A simulation study of the influence of vector and scalar filters on the matching precision of stereo images

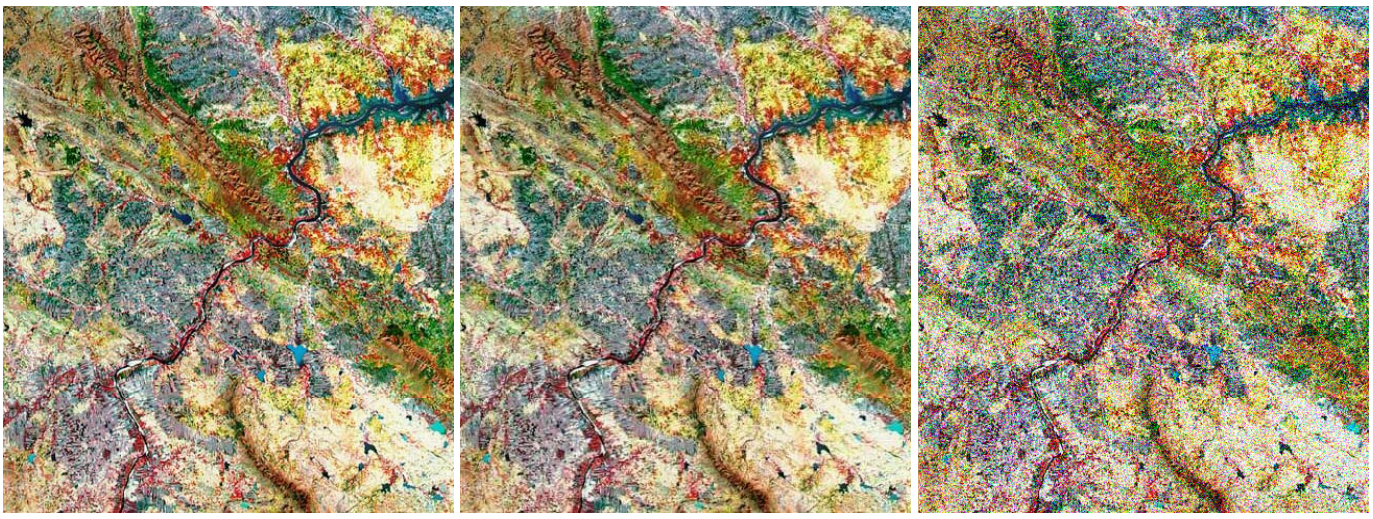
JERZY SIUZDAK, TOMASZ CZARNECKI
Institute of Telecommunications
Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warszawa
POLAND

Abstract:-Investigated is the problem of points matching in stereo images (both color and monochrome). The correlation method is applied to the areas around the points being matched. The influence of noise on the matching errors and the error reduction by vector and scalar filters are analyzed. The obtained results indicate that there is only a small difference between the matching precision of color and monochrome images as well as that only two of several analyzed filters are suitable for the precision increase. These filters are an adaptive filter and an impulse noise filter designed by the authors.

Keywords:-Image processing, stereo images, vector filters, scalar filters, correlation, points matching.

1 Introduction

The basic principle used in the extraction of depth (height) from the pair of stereo images is triangulation. In this technique a correspondence needs to be established between points (generally: features) from two images that correspond to some physical points (features) in space [1]. Then, provided that geometric parameters (such as camera spacing and focal length) are known the depth can be reconstructed. The main steps involved in the extraction of structure from stereo are: image preprocessing, establishing correspondence, and recovering depth. Little is known, however, about the influence of preprocessing on the precision of points (feature) matching. An exception is the paper of Abramson and Schowengerdt, [2], where the authors evaluated the performance of six edge preserving noise smoothing filters. The goal of the current work is to find out how the initial preprocessing affects the precision of points matching. We shall treat both the monochrome and color images and determine how the noise influences this precision. We shall compare the mean matching error reduction obtained by the employment of various types of vector and scalar filters. In this work we shall use an area based matching technique. The cross correlation is calculated between the local neighborhood intensities (colors) of a pixel in one image and a corresponding neighborhood intensities (colors) in the other image. The search for the cross correlation function maximum is limited to the epipolar lines.



a b c

Fig. 1. a) Example of an image used in calculation, b) generated stereo image, c) image after noise addition

2 Theoretical background

The image processing as used in this work involved the following steps: image pair generation, noise addition, filtering, area based points matching, and matching error calculation. These steps are described in the sequel.

