Pattern Recognition Letters 6 (1987) 155–162
North-Holland

August 1987

A new class of edge-preserving smoothing filters

David HARWOOD, Muralidhara SUBBARAO, Hannu HAKALAHTI and Larry S. DAVIS
Center for Automation Research, University of Maryland, College Park, MD 20742, USA

Received 18 November 1986

Abstract: A new class of edge-preserving noise-cleaning filters are introduced which use both spatial and nearest-neighbor constraints on image pixels to smooth an image. They are simple, fast and good at preserving edges and thin structural details in images. Edges and corners of varying contrasts that may be present in real images have been simulated in synthesized random checkerboards. The performance of the new smoothing filters on the blurred and noisy checkerboards has been quantitatively compared with that of four other filters.

Key words: Image processing, image enhancement.

1. Introduction

A number of edge preserving noise-cleaning non-linear filters have been proposed in the literature [1,2]. The median filter [3], K-nearest neighbor mean filter [4], gradient inverse weighted filter [5] and maximum homogeneity smoothing filter [6,7] are some examples. These methods involve some type of local operation such as selective averaging, weighted averaging, etc., They attempt to remove the effects of noise, while retaining the edges and other structural details in the image.

The purpose of our study was to find appropriate smoothing filters for preprocessing Landsat images. Landsat images have subtle details and thin structures like roads, rivers etc., and are sometimes of low contrast, noisy and blurred (Figure 9); the volume of the data to be processed requires that the smoothing filters should be simple and fast. The filters should be capable of effectively flattening the interiors of homogeneous regions while simultaneously enhancing blurred edges and corners. Our efforts in this direction resulted in the development of a new class of smoothing filters, which we call the Symmetric Nearest Neighbor (SNN) filters. The SNN filters described here avoid exhaustive search of the location and orientation of straight edges within a square window corresponding to the size of the smoothing operator by using spatial-symmetry and nearest-neighbor constraints. The SNN filters are described in Section 3 and the other filters used in our study are described in the next section.

A simple quantitative comparison of several smoothing filters requires knowledge of the underlying ideal image of the noisy and blurred test sample. For this purpose, we used checkerboards with the mean gray values of the checkers drawn randomly from a normal distribution. Test samples were obtained by adding Gaussian noise to these images. Experiments were also carried out on the blurred versions of these images, obtained by convolving these images with a center-weighted 3×3 mask. The basic idea is to simulate edges and corners of varying contrasts encountered in real images.

Note: The support of the Texas A&M Research Foundation under Subcontract L200077 (NASA Contract 9-1664) is gratefully acknowledged.

2. Edge-preserving noise-cleaning filters

A recent comparative study of various edge-

0167-8655/87/$3.50

preserving noise smoothing techniques [1] indicates that the best results are obtained by using the median and the K-nearest neighbor mean filters. We have chosen these two filters and two other filters, the K nearest neighbor median filter and the recently proposed sigma filter [8], to compare with the performance of the SNN filters. Brief descriptions of all the smoothing filters used in our study are given below.

The median filter [3] replaces the center pixel in a square window with the median gray value of the pixels in the window. This filter flattens the interior regions well and sharpens the edges in the first few iterations, but repeated iteration blurs the edges and removes thin details.

The K-nearest neighbor (KNN) mean filter [4] substitutes the center pixel in a square window with the mean gray value of the first K pixels nearest in gray value to the center pixel (the lower the absolute difference between the gray value of the center pixel and a neighboring pixel, the nearer the pixel is to the center pixel). K is a parameter of the filter and was chosen to be the maximum number of pixels (excluding the center pixel) which lie on the same side of a straight edge that passes through the center of the window (5 for a 3×3 filter and 14 for a 5×5 filter). This usually avoids averaging across the edges, thus preventing them from being blurred.

The KNN-median filter is similar to the KNN-mean filter, except that the median of the K selected pixels is substituted as the new value of the center pixel instead of the mean. To some extent, this has the same advantages over the KNN-mean filter as the simple median filter has over the mean filter; it pulls the substituted value to one side or the other of the mean and sharpens the edges better and smears the details less.

