SUSAN – A New Approach to Low Level Image Processing

Technical Report TR95SMS1c
(Shorter versions have now been published in IJCV [66] and ICPR96 [64],
and the relevant patent is [60]. To reference this research, cite [66])

1995

S.M. Smith
Oxford Centre for Functional Magnetic Resonance Imaging of the Brain (FMRIB),
Department of Clinical Neurology, Oxford University, Oxford, UK
(Previously in Computer Vision and Image Processing Group, DRA Chertsey, DERA, UK)
steve@fmrib.ox.ac.uk
www.fmrib.ox.ac.uk/~steve

J.M. Brady
Department of Engineering Science, Oxford University, Oxford, UK

Abstract

This paper describes a new approach to low level image processing; in particular, edge and corner detection and structure preserving noise reduction.

Non-linear filtering is used to define which parts of the image are closely related to each individual pixel; each pixel has associated with it a local image region which is of similar brightness to that pixel. The new feature detectors are based on the minimization of this local image region, and the noise reduction method uses this region as the smoothing neighbourhood. The resulting methods are accurate, noise resistant and fast.

Details of the new feature detectors and of the new noise reduction method are described, along with test results.

Keywords: edge detection, feature detection, univalue areas, noise reduction, smoothing.

1 Introduction

This paper describes an entirely new approach to low level image processing, specifically, edge detection (one dimensional feature detection), "corner" detection (two dimensional feature detection, including, therefore, corners, junctions, etc.) and structure preserving noise reduction. The new approach represents a significant departure from feature extraction and noise reduction methods previously developed.

The paper begins with an explanation of the SUSAN feature detection principle, and continues with details of the applications of it, including reviews of relevant past research and results of testing the applications. Because the research is based on fundamentally non-linear filtering, the approach to theoretical justification is necessarily different from that traditionally applied, for example, to "optimal" edge filtering.

2 The SUSAN Principle for Feature Detection

The SUSAN principle is now introduced, from which the research described in this paper is derived. Consider Figure 1, showing a dark rectangle on a white background. A circular mask (having a centre pixel which shall be known as the "nucleus") is shown at five image positions. If the brightness of each pixel within a mask is compared with the brightness of that mask's nucleus then an area of the mask can be defined which has the same (or similar) brightness as the nucleus. This area of the mask shall be known as the "USAN", an acronym standing for "Univalue Segment Assimilating Nucleus". In Figure 2 each mask from Figure 1 is depicted with its USAN shown in white.

This concept of each image point having associated with it a local area of similar brightness is the basis for the SUSAN principle. The local area or USAN contains much information about the structure of the image. It is effectively region finding on a small scale. From the size, centroid and second moments of the USAN two dimensional features and edges can be detected. This approach to feature detection has many differences to the well known methods, the most obvious being that no image derivatives are used and that no noise reduction is needed.

The area of an USAN conveys the most important information about the structure of the image in the region around any point in question. As can be seen from Figures 1 and 2, the USAN area is at a maximum when the nucleus lies in a flat region of the image surface, it falls to half of this maximum very near a straight edge, and falls even further when inside a corner. It is this property of the USAN's area which is used as the main determinant of the presence of edges and two dimensional features. Consider now Figure 3, where a small part of a test image has been processed to give USAN area as output.[1] Each point in the input image is used as the nucleus of a small circular mask, and the associated USAN is found. The area of the USAN is used in the three dimensional plot shown. The USAN area falls as an edge is approached (reaching a minimum at the exact position of the edge), and near corners it falls further, giving local minima in USAN area at the exact positions of image corners. Figure 4 shows a small part of a real noisy image, and the resulting output from USAN area processing. (The variation in brightness within the "flat regions" is of the order of 15 – out of 256 – greyscale levels.) Again there is edge and corner enhancement, with the noise having no visible effect on the final plot.

Consideration of the above arguments and observation of the examples and results shown in Figures 1, 2, 3 and 4 lead directly to formulation of the SUSAN principle:

> An image processed to give as output inverted USAN area has edges and two dimensional features strongly enhanced, with the two dimensional features more strongly enhanced than edges.

This gives rise to the acronym SUSAN (Smallest Univalue Segment Assimilating Nucleus). Mathematical analyses of the principle are given after the algorithms have been described in detail.

The fact that SUSAN edge and corner enhancement uses no image derivatives explains why the performance in the presence of noise is good. The integrating effect of the principle, together with its non-linear response, give strong noise rejection. This can be understood simply if an input signal with identically

[1] Note that the scale on the vertical axis is inverted.

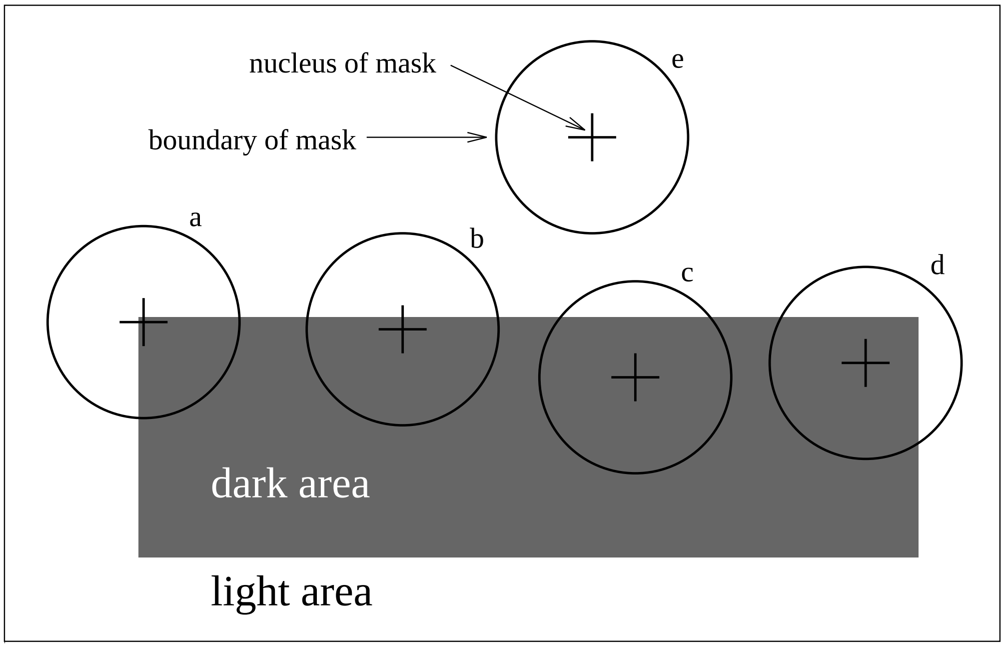

Figure 1: Four circular masks at different places on a simple image.

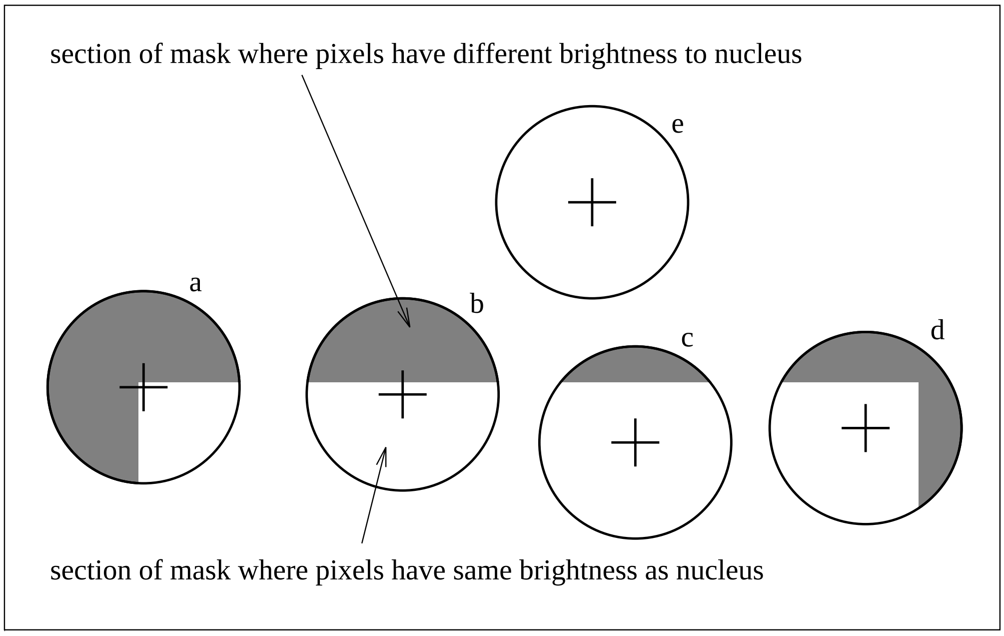

Figure 2: Four circular masks with similarity colouring; USANs are shown as the white parts of the masks.

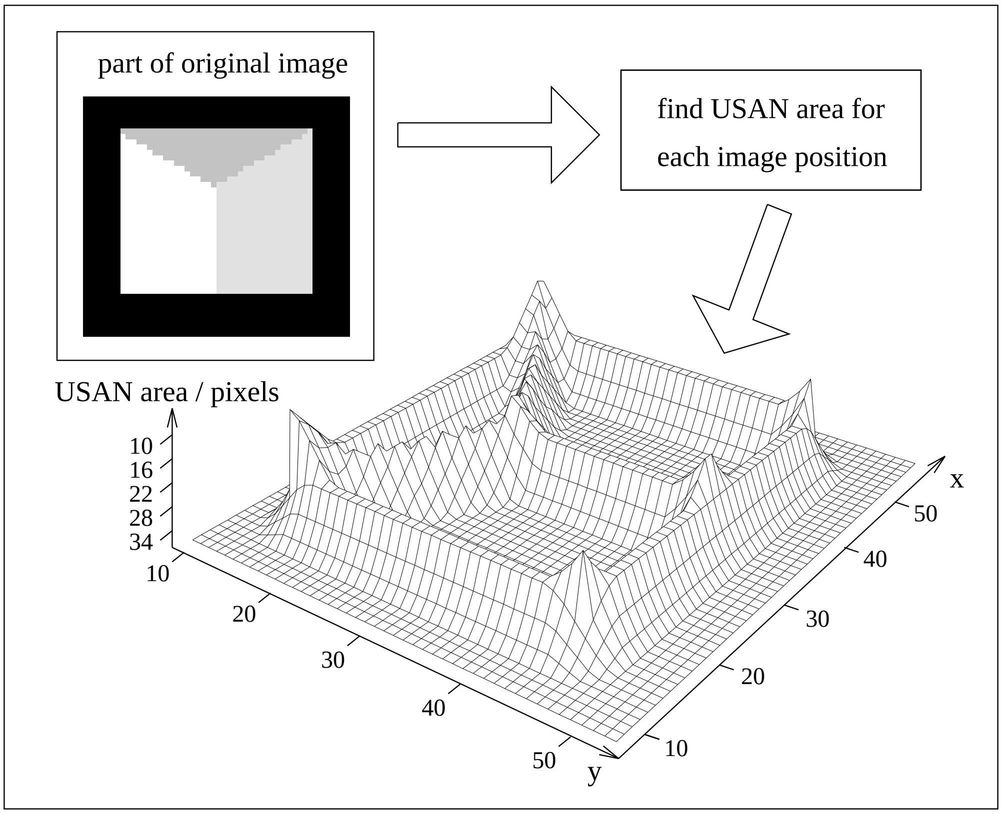

Figure 3: A three dimensional plot of USAN area given a small part of a test image, showing edge and corner enhancement.

independently distributed Gaussian noise is considered. As long as the noise is small enough for the USAN function (see Figure 5) to contain each "similar" value, the noise is ignored. The integration of individual values in the calculation of areas further reduces the effect of noise. Another strength of the SUSAN edge detector is that the use of controlling parameters is much simpler and less arbitrary (and therefore easier to automate) than with most other edge detection algorithms.