2.1 Image generation

To attain the objects of this work pairs of stereo images are required. The first image is a real satellite image of an area. The example of an image used in this paper is shown in Fig.1a. The second (stereo) image was generated artificially from the first image in the way described below. First, the disparity z(x,y) between images was assumed to have the following form

$$z(x,y) = N \sin k_1 x \sin k_2 y \qquad (1)$$

Here, N is the disparity amplitude, k_1, k_2 are the spatial frequency parameters, and x, y are the pixel coordinates. The disparity was limited to one coordinate (x) and the other coordinate (y) was preserved due to the epipolar constraint. Second, the stereo image was generated based on this disparity. The values of the RGB(x,y) components of the resulting image were calculated based on the RGB of pixels located near (x+z(x,y), y) in the original image. Since the disparity z(x,y) may not be integer an appropriate interpolation was employed. It did not lead to the color changes in the generated image (see Fig. 1b).

2.2 Noise generation

In order to verify the performance of various filters two types of noise were generated and added to the image colors (intensities): a white gaussian noise of power of δ^2 that modeled the noise of opto-electronic converter (camera) and impulse noise that stood for relatively rare (with probability r<<1) interference event of high amplitude δ. In each case the noise power depended on the δ parameter and it is expressed via the color (intensity) control value in the range (0, 255). The noise was always added to **both** images of the processed pair but its realization was different in each image. An example of the noisy image is shown in Fig 1c.

2.3 Filtering

Many types of filters were tested including those evaluated in [2]. The scalar filters investigated in [2] were generalized to vector filters and novel vector filters were tested as well. A high performance vector filter was proposed by the authors. We shall briefly describe the tested filters in the sequel.

Median filters (MF)

The output of the vector median filter is that input vector whose sum of distances from other vectors in the processed window (usually 3x3 or 5x5) is the smallest. The choice of metric determines the filter properties. The following metrics were used: L_1 [3] (rgb1), L_2 euclidean (rgb2), vector directional [4] (rgb3), content based [5] (rgb4).

For a monochrome image the median filtering involves ordering pixel intensities in the processed window from the smallest to the largest and selecting the middle intensity (5th for a 3x3 window and 13th for a 5x5 window).

Order statistic filters (OSF)

These filters are generalized median filters. Here again, we use the metrics that are mentioned above. A value of d_i is defined for each RGB vector in the 3x3 (5x5) window. It is the sum of distances between a given i-th pixel and all other pixels in the window. The output of the specific filter used here is defined as [4] a weighted sum of RGB values of pixels in the processed window where the weights are given by

$$w_j = \frac{d_{\max} - d_j}{d_{\max} - d_{\min}} \qquad (2)$$

Here d_{max} is the maximum and d_{min} minimum value of d_j (j=1..N, N is the window size).

Adaptive filter (AF)

The adaptive filter used in this work is based on the Bayesian estimator in which the conditional probability (known in Bayesian model) is approximated by exponential functions determined from the vector signals in the window around the processed point [6].

Nagao filter (NF)

This filter searches for a hypothetical edge within a 5x5 window. To this end an elongated bar mask is rotated around the central pixel and a mean and a variance of the colours (intensities) of pixels within the actual mask position are calculated [7]. The output of the filter is the mean of the colors (intensities) of the pixels within the subarea having the smallest variance. The filter was originally used for monochrome images and it is extended here to color (vector) images.

K-nearest neighbors filter (KNNF)

The output of this filter is an unweighted average of K-pixels taken from 3x3 (or 5x5) window [8]. Included are only the pixels with the smallest color vector distances (L_1 metric for color images) or absolute intensity difference (for monochrome images) from the central pixel. Here K=4 and the window size was 3x3.

Gradient inverse filter (GIF)
The output of this filter is a weighted neighbourhood (3x3) average where the weight of each pixel is inversely proportional to the colour vector (or intensity for monochrome image) distance between that pixel and the central pixel of the window [9]. The pixel distance for the monochrome image is the absolute intensity difference and for colour image is given by any of the mentioned metrics.