The recently proposed sigma filter [8] is motivated by the sigma probability of the normal distribution; it smooths the image noise by averaging only those pixels in the neighborhood which have their gray values within a fixed sigma range from the center pixel. Consequently, edges are preserved and thin details retained. In [8], a comparison of the sigma filter with the gradient inverse filter [5], Nagao's [7] filter and the median filter indicated that it was better than the other filters.

Some of the difficulties associated with the sigma filter are the estimation of the noise standard deviation, the choice of the sigma range for averaging and the question of deciding what to do when there are very few or no pixels within the chosen sigma range. In our experiments, the noise standard deviation was estimated by computing the frequency distribution of the standard deviations in small neighborhoods (of the size of the smoothing filter) over the entire image and taking the mode of the distribution as the noise standard deviation; this gives a reliable estimate of the noise level, provided the proportion of border pixels is low compared to that of the interior pixels. The noise level was recomputed afresh for each iteration of smoothing. Again, as in [8], the sigma range was chosen to be two standard deviations of the noise distribution (normally distributed) on either side of the gray level of the center pixel. This includes about 95.5 percent of the pixels which come from the same population.

A solution suggested in [8] when there are less than a fixed number of pixels in the given sigma range is to replace the center pixel with the mean gray value of the immediate neighbors. In our experiments the fixed minimum of pixels was chosen to be 2 (including the center pixel) for a 3×3 filter and 3 for a 5×5 filter; whenever the number of neighboring pixels in the two-sigma range was less than the fixed minimum, the center pixel was replaced by the mean gray value of the pixels in a 3×3 neighborhood around the center pixel.

3. Symmetric nearest neighbor (SNN) filters

The SNN filters make use of both the spatial and gray value information in the neighborhood of a pixel to be processed. To compute the new gray value for the center pixel in a square window, they select half the number of pixels in the square window by selecting one pixel nearest in gray value to the center pixel from each pair of pixels located symmetrically opposite the center. In case of ties, the mean of the symmetric pair is used.

More formally, for a $(2n+1)\times(2n+1)$ window centered at the pixel (x, y) in the image, from each pair of pixels $\{(x+i, y+j), (x-i, y-j)\}$ where

$-n \le i, j \le +n$,

select $(x+i, y+j)$ if $|g(x, y) - g(x+i, y+j)|$
$< |g(x, y) - g(x-i, y-j)|$;

select $(x-i, y-j)$ if $|g(x, y) - g(x+i, y+j)|$
$> |g(x, y) - g(x-i, y-j)|$;

otherwise select (x, y).

Here, $g(p, q)$ is the gray value of the pixel (p, q). See Figure 1.

The advantage of this filter is that it is simple and fast (since it involves only comparisons) and usually computes statistics based only on those pixels which lie on the same side of a straight edge as the center; interestingly, this is true for *any* position and orientation of the edge through the window whenever the local gray value distributions on either side of the edge are relatively well separated (Figure 1). We have considered two variations of the SNN filter, the SNN-mean and the SNN-median; the SNN-mean filter assigns the mean of the gray values of the set of pixels selected to the center pixel, whereas the SNN-median filter assigns the median of these pixels to the center pixel.

SNN filters more general than the ones described above can be constructed. The basic assumption involved is that, locally (within a window corresponding to the size of the filter), there are at least K pixels in the neighborhood which come from the same gray level distribution as the center pixel and if there are pixels with gray values in this neighborhood which come from other distributions, these distributions are well separated from the center pixel's distribution. When this assumption is valid, then the simple KNN filters can preserve any patterns of K pixels or more within the window of the operator. The SNN filters, in addition to the nearest neighbor constraint, use a spatial constraint to restrict the patterns that are preserved. The SNN filters used in our study use a spatial constraint which corresponds to the preservation of locally straight edges; an example of a SNN filter which can preserve corners (two edges intersecting at right angles) is given in Figure 2.

