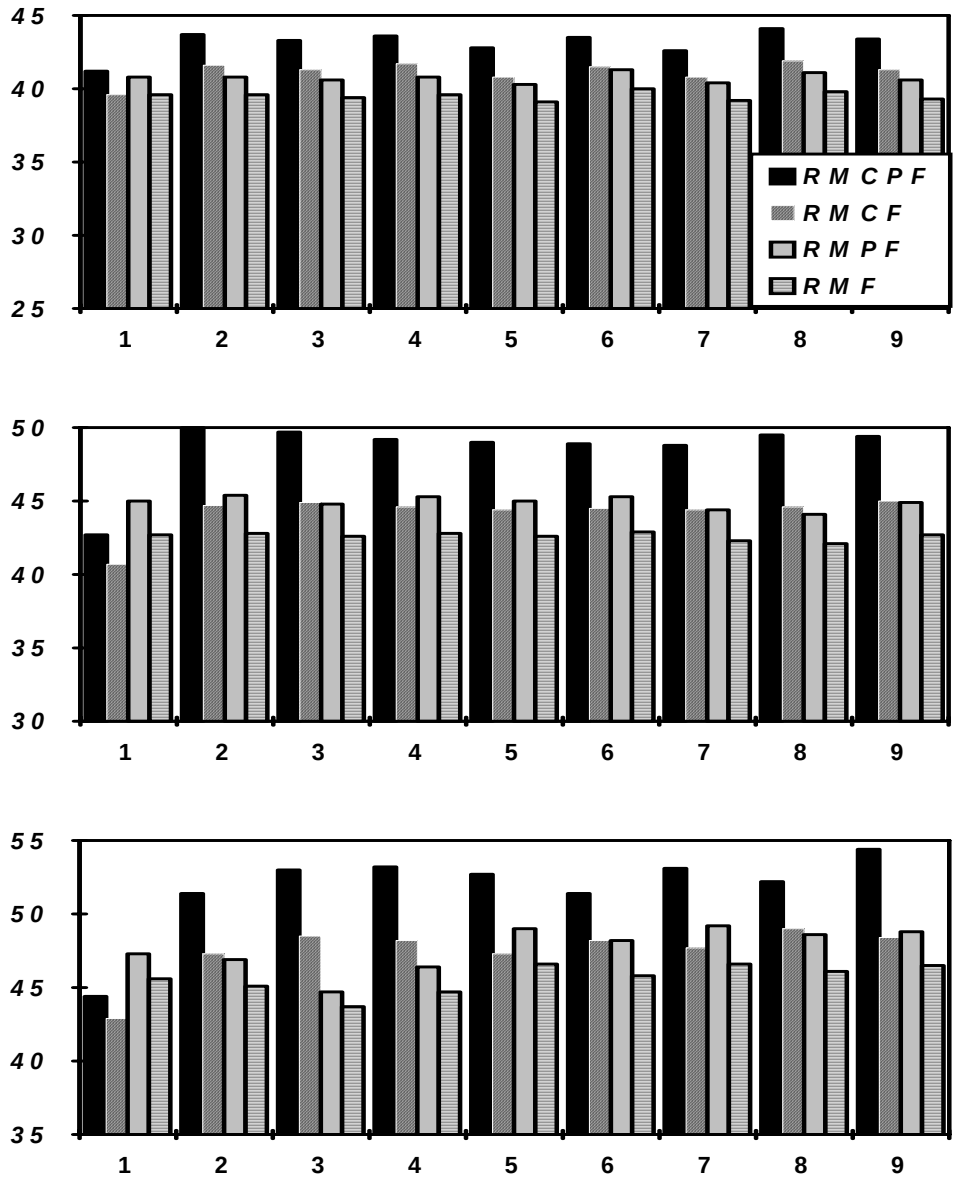

Figure 6.5. Selected frame of *susie* image: a) the original, b) corrupted with $p$ = 5% impulse noise, c) two-dimensional median filter output, d) RVMPCF output.

Fig. 6.6 shows a same comparison of the filter output by the use of other test sequence in CIF format, same conclusion is obtained. The experiment is done by the use of scalar median filter as a reference filter. The improvements could be observed at all the color components.

The results lead to the opinion that using the prediction error processing, motion-compensation, and the combination of both improve the filter output. Prediction error processing with motion-compensation leads to superior results in the filter performance. The same conclusion is obtained when vector median filter is used as a reference filter as shown in Fig. 6.7. The results when vector reference filter is used seem to be better than scalar reference filter. All the results prove that:

- Filters with prediction error processing are superior to classic filters,
- Three-dimensional motion-compensated filters perform better than intra-frame two-dimensional filters.
- The combination of prediction error processing and motion-compensation leads to better improvement in the filter output.

*PSNR* [dB]




*Frame no.*

Figure 6.6. *PSNR* at the output of recursive median for $Y$, $C_B$, and $C_R$ (top to bottom), for output frames from the *Claire* test sequences in CIF format. Noise probability is $p = 5\%$.

*PSNR* [dB]

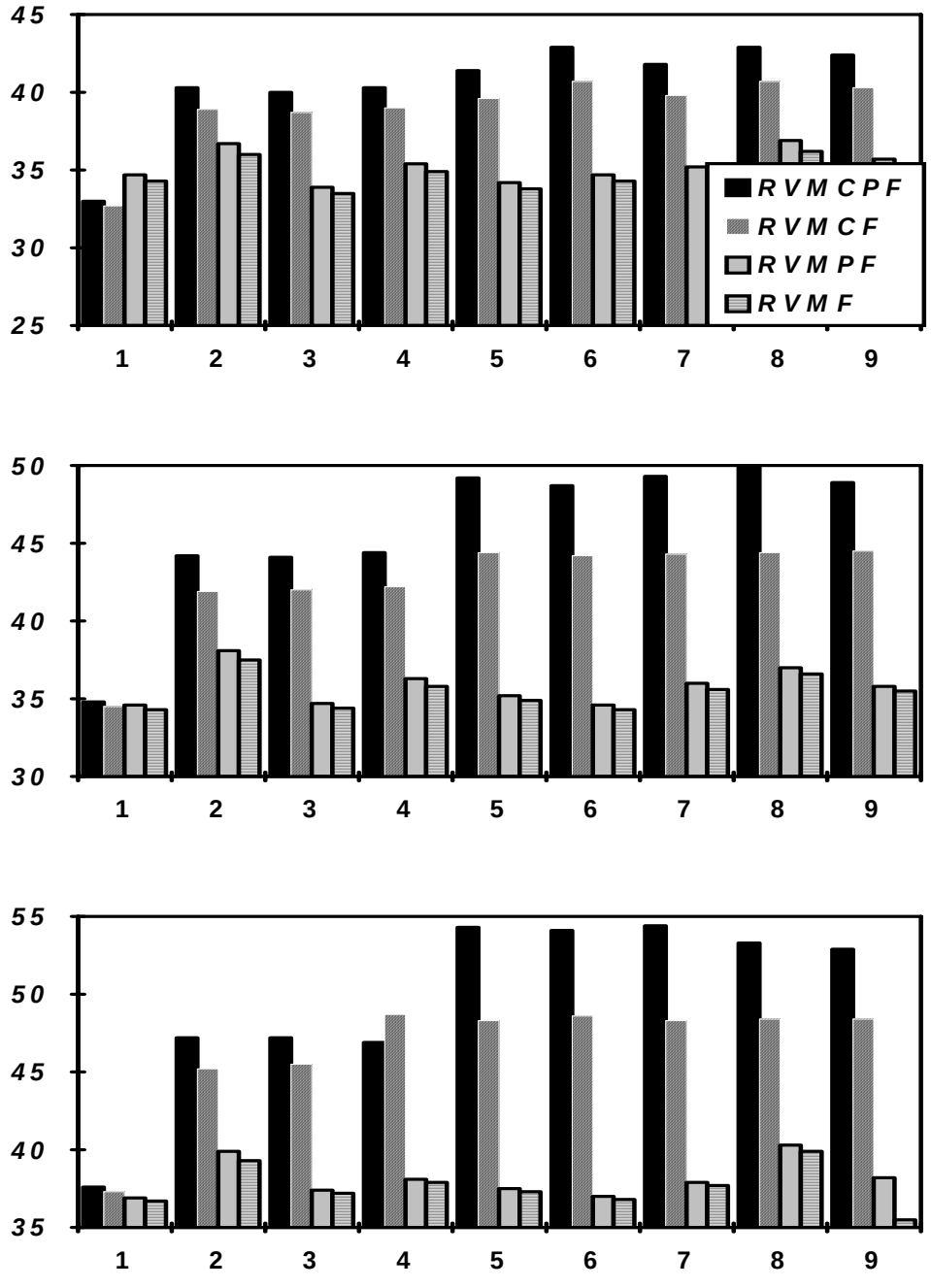



Figure 6.7. *PSNR* at the output of recursive vector median for $Y$, $C_B$, and $C_R$ (top to bottom), for output frames from the *Claire* test sequences in CIF format. Noise probability is $p$=5%.