Double-window modified trimmed mean filter (DWMTMF)
This filter used two nested windows of different size (3x3 and 5x5). At each image pixel, the algorithm finds the median (scalar for monochrome and vector for color images) in the smaller window [10]. The output is then calculated as the arithmetic average (of scalars or vectors) of all pixels in the second (larger) window that do not differ (i.e. the distances must be smaller) from the median more than a specified value.

Sigma filter (SF)
This filter is somewhat similar to the previous one but only one window (5x5) is considered. First, one determines the set of all pixels whose colour vectors (intensities) differ from the central pixel less than a specified threshold [11]. If there are more than 3 such pixels then the output is defined as the arythmetic mean of colour vectors (intensities) over this set. Otherwise, the output is the arythmetic mean of the colour vectors (intensities) of eight immediate neighbours of the central pixel.

Extended median filter (EMF)
Here, the values of arithmetic mean and median of color vectors (intensities for monochrome images) are calculated in the 3x3 window [12]. The color vector (intensity) distances from the mean and median are calculated for all pixels in the window. Then, all distances from the mean are added. The same is done with the distances from the median. The filter output is that value (either mean or median) for which this sum is smaller.

Modified median filter (MMF)
A very similar approach is employed in the so called modified median filter [13]. However, instead of the mean the sum of distances is calculated for the central pixel. If this sum differs less than a threshold from the similar sum calculated for the median then the filter output is unchanged (i.e. it is the color or intensity of the central pixel). Otherwise, the output is the median (vector for colors and scalar for monochrome).

Fuzzy set filters (FSF)
These are weighted filters where the weighting factors are calculated based on the fuzzy sets theory [14]. Two such filters were tested here, but they performed very similarly so results for only one are referred to in the sequel.

Impulse noise filter (INF)
This filter was designed by the authors. It was initially meant for the impulse noise filtering but it turned out that it performed equally well for the white noise. Here, a value of d_0 being the sum of the color distances of all pixels from the central pixel, is calculated in the 3x3 window according to the formula

$$d_0 = \sum_j D(RGB_0, RGB_j) \tag{3}$$

Also, a sum P in this window is determined from

$$P = \sum_{i=1}^{9} d_i \tag{4}$$

where d_i is given by a formula similar to (3). The filter output is then

$$R'GB = \begin{cases} [R_0, G_0, B_0] \; for \;\; d_0 \le \beta \; P, \\ \frac{1}{8}\sum_{j=1}^{8}[R_j, G_j, B_j] \;\; for \;\; d_0 > \beta \; P \end{cases} \tag{5}$$

The value of β is experimentally chosen (one value for all images) and the addition extends over all pixels except for the central one. That is, the central pixel is not changed if $d_0<\beta P$. Otherwise, it is replaced with arithmetic mean of all immediate neighbors.

2.4 Matching

In order to find matching points in both images it is necessary to calculate the cross covariance between two square areas: one from the reference image and one from the synthetic image. They may be shifted by p pixels along the x coordinate. This cross covariance for color images is given by

$$Cov(p) = \sum_{i=-k}^{k}\sum_{j=-k}^{k} RGB_1(x+i, y+j) \cdot RGB_2(x+p+i, y+j) - \left(\sum_{i=-k}^{k}\sum_{j=-k}^{k} RGB_1(x+i, y+j)\right) \cdot \left(\sum_{i=-k}^{k}\sum_{j=-k}^{k} RGB_2(x+p+i, y+j)\right) \tag{6}$$

Here (2k+1)(2k+1) is the correlation window size, RGB_1, RGB_2 are the color vectors in corresponding images and a dot denotes the scalar multiplication. The cross covariance for monochrome images is calculated in the similar way except for the fact that the intensities are scalars.