The SUSAN noise reduction algorithm is related to the SUSAN principle in that the USAN is used to choose the best local smoothing neighbourhood.

3 Criteria for Determining the Quality of Feature Detectors

In this section the desired qualities of feature detectors are explained. The criteria eventually given have a slightly different emphasis from those previously used.

In [9] three criteria for edge detection are given. These have been used in similar form in a good deal of vision research. They are:

1. Good detection. There should be a minimum number of false negatives and false positives.[2]

[2] Canny develops Criterion 1 to mean that the edge enhancing filter should maximize the signal-to-noise ratio. This makes sense in the case of a linear filter which has features thresholded at a later stage than the enhancement, and Canny formulates a signal-to-noise functional which expresses this requirement. However, the SUSAN principle is fundamentally non-linear. (This can be simplistically interpreted as performing thresholding at an earlier stage.) In fact, it is clear from Figure 4 that the term "enhancement" is a little weak, given the obvious signal-to-noise ratio. (The plot of the operator output shows no visible noise.) Thus the criterion is most appropriate as it stands.

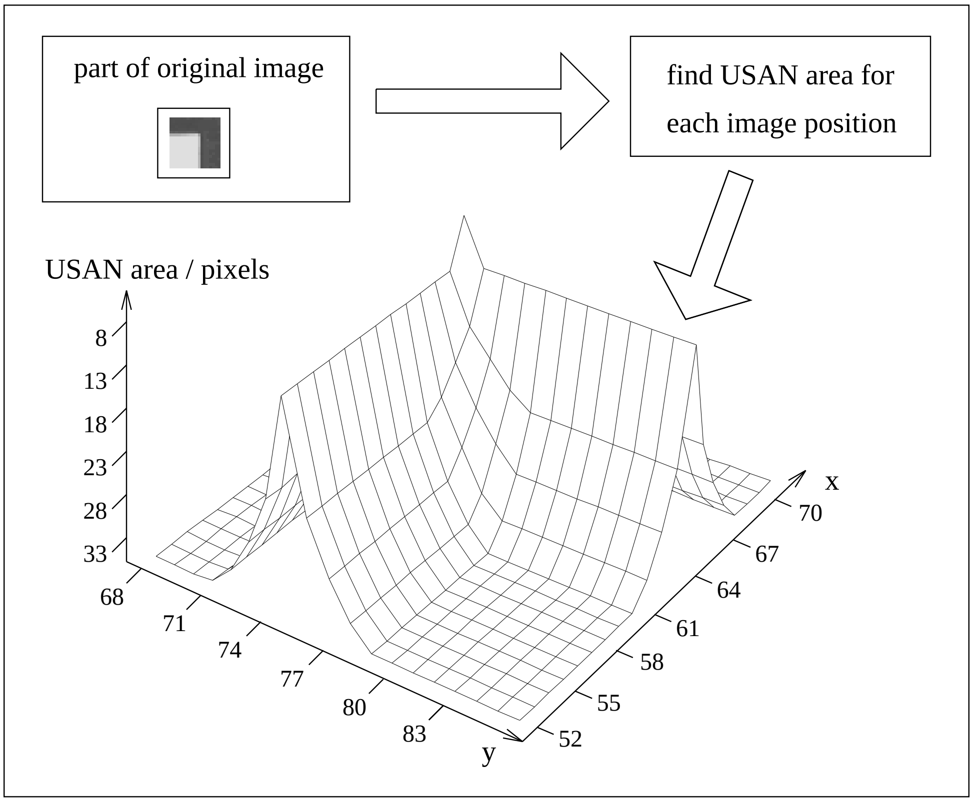

Figure 4: A three dimensional plot of USAN area given a small part of a real noisy image, showing edge and corner enhancement.

2. Good localization. The edge location must be reported as close as possible to the correct position.

3. Only one response to a single edge.

These criteria are equally appropriate for two dimensional feature detection, except for the complication that arises from using the term "correct position" in Criterion 2. There are many mathematical descriptions of the two dimensional image structure which define a "corner" or more general feature, leading to a lack of one agreed definition of the exact location of a feature. Thus in the case of two dimensional features, measuring fine localization error is not appropriate. However, it will be seen later that the error in localizing two dimensional features is often several pixels with some existing algorithms, greater even than the expected variation in positions determined by hand.

The purpose of developing new feature detectors was that they should be appropriate to being used as part of a *real time* system using *real* image sequences. Therefore it has become apparent that for the purpose of this work, one final criterion is important;

4. Speed. The algorithm should be fast enough to be usable in the final image processing system.

Obviously this criterion must not be allowed to dominate the development of a feature detector to an extent that the quality of the results is adversely affected. However a fast algorithm performing well in the other criteria is more desirable than a slow one.

Canny has formulated his criteria mathematically (in [8] and [9]). He uses the criteria functionals to derive "optimized" edge filters for each image type. Criteria 1 and 2 are clearly the most important ones in terms of "optimizing" the filter. Later analysis shows how the SUSAN brightness comparison function has been optimized to give the lowest number of false negatives and false positives. In the case of localization, the spatial domain part of the SUSAN detectors does not introduce any new concepts, and, following Canny and others, either a Gaussian or square distribution may be used (see later). Criterion 3 has not been of relevance in this work, as multiple responses have not been a problem at all. Finally, as will be seen, the SUSAN feature detectors are extremely computationally efficient.

In [71] a similar set of criteria are defined, not for optimizing filters, but for comparing the outputs of different edge detectors. There has been very little work on the area of objective quantitative tests for feature detectors. In [71] four quantitative criteria are defined, those being proportional to the number of false negatives, the number of false positives, the number of multiple detections and the number of incorrectly localized pixels. A linear combination of these measures is then used to obtain one quantity, the "failure measure" or "FM". The weights used to create this weighted sum are found according to a synthesis of some simple minimization rules with the constraint that the resulting edges found using this sum correspond to the best edges judged by eye. The weights will also depend on the proposed use of the reported edges.

The authors are of the opinion that this method of calculating the quality of an edge filter is not very meaningful, as the measures defined in [71] are found before post-processing takes place. Non-maximum suppression (a common enough operation) will normally eliminate multiple detections of a single edge completely. Binary thinning (particularly when continuously taking the initial response into account, as described in [63]) will greatly reduce the number of false negatives and false positives. All that remains is the localization error. Unfortunately, in the scheme described, the importance given to this error is between three and ten times less than that given to the number of false negatives or false positives.

However, in the absence of a better comparative method in the literature, quantitative tests have been carried out using the FM scheme, comparing the output from the first stage of the SUSAN edge detector with the four filters tried in [71], namely, Sobel, Prewitt, Roberts and the "three-point energy model"[3] (see [67], [50], [51] and [72]). A test image identical to the image described was created (i.e. a vertical step edge with added Gaussian noise) giving a signal to noise ratio of 14.79. The performance of the initial stage of the SUSAN detector was evaluated for the four criteria defined; the result was that the SUSAN edge detector gave FMs which were better than those obtained with the other four detectors, independent of the set of weights used. Note that the algorithms compared here with the first stage of SUSAN represent the first (or enhancement) stage of edge detection, rather than complete edge detectors, as this is appropriate to the comparative method developed in [71].

It is worth noting here that in the absence of multiple features the SUSAN principle bypasses the "uncertainty principle" which applies to most feature detectors (and most obviously the Gaussian based ones) with respect to Canny's first and second criteria. This well understood problem means that the better the detection quality (including noise suppression) then the worse the localization of the detected feature. In the case of the SUSAN feature detectors, however, it is clear that the localization of the features is independent of the mask size, as long as no other features are found within the mask's region. Thus there is no conflict here between the two most "important" criteria.

4 The SUSAN Edge Detector

The details of the SUSAN edge finding algorithm are given, followed by an analysis of the algorithm's validity. Finally, examples of the output of the edge detector are presented and discussed. Firstly, however, a brief review of existing approaches is given. (For a longer review of edge finding, see [63].)

[3]This edge detector combines the outputs of two filters to attempt to find both step edges and ridge/roof edges. It basically performs a simple test to find maxima in the first or second derivatives.

4.1 Review

There has been an abundance of work on different approaches to the detection of one dimensional features in images. The wide interest is due to the large number of vision applications which use edges[4] and lines as primitives, to achieve higher level goals.

Some of the earliest methods of enhancing edges in images used small convolution masks to approximate the first derivative of the image brightness function, thus enhancing edges; e.g., see [50] and [67]. These filters give very little control over smoothing and edge localization.

In [34] Marr and Hildreth proposed the use of zero crossings of the Laplacian of a Gaussian (LoG). Contours produced using the LoG filter have the property, convenient for some purposes, of being closed. However, connectivity at junctions is poor, and corners are rounded. Also, the use of non-directional derivatives means that the edge response parallel to an edge is always measured (as well as the expected response perpendicular to it), reducing the signal to noise ratio. The use of directional first and second derivatives improves on this. Finally, the LoG filter gives no indication of edge direction, which may be needed by higher level processes.

In [8] Canny described what has since become one of the most widely used edge finding algorithms. The first step taken is the definition of criteria which an edge detector must satisfy (see Section 3). These criteria are then developed quantitatively into a total error cost function. Variational calculus is applied to this cost function to find an "optimal" linear operator for convolution with the image. The optimal filter is shown to be a very close approximation to the first derivative of a Gaussian.[5] Non-maximum suppression in a direction perpendicular to the edge is applied, to retain maxima in the image gradient. Finally, weak edges are removed using thresholding. The thresholding is applied with hysteresis. Edge contours are processed as complete units; two thresholds are defined, and if a contour being tracked has gradient magnitude above the higher threshold then it is still "allowed" to be marked as an edge at those parts where the strength falls below this threshold, as long as it does not go below the lower value. This reduces streaking in the output edges.

The Gaussian convolution can be performed quickly because it is separable and a close approximation to it can be implemented recursively. However, the hysteresis stage slows the overall algorithm down considerably. While the Canny edge finder gives stable results, edge connectivity at junctions is poor,[6] and corners are rounded, as with the LoG filter. The scale of the Gaussian determines the amount of noise reduction; the larger the Gaussian the larger the smoothing effect. However, as expected, the larger the scale of the Gaussian, the less accurate is the localization of the edge.

Canny also investigated the synthesis of results found at different scales; in some cases the synthesis improved the final output, and in some cases it was no better than direct superposition of the results from different scales.

Finally, Canny investigated the use of "directional operators". Here several masks of different orientation are used with the Gaussian scale larger along the direction parallel to the edge than the scale perpendicular to it. This improves both the localization and the reliability of detection of straight edges; the idea does not

[4]Much research in this field has assumed that the only one dimensional features of interest are step edges. However, there exist many other types of feature. These include lines (ridges in the image surface), ramp ends and roof edges; see Figure 9 for examples of these. There are three main reasons for the concentration on step edges. The first is that they are the most common type of one dimensional change. The second is that edges containing a step component are the most well localized one dimensional features, that is, they are formed by a "first order" change. The third reason for working only with step edges is that some proposed edge finders (such as Canny's) are easily extended to finding other types of change once the theory for step edges has been completed. Thus many detectors have been developed using rigorous derivations of optimal algorithms using various criteria based on the model of the ideal step edge.