Table 6.6 shows that application of prediction error processing leads to improved quality of the output images. The results show a comparison of the filters output at various percentage of noise probability applied to *Claire* sequence.

Table 6.6. An average improvement in *PSNR* [dB] caused by application of prediction error processing for two-dimensional and three-dimensional scalar and vector median filters for nine frames of *Claire* test sequence. Three-dimensional filters are motion-compensated. Filters are used for Type A noise rejection.

| Color Component | Filter type | | | |
|---|---|---|---|---|
| | MF | MCF | VMF | VCMF |
| for $p$=3% | | | | |
| $Y$ | 1.8 | 1.8 | 0.8 | 2.4 |
| $C_R$ | 4.3 | 5.9 | 0.5 | 3.8 |
| $C_B$ | 5.3 | 5.3 | 0.2 | 2.3 |
| for $p$=5% | | | | |
| $Y$ | 1.2 | 2.1 | 0.5 | 1.9 |
| $C_R$ | 2.2 | 4.8 | 0.4 | 3.9 |
| $C_B$ | 2.0 | 4.9 | 0.2 | 4.3 |
| for $p$=10% | | | | |
| $Y$ | 0.6 | 0.8 | 0.3 | 1.8 |
| $C_R$ | 1.9 | 2.7 | 0.2 | 1.7 |
| $C_B$ | 2.5 | 2.9 | 0.2 | 1.8 |
| for $p$=15% | | | | |
| $Y$ | 0.4 | 0.5 | 0.2 | 0.8 |
| $C_R$ | 1.1 | 1.7 | 0.1 | 1.2 |
| $C_B$ | 1.7 | 2.2 | 0.1 | 1.4 |
| for $p$=20% | | | | |
| $Y$ | 0.2 | 0.4 | 0.2 | 0.6 |
| $C_R$ | 0.7 | 1.2 | 0.0 | 0.8 |
| $C_B$ | 1.1 | 1.6 | 0.1 | 0.8 |
| for $p$=25% | | | | |
| $Y$ | 0.1 | 0.2 | 0.2 | 0.5 |
| $C_R$ | 0.5 | 0.8 | 0.0 | 0.5 |
| $C_B$ | 0.7 | 1.1 | 0.0 | 0.5 |

Same results are shown in Table 6.7 by application of three-dimensional filtering with motion-compensation. This processing increases the filter improvements into more betters as compared with two-dimensional processing.

Table 6.7. An average improvement in *PSNR* [dB] caused by application of motion-compensated three-dimensional filtering as compared to the two-dimensional scalar and vector median filters for nine frames of *Claire* test sequence. Filters are used for Type A noise rejection.

| Color component | Filter type | | | |
|---|---|---|---|---|
| | MF | MPF | VMF | VPMF |
| for $p$=3% | | | | |
| $Y$ | 0.4 | 0.7 | 4.2 | 5.7 |
| $C_R$ | 1.4 | 3.0 | 6.1 | 9.3 |
| $C_B$ | 0.1 | 0.1 | 7.1 | 10.8 |
| for $p$=5% | | | | |
| $Y$ | 1.9 | 2.8 | 4.8 | 6.2 |
| $C_R$ | 2.3 | 4.9 | 8.0 | 11.7 |
| $C_B$ | 2.5 | 5.4 | 9.3 | 13.4 |
| for $p$=10% | | | | |
| $Y$ | 0.8 | 1.0 | 4.3 | 5.7 |
| $C_R$ | 1.9 | 2.7 | 6 .1 | 8.3 |
| $C_B$ | 0.7 | 1.1 | 9.3 | 11.0 |
| for $p$=15% | | | | |
| $Y$ | 1.1 | 1.3 | 4.6 | 5.1 |
| $C_R$ | 2.1 | 2.7 | 8.7 | 9.8 |
| $C_B$ | 1.2 | 1.7 | 11.0 | 12.3 |
| for $p$=20% | | | | |
| $Y$ | 1.4 | 1.5 | 4.7 | 5.0 |
| $C_R$ | 2.3 | 2.8 | 9.3 | 10.1 |
| $C_B$ | 1.7 | 2.2 | 11.3 | 12.1 |
| for $p$=25% | | | | |
| $Y$ | 1.4 | 1.5 | 4.3 | 4.6 |
| $C_R$ | 2.3 | 2.6 | 9.0 | 9.5 |
| $C_B$ | 1.6 | 1.9 | 1.9 | 11.4 |

Subjective quality is clear also from Fig. 6.8. The preservation of the texture is clear at Fig. 6.8d which shows the output of three-dimensional filter with prediction error processing and motion-compensation compared with the two-dimensional filter output in Fig. 6.8c. Recursive vector processing is used in this figure.

The output of filter with prediction error processing is the tradeoff between the noise rejection and detail preservation. As a result of this, the output of the filter may sometimes include a little low level noise. This conclusion is clear in Fig. 5.9. In case of three-dimensional filtering with motion-compensation, the output has a good rejection of noise with less image artifact caused by median filtering compared with two-dimensional filtering. The combination of these two procedures has better resulted in both objective and subjective point of view, as shown in the final image of Fig 6.9.


Figure 6.8. Selected frame of *Missa* sequence in CIF format: a) the original image, b) corrupted with impulse noise ($p$ = 5%), c) RVMF output, d) RVMCPF output.

Original image

Corrupted image

Figure 6.9. A selected frame from *Claire* sequence in CIF format, before and after processing.

VMF output

*Two-dimensional vector median filter without prediction error processing*

VMCF output

*Three-dimensional vector median filter with motion-compensation*

(Cont. Fig. 9)

VMPF output

*Two-dimensional vector median filter with prediction error processing*

VMCPF output

*Three-dimensional motion-compensated vector median with prediction error processing*

(Cont. Fig. 9)

## 6.3. Interlaced video processing

Video systems create an electrical signal from a two-dimensional picture by the process of scanning, which is done by imaging devices in a camera. Scanning has a form of sampling of a continuously varying two-dimensional signal. Raster scanning is the most commonly used spatial sampling representation [HAS97a, NET88a]. It converts two-dimensional image intensity into one-dimensional waveform. The brightness of points in a picture is reproduced in the imaging device by beginning at the top left of the picture and reading points horizontally across the picture. At the right side of the picture, scanning moves back to the left and down to read the next line of points. This continues until the bottom of the picture is reached. Each complete scan of the picture is a frame. Frames are scanned rapidly enough to allow smooth motion in the picture to be reproduced [HAS97a, LUT99a].

Scanning all the points of the picture in a single vertical scan as just described is called *progressive* scanning. An alternative scanning method that scans only half of the lines in each vertical scan is *interlaced* scanning.

Broadcast television standards use the technique of interlacing, which is a useful method of bandwidth reduction [BRU99a, DAM96a, DUB94a]. Interlaced frames theoretically offer the same vertical resolution as that of progressive formats while permitting a saving of the bandwidth. At present, much attention is paid to the progressive format. Both interlaced and progressive formats have their respective advantages and drawbacks.