4. Experiments

Ten checkerboards of size 48×48 were generated, five with checkers of size 4×4 and the other five with checkers of size 8×8. The mean gray values of the checkers were drawn randomly from a normal distribution with mean 128.0 and standard deviation 40.0 (the gray level range of all images were 0 to 255). Zero mean Gaussian noise was added to these checkerboards with standard deviations of 10.0 and 20.0. Experiments were carried out on these samples and also on their blurred

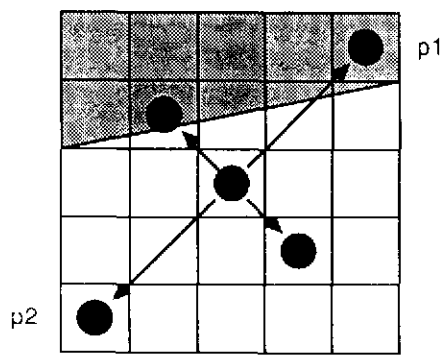

Figure 1. Symmetric Nearest Neighbor algorithm selects one pixel nearest in gray value to the center pixel from every pair of pixels located symmetrically opposite the center (e.g. [p1,p2]). If a window contains a straight gray-level edge between two regions with different gray value distributions, then most of the pixels selected by the Symmetric Nearest neighbor criterion will lie on the same side as the center.

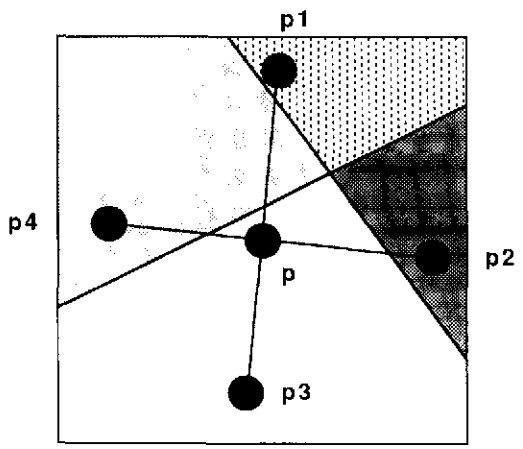

Figure 2. A Symmetric Nearest Neighbor filter designed to preserve edges intersecting at right angles. One nearest pixel from every symmetric quadruple such as (p1,p2,p3,p4) is selected for computing a new gray value for the center pixel.

versions. The samples were blurred (after adding noise) by convolving with the center weighted mask shown in Figure 3.

$$\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$

Figure 3. A mask which approximates Gaussian spread.

All samples were iteratively filtered three times and to avoid border effects, were reduced in size to 32×32 by discarding the outer eight pixels on all sides. Checkerboards with 4×4 checkers were smoothed with 3×3 filters and those with 8×8 checkers were smoothed with 5×5 filters. For each filter, the composite distribution of differences between five filtered images and their corresponding original checkerboards (uncorrupted by noise or blur) was computed. These distributions were thresholded at one standard deviation of the distribution of the differences between the five unfiltered test samples and the original checkerboards. The percentage of pixels below the threshold are given in Tables 1 and 2.

In Table 1, we see that the SNN filters attain their near optimal performance after one or two iterations and then saturate; the same trend was observed for higher numbers of iterations for both the SNN filters and the KNN filters. the performance of the sigma filter and the median filter begins to deteriorate after two or three iterations. Similar trends were observed for the blurred samples in Table 2. The SNN filters are slightly better than the KNN filters for the unblurred samples while the difference between their performance is wider for the blurred samples.

Table 1
Percentage of pixels below the threshold for unblurred noisy pictures

Iteration	Median	KNN median	KNN mean	Sigma	SNN median	SNN mean
(a) Checker size: 4×4, noise std: 10.0.						
1	78.8	77.5	84.6	87.4	89.0	84.8
2	76.1	85.0	86.7	88.6	92.4	86.9
3	72.5	87.4	85.7	70.8	92.6	87.1
(b) Checker size: 4×4, noise std: 20.0.						
1	83.0	77.1	87.0	84.4	87.2	87.1
2	82.3	82.1	89.9	88.6	90.6	89.9
3	79.9	83.9	90.9	89.3	91.0	90.2
(c) Checker size: 8×8, noise std: 10.0.						
1	79.8	89.2	92.3	90.9	95.8	91.2
2	73.7	95.6	93.2	95.2	97.6	91.4
3	67.3	97.2	92.9	94.9	97.6	91.1
(d) Checker size: 8×8, noise std: 20.0.						
1	86.3	87.5	92.7	90.6	93.9	93.0
2	81.8	92.9	94.7	94.2	95.5	94.4
3	77.6	94.5	95.1	94.3	95.4	94.4