[5]The problem with using image derivatives is that differentiation enhances noise as well as edge structure, so most edge detectors include a noise reduction stage. Thus the use of the derivative of a Gaussian enables differentiation to take place at the same time as the smoothing; this is allowable, as the two processes commute (exactly in the continuous case, and approximately in the discrete case). The problem of noise enhancement is even worse when differentiation is performed twice.

[6]Some higher level algorithms use Canny's method *because* of this characteristic, as they work better with simple unconnected edges. However, achieving full connectivity at junctions is clearly a worthwhile goal as it correctly represents the scene. In [32] Li et. al. suggest heuristic extensions to the Canny algorithm to enable the joining of open contour ends with nearby contours. This however produces some false edge extensions.

work well on edges of high curvature.

In [14] and [56] similar analytical approaches to that of Canny are taken, resulting in efficient algorithms which have exact recursive implementations. These algorithms give results which are very similar to Canny's.

In [22], Haralick proposes the use of zero crossings of the second directional derivative of the image brightness function. This is theoretically the same as using maxima in the first directional derivatives, and in one dimension is the same as the LoG filter. The zero crossings are found by fitting two dimensional functions to the image brightness surface and then analyzing the resulting fit (see the following section). The functions used are "discrete orthogonal polynomials of up to degree three". There is a problem with "phantom edges" created by the second directional derivative at "staircase structures". Connectivity at junctions is poor. In [18] Fleck describes the use of the second directional derivative for edge finding, with various extensions to the basic use of zero crossings. The problem of phantom edges is reduced with a test using the first and third derivatives. Image noise is reduced using topological sums; smoothing is achieved at potential edge points by measuring the local support they have for their "edge point type" classification. The overall algorithm (and in particular the noise reduction part) is computationally expensive.

The approach of surface fitting defines a two dimensional function which should be "flexible" enough to provide a good approximation to the image brightness function. The best fit of the function to the image is then found at each image position, and the parameters of the fit are used either to estimate image derivatives directly or to find edges using alternative definitions. The problem with the mathematical model approach is that deviations from the surface model cannot (by definition) be accommodated by varying the model's parameters. Work taking this approach includes [28], [42] and [59].

In [43] Noble uses mathematical morphology to find image structure. Several different morphological operations are described; these are used to enhance edges and find two dimensional features. The "erode-dilate" operator is similar to a first order derivative, and the "open-close" operator is similar to a second order derivative. Good quality edges are found by tracking both sides of each edge and then stitching these "half boundaries" together. Connectivity at junctions is good, although spurious short "tails" sometimes appear at structures such as "T" junctions. The algorithm, including edge tracking, is fairly computationally expensive.

In [72] and [70] Venkatesh, Owens et. al. describe the approach of using "local energy" (in the frequency domain) to find features. The local energy is found from quadrature pairs of image functions, such as the image function and its Hilbert transform. Fourier transforms of the image were initially used to find the required functions, at large computational cost. In a more "realistic" implementation also described, processing is performed using a function pair very similar to a first and a second derivative; this is shown to be equivalent to the method proposed originally. This approach has the advantage of being able to find a wider variety of edge types than that typically found using antisymmetric linear filters, however, this is at the expense of optimizing the signal to noise ratio. (In [9] Canny makes a similar point, when discussing the symmetric and antisymmetric components of a filter.) Other work taking this approach includes [48] and [53].

In [6] Blake and Zisserman used a weak membrane model in the framework of statistical regularization. The elements of the cost functional (which is to be globally minimized) are the smoothness of the estimated surface, the quality of the fit of the estimated surface to the original data and the "line processes", i.e., the discontinuities in the membrane. This method does not pick up fine complicated detail very well, and is computationally expensive. This approach has been developed further, for example, in [21], [44] and [68]. Other methods of using "global" support for edge detection include [45] and [23].

The edge detector described in this paper uses a completely new definition of edges, and in doing so, solves many of the problems which existing algorithms have failed to overcome.

4.2 The SUSAN Edge Detector in Detail

The edge detection algorithm described here follows the usual method of taking an image and, using a predetermined window centred on each pixel in the image, applying a locally acting set of rules to give an edge response. This response is then processed to give as the output a set of edges.

The SUSAN edge finder has been implemented using circular masks (sometimes known as windows or kernels) to give isotropic responses.[7] Digital approximations to circles have been used, either with constant weighting within them or with Gaussian weighting – this is discussed further later. The usual radius is 3.4 pixels (giving a mask of 37 pixels), and the smallest mask considered is the traditional three by three mask. The 37 pixel circular mask is used in all feature detection experiments unless otherwise stated.

The mask is placed at each point in the image and, for each point, the brightness of each pixel within the mask is compared with that of the nucleus (the centre point). Originally a simple equation determined this comparison – see Figure 5(a);

$$c(\vec{r},\vec{r_0}) = \begin{cases} 1 & \text{if } |I(\vec{r}) - I(\vec{r_0})| \le t \\ 0 & \text{if } |I(\vec{r}) - I(\vec{r_0})| > t, \end{cases} \tag{1}$$

where $\vec{r_0}$ is the position of the nucleus in the two dimensional image, $\vec{r}$ is the position of any other point

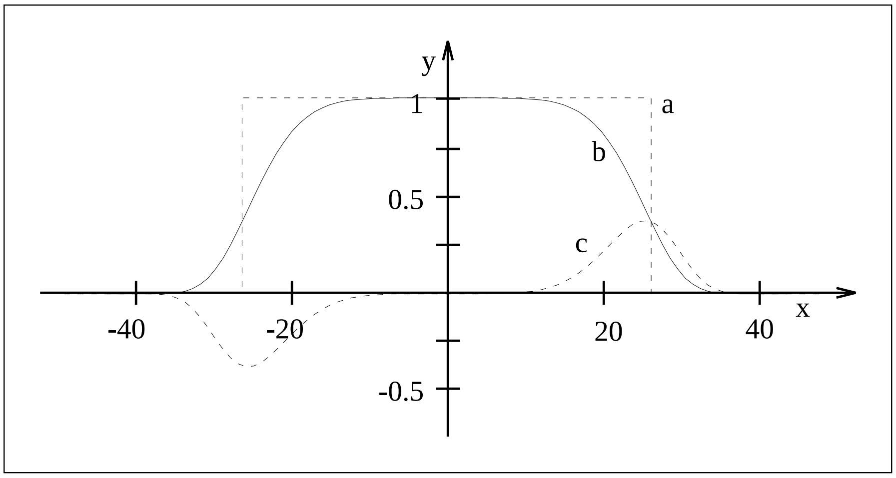

Figure 5: a) The original similarity function (y axis, no units) versus pixel brightness difference (x axis, in greylevels). For this example the pixel brightness difference "threshold" is set at ± 27 greylevels. b) The more stable function now used. c) The boundary detector B (see later text).

within the mask, $I(\vec{r})$ is the brightness of any pixel, t is the brightness difference threshold and c is the output of the comparison.[8] This comparison is done for each pixel within the mask, and a running total, n, of the outputs (c) is made;

$$n(\vec{r_0}) = \sum_{\vec{r}} c(\vec{r},\vec{r_0}). \tag{2}$$

This total n is just the number of pixels in the USAN, i.e. it gives the USAN's area. As described earlier this total is eventually minimized. The parameter t determines the minimum contrast of features which will be detected and also the maximum amount of noise which will be ignored. Further discussion later will show why performance is not dependent on any "fine-tuning" of the value of t.

Next, n is compared with a fixed threshold g (the "geometric threshold"[9]), which is set to $3n_{\max}/4$, where $n_{\max}$ is the maximum value which n can take. The initial edge response is then created by using the

[7] In [9] the circular Gaussian mask is developed into a non-circular one once edge direction has been found. This reduces the contribution of noise to the edge signal. This step could be mirrored in the SUSAN algorithm, but there is no need, as no signal as such is being filtered, and the noise reduction is already large due to integration over the mask.

[8] The software implementation uses a value of 100 rather than 1 for the maximum similarity, so that integer arithmetic may be used rather than floating point, for speed of computation.

[9] Note that the *only* threshold introduced in this paper which is *not* optimally established by careful analysis is t, the brightness difference threshold. This is the parameter which controls the sensitivity of the feature detection algorithms. Feature detection applications almost always require that sensitivity can be controlled by variation of one or more parameters, preferably only one.

following rule:

$$R(\vec{r_0}) = \begin{cases} g - n(\vec{r_0}) & \text{if } n(\vec{r_0}) < g \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $R(\vec{r_0})$ is the initial edge response. This is clearly a simple formulation of the SUSAN principle, i.e., the smaller the USAN area, the larger the edge response. When non-maximum suppression has been performed the edge enhancement is complete.

When finding edges in the absence of noise, there would be no need for the geometric threshold at all. However, in order to give optimal noise rejection g is set to $3n_{\max}/4$. This value is calculated from analysis of the expectation value of the response in the presence of noise only – see later. The use of g should not result in incorrect dismissal of correct edges for the following reasons. If a step edge (of general curvature) is considered, it can be seen that n will always be less than (or equal to) $n_{\max}/2$ on at least one side of the edge. In the case of a curved edge, this will correspond to the boundary of the region which is convex at the step edge. Thus valid edges should not be rejected. If the edge is not an ideal step edge but has a smoother profile then n will have even lower minima so that there is even less danger of edges being wrongly rejected.

The algorithm as described gives quite good results, but a much more stable and sensible equation to use for c in place of Equation 1 is

$$c(\vec{r}, \vec{r_0}) = e^{-\left(\frac{I(\vec{r}) - I(\vec{r_0})}{t}\right)^6}. \tag{4}$$

This equation is plotted in Figure 5(b). The form of Equation 4 was chosen to give a "smoother" version of Equation 1. This allows a pixel's brightness to vary slightly without having too large an effect on c, even if it is near the threshold position. The exact form for Equation 4, i.e., the use of the sixth power, can be shown to be the theoretical optimum; see later for analytic comparison of shapes varying from one extreme (Gaussian) to the other (square function, as originally used). This form gives a balance between good stability about the threshold and the function originally required (namely to count pixels that have similar brightness to the nucleus as "in" the univalue surface and to count pixels with dissimilar brightness as "out" of the surface). The equation is implemented as a look up table for speed. (The threshold t determines, of course, the minimum contrast of edges which will be picked up, and provides an easy way of controlling this, even automatically, if necessary.)

Computation of the edge direction is necessary for a variety of reasons. Firstly, if non-maximum suppression is to be performed the edge direction must be found. It is also necessary if the edge is to be localized to sub-pixel accuracy. Finally, applications using the final edges often use the edge direction for each edge point as well as its position and strength. In the case of most existing edge detectors, edge direction is found as part of the edge enhancement. As the SUSAN principle does not require edge direction to be found for enhancement to take place, a reliable method of finding it from the USAN has been developed. This method is now described.

The direction of an edge associated with an image point which has a non zero edge strength is found by analyzing the USAN in one of two ways, depending on the type of edge point which is being examined. For examples of the two types of edge points, see Figure 6. It can be seen that points (a) and (b) have USAN shapes which would be expected for an ideal step edge. In this case (which shall be known as the "*inter*-pixel edge case") the vector between the centre of gravity $\bar{\vec{r}}$ of the USAN and the nucleus of the mask is perpendicular to the local edge direction. The centre of gravity is found thus;

$$\bar{\vec{r}}(\vec{r_0}) = \frac{\sum_{\vec{r}} \vec{r}\, c(\vec{r}, \vec{r_0})}{\sum_{\vec{r}} c(\vec{r}, \vec{r_0})}. \tag{5}$$

This simple rule allows the edge direction to be found for this type of edge point.