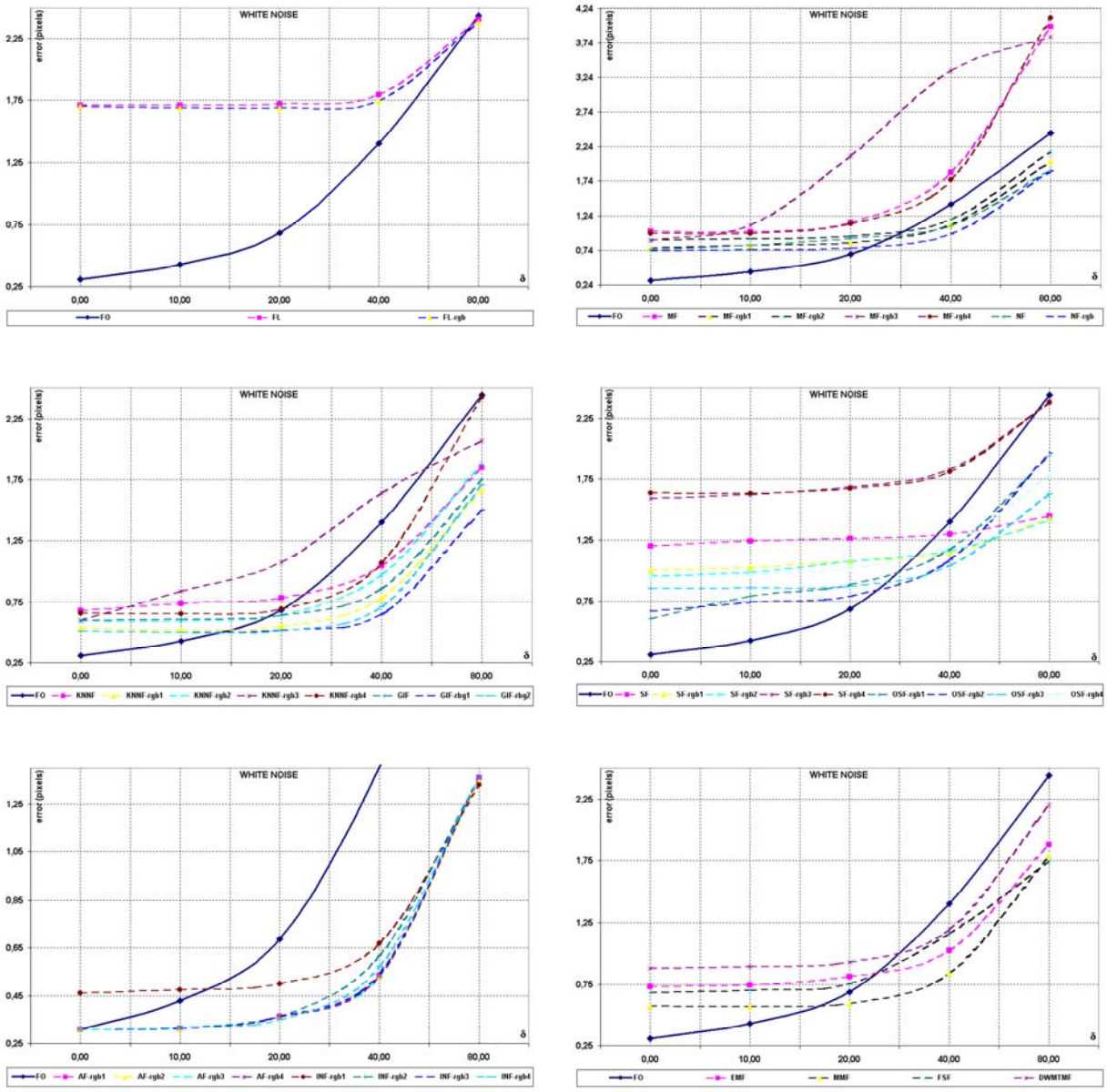


Fig. 2. Matching errors for various filters and white gaussian noise. Solid lines refer to the no filter case and they are used as references; rgbi, i=1,2,3,4 refers to the metric used in vector calculation

The matching precision is a function of the window size. Here we assumed a 15x15 window since its further enlargement did not lead to the substantial precision increase. The cross covariance values are calculated for pε(-10,10) and a maximum is found. The subpixel accuracy of matching is obtained by square interpolation of the cross covariance values near this maximum.

2.5 Matching error determination

The rms matching error was determined based on the difference between the actual value of the shift $z(x_i,y_j)$ and the corresponding value calculated by the correlation method $p'(x_i,y_j)$ according to the formula

$$\sigma = \sqrt{\frac{1}{I \cdot J} \sum_{i=1}^{I} \sum_{j=1}^{J} (p'(x_i, y_j) - z(x_i, y_j))^2} \qquad (7)$$

Here IxJ=625 and the addition extends over a rectangular grid of the pixels.