Table 2
Percentage of pixels below the threshold for blurred noisy pictures

Iteration	Median	KNN median	KNN mean	Sigma	SNN median	SNN mean
(a) Checker size: 4×4, noise std: 10.0.						
1	51.3	72.0	68.6	61.9	80.8	76.9
2	45.7	76.7	71.2	61.3	89.4	83.8
3	42.6	78.4	70.8	52.8	90.7	85.0
(b) Checker size: 4×4, noise std: 20.0.						
1	48.5	58.8	58.5	55.2	64.4	62.0
2	44.5	61.9	60.9	56.7	69.8	66.7
3	42.0	63.8	61.1	54.8	70.8	67.6
(c) Checker size: 8×8, noise std: 10.0.						
1	62.6	87.5	81.3	71.1	90.5	85.0
2	56.5	92.9	87.0	71.6	95.0	88.3
3	50.0	94.9	88.4	66.4	96.2	89.0
(d) Checker size: 8×8, noise std: 20.0.						
1	54.2	68.4	66.0	63.0	73.7	70.6
2	48.5	75.4	72.5	67.9	80.7	76.7
3	42.2	79.2	75.5	69.3	82.4	79.2

Figures 4 to 8 show the results of smoothing with various filters. We observe that as expected, most of the errors occur at the corners. Figure 9 shows the result of smoothing an actual Landsat image with the filters compared. All pictures have been magnified by a linear factor of 8 to make each pixel visible.

5. Comments on performance and efficiency

The SNN-mean and SNN-median filters are nearly as fast as the mean and median filters respectively. They use a nearest neighbor algorithm, but only with respect to pairs, hence are much faster than the KNN filters which use an

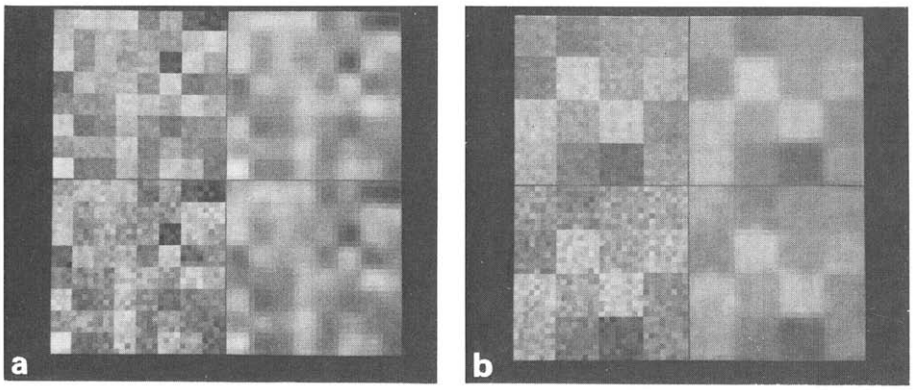

Figure 4. Synthesized 32×32 checkerboards. All pictures are magnified 8 times to make each pixel visible. (a). Samples with 4×4 checkers. The left two are unblurred and at the right are their blurred versions. The top two pictures correspond to noise standard deviation of 10.0 and the bottom two to 20.0. (b) Same as in (a) for 8×8 checkers.