The point shown in case (c) lies on a thin band which has a brightness roughly half way between the brightnesses of the two regions which generate the edge. This occurs when the real edge projects very close to the centre of a pixel rather than in between pixels (or when the edge is not a sharp step edge in the first place) and when the edge contrast is high. In this case, (which shall be known as the "*intra*-pixel edge case")

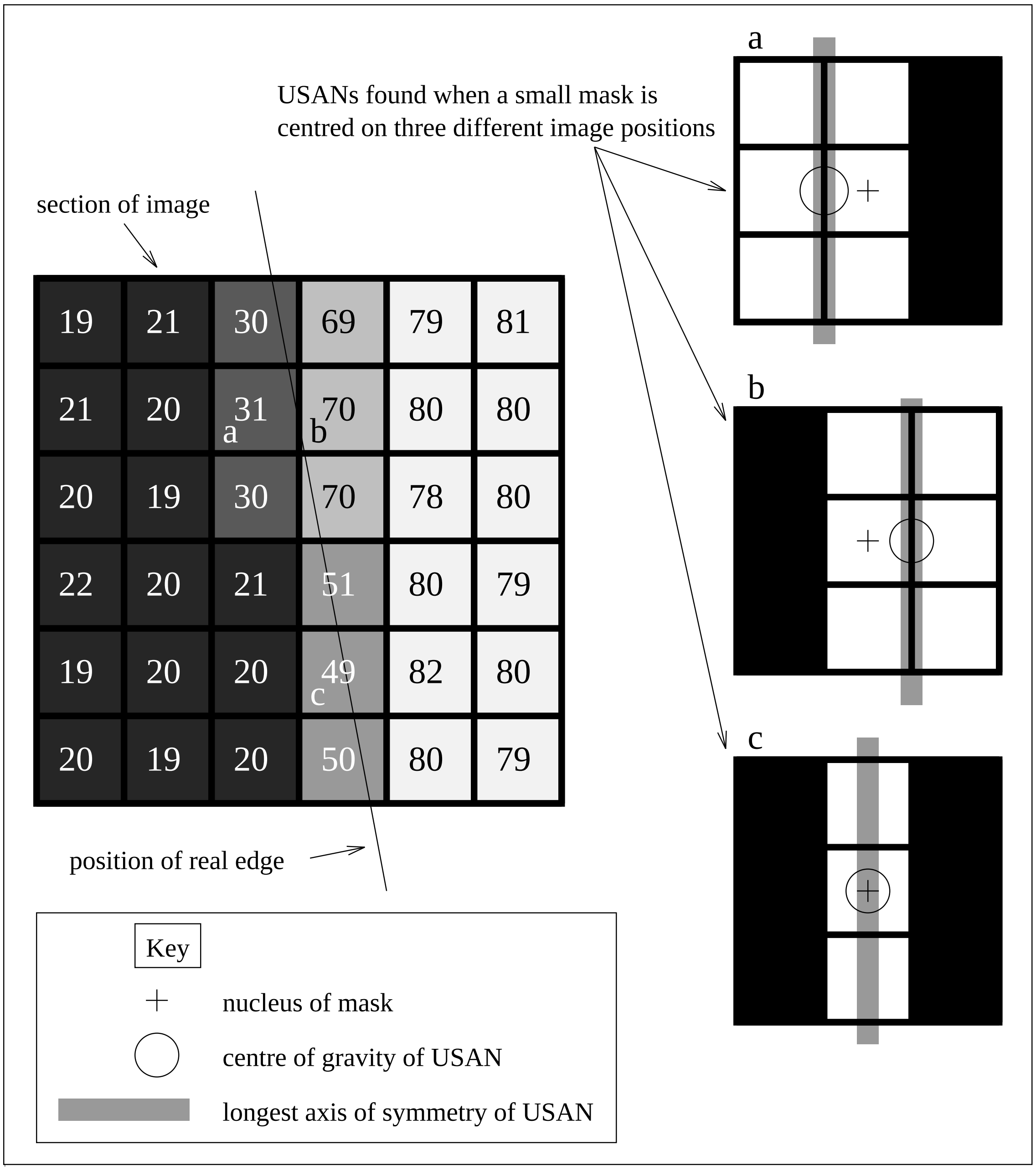

Figure 6: The two main edge types; a typical straight edge in a section of a real image, with brightness indicated by the numerical text as well as the shading of the pixels. The USANs for three points of interest are shown as the white regions of a small (3 by 3) mask. Points (a) and (b) are standard edge points, lying definitely on one side of the edge or the other. Point (c) lies on a band of brightness half way between the brightnesses of the two regions generating the edge. It therefore has a differently shaped USAN, and a centre of gravity coinciding with the nucleus.

the USAN formed is a thin line in the direction of the edge, as can be seen in Figure 6. The edge direction is thus calculated by finding the longest axis of symmetry. This is estimated by taking the sums

$$\overline{(x-x_0)^2}(\vec{r_0}) = \sum_{\vec{r}} (x-x_0)^2 \; c(\vec{r},\vec{r_0}), \tag{6}$$

$$\overline{(y-y_0)^2}(\vec{r_0}) = \sum_{\vec{r}} (y-y_0)^2 \; c(\vec{r},\vec{r_0}) \tag{7}$$

and

$$\overline{(x-x_0)(y-y_0)}(\vec{r_0}) = \sum_{\vec{r}} (x-x_0)(y-y_0) \; c(\vec{r},\vec{r_0}). \tag{8}$$

(No normalization of the sums is necessary with the second moment calculations.) The ratio of $\overline{(y-y_0)^2}$ to $\overline{(x-x_0)^2}$ is used to determine the orientation of the edge; the sign of $\overline{(x-x_0)(y-y_0)}$ is used to determine whether a diagonal edge has positive or negative gradient. Thus in the case of edge points like (c) the edge direction is again found in a simple manner.

The remaining question is how to automatically determine which case fits any image point. Firstly, if the USAN area (in pixels) is smaller than the mask diameter (in pixels) then the intra-pixel edge case is assumed. Figure 6 shows clearly the logic behind this. If the USAN area is larger than this threshold, then the centre of gravity of the USAN is found, and used to calculate the edge direction according to the inter-pixel edge case. If however, the centre of gravity is found to lie less than one pixel away from the nucleus then it is likely that the intra-pixel edge case more accurately describes the situation. (This can arise for example if the intermediate brightness band is more than one pixel wide, when larger mask sizes are used.)

The edge direction can be found to varying accuracy using this method, depending on the intended application. If the final output desired is simply a binarized edge image it may be enough to simply categorize an edge element as being "vertical" or "horizontal".

An interesting point about the SUSAN edge detector is shown by the USAN areas in Figure 6. It can be simply seen that as an edge becomes blurred, the area of the USAN at the centre of the edge will decrease. Thus we have the interesting phenomenon that the response to an edge will increase as the edge is smoothed or blurred. This is most unusual for an edge detector, and is not an undesirable effect.

Finally, therefore, the edge response image is suppressed so that non-maxima (in the direction perpendicular to the edge) are prevented from being reported as edge points. Following this, the "strength thinned" image can be "binary thinned". This means that standard thinning processes can be used to ensure that the edges obey required rules about number-of-neighbour connectivity, so that remaining noise points can be removed, and so that edge points incorrectly removed by the non-maximum suppression can be replaced. A set of rules for binary thinning has been implemented (still making use of the strengths in the non-suppressed edge response image) which work well to give a good final binarized edge image. These rules are described in [63].

If the position of an edge is required to accuracy greater than that given by using whole pixels (the accuracy which would be achieved by using the theory presented so far), it may be calculated in the following way. For each edge point, the edge direction is found and the edge is thinned in a direction perpendicular to this. The remaining edge points then have a 3 point quadratic curve fit (perpendicular to the edge) to the initial edge response, and the turning point of this fit (which should be less than half a pixel's distance from the centre of the thinned edge point) is taken as the exact position of the edge.

With respect to the scale-space behaviour of the SUSAN edge detector, scale-space graphs showing edge localization against mask size (e.g., plotting a single horizontal line from the edge image against mask size, in the manner of Witkin [75]) give vertical lines. (Most edge detectors do not give scale-invariant edge positions, thus producing curves in scale-space graphs.) This is obviously a desirable feature, as it means that accuracy does not depend on mask size. This is to be expected; the minimum USAN area when approaching an edge occurs on top of the edge regardless of the mask size.

In summary then, the algorithm performs the following steps at each image pixel:

1. Place a circular mask around the pixel in question (the nucleus).
2. Using Equation 4 calculate the number of pixels within the circular mask which have similar brightness to the nucleus. (These pixels define the USAN.)
3. Using Equation 3 subtract the USAN size from the geometric threshold to produce an edge strength image.
4. Use moment calculations applied to the USAN to find the edge direction.
5. Apply non-maximum suppression, thinning and sub-pixel estimation, if required.

4.3 Analysis of the SUSAN Edge Detector

A simple derivation is now given which shows the theoretical coincidence of the exact SUSAN edge position and the zero crossing of the second derivative of the image function.

If we assume a one dimensional input signal which is monotonically increasing then it can be simply shown that a minimum in the USAN area (in this case USAN length) will be equivalent to the edge definition which places the edge at the position of inflection of the curve, that is, where the second derivative of the image function is zero. See Figure 7 for an example image function and three one dimensional mask positions.

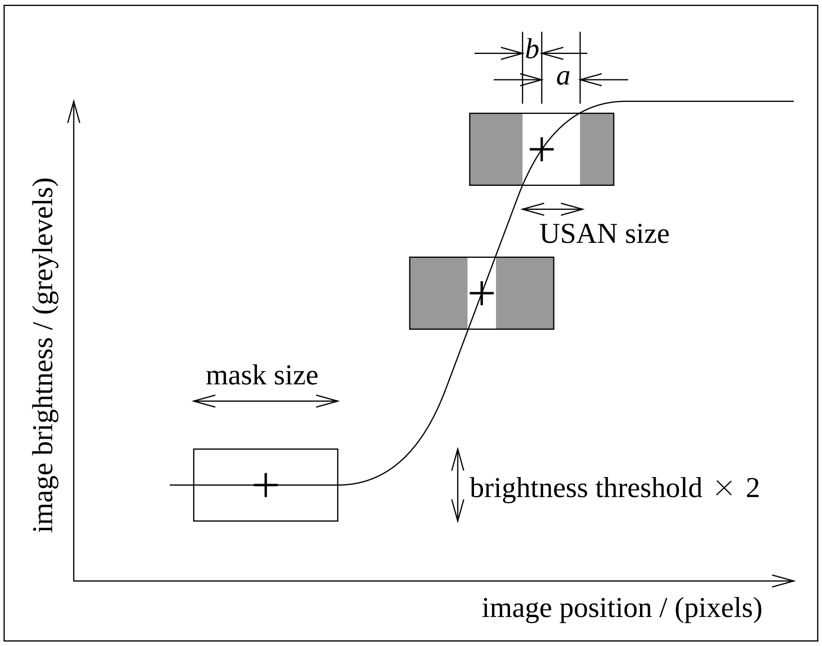

Figure 7: A monotonically increasing one dimensional input signal and three mask positions showing USAN size varying with signal gradient. The USAN is shown as the white portion of the mask. The mask centred on the signal inflection has the smallest USAN.