Concerning the picture quality, progressive scanning offers the benefits of an improved vertical resolution, especially on moving parts of the picture for which intra-field aliasing is avoided. Interlaced video has been used around for quite some time, and along the way many of the problems associated with it have been discovered, such as crawl and inter-line flicker. Moreover, interlaced video makes motion-based processing very difficult, resampling hard, and it does not make much sense displaying a single frame out of a sequence. For these and other reasons, the current trend is towards progressive video. However, still there exist very good interlaced cameras, due to physical limitation in the current technology. The improved quality of sources and displays makes the viewer much less tolerant to the interlaced picture, especially for large displays, at close viewing

distance and high brightness levels. The US HDTV Grand alliance has put forward a proposal containing both interlaced and progressive formats. Therefore, there exists a need to consider the interlacing during the video processing [BAG96a, KOV97a, SAN98a].

In interlacing format, the frame is generated in two halves (fields). Each field contains half of the total number of lines. Nevertheless, these two fields are in slightly different time instants. In a moving scene, fast motion of camera or viewed object is related to some artifact in the frame directly restored from two fields. This phenomenon has to be considered during the filtering.

When a standard window is applied as defined by (6.1), the data of neighboring pixels will contain information from the odd and even lines of the image i.e. this data includes mixed information from different time instances. As a result of this, undesirable output will be expected. In this case, (6.1) must be extended to,

$$\begin{aligned} v(n_1,n_2) = Median\{&u(n_1+hp_1+np_1, n_2+hp_2+np_2, t-1),\\ &u(n_1+nc_1, n_2+nc_2, t),\ u(n_1+h_{f1}+nf_1, n_2+h_{f2}+nf_2, t+1)\} \end{aligned} \tag{6.2}$$

where,

$np_1$= -($NP_1$-1)/2, ..., -2, -1, 0, 1 , 2, ..., ($NP_1$-1)/2,

$np_2$= -($NP_2$-1)/2, ..., -4, -2, 0, 2 , 4, ..., ($NP_2$-1)/2,

$nc_1$= -($NC_1$-1)/2, ..., -2, -1, 0, 1 , 2, ..., ($NC_1$-1)/2,

$nc_2$= -($NC_2$-1)/2, ..., -4, -2, 0, 2 , 4, ..., ($NC_2$-1)/2,

$nf_1$= -($NF_1$-1)/2, ..., -2, -1, 0, 1 , 2, ..., ($NF_1$-1)/2,

$nf_2$= -($NF_2$-1)/2, ..., -4, -2, 0, 2 , 4, ..., ($NF_2$-1)/2,

$NP_1 \times NP_2$= The window size in the previous frame,

$NC_1 \times NC_2$= The window size in the current frame,

$NF_1 \times NF_2$= The window size in the future frame,

The selection of the vertical samples in (6.2) is shown to be taken from the odd lines when the processed pixel is located on an odd line, and when the processed pixel is located on an even line the selected samples will be taken from the even lines.

In a simple case, we try to define that progress-wise window is used for progressive images and interlace-wise window is used for interlaced images.

Fig. 6.10 shows the filter output for an interlaced television sequence received by a satellite receiver when the antenna is not directed in a proper direction. Each image in the sequence is assumed to be corrupted by noise with percentage about 10%. The same

variant of Section 6.2 is considered in the filter. The results show that the image degradation caused by median filter is recovered. The fine textures and details are clearly observed. And, the supposed procedure by using the interlace-wise window in the filter operation has improved the filter into better output observation compared with the way of using progress-wise window.

a)  b) 

c)  d) 

Figure 6.10. A selected frame from a television satellite receiver before and after processing, a) a corrupted frame with comet-like impulse noise, b) an output of two-dimensional recursive vector median filter with progress–wise window, c) an output of two-dimensional recursive vector median filter with interlace-wise window, d) and an output of three-dimensional recursive vector median with prediction error processing using interlace-wise window.

## 6.4. Scratch rejection

As presented in chapter 2, impulse noise appears as single spots or bright tailed spots, which look like comets. This kind of noise is called sometimes video scratches. The video scratches could be caused in different situations,

- during transmission of an image over noisy digital channel as discussed in chapter 2,
- mechanical damage of the tape guide in the video player [HAR97a]

The scratches are running horizontally along the image. They are a kind of salt and pepper noise but with non-constant impulse values. Since a pixel value is transmitted as its binary representation, any of its bits can be corrupted.

In case of a spot-like impulse noise, we can see that we are able to determine these spots due to the fact that the respective pixel value is very different from the majority of the values of the surrounding pixels. For such a type of noise, a progress-wise supporting window of the predictor in (6.1) for median filter is used.

In the case of a comet-like impulse noise, each impulse consists of a number of pixels and not a single pixel as in the spot-like noise as shown in Fig. 6.11. The noise appears here as bright lines, which may be classified by the impulse detector as texture. In this case, many corrupted pixels could be classified as uncorrupted pixels and left without change as shown in middle image of Fig. 6.11. For this type of noise an interlace-wise window for the predictor in (6.2) is proposed. The idea is to make a certain shift between the processed pixel and its neighbors. In this case, the processed window will contain other samples from uncorrupted pixels, which lead to improve the filter output.

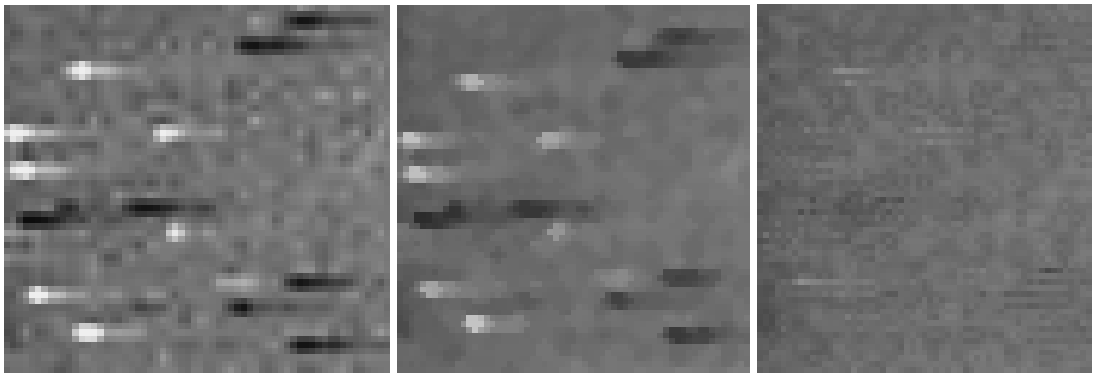


Figure 6.11. Enlarged impulse noise example (comet-like), received impulses (left), output of classic 3×3 median filter with progress-wise window (middle), output of median filter using interlace-wise window (right).