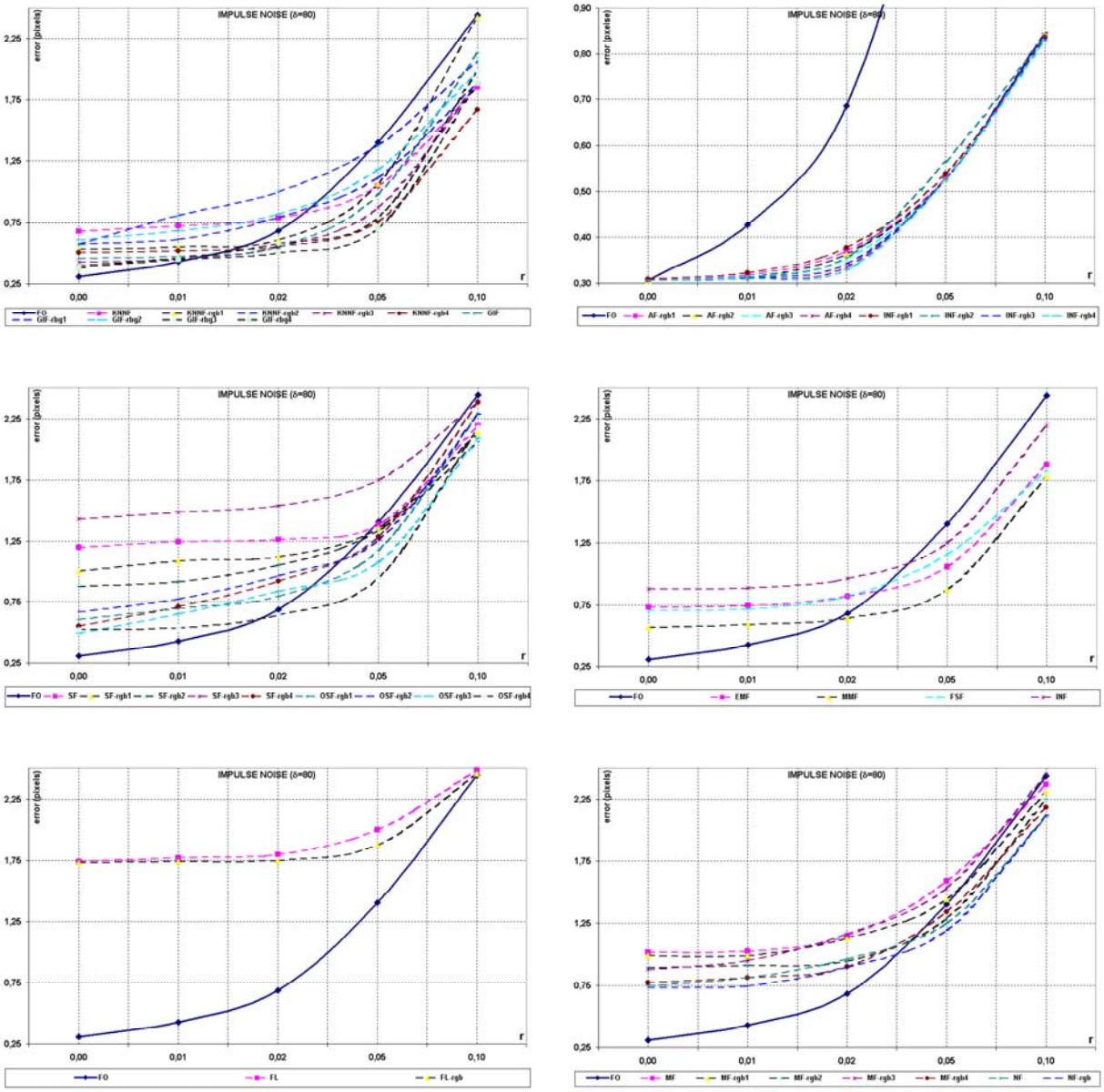


Fig. 3. Matching errors for various filters and impulse noise. Solid lines refer to the no filter case and they are used as references; rgbi, i=1,2,3,4 refers to the metric used in vector calculation