If we have general image position x, image function $I(x)$ and brightness threshold t, then a and b (the "right" and "left" widths of the USAN – see Figure 7), for any specific image position x_0, are, by definition, given by

$$I(x_0 + a(x_0)) = I(x_0) + t \tag{9}$$

and

$$I(x_0 - b(x_0)) = I(x_0) - t. \tag{10}$$

The SUSAN principle is formulated in the following equation, where $n(x_0)$ is the USAN size at x_0 (the size being a length in this one dimensional case);

$$\frac{dn(x_0)}{dx_0} = \frac{d(a(x_0) + b(x_0))}{dx_0} = 0, \tag{11}$$

assuming that a check is made for n being a minimum and not a maximum. This gives

$$\frac{d}{dx_0}(x(I(x_0) + t) - x(I(x_0) - t)) = 0, \tag{12}$$

giving

$$\left(\frac{dx}{dI}(I(x_0) + t)\ \frac{dI}{dx_0}(x_0)\right) - \left(\frac{dx}{dI}(I(x_0) - t)\ \frac{dI}{dx_0}(x_0)\right) = 0, \tag{13}$$

that is,

$$\frac{dI}{dx}(x_0 + a(x_0)) - \frac{dI}{dx}(x_0 - b(x_0)) = 0, \tag{14}$$

unless the image function is constant, an uninteresting case. This is equivalent, in the continuous limit (that is, as t tends to zero), to

$$\frac{d^2 I}{dx^2}(x_0) = 0, \tag{15}$$

and we have equivalence between the two definitions of exact edge position.

This mathematical equivalence in no way means that the SUSAN edge finding method is performing the same function as a derivative based edge detector. No derivatives are taken; indeed, no direction of maximum gradient is identified (in order for differentiation to be performed) in the first place. However, the preceding argument shows why the SUSAN edge detector gives edges to good sub-pixel accuracy even with edges that are not perfect step edges.

It can be seen from the preceding discussion that the initial response of the SUSAN edge detector to a step edge will be increased as the edge is smoothed. The response is also broadened.

As mentioned earlier, the form of the brightness comparison function, the heart of USAN determination (Equation 4), can be shown to achieve the optimal balance between stability and enhancement. This is equivalent to satisfying the criterion that there should be a minimum number of false negatives and false positives. This criterion is formulated in the following expression:

$$F(d, t, s) = \frac{\sqrt{\mathrm{var}(R_S)} + \sqrt{\mathrm{var}(R_N)}}{<R_S> - <R_N>}, \tag{16}$$

where F is proportional to the number of false positives and false negatives that will be reported, s is image noise standard deviation, R_N is the SUSAN response strength with no edge present and R_S is the SUSAN response strength when the mask is centred on an edge of strength d. Thus this expression simply divides the expected total "noise" by the expected "signal" in the most logical way.

The value of F will depend on the relative values of d, t and s. Thus fixing t at 1 and varying the other two variables will cater for all eventualities. If this expression is averaged over all likely values of d and s, the overall quality can be evaluated. The (wide) ranges of d and s chosen were $1 \to 10$ and $0 \to 1/\sqrt{2}$ respectively; the reason for choosing the the upper limit on s will shortly become apparent.

All of the calculations within the SUSAN feature detectors are based around the use of the equation

$$\Delta = I(\vec{r}) - I(\vec{r_0}), \tag{17}$$

where I is defined as before. If the image noise is Gaussian with standard deviation s then the noise on Δ will also be Gaussian with standard deviation $s_\Delta = \sqrt{2}s$. Thus F is evaluated over the interval $s_\Delta = 0 \to 1$.

To evaluate F given d, t and s, it is necessary to find the expectation values and variances of R_N and R_S, using

$$<R_N>= 1- <N>, \tag{18}$$

$$<R_S>= 1 - \frac{1}{2} <N> - \frac{1}{2} <S>, \tag{19}$$

$$\mathrm{var}(R_N) = \mathrm{var}(N), \tag{20}$$

and

$$\mathrm{var}(R_S) = \frac{1}{4}\mathrm{var}(N) + \frac{1}{4}\mathrm{var}(S), \tag{21}$$

where

$$N = e^{-(\frac{\Delta}{t})^J} \tag{22}$$

and

$$S = e^{-(\frac{\Delta - d}{t})^J}, \tag{23}$$

with J being the even number determining the shape of the brightness comparison function; it is 2 for a Gaussian and infinity for a square function. (N and S are responses due to individual pixels with no edge present and across an edge respectively, using the brightness comparison function. The form of R_S arises from the fact that in the presence of an edge, half of the response will be due to noise only, and half of the response will be due to the edge as well as noise.) The expectation values and variances are computed using the integrals:

$$<X>= \frac{1}{s_\Delta \sqrt{2\pi}} \int_{-\infty}^{\infty} X(\Delta)\, e^{-\frac{1}{2}\left(\frac{\Delta}{s_\Delta}\right)^2} d\Delta \tag{24}$$

and

$$\mathrm{var}(X) = \frac{1}{s_\Delta \sqrt{2\pi}} \int_{-\infty}^{\infty} X^2(\Delta)\, e^{-\frac{1}{2}\left(\frac{\Delta}{s_\Delta}\right)^2} d\Delta - <X>^2, \tag{25}$$

where the integral includes the Gaussian model of noise. The integrals are calculated over the intervals of d and s_Δ specified and the mean result for F is thus found. The resulting plot of F against J is shown in Figure 8. It is clear that the optimal value for J is 6. This gives the form of the function used in the SUSAN filters.

The optimal value for g, the geometric threshold, is found by calculating the mean expectation value for N over the same realistic range of image noise as before. With no noise present, $<N>$ is 1; its mean value in the presence of noise is calculated (using the integral shown in Equation 24) to be close to 0.75. Therefore the value of g which is likely to provide the correct amount of noise rejection is 3/4 of the maximum possible USAN size.

Finally, the spatial part of the filter is of interest. In [9] the "difference of boxes" edge detector is compared unfavourably with a "derivative of Gaussian" edge detector, due to the sharp cutoff in the spatial profile of the "box" filter. Because the Gaussian based filter has smooth edges, image noise is suppressed more effectively than using a box (or square) shaped filter. Therefore it may seem that a similar approach to the design of the mask used in SUSAN feature detection would be preferable. Indeed, in the brightness domain, it was found worthwhile to use a smooth brightness comparison function (as shown in Equation 4), rather than having thresholding done with a sharp cutoff. In the case of the spatial extent of the mask, the equivalent process would be achieved by using

$$n(\vec{r_0}) = \sum_{\vec{r}} e^{\frac{-(\vec{r} - \vec{r_0})^2}{2\sigma^2}} c(\vec{r}, \vec{r_0}) \tag{26}$$

in place of Equation 2, where σ is a suitable scaling factor. This would give a circular mask with pixels contributing less to the calculations as their distance from the nucleus increased – a Gaussian profile. An edge detector based on this kind of mask has been fully implemented and tested.

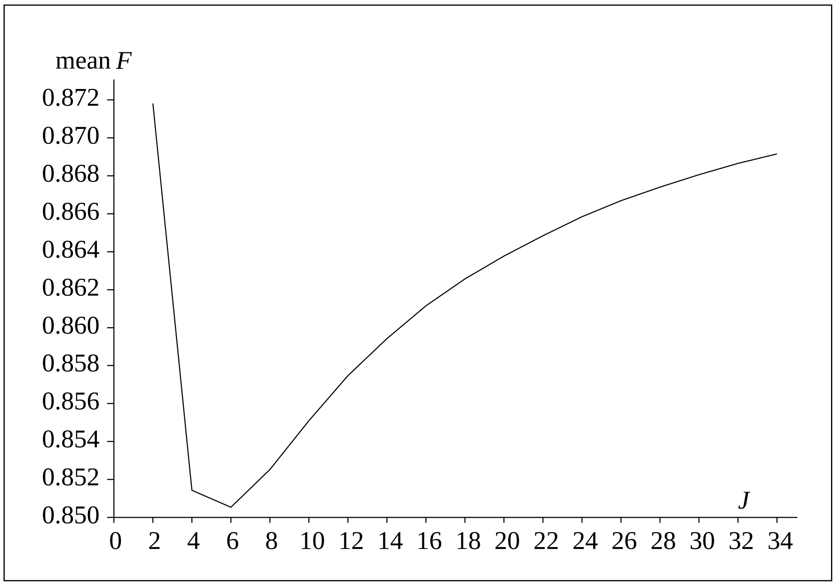

Figure 8: A plot of an objective formulation of expected false negatives and positives against the J factor in the brightness comparison function.

The difference in practice between the Gaussian spatial SUSAN detector and a sharp cutoff SUSAN detector turns out to be minimal; there is very little difference in the final reliability or localization of reported edges. The explanation for this possibly surprising result is that, unlike the differential edge detector, no image derivatives are used, so that the problem of noise is vastly reduced in the first place. Also, a linear filter is performing a different task to the initial SUSAN response. In the former, a filter has a Gaussian profile in a "brightness versus distance" plot. However, the latter uses either a sharp circular mask or a Gaussian profile in the spatial domain and smoothed thresholding in the brightness domain. Thus "half" of the algorithm already has a smooth cutoff action. From the results, it seems that it is this part (the brightness domain) which is more important. Thus it does not appear necessary to use a Gaussian spatial mask, although it is easy to do so if desired.

4.4 Testing of the SUSAN Edge Detector

A quantitative test of the initial response of the SUSAN detector compared with four other edge enhancement algorithms was described in Section 3. To give a more meaningful explanation of the tests described in [71] than the short one given earlier would require an undeserved amount of space. Suffice it to say that the initial response given by SUSAN was better than the best results of the four detectors used in these tests, using all six suggested "failure measures".

A one dimensional test of the SUSAN response to various edge types has been carried out using the input signal shown in Figure 9. The results are good. All the edge types are correctly reported apart from the roof edge. (Note that even the roof edge produces a very small local maximum in the response.) The ridge edges are picked up correctly, and produce only one response, as desired. Thus the SUSAN edge detector is shown to report lines correctly and not just edges. With the two pixel width ridge it is of course a matter of definition whether one ridge or two edges should be reported. The response is symmetrical about each edge type, which results in sub-pixel accuracy giving excellent results.