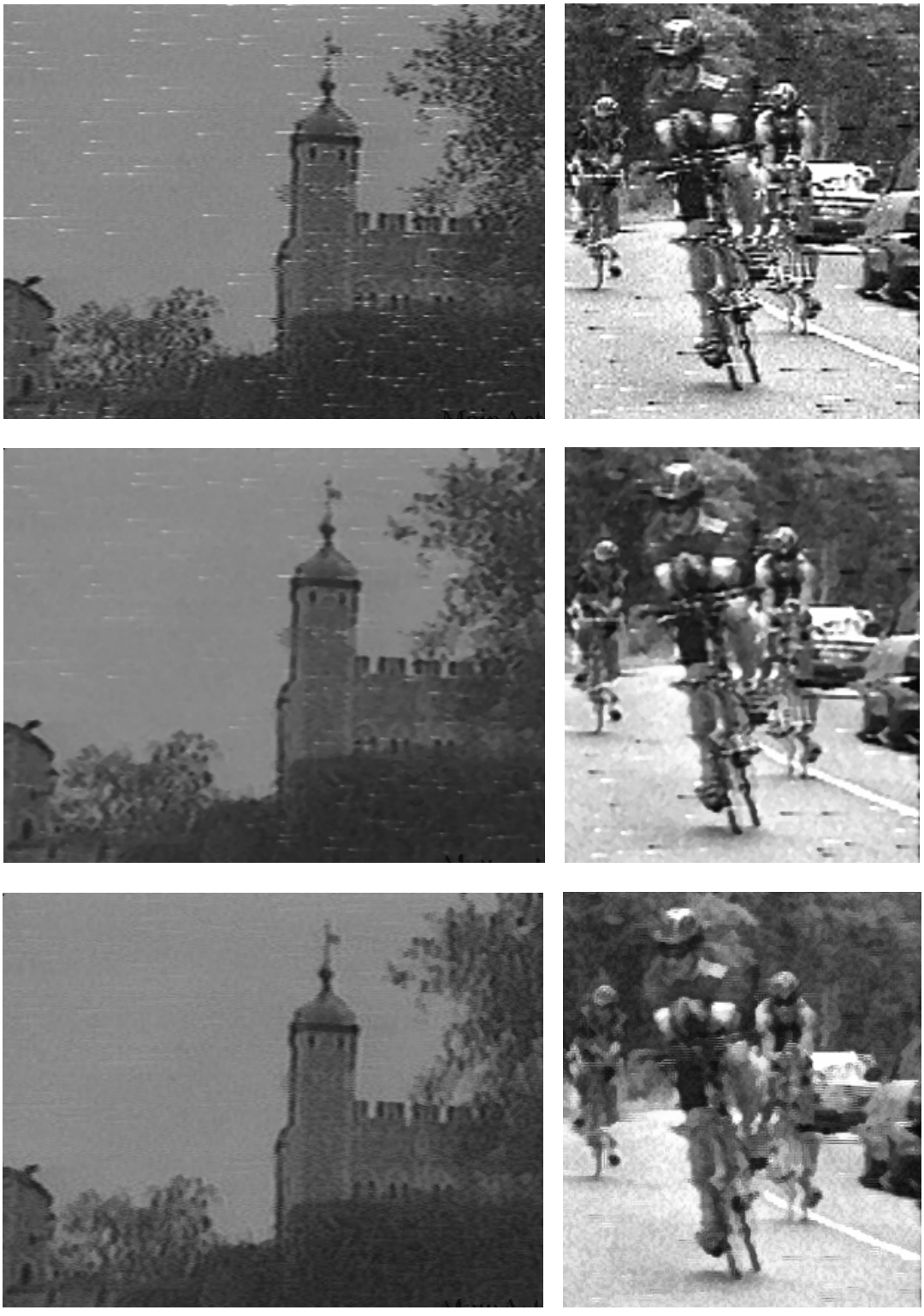


Figure 6.12. A selected frame from a television satellite receiver before and after processing: corrupted frame with comet-like impulse noise (up), an output of two-dimensional recursive vector median filter using progress-wise window (middle), and an output of three-dimensional recursive vector median with prediction error processing when an interlace-wise window is used (down).

a)  b) 

c)  d) 

Figure 6.13. A selected frame from a television satellite receiver before and after processing, a) a corrupted frame with positive and negative comet-like impulse noise, b) an output of two-dimensional 3×3 scalar median filter using progress-wise window, c) an output of two-dimensional 5×5 scalar median filter using progress-wise window, d) and an output of three-dimensional recursive vector median with prediction error processing using interlace-wise window.

To examine the proposed structure of three-dimensional motion-compensated filter with prediction error processing for comet-like scratches rejection interlace-wise window is used. A selected test sequence is taken from a satellite receiver corrupted by comet-like

noise. The images of the sequence are estimated to be corrupted by 10% of noise. The output is compared with median filter output with different size of window.

The result shows that, the proposed method leads to better results than these shown for classical filters with different sizes using interlace-wise window. The proposed method shows that negative and positive scratches are well removed while the fine textures and details are clearly observed at the output of the filter.

Another example is shown in Fig. 6.13. The received video sequence was highly contaminated by positive and negative impulse comet-like scratched due to a non-proper directing of the satellite antenna. The proposed filter structure shows an adequate removing of noise while keeping the final details unchanged. From the output images, we can conclude that the proposed interlace-wise window is very useful when it is used both in removing such a type of noise and to treat with interlaced video sequence.

## 6.5. An empirical choice of threshold for video processing

It is clear shown from the experimental results that self-adaptation by use of the total information of the histogram of the prediction error posses excellent noise removing capabilities. But we have to know that previous information of the noise level $p$ is needed. Mostly, in practical situation of video processing we do not have an idea about the transmitted signals. In this case, an automatic noise level detection is needed. In order to eliminate such additional computational cost added to the filter, an automatic adaptation depends on the local statistics of the processed data is proposed. The adapted algorithm depends on the empirical test of the local statistics of data inside the processed window.

The standard deviation is a very attractive tool for impulse noise detection. The fixed sample size FSS test may be the simplest and easiest choice. This approach depends on a simple statistical hypothesis [LEE98a, PIT90a]. The standard deviation is used to determine the threshold value, which will be adapted to the filter structure [BOV83, MAC92a, TAG95, VEL98a]. The threshold value depended on this parameter also has a limitation. The noise variance in this case is needed for outlayer detection. The outlayer is determined according to the amount of the standard deviation shift, of the processed samples, from the noise variance.

Usually the noise variance and the amount of the shift are unknown. These two facts justify the main idea of the proposed algorithm. The proposed algorithm depends on two main factors:

- The median and the median deviation can be used instead of the arithmetic mean and the standard deviation. The resulting test is more robust,

$$a = \frac{1}{M \cdot N} \sum_{i=-(M-1)/2}^{(M-1)/2} \sum_{j=-(N-1)/2}^{(N-1)/2} \left\| v(n_1, n_2) - u(n_1 - i, n_2 - j) \right\| \tag{6.3}$$

The median value (the output of the predictor, $v$) is used instead of the mean value of the processed sample, and the median deviation is the mean distance between the median value (or vector) and all the values (or vectors) in the sample.

- The maximum data shift is determined by the comparison between the predicted value (or vector) and the previous prediction data,

$$b = \max\left\{ \left\| v(n_1, n_2) - v(n_1 - i, n_2 - j) \right\|,\ i = 0,1,...,M,\ j = 0,1,...N. \right\} \tag{6.4}$$

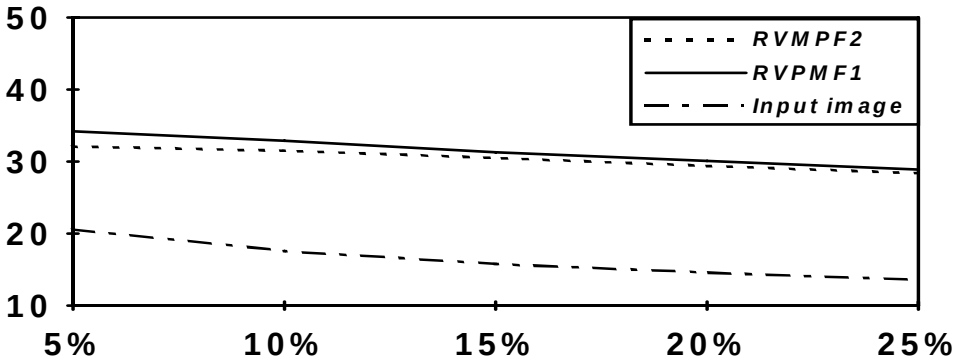
where $u$ and $v$ are the input and the prediction filter output data, respectively, and $M \times N$ is the window size. The estimated threshold value is calculated in this case as,