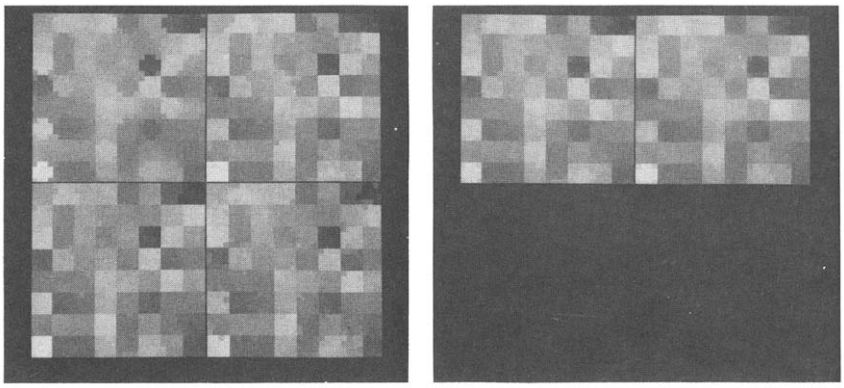

Figure 5. Results of smoothing unblurred 4×4 checkers with a noise standard deviation of 10.0. All samples are iteratively filtered three times with 3×3 filters. (a) Clockwise from top left: Median, KNN-median, Sigma and SNN-median. (b) Left: SNN-mean, right: KNN-mean.

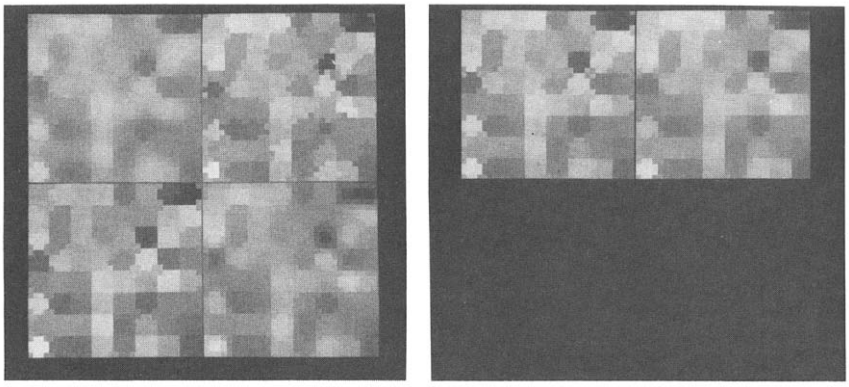

Figure 6. Same as Figure 5, but for the corresponding blurred samples.

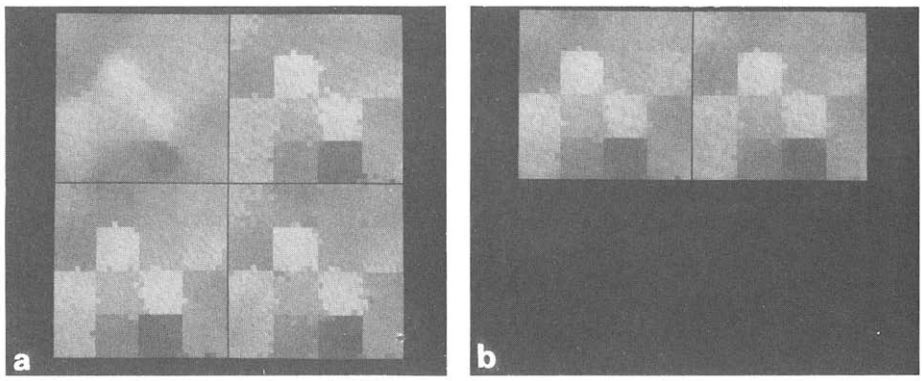

Figure 7. Results of smoothing unblurred 8×8 checkers with a noise standard deviation of 20.0. All samples are iteratively filtered three times with 5×5 filters. (a) Clockwise from top left: Median, KNN-median, Sigma and SNN median. (b) Left: SNN-mean, right: KNN-mean.