The roof edge obviously does not conform to the (fairly lenient) model of an edge which the SUSAN detector is based on. In [9] Canny states that

"A roof edge detector has not been incorporated into the implementation of the edge detector

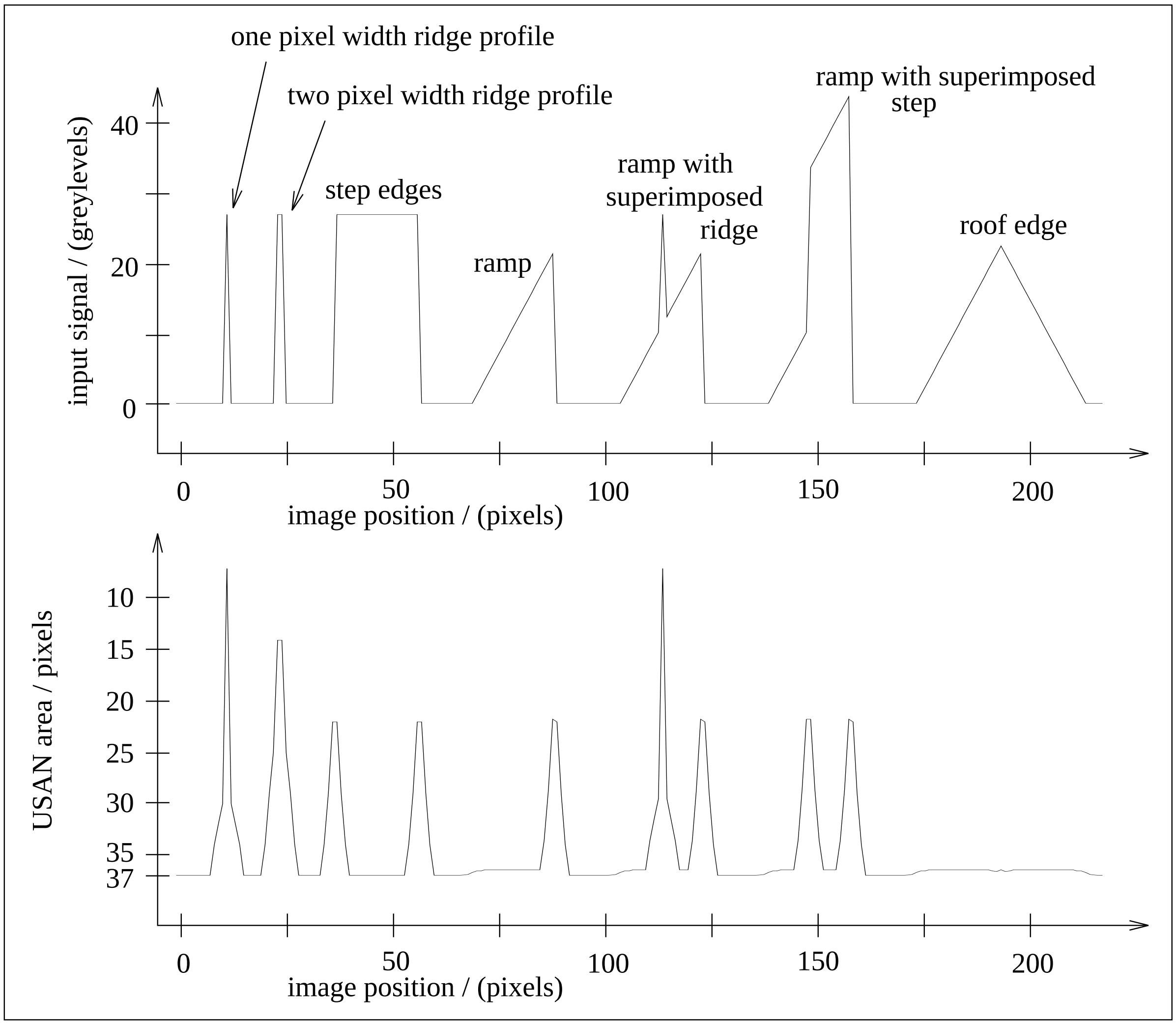

Figure 9: The initial response of the SUSAN edge finder to various edge types.

> because it was found that ideal roof edges were relatively rare."

This is a common approach, and indeed, only second order derivative detectors (including Noble's edge detector, but not including second order zero-crossing detectors) will give a response to the roof edge. If edge finding is to be based on the human visual system, then roof edges (and step changes in the first derivative in general) should not be counted as edges at all anyway; humans can rarely detect this sort of image feature, and find it very hard to localize.

To test both edge detectors and two dimensional feature detectors a test image which includes two dimensional structures of many different types has been designed.[10] This is shown in Figure 10. Note the corners of widely varying angles, various junctions with more than two regions meeting at a point, and the region created by taking a brightness ramp and raising a rectangle in its centre by a uniform amount. An exact description of this test image can be found in [63].

Figure 11 shows the output of the SUSAN edge detector. Note that where a reported edge appears to

[10] All images shown in this paper are 256 by 256 pixels in size, except in a few cases where small sections of images have been used, or where otherwise stated. Where appropriate, images have been scaled horizontally for display.

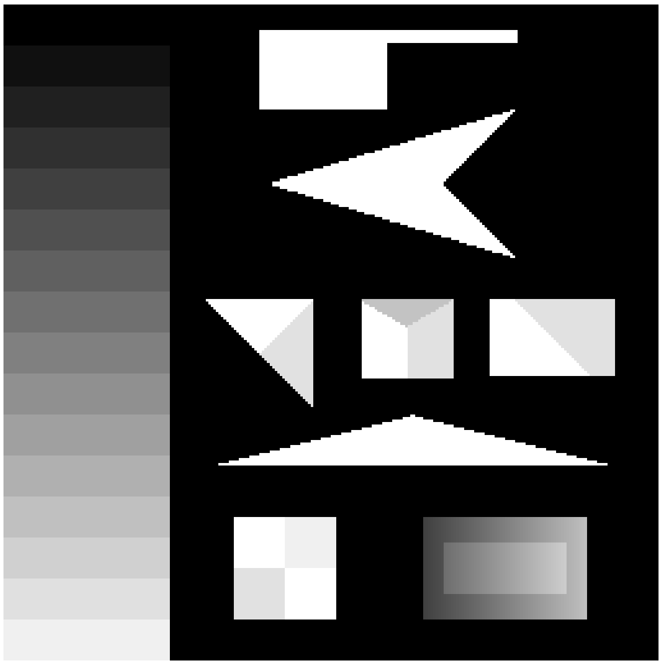

Figure 10: An image designed to test corner and edge detectors.

jitter between one side of an image edge and the other, the effect is due simply to quantization of the marked edge position. In reality, giving the figure at a higher resolution (i.e. with sub-pixel accuracy) would show the edges to lie on the image edges. It can be seen from this diagram that the SUSAN edge detector possesses the following attributes:

1. Edge connectivity at junctions is complete.
2. The reported edges lie exactly on the image edges. (Because of image quantization, they lie on pixels on one side of the edge or the other.)
3. The edges around and inside the brightness ramp are correctly found and no false edges are reported. Thus it is shown that the edge detector copes with finding edges of regions with low or smooth variation of brightness.

The output of a standard implementation of the Canny edge detector using a very small Gaussian smoothing value (to give the best possible output) and with the final threshold optimized to just pick up the weakest correct edges is shown in Figure 12. Note the following points of interest:

1. Very few junctions involving more than two edges are correctly connected.
2. Some sharper corners have broken edges.
3. Incorrect edges within the brightness ramp were nearly as strong as the weakest correct edges. They have been eliminated by selecting exactly the right final threshold for binarization.
4. For normal real images a larger value of σ is needed, usually at least $\sigma = 1.0$. This results in even greater breaks at junctions, and incorrectly rounded corners.
5. The Canny edge detection is approximately 10 times slower than SUSAN edge detection.

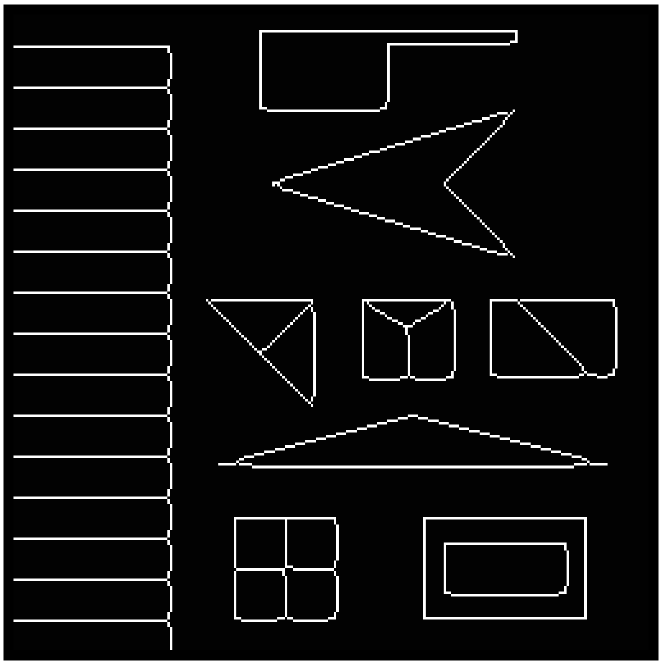

Figure 11: Output of the SUSAN edge finder (t=10) given the test image.

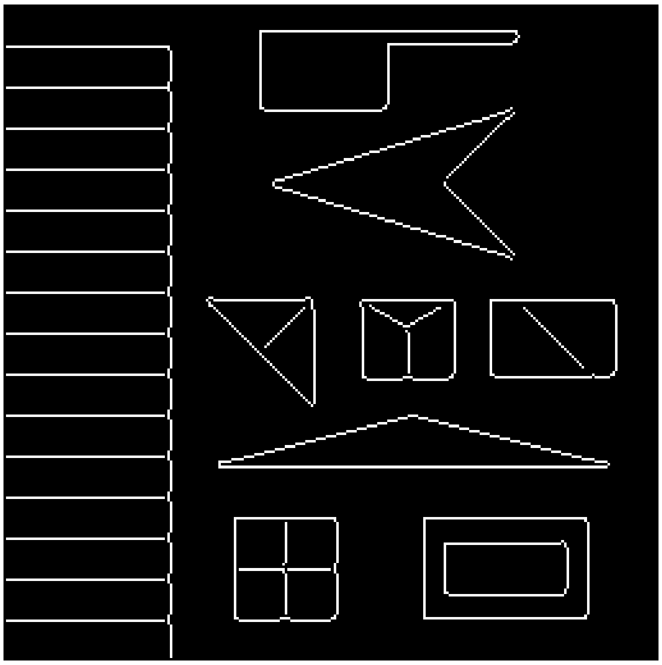

Figure 12: Output of the Canny edge finder (σ=0.5) given the test image.

Figure 13 shows the output of the SUSAN edge finder when Gaussian noise is added to the test image. The noise added gives a minimum signal to noise ratio[11] of 5. This amount of noise is greater than would be expected in a normal real image by a factor of about 3. The results are good. A lot of the noise in the reported edges is only apparent, as the noise simply forces the edge to toggle between lying on one side of the real edge or the other, with no loss of accuracy. Continuity is still preserved, in even the sharpest angles (note, of course, that pixels touching diagonally constitute continuity). The output of the Canny edge finder degrades slightly more than the output of the SUSAN algorithm; see Figure 14.

Figures 15 and 16 show the result of running the SUSAN edge finder on a digitized real image of some printed text. Note that edges (here and in some of the other examples) have been marked in white with a black border so that all the marked edges are visible. The small (three by three) mask has been used for the SUSAN edge detector, and Canny has been carefully optimized to give the best possible results, by choosing a small value for σ and the best response threshold. Note that not only does the Canny detector have broken edges at several places, but the boundaries of the dots comprising the colon are not very clean circles. The original dots' boundaries *are* fairly good circles, and are better represented by the output of the SUSAN edge finder. This suggests that the SUSAN edge detector is more isotropic than a derivative based detector, as one would expect.

Two more real examples of testing the SUSAN edge detector are given in Figures 17 and 18. Figure 17 is taken from the testing of the motion segmentation and tracking system ASSET (see [65]), where the SUSAN edge detector is used to fine-tune motion segmentation boundaries, enhancing the final accuracy. In Figure 18 note that junctions are correctly handled. In both pictures very slight breaks in the reported edges appear due to the dithering process used by the printer.

The results of the tests described show the SUSAN edge detector to be accurate, stable and very fast on both test and real images. It has given very good results in all the tests.

Finally, the idempotence of the SUSAN principle is investigated. It has been suggested (e.g., see [70]) that feature enhancement should be idempotent. This means that after one application of the enhancement process, further applications should make no change to the processed image. The supposed desirability of this property is based on three notions:

1. Human vision interprets a scene as a series of edges, and will interpret a line drawing of the same scene in the same way. Thus it may be argued that the human visual process when fed by its interpretation of a scene will give the same interpretation. See [70].