$$T = \begin{cases} a & if \quad a > b \\ b & else \end{cases} \tag{6.5}$$

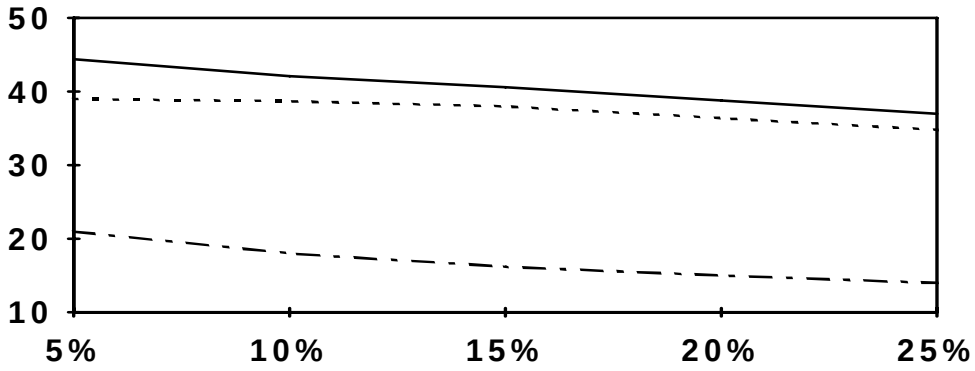
The actual value of the threshold to be adapted is ($\alpha_E$). For soft threshold, the adaptive threshold value ($\alpha_E$) is assumed to be $0.667T$. A comparison is done between the prediction error processing filter which depends on this empirical threshold $\alpha_E$ (RVMPF2), and the previous prediction error processing filter which depends on the objective calculation of threshold $\alpha$ (RVMPF1), using recursive vector median filter RVMF as predictor. The results are shown in Fig. 6.14 and 6.15. The results show that this algorithm leads to good filter performance with a little less objective quality in *PSNR* as compared with the prediction error processing filters using $\alpha$. The loss in the filter output is due to some details in the image, which are considered by median filter as noise.

In the subjective point of view, Fig. 6.16 shows the output of the proposed filter compared with recursive median filter without prediction error processing. The result is a rejection of noise with preservation of fine textures and details.

*PSNR* [dB]



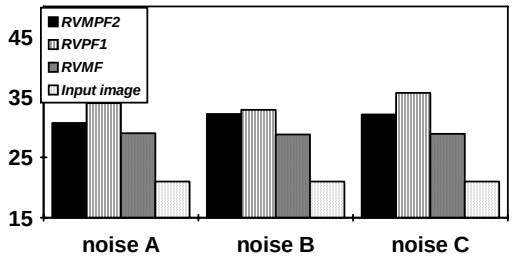

*Clown* image




*Lena* image

Figure 6.14. A comparison between RVMPF2 and RVMPF1 in *PSNR* for corrupted *Clown* image and *Lena* image, with various noise probability of Type C.

*PSNR* [dB]




*PSNR* [dB]

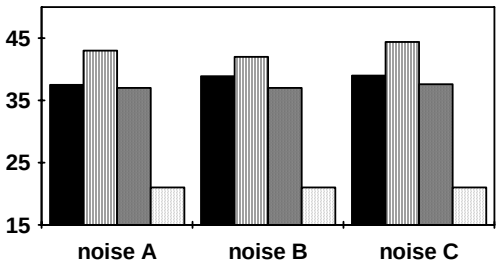



Figure 6.15. A comparison between RVMPF2, RVMPF1 and RVMF, with noise probability of $p$ = 5% is applied to *Clown* image (left) and *Lena* image (right).

a)

b)

Figure 6.16a. A selected frame from a television satellite receiver before and after processing, corrupted frame with comet-like impulse noise (top), an output of two-dimensional recursive vector median filter by using of progress-wise window (bottom).

c) 

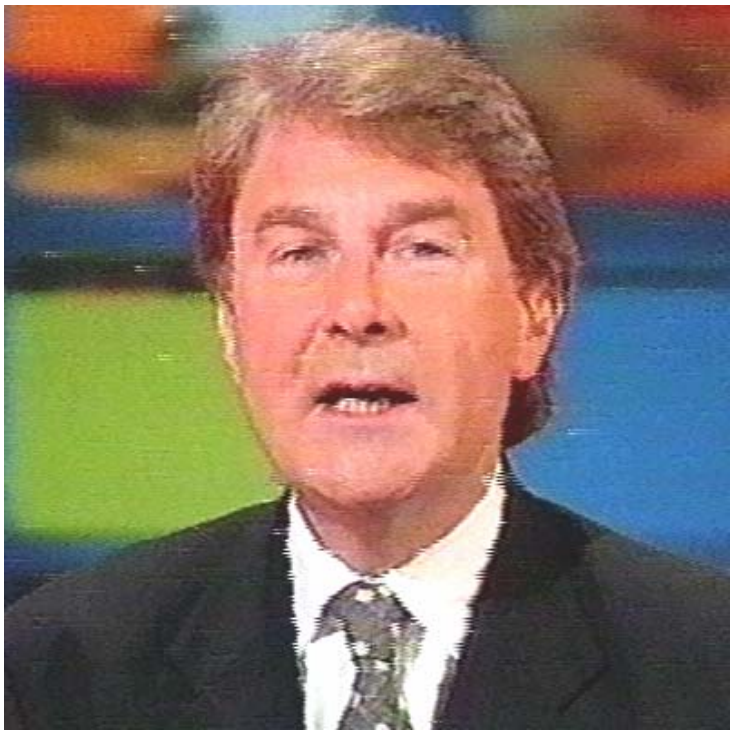
d) 

Figure 6.16b. A selected frame from a television satellite receiver before and after processing: an output of two-dimensional recursive vector median filter when an interlace-wise window is used (top), and an output of three-dimensional recursive vector median with prediction error processing when an interlace-wise window is used with adaptive threshold $\alpha_E$ (bottom).

# 7. Final remarks and conclusions

In this dissertation, filters with prediction error processing have been examined as a tool for denoising of color images and video sequences. In the current implementation, the proposed filters have been used for rejection of scratches and impulse noise from static color images, interlaced color video sequences, and progressive color video sequences. Prediction error processing has been shown as a very effective and flexible ability for this task. The major steps of this dissertation were:

- A decision-based filter is used for removing the impulse noise from various images while preserving edges and fine textures.
- The modified filter is developed to preserve lines while keeping the same performance of the first filter.
- Another approach is used that the processed pixel is not used to estimate the output value when it is detected as a corrupted pixel. This approach gave an advantage to increase the performance of the filter.
- The filter is applied to the progressive video sequences.
- Then, the filter is extended to motion-compensated three-dimensional filter.
- After that, the filter is developed to treat with interlaced television signals.
- For each step, there were suggestions supported by practical results (as shown in Chapter 5 and 6) to choose the proper filter parameters and to increase the filter performance.

Clear conclusions from the experimental results are:

- Application of structures with prediction error processing posses excellent noise removing capabilities as a result of their use of median-based filters as predictors,
- Application of a scheme with prediction error processing results in a significant improvement of subjective quality of all considered filters: scalar and vector median, two-dimensional and three-dimensional with motion compensation,
- Detailed inspection of test images and video sequences proves significantly better reproduction of small details and fine textures in color pictures,
- The improvement resulting from application of prediction error processing is sometimes even more significant than that resulting from application of three-dimensional processing with motion compensation instead of two-dimensional processing. Nevertheless the computational cost of prediction error processing is negligible while the computational cost of motion estimation is enormous and greater than that of median filter itself. Of course, three-dimensional motion-compensated filters with prediction error processing can produce images with the best quality.

It is possible to improve the performance of prediction error processing filters using different kind of features for future work and developments:

- The adaptive algorithm could be developed to treat with other types of noise such as a Gaussian noise or other image artifacts. This could be done by choosing other types of nonlinear filters, which have an ability to detect such a type of noise.
- Motion-compensation is an effective technique to reduce temporal redundancy between frames of a video sequence. In this work, full-search block-matching technique is used for motion-estimation in order to use it in motion-compensation. This technique has a simple implementation and high accuracy but it takes quite long time to calculate the motion-vectors. To speedup the computation of motion-estimation while keeping the motion-estimation accuracy as high as possible other techniques could be used, e.g. cross-search or three-step search motion-estimation.

# References


[ABB00a] J. Abbas, M. Domański, Median-based filters with prediction error processing for video restoration, IEEE International Symposium on Circuit and Systems, Geneva, Switzerland, 2000, (in print).

[ABB00b] J. Abbas, M. Domański, *Rejection of scratches from video by use of filters with prediction error processing,* in: Signal Processing X: Theories and Applications, Tampere, Finland, 2000, (to be published).