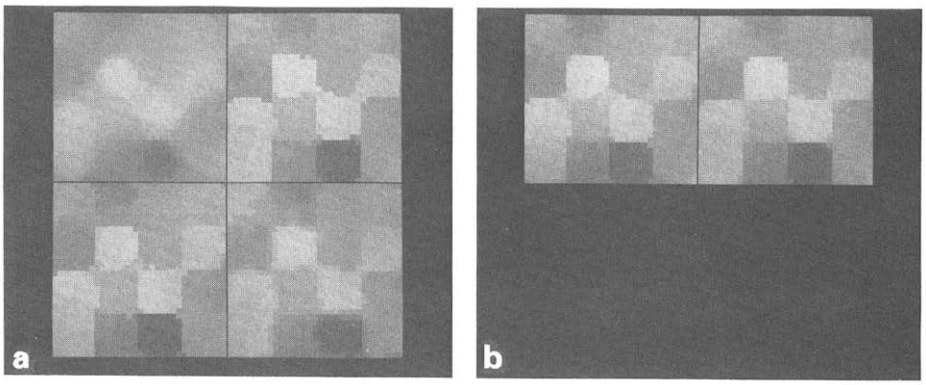

Figure 8. Same as Figure 7, but for the corresponding blurred samples.

ordering algorithm. Also, the SNN filters have more flattening power than the KNN filters in the interiors of regions because they choose nearest values only with respect to pairs, rather than the whole sample. They preserve edges by choosing pixels from the same side and sharpen them better since they must choose points across and away from edges vis a vis KNN filters.

Although the SNN-median filter is marginally better than the SNN-mean filter in terms of edge preservation, computationally the latter is much more efficient. The SNN-mean filter has almost all the desirable characteristics: it is fast, reduces noise and minimizes artifacts (like the mean filter), preserves thin structures, retains edges using a nearest neighbor constraint and converges with iterations (like KNN filters) and sharpens edges and flattens interiors of regions well (unlike KNN filters).

An efficient algorithm used to implement SNN filters which avoids computing absolute differences is outlined in the Appendix.

6. Conclusions

Edges and corners of varying contrasts were simulated in noisy and blurred random checkerboards and the performance of six edge-preserving noise-cleaning filters was compared. The new Symmetric Nearest Neighbor filters introduced in this paper performed better than the other filters; they are simple, fast and retain the edges and details well.