2. If a process can be applied to its output with no change then its non-linear parts (e.g. thresholding) are functioning correctly.

3. If a process is idempotent then it is clear that only one application of the process is necessary to achieve the maximum "enhancement" possible. This *could* be seen as signifying efficient design.

Although the authors are of the opinion that the property of idempotence is not of very great importance, for the sake of interest, and of completeness, testing has been carried out which shows SUSAN edge detection to be idempotent. See [70] for a theoretical analysis of the Sobel, Marr-Hildreth and local energy (three-point) operators. It is shown theoretically that the local energy operator is idempotent. However when tested on a chequerboard pattern it was found that none of the tested operators was idempotent. On a real image the Sobel and energy operators gave similar results; each edge gave rise to multiple responses by the third application of the operator.

Investigation of idempotence can be applied at two very different levels for feature detectors. Repeating the initial enhancement on itself is one level, that is, only using the first stage of a feature detector (for example, Sobel enhancement). Taking the final output of the complete algorithm (including non-maximum suppression and thinning if appropriate) and feeding this back into the complete algorithm is the other level (for example, the Canny edge detector).

[11] The minimum signal to noise ratio is taken to mean the ratio of the smallest edge height to the standard deviation of the added Gaussian noise.

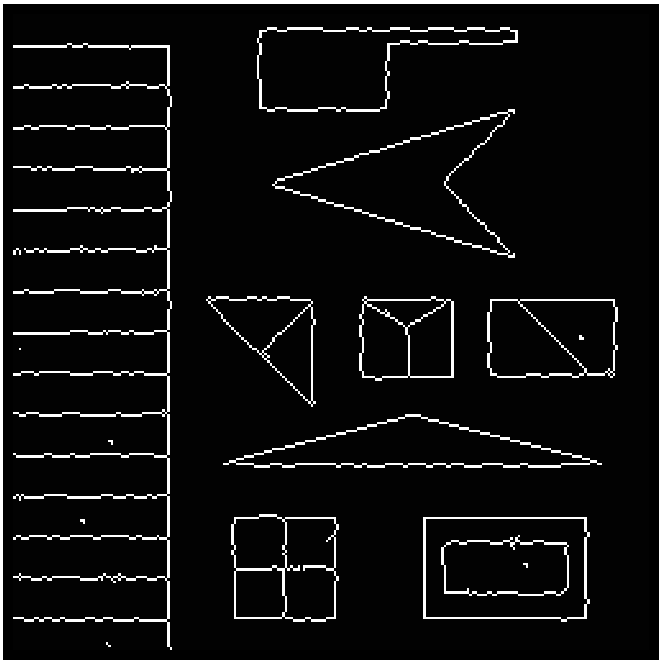

Figure 13: Output of the SUSAN edge finder (t=10) given a test image with Gaussian noise of minimum SNR=5 added.

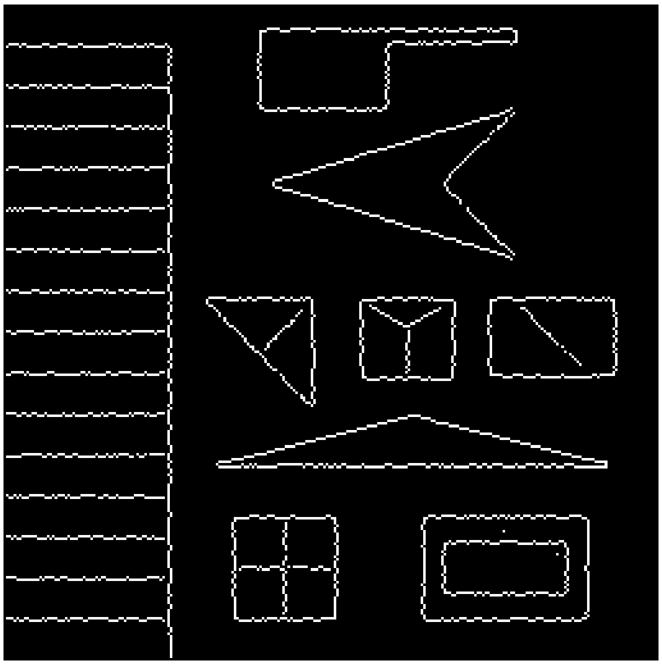

Figure 14: Output of the Canny edge finder (σ=0.5) given the test image with Gaussian noise of minimum SNR=5 added.

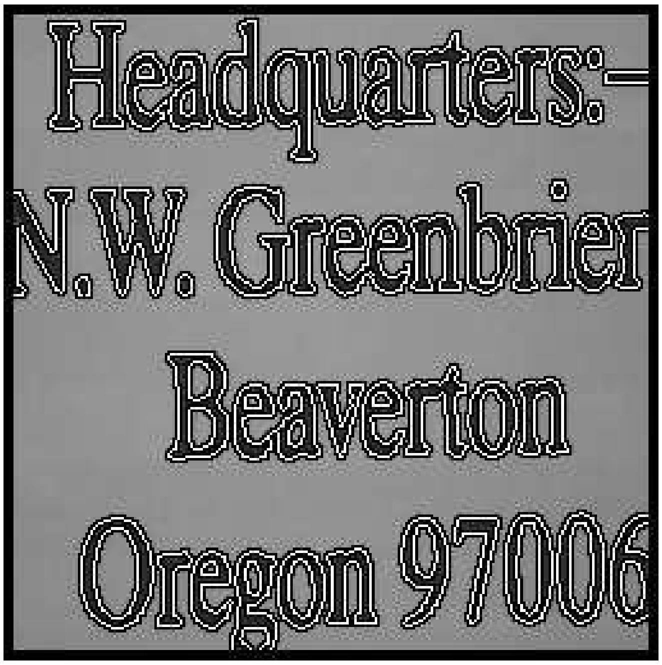

Figure 15: Output of the SUSAN edge finder (t=20, 3 by 3 mask) given the “letters” image.

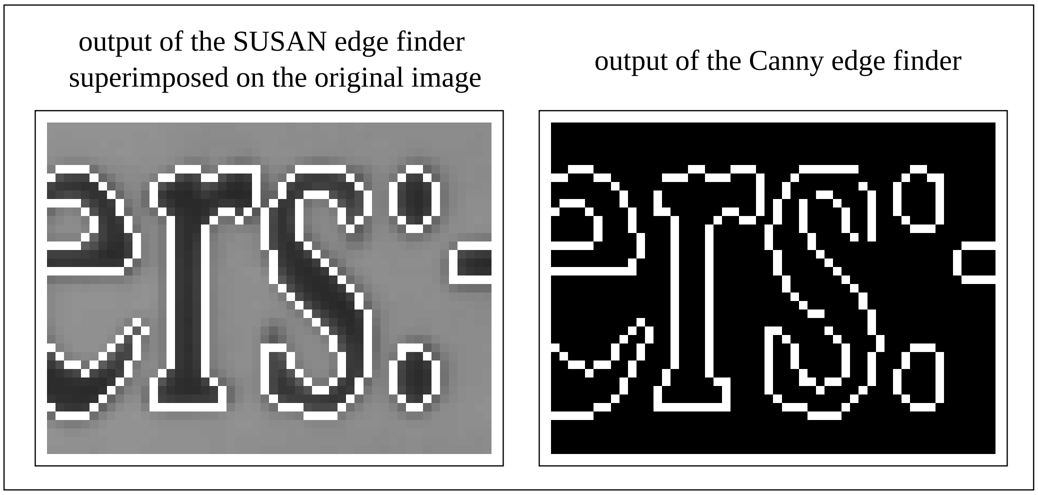

Figure 16: Small sections of the outputs of the SUSAN (t=20, 3 by 3 mask) and Canny (σ=0.5) edge finders when given the “letters” image.

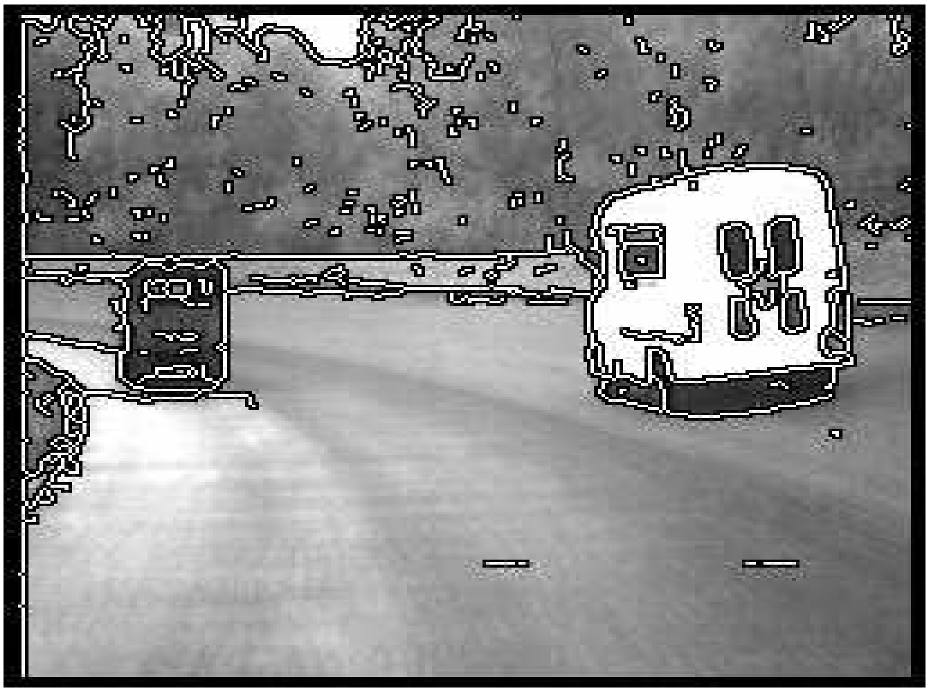

Figure 17: Output of the SUSAN edge finder (t=25) given the road image.

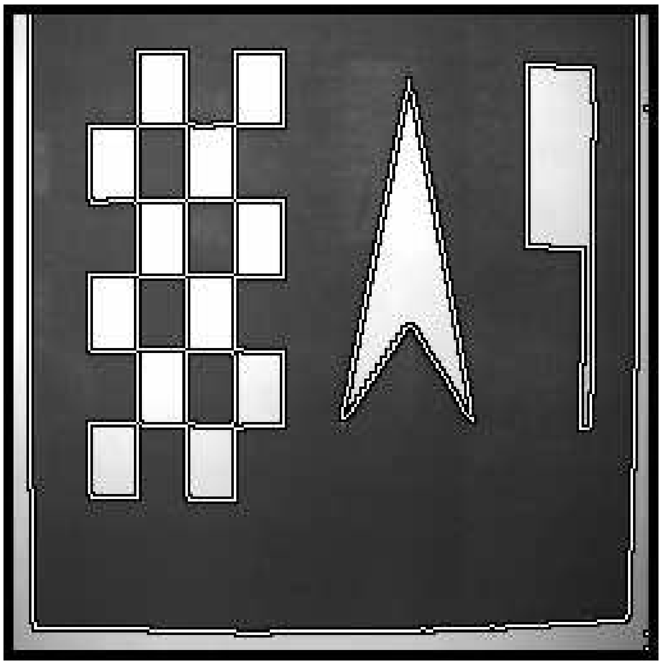

Figure 18: Output of the SUSAN edge finder (t=20) given a real test image. The image was created by laser printing a synthetic test image and then digitizing it using a video camera.