[ABB00c] J. Abbas, M. Domański, *A family of efficient nonlinear filters for video restoration,* 6th International Conference on Computer Graphics and Image Processing, Podlesice, 2000, (in print).

[ABB98a] J. Abbas, M. Domański, *Stable nonlinear filters with spatial prediction*, in: Signal Processing IX: Theories and Applications, S. Theodoridis, I. Pitas, A. Stouraitis, N. Kalouptsidis (editors), Typorma, Greece, 1998, pp. 28-31.

[ABB98b] J. Abbas, M. Domański, *Nonlinear recursive filters for image processing*, in: IWSSIP '98, 5th International Workshop on System, Signals, and Image Processing, Zagreb, 1998, pp. 28-31.

[ABB98c] J. Abbas, M. Domański, *Nieliniowe filtry rekursywne do usuwania zniekształcen z obrazów kolorowych*, Krajowe Sympozium Telekomunikacji, Bydgoszcz, 1998, pp. 120-124.

[ABB98d] J. Abbas, M. Domański, *Image denoising using nonlinear two-dimensional filters with prediction error processing*, XXI National Conference of Circuit Theory and Electronics Networks, Poznań, 1998, pp. 431-436.

[ABB98e] J. Abbas, *Motion-compensated median filters for video restoration*, Poznańskie Warsztaty Telekomunikacyjne - PWT '98, Poznań, 1998, pp. 3.1/1 - 3.1/4.

[ABB99a] J. Abbas, M. Domański, *Motion-compensated nonlinear filters for video restoration*, Proceedings of the SPIE: Nonlinear Image Processing X, vol. 3646, San Jose, CA, 1999, pp. 217-227.

[ABB99b] J. Abbas, M. Domańnski, *Digital video restoration by use of nonlinear filters*, Poznańskie Warsztaty Telekomunikacyjne - PWT '99, Poznań, 1999, pp. 3.1/1 - 3.1/5.

[ABB99c] J. Abbas, M. Domański, *Adaptive motion-compensated filters for video restoration*, ECMCS '99, EURASIP Conference: DSP for Multimedia Communication and Services, Kraków, 1999.

[ABB99d] J. Abbas, M. Domański, *Median-based three-dimensional filters for video restoration*, IX Krajowe Sympozjum Nauk Radiowych, URSI '99, Poznań, 1999, pp. 135-140.

[ABB99e] J. Abbas, M. Domański, *Vector nonlinear recursive filters for color image denoising*, in: IWSSIP '99, 6th International Workshop on System Signals and Image processing, Bratislava, Slovakia, 1999, pp. 30-33.

[ABR96a] E. Abreu, M. Lightstone, S. K. Mitra, K. Arakawa, *A new efficient approach for the removal of impulse noise from highly corrupted images*, IEEE Trans. on Image Processing, vol. 5, no. 6, 1996, pp. 1012-1025.

[ALP91a] M. B. Alp, Y. Neuvo, *Three-dimensional median filters for image sequence processing*, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 1991, pp. 2917-2920.

[ALP96a] L. Alparone, M. Barni, F. Bartolini, and V. Cappellini, *Adaptively weighted vector-median filters for motion-fields smoothing*, Proceeding of the IEEE International Conference on Acoustic, Speech, and Signal Processing, 1996.

[AMI92a] H. Amir, P. Zamperoni, *Multiform enhancement by means of locally adaptive rank-order filters for the preprocessing of natural images*, in: Signal Processing VI: Theories and Applications, 1992, pp. 1401-1404.

[AND77A] H. C. Andrews, B. R. HUNT, *Digital image restoration*, Prentice-Hall, USA, 1977.

[AST90a] J. Astola, P. Haavisto, Y. Neuvo, *Vector median filters*, Proceedings of the IEEE, vol. 78, no. 4, 1990, pp. 678-688.

[AST99a] D. Astely, B. Ottersten, *The effects of local scattering on direction of arrival estimation with music*, IEEE Trans. on Signal Processing, vol. 47, no. 12, 1999, pp. 3220-3234.

[AUR95a] V. Aurich, J. Weule, *Non-linear Gaussian filters performing edge preserving diffusion*, Proceedings of the 17 DAGM Symposium, Bielefeld, 1995, pp. 538-545.

[BAG96a] D. Bagni, P. Muzio, P. Carrai, V. Riva, S. Vedani, *Motion compensated de-interlacing*, in: Signal Processing VIII: Theories and Applications, 1996.

[BAR95a] M. Bartkowiak, M. Domański, *Vector median filters for processing of color images in various color spaces*, 5th International Conference on Image Processing and Its Applications, 1995, pp. 833-836.

[BAR97a] A. J. Bardos, S. J. Sangwine, *Recursive vector median of colour images,* in: IWSSIP '97, Proceedings of the 4th International Workshop on Systems, Signal, and Image Processing, Poznań, 1997, pp. 187-190.

[BAR99a] S. Barbarossa, A. Scaglione, *Adaptive time-varying cancellation of wideband interferences in spread-spectrum communications based on time-frequency distributions*, IEEE Trans. on Signal Processing, vol. 47, no. 4, 1999, pp. 957-965.

[BAU91a] P. H. Bauer, M. Sartori, *2-D nonlinear IIR-filters for image processing - an exploratory analysis*, IEEE International Symposium Circuits Systems 1991, pp. 416-419.

[BEL97a] S. Bellofiore, L. Karam, W. Metz, T. Acharya, A flexible and user- friendly image quality assessment system, Proceedings of the IASTED International Conference: Signal and Image Processing, 1997.

[BER93a] J. Bernd, *Digital image processing*, Springer-Verlag, Berlin, 1993.

[BLU99a] R. S. Blum, R. J. Kozick, B. M. Salder, *An adaptive spatial diversity receiver for non-Gaussian interference and noise*, IEEE Trans. on Signal Processing, vol. 47, no. 8, 1999, pp. 2100-2111.

[BLU99b] H. Blume, *Nonlinear vector error tolerant interpolation of intermediate video images by weighted medians*, IEEE Trans. on Signal Processing: Image Communication, 14, 1999, pp. 851-868.

[BOV83a] A. C. Bovik, T. S. Huang, D. C. Munson, *A generalization of median filtering using linear combinations of order statistics*, IEEE Trans. on

Acoustics, Speech, and Signal Processing, vol. ASSP-31, no. 6, 1983, pp. 1342-1350.

[BRU99a] R. de Bruin, J. Smits, *Digital video broadcasting*, Artech House, London, UK, 1999.

[CHA77a] H. Chang, J. K. Aggarwal, *Design of two-dimensional recursive filters by interpolation*, IEEE Trans. on Circuit and Systems, vol. CAS-24, no. 6, 1977, pp. 281-291.

[COM92a] M. L. Comer, E. J. Delp*, An empirical study of morphological operators in color image enhancement*, Proceedings of the SPIE: Image Processing Algorithms and Techniques III, vol. 1657, 1992, pp. 314-325.

[CON97a] C. Connolly, T. Fliess, *A study efficiency and accuracy in the transformation from RGB to CIELAB color space*, IEEE Trans. on Image Processing, vol. 6, no. 7, 1997, pp. 1046-1048.

[COY91a] E. J. Coyle, M. Gabbouj, J. -H. Lin, *From median filters to optimal stack filtering*, IEEE International Symposium Circuits and Systems, 1991, pp. 9-12.

[DAM96a] P. Dambacher, *Digital broadcasting*, J. E. Flood, C. J. Hughes, J. D. Parsons (series editors), IEE Telecommunications Series, vol. 45, 1996.

[DES93a] S. Desmet, B. Denknuydt, N. Li, L. Van Eycken, A. Oosterlink*, A simple algorithm to extract realistic motion field out of video sequences*, Proceedings of the EOS-SPIE International Symposium on Fiber Optics, Networks, and Video Communications, Berlin, Germany, vol. 1977, 1993, pp. 248-254.

[DOM86a] M. Domański*, 2-D digital $l_1$-pseudopassive filters,* in: Signal Processing III: Theories and Applications, I. Young et al. (editor), Elsevier Science, 1986, pp. 721-724.