3 Discussion

In order to perform all the operations described in the previous paragraph a special software was written. It is necessary to stress that filter parameter(s) were globally optimized to reach the best precision of matching for all images (i.e. the parameter(s) were the same for all images). Gaussian white and impulse noise influence on the matching precision were tested separately. The noise parameter δ varied between 0 and 80 for the white noise and between 10 and 100 for the impulse noise whereas the parameter r was between 0 and 0.1 for the latter (for comparison the color dynamics was between 0 and 255). The calculations were conducted for various images. However, the results obtained for different images were similar, therefore only those calculated for the image in Fig. 1a are depicted here in Fig. 2 and Fig. 3. Each figure presents rms matching error against the noise parameters. Results obtained with no filter are used as a reference (a solid lane in each image). Filters are denoted by abbreviations used in paragraph 2.3. and the extensions rgb1...rgb4 refer to the vector metric used. All the curves have similar character: the error increases with the noise level increase and the filtering is more suitable for higher noise levels. The following conclusions may be drawn from these results:

1. As compared with scalar filtering the vector filtering (colour images) leads to a small (a few percent) matching precision increase only for some types of filters.

2. Only few employed filters improve the matching precision. This improvement is more pronounced for higher noise levels. For lower noise levels however, many filters can not be recommended as they increase the matching error.
3. Two filters can be recommended for both types of noise and all noise levels, namely the adaptive filter and impulse noise filter. Both filters allow for a vast error reduction (up to three times) for high noise levels and do not cause error increase for low noise levels. Especially the impulse noise filter designed by the authors is worth recommendation: it is fast and has the lowest errors of all filters for the impulse noise.

4 Conclusion

This work was to answer two principal questions:
a) do the vector correlation and filtering lead to the matching precision improvement as compared with the monochrome case?
b) which are the most suitable filters for noise errors reduction?
The answer to the first question is rather negative: whenever the comparison between the scalar and vector processing was possible the errors did not differ much.
It is necessary to stress however, that the best noise properties are obtained for two **vector** filters: the adaptive filter and impulse noise filter which is faster. Only these filters may be recommended as they perform well for various noise types and levels.
However, one should bear in mind that these results are based partly on synthetic images and simulated noise. Further verification requires true stereo images be used.

References:

[1] U.R. Dhond, and J.K. Aggarwal, Structure from stereo- a review. *IEEE Trans. Syst. Man & Cybernetics*, vol. 19, no 6, 1989, pp. 1489-1510
[2] S.B. Abramson, and R.A. Schowengerdt, Evaluation of edge-preserving smoothing filters for digital image mapping, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 48, no 23, 1993, pp. 2-17
[3] J. Astola, P. Haavisto, and Y. Neuvo, Vector median filters. *Proc. IEEE*, vol. 78, 1990, pp. 678-689
[4] K.N. Plataniotis et al., Nearest-neighbour multichannel filter. *Electronics Letters*, vol. 31, no. 22, 1995, pp. 1910-1911
[5] K.N. Plataniotis et al., Content based colour image filters. *Electronics Letters*, vol. 33, no.3, 1997, pp. 202-203
[6] K.N. Plataniotis et al., Adaptive multichannel filters for colour image processing. *Signal processing: Image communication*, vol.11, 1998, pp. 171-177
[7] M. Nagao, and T. Matsuyama, Edge preserving smoothing. *Computer Graphics Image Process*., vol.9, 1979, pp.394-407
[8] L.S. Davis, and A. Rosenfeld, Noise cleaning by iterated local averaging. *IEEE Trans. Syst. Man Cybern*., vol. 8, no 9, 1978, pp. 705-710
[9] D.C.C. Wang, and A.H. Vagnucci, Gradient inverse weighted smoothing scheme and the evaluation of its performance, *Computer Graphics Image Process*., vol. 15, 1981, pp. 167-181
[10] Z. Mao, and R.N. Strickland, Image sequence processing for target estimation in forward-looking infrared imagery, *Opt. Eng*., vol. 27, no. 7, 1988, pp. 541-549
[11] J. Lee, Digital image smoothing and the sigma filter. Comput. *Vision Graphics Image Process*., vol. 24, 1983, pp. 255-269
[12] K. Ostamo, and Y. Neuvo, Vector median operation for color image processing, *Proceed. SPIE*, vol.1247, 1990, pp. 2-12
[13] L.W. Chang, and J.S. Lee, A new hardware implementation of fast vector median filters. *Proceed. SPIE*, vol.2094, 1993, pp. 602-612
[14] K.N. Plataniotis et al., Multichannel filters for image processing. *Signal processing: Image communication*, vol. 9, 1997, pp. 143-158