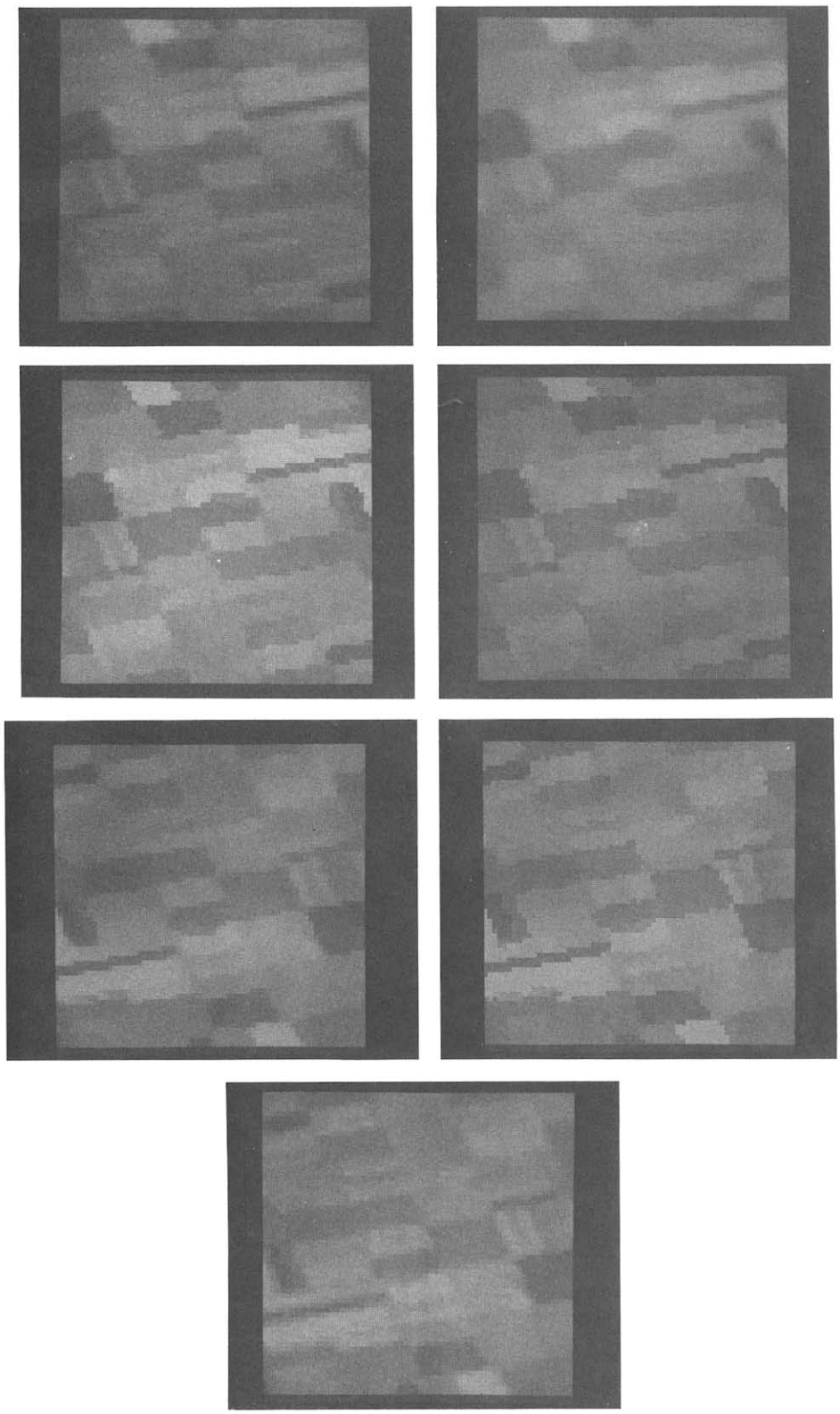

Figure 9. Results of smoothing a Landsat image. Clockwise from top left: Original, Median, KNN-mean, KNN-median, SNN-median and SNN-mean.

Appendix

In the straightforward implementation of SNN filters, the sources of computational overhead are indexing symmetric pairs of pixels (e.g. image$[x+i][y+j]$, image$[x-i][y-j]$) and selecting one pixel from every such pair which is nearest in gray value to the center pixel. In a language like C which allows efficient manipulation of pointers, indexing symmetric pairs can be made just as efficient as indexing pixels for a mean filter. To reduce the overhead associated with the selection of a nearest pixel from a pair, a method which avoids the computation of absolute differences was used. Let c be the gray value of the center pixel and b and d be those of a symmetric pair. First compute $cc = c + c$ for the center pixel. Then from each symmetric pair, select one as follows:

```
if (b+d>cc)
  {if (b>d) select d
   else select b}
else {if (b+d<cc)
       {if (b>d) select b
        else select d}
       else select c}
```

References

[1] Chin, R.T. and C. Yeh (1985). Quantitative evaluation of some edge-preserving noise-smoothing techniques. *Computer Vision, Graphics and Image Processing* 23, 67–91.

[2] Davenport, J.P. (1978). A comparison of noise cleaning techniques. TR-689, Computer Vision Laboratory, Computer Science Center, University of Maryland, Sept.

[3] Rosenfeld, A. and A.C. Kak (1982). *Digital Picture Processing*, Vol. 1. Academic Press, New York.

[4] Davis, L.S. and A. Rosenfeld (1978). Noise cleaning by iterated local averaging. *IEEE Trans. Syst. Man Cybern.* 8, 705–710.

[5] Wang, D.C. A.H. Vagnucci and C.C. Li (1981). Gradient inverse weighted smoothing scheme and the evaluation of its performance. *Computer Graphics and Image Processing* 15, 167–181.

[6] Tomita, F. and S. Tsuji (1977). Extraction of multiple regions by smoothing in selected neighborhoods. *IEEE Trans. Syst. Man Cybern.* 107–109.

[7] Nagao, M. and T. Matsuyama (1979). Edge preserving smoothing. *Computer Graphics and Image Processing* 9, 394–407.

[8] Lee, Jong-Sen (1983). Digital image smoothing and the sigma filter. *Computer Vision, Graphics and Image Processing* 24, 255–269.