[DOM89a] M. Domański, A. Fettweis, *Pseudopassive 2-D recursive digital filters for image processing*, International Journal on Circuit Theory and Applications, vol. 17, 1989, pp. 191-195.

[DOM92a] M. Domański, *Synthesis of inherently stable and low- sensitive 2-D IIR filters using the $l_1$-passivity concept*, in: Signal Processing VI: Theories and Applications, 1992, pp. 1161-1164.

[DOM94a] M. Domański, *Some results in nonlinear $l_1$-passive digital filters,* XIII Symposium on Electromagnetic Phenomena in Nonlinear Circuits, Poland, 1994, pp. 515-520.

[DOM96a] M. Domański, *Solutions of nonlinear problems in multidimensional digital filter design via circuit-theoretical analogy*, XIV Symposium on Electromagnetic Phenomena in Nonlinear Circuits, Poznań, 1996, pp. 327-332.

[DUB93a] E. Dubois and J. Konrad, *Estimation of 2-D motion fields from image sequences with application to motion-compensated processing*, in: Motion Analysis and Image Sequence Processing (M. Sezan and R. Lagendijk), Kluwer, 1993.

[DUB93b] E. Dubois and J. Konrad, *Motion estimation and motion-compensated filtering of video signals*, Proceedings of the IEEE International Conference on Acoustics, Speech, Signal Processing, Minneapolis, MN, 1993, pp. I-95 - I-98.

[DUB94a] E. Dubois, G. De Haan, T. Kurita, *Special issue on motion estimation and compensation technologies for standard conversion*, Signal Processing: Image Communication, 6, 1994.

[DUD84a] D. E. Dudgeon, R. M. Mersereau, *Multidimensional digital signal processing*, Prentice-Hall, USA, 1984.

[FAI97a] M. D. Fairchild, *Color appearance models*, Addison-Wesley, Reading, MA, 1997.

[GAR98a] L. Garcia-Cabrera, P. L. Luque-Escamilla, J. Martinez-Aroza, A. M. Robles-Perez, R. Roman-Roldan, *Two pixel-preselection methods for median-type filtering*, IEE Proceedings: Vision, Image, and Signal Processing, vol. 145, no. 1, 1998, pp. 30-40.

[GAS98a] A. Gasteratos, I. Andreadis, Ph. Tsalides, *Fuzzy soft mathematical morphology*, IEE Proceedings: Vision, Image, and Signal Processing, vol. 145, no. 1, 1998, pp. 41-49.

[GIR96a] B. Girod, K. B. Younes, N. Faerber, E. Steinbach*, Recent advances in mobile video communications*, Proceedings of the IEEE VLSI Signal Processing, San Francisco, USA, 1996, pp. 3-12.

[GON92a] R. Goñzalez, R. Winz, *Digital Image Processing*, Addison-Wesley, Reading, MA, 1992.

[HAR95a] R. C. Hardie, C. G. Boncelet, *Gradient-based edge detection using nonlinear edge enhancing prefilters*, IEEE Trans. on Image Processing, vol. 4, no. 11, 1995, pp. 1572-1577.

[HAR97a] N. R. Harvey, S. Marshall, *Application of non-linear image processing: digital video restoration*, IEEE Workshop on Nonlinear Signal and Image Processing, USA, 1997.

[HER95a] N. Herodotou, A. N. Venetsanopoulos, *Colour image interpolation using nonlinear filters*, Proceedings of the International Conference on Digital Signal Processing, Cyprus, 1995, pp. 620-625.

[HAS97a] B. G. Haskell, A. Puri, A. N. Netravali, *Digital video: an introduction to MPEG-2*, Chapman and Hall, USA, 1997.

[HOH81a] K. H. Hohne, *Digital image processing in medicine*, Springer-Verlag, Berlin, 1981.

[IMA96a] G. R. Arce, P. Maragos, Y. Neuvo, I. Pitas (editors), *Special issue on nonlinear image processing*, IEEE Trans. on Image Processing, vol. 5, no. 6, 1996.

[ITU94a] ITU-R Recommendation, BT.500-6, *Methodology for the subjective assessment of the quality of television pictures*, 1994 BT Series Volume: Broadcasting Service (Television), International Telecommunication Union, 1994, pp. 348-370.

[JAI89a] A. K. Jain, *Fundamentals of digital image processing*, Prentice-Hall, USA, 1989.

[JAY84a] N. Jayant, P. Noll, *Digital coding of waveforms*, Prentice-Hall, USA, 1984.

[KAR95a] M. Karaman, M. A. Kutay, G. Bozdagi, *An adaptive speckle suppression filter for medical ultrasonic imaging*, IEEE Trans. on Medical Imaging, vol. 14, no. 2, 1995, pp. 283-292.

[KIM95a] S. R. Kim, A. Efron, *Adaptive robust impulse noise filtering*, IEEE Trans. on Signal Processing, vol. 43, no. 8, 1995, pp. 1855-1866.

[KIN89a] R. King, M. Ahmadi, R. Gorgui-Naguib, A. Kwabwe, M. Azimi-Sadjadi, *Digital filtering in one and two dimensions: design and Applications*, Plenum Press, New York, USA, 1989.

[KOK96a] A. Kokaram, *Detection and removal of line scratches in degraded motion picture sequences,* in: Signal Processing VIII: Theories and Applications, 1996.

[KOV97a] J. Kovaceviæ, R. J. Safranek, E. M. Yeh, *Deinterlacing by successive approximation*, IEEE Trans. on Image Processing, vol. 6, no. 2, 1997, pp. 339-344.

[KUN84a] A. Kundu, S. K. Mitra, P. P. Vaidyanathan, *Application of two-dimensional generalized mean filtering for removal of impulse noises from images*, IEEE Trans. on Acoustics, Speech, and Signal Processing, vol. ASSP-32, no. 3, 1984, pp. 600-609.

[LAB99a] F. Labeau, L. Vandendorpe, B. Macq, *Gaussian modeling for channel error diagnosis in image transmission*, ICASSP '99, IEEE International Conference on Acoustics, Speech, and Signal Processing, Phoenix, USA, 1999, pp. 2455-2458.

[LAT89a] B. P. Lathi, *Modern digital and analog communication systems*, Holt, Rinehart and Winston, USA, 1989.

[LEE98a] Y. H. Lee, S. J. Kim, *Robust sequential detectors employing an order statistics prefilter*, IEE Proceedings: Vision, Image, and Signal Processing, vol. 145, no. 2, 1998, pp. 82-87.

[LEU97a] L. T. Leung, L. M. Po, *Low bit-rate video coding using 2-D adaptive sampling motion compensation*, Proceedings of the IEEE International Symposium on Circuits and Systems, vol. 2, Hong Kong, 1997, pp. 1169-1172.

[LIN88a] H. M. Lin, A. N. Wilson, *Median filters with adaptive length*, IEEE Trans. on Circuit and Systems, vol. 35, no. 6, 1988, pp. 675-390.

[LIN93a] J. N. Lin, X. Nie, R. Unbehauen, *Two-dimensional LMS adaptive filter incorporating a local-mean estimator for image processing*, IEEE Trans. on Circuits and Systems - II: Analog and Digital Signal Processing, vol. 40, no. 7, July 1993, pp. 417-428.

[LIN97a] C. T. Lin, C. F. Juang, *An adaptive neural Fuzzy filter and its applications*, IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics, vol. 27, no. 4, 1997, pp. 635-656.

[LU92a] W. S. Lu, A. Antoniou, *Two-dimensional digital filters*, M. Dekker, New York, 1992.

[LUK82a] F. X. J. Lukas, Z. L. Bodrikis, *Picture quality prediction based on a visual model*, IEEE Trans. on Communications, vol. 30, no. 7, 1982, pp. 1679-1700.

[LUT99a] A. C. Luther, *Video recording technology*, Artech House, Norwood, MA, 1999.

[MAC92a] M. D. Macleod, *Non-linear recursive smoothing filters and their use for noise floor estimation*, Electronics Letters, vol. 28, no. 12, 1992, pp. 1952-1953.

[MAC94a] M. D. Macleod, *Performance analysis of simple non-linear recursive smoothing filters*, Electronics Division Colloquium on Nonlinear Filters, The Institution of Electrical Engineers, London, UK, 1994.

[MAD97a] V. K. Madisetti, *The digital signal processing handbook*, CRC Press and IEEE Press, 1997.

[MIT93a] S. K. Mitra, J. F. Kaiser (editors), *Handbook of digital signal processing*, J. Wiley and Sons, USA, 1993.

[NET88a] A. N. Netravali, B. G. Haskell, *Digital pictures*: *representation and compression*, Plenum Press, New York, USA, 1988.

[NIE98a] M. Nieniewski, *Morfologia matmatyczna w przetwarzaniu obrazów*, Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1998.

[OLI97a] J. L. Olives, B. Lamiscarre, M. Gazalet, *Optimization of electro-optical imaging system with an image quality measure*, EI '97, Electronic Imaging: Science and Technology, IA&T/SPIE's Symposium, San Jose, 1997.

[PAN94a] K. K. Pang, T. K. Tan, *Optimum loop filter in hybrid coders*, IEEE Circuits and Systems for Video Technology, vol. 4, no. 4, 1994, pp. 158-167.

[PAR99a] J. M. Park, W. J. Song, W. A. Pearlman, *Speckle filtering of SAR images based on adaptive windowing*, IEE Proceedings: Vision, Image, and Signal Processing, vol. 146, no. 4, 1999, pp. 191-197.

[PIT88a] I. Pitas, A. N. Venetsanopoulos, *A new filter structure for the implementation of certain classes of image processing operations*, IEEE Trans. on Circuits and Systems, vol. 35, no. 6, 1988, pp. 636-647.

[PIT90a] I. Pitas, A. Venetsanopoulos, *Nonlinear digital filters*, Kluwer, UK, 1990.

[PLA98a] K. N. Platanitis, D. Androoutsos, A. N. Venetsanopoulos, *Color image processing using adaptive vector directional filters*, IEEE Trans. on Circuits and Systems-II: Analog and Digital Processing, vol. 45, no. 10, 1998, pp. 1414-1419.

[RAM88a] G. F. Ramponi, G L. Sicuranza, W. Ukovich*, A Computational method for the design of 2-d nonlinear Volterra filters*, IEEE Trans. on Circuit and Systems, vol. 35, no. 9, 1988, pp. 1095-1102.

[RAM97a] G. Ramponi, C. Moloney, *Smoothing speckled image using an adaptive rational operator*, IEEE Signal Processing Letters, vol. 4, no. 3, 1997, pp. 68-71.

[SAI99a] T. Saito, T. Komatsu, T. Ohuchi, T. Hoshi, *Practical nonlinear filtering for removal of blotches from old film*, ICIP '99, International Conference on Image Processing, 1999.

[SAN98a] S. J. Sangwine, R. E. N. Horne, *The colour processing handbook*, Chapman and Hall, UK, 1998.

[SAW96a] J. Sawicki, *Eliminating impulse noise using adaptive filtering algorithm*, Proceedings of the 19th National Conference: Circuit Theory and Electronics Circuits, Krynica, 1996, pp. II/417-422.

[SAW97a] J. Sawicki, *Reducing impulsive noise in adaptive nonlinear circuits*, in IWSSIP '97, Proceedings of the 4th International Workshop on Systems, Signal, and Image Processing, Poznań, 1997, pp. 21-24.

[SCH86a] R. L. Scheaffer, J. T. McClave, *Probability and statistics for engineers*, PWS-Kent, Boston, USA, 1986.

[SMI97a] S. M. Smith, J. M. Brady, *Susan - a new approach to low level image processing*, International Journal of Computer Vision, 23, 1, 1997, pp. 45-78.

[SPI72a] M. R. Spiegel, *Statistics*, McGraw-Hill, UK, 1972.

[STE92a] R. L. Stevenson, S. M. Schweizer, *Nonlinear filtering structures for image smoothing in mixed-noise environments*, Journal of Mathematical Imaging and Vision, vol. 2, no. 2/3, 1992, pp. 137-157.

[SUC94a] R. Sucher, *Removal of impulse noise by selective filtering*, Proceedings of the IEEE International Conference on Image Processing, vol. 2, Austin, 1994, pp. 502-506.

[SUC96a] R. Sucher, *A distortion measure for impulse noise in images*, Proceedings of the IEEE Workshop on Image and Multimedia Digital Signal Processing, Belize, 1996, pp. 62-63.

[SUN91a] S. Ko, Y. H. Lee, *Centre weighted median filters and their applications to image enhancement*, IEEE Trans. on Circuit And Systems, vol. 38, no. 9, 1991, pp. 984-993.

[SUN92a] T. Sun, Y. Neuvo, *A simple synthesis method for weighted median filters*, in: Signal Processing VI: Theories and Applications, 1992, pp. 1405-1408.

[SUN94a] M. O. Sunay, M. M. Fahmy, *An orthogonal approach to the spatial-domain design of 2-D recursive and nonrecursive nonlinear filters*, IEEE Trans. on Circuit and Systems - II: Analog and Digital Signal Processing, vol. 41, no. 10, 1994, pp. 669-677.

[SUN95a] T. Sun, M. Gabbouj, Y. Neuvo, *Analysis of two-dimensional center weighted median filters*, Multi-Dimensional Systems and Signal Processing, 6, 1995, pp. 159-172.

[SUN95b] C. Sun, P. D. Rabinwitz, *Recursive approaching signal filter*, IEEE Signal Processing Letters, vol. 2, no. 5, 1995, pp. 85-88.

[TEK95a] A. M. Tekalp, *Digital video processing*, Prentice-Hall, USA, 1995.

[TSA88a] T. Tsai, D. Anastassiou, *Stable nonlinear recursive filters for edge enhancing noise smoothing of images*, IEEE International Symposium on Circuits and Systems, 1988, pp. 2561-2564.

[TSA88b] T. Tsai, D. Anastassiou, *Nonlinear recursive filters and applications to image processing*, IEEE International Conference on Acoustics, Speech, and Signal Processing, 1988, pp. 828-831.

[VAN91a] R. E. Van Dyck, S. A. Rajala, *Sensitivity to color errors introduced by processing in different color spaces*, IEEE Workshop on Visual Processing and Communications, Taiwan, China, 1991, pp. 192-195.

[VEL98a] T. L. Veldhuizen, M. E. Jernigan, *Grid filters for local nonlinear image restoration*, International Conference on Acoustics, Speech, and Signal Processing, Seattle, Washington, 1998.

[VIE94a] T. Viero, Y. Neuvo, *Three-dimensional median-related filters for color Image sequence filtering*, IEEE Trans. on Circuit and Systems for Video Technology, 4, 1994, pp. 129-142.

[WAN99a] Z. Wang, D. Zhang, *Progressive switching median filter for the removal of impulse noise from highly corrupted images*, IEEE Trans. on Circuit and Systems - II: Analog and Digital Signal Processing, vol. 46, no. 1, Jan 1999, pp. 78-80.

[WAT99a] H. Watabe, Y. Arakawa, K. Arakawa*, Nonlinear filters for multimedia applications*, ICIP '99, International Conference on Image Processing, 1999.

[YAN94a] R. Yang, M. Gabbouj, Y. Neuvo, *Fast algorithms for analyzing and designing weighted median filters,* in: Signal Processing VII: Theories and Applications, M. J. J. Holt, C. F. N. Cowan, P. M. Grant, W. A. Sandham (editors), Lausanne, Switzerland, 1994, pp. 1559-1562.

[YIN96a] L. Yin, R. Yang, M. Gabbouj, Y. Neuvo, *Weighted median filters: a tutorial*, IEEE Trans. on Circuit and Systems - II, vol. 43, No. 3, 1996, pp. 157-192.

[YOO99a] J. Yoo, K. L. Fong, J. J. Huang, E. J. Coyle, G. B. Adams, *Fast algorithms for designing stack filters*, IEEE Trans. on Image Processing, vol. 8, no. 7, 1999.