In the testing performed the test image was fed into an edge enhancement algorithm, this output was fed back into the algorithm, and the output of this was fed back in once more. Thus each algorithm was run a total of three times. For each algorithm the outputs after one pass and after three passes were examined. Due to lack of space the visual results must be seen in [63]. The results show that the initial response of the SUSAN enhancement gives maximal ridges (which would usually be suppressed to be one pixel thick) which are still evident, and no double edges have been produced. In contrast, a 3 by 3 Sobel operator produces multiple responses at nearly all of the edges.

Testing the *complete* SUSAN algorithm shows that after three passes the results are virtually unchanged from the initial thinned edge map.[12] In contrast, the Canny algorithm produces multiple responses at nearly all of the edges.

The results show that, as would be expected from the nature of SUSAN feature enhancement, the SUSAN edge detector is idempotent.

4.5 Conclusions

This section has described application of the SUSAN principle to the detection of image edges and lines accurately and quickly. The localization of the features is independent of the mask size used, and noise suppression is good. Connectivity of edges and lines at junctions is good. The SUSAN principle can be viewed as an efficient way of finding features using local information from a pseudo-global viewpoint; it is the image regions which define the features at their edges and boundaries, and the features themselves are not the *primary* objects of investigation.

5 The SUSAN Two Dimensional Feature Detector

A brief review of existing feature detectors is now given. (For a longer review, see [63].) This is followed by details of the SUSAN "corner" finding algorithm, where they differ from edge finding, and an analysis of the algorithm's validity. Finally, examples of the output of the algorithm are presented and discussed.

5.1 Review

Mathematical definitions of image structures that are well localized in two dimensions have been even more diverse in the past than definitions of edges. Much of the work on two dimensional features has concentrated solely on "corners", in name and in practice, that is, features formed at boundaries between only two image brightness regions, where the boundary curvature is "sufficiently high". Much theory has been presented which is only strictly accurate for such simple image structures. However, many other types of localized structure arise, for example, "T", "Y" and "X" junctions, as well as less rigidly defined structures. These will not usually fit into the model[13] of corner finding theory, and so will not be correctly dealt with.

A few methods which find *points of localized image structure* directly (i.e., take the italicized phrase as the "brief" for developing an algorithm) have been used, for example the Moravec interest operator and the Plessey algorithm.[14] These are described below. This relaxation of the corner "definition" is to be commended; however, they each have their own shortcomings. In the case of the Plessey feature point detector, the increase in reliability of point detection and in the variety of points which may be detected (when compared with "corner" based methods, such as Kitchen-Rosenfeld and Dreschler-Nagel) is balanced by a loss in localization accuracy (for reasons explained below).

[12] This result also shows the success of the SUSAN edge detector when used for finding lines in the image.

[13] Here *model* means the type of image structure assumed to be present when discussing any particular aspect of corner finding theory. Note that almost all approaches to corner finding assume a simple corner *model*, whether or not they are "model based".

[14] The Plessey corner detector (also known as the Harris detector) should perhaps be referred to as the (very closely related) Förstner detector, as the latter appears in earlier literature. However, the Plessey detector is more widely known and referenced, so for ease of understanding the name Plessey is used here.

The SUSAN detector makes no assumptions about the form of the local image structure around a well localized point. Neither does it look for "points of interest". Instead, it analyzes different regions separately, using direct local measurements, and finds places where individual region boundaries have high curvature, i.e., finds corners formed by single regions. Thus at junctions involving more than two regions, for example, "T" junctions, more than one region may contribute to the detection of a "corner", and all junctions will be correctly processed, no matter how complicated they are. Also, no assumptions about the internal structures of the regions around a "corner" are made; it is common to assume that the regions are constant in value, but in the case of the SUSAN detector, sloping regions are allowed, as shown in later examples.

A few methods of using binary edge maps to find corners have been suggested. The edges are detected and then edge curvature is calculated in order to find corner locations. See, for example, [20], [3] and [36]. Clearly this sort of approach has severe problems with junctions.

In [38] and [39], Moravec developed the idea of using "points of interest". These are defined as occurring when there are large intensity variations in every direction. This definition is realized by computing an un-normalized local autocorrelation in four directions and taking the lowest result as the measure of interest. This response is thresholded and then local non-maxima are suppressed. However, some problems with the Moravec operator have been identified (principally by the proponents of the Plessey detector, which builds on the same feature definition): the response is anisotropic, due to the use of only four directions for finding the local autocorrelation, the response is noisy, as the profile of the "window" used for finding the autocorrelation is square, and the operator is sensitive to strong edges, as the response is given by the minimum in the autocorrelation measurements, and not by the variation in them.

In [4] Beaudet enhanced high curvature edges (i.e., looked for saddle points in the image brightness surface) by calculating image Gaussian curvature (the product of the two principle curvatures). In [29] Kitchen and Rosenfeld used a local quadratic surface fit to find corners. The parameters of the surface were used to find the gradient magnitude and the rate of change of gradient direction; the product of these quantities was used to define "cornerness", and local maxima were reported as corners. In [16] Dreschler and Nagel defined corners to be points lying between extrema of Gaussian curvature. Later, (for example, see [41]) Nagel used the definition (of corner position) of "the point of maximum planar curvature for the line of steepest slope". In [76] Zuniga and Haralick found corners at significant changes in curvature along edges.

In [43] Noble shows that these four approaches are all making the same underlying measurement, that is, the product of the gradient magnitude and the edge contour curvature. They are all sensitive to noise, and have relatively low reliability, as shown in various tests in [24].

In [15] Deriche and Giraudon use Beaudet's "cornerness" measure in conjunction with edge detection to obtain accurate corner localization. Corners are found at two different scales, and lines are drawn between a corner at one scale and the same corner at the other. Then the intersection of this line with the nearest zero crossing of the Laplacian edge is defined as the correct position of the corner. It is not clear whether sensitivity to noise and reliability of detection is improved with this method.

In [74] Wang and Brady develop the above curvature based methods. As well as requiring that curvature be a maximum and above a threshold, they require that gradient perpendicular to the edge be a maximum and above a threshold. Also false corner response suppression is performed to prevent corners being wrongly reported on strong edges. However, in the presence of large amounts of noise, this is not sufficient to prevent false responses on strong diagonal edges. The corners are found at different smoothing levels allowing an estimation of the corner positions at zero smoothing.

In [26] Harris and Stephens described what has become known as the Plessey feature point detector. This is built on similar ideas to the Moravec interest operator, but the measurement of local autocorrelation is estimated from first order image derivatives. The variation of the autocorrelation over different orientations is found by calculating functions related to the principle curvatures of the local autocorrelation. This well conditioned algorithm gives robust detection, that is, feature points are reliably detected. However, localization accuracy is poor, particularly at certain junction types (see results presented in Section 5.4). This occurs for two reasons. The first is that the method can only work accurately at most junctions if the masks used to find the derivatives and to perform smoothing have zero size. This can be easily understood in the case of “T” junctions, where the two “smaller” regions at the junction have similar brightness, and the “larger” region has a very different brightness from the other two. See, for example, the junction near the lowest and leftmost marker shown in Figure 21. Clearly, for finite mask sizes, one curvature (that measured perpendicular to the strong edge) will have a maximum value all along that edge. However, the other curvature (that parallel to the strong edge) will have a value which is constant at all points along the weak edge and which starts to drop as the strong edge is approached, the distance at which this begins to occur being half the size of the masks used. Thus the position at which both curvatures are largest is not well defined, and when some compromise position is chosen, it will be displaced away from the real junction position. The other reason for localization error is explained fully in [62], where it is shown that the actual measure of interest depends to a large extent on the relative values of the two curvatures, and is greatest when they are equal. In the case described above, the two curvatures will be equal at some point along the weak edge, away from the strong edge, and therefore away from the correct position of the junction.

In [43] Noble relates the Plessey operator to the four “greylevel corner detectors” mentioned above, and shows that it can be viewed as measuring image curvature also.

In [19] Förstner and Gülch describe a method which appears to use exactly the same measure of “cornerness” as the Plessey operator (note the earlier statement that this work predates the Plessey paper). However, a more complicated implementation is used; the detection and localization stages are separated, into the selection of windows, in which features are known to reside, and feature location within selected windows. This results in improved localization accuracy with some corner types, but the algorithm is, as one would expect, slower than the Plessey detector.

In [12] Cooper et. al. use image patch similarity tests in two edge based corner finding algorithms. In the first method the similarity function (an un-normalized local autocorrelation function) is found at positions in both directions along an edge; if the function gives a high result in either direction parallel to the edge then a corner is reported. This is clearly similar to the principle behind the Moravec and Plessey detectors. In the second method the results of the similarity tests performed along the edge are used to estimate image curvature directly. In both cases the localization accuracy at junctions is poor.

In [43] Noble uses mathematical morphology (as mentioned earlier), applied to the approach of facet based structure modelling, to propose a two dimensional feature detector. Gaussian curvature is used to determine whether local patches are elliptic, hyperbolic, parabolic or flat; this information is extracted from the image using morphological techniques. It is not clear how well junctions will be dealt with, as no detector is actually developed, and results of testing the morphological approach on two dimensional features are not presented.

In [70] Venkatesh uses the local energy approach discussed earlier to find two dimensional features. Maxima in local energy are searched for along the edge direction. Only tests performed on very simple (and noise free) images are presented.

In [53] Rosenthaler et. al. use an integrated edge and “keypoint” detector to find two dimensional features. Oriented directional energy filters are used to find first and second derivatives of energy along the direction of each filter; these are combined to give a final response which is a maximum at “keypoints”. The results presented are good, but performance in the presence of noise is not investigated.

In [33] Liu and W.-H. Tsai assume that only two regions form a corner. The image is divided up into patches, and each patch is segmented into two regions using grey level moment preserving thresholding. The two regions are then analyzed to determine whether a corner exists anywhere within the patch. If it does, the corner's position and angle are found. The method clearly cannot cope with junctions. It has apparently only been tested on very simple images, and does not function very well in the presence of noise.

In [37] Mehrotra et. al. use half-edge detection to find corners. Directional masks which are zero in one half are used to find either gradient maxima or zero crossings of the second derivative. (The line splitting the mask into “zero” and “non-zero” parts is perpendicular to the measured edge direction. Large masks are used at eight orientations.) Then for each edge point the difference in orientation between the neighbouring edge points is used to find corners. Results are presented for very simple images only.

In [52] Rohr uses strict models to detect and localize corners and junctions. The assumptions are made that regions making up junctions have constant underlying values and that junctions are well isolated in the image. Large masks are used to fit local structures to models with several parameters, including the feature position and orientation, the intensities of regions touching at the junction, the amount of blurring, and the relative positions of the edges forming the junction. Thus feature detection from a large scale viewpoint is performed. Results are only presented for very simple low noise images.

In [58] Singh and Shneier attempt to overcome the problem of the trade-off between reliable detection and accurate localization by fusing various corner finding methods. Initially a template based corner detector is described. Three different methods for performing the matching between the image and the template are given, and it is shown that they have differing degrees of reliability and accuracy. (These two quantities are related to each other for any one method by a qualitative "uncertainty principle".) Fusing the three methods' outputs should give both reliable and accurate results. In the second part of the paper, a gradient based approach is taken, similar to that of Kitchen and Rosenfeld. Different methods are described, each using a slightly different definition of "cornerness". Again, the different methods are fused to provide a final list of detected corners. Results presented appear poor.

In [46] Paler et. al. note that at a corner the median of local brightness values taken over a small mask is significantly different from the centre value. Thus the difference between the median and the centre value is used to produce a corner response. This approach is limited to situations where the edge widths and the contrast between "the object and the background" can be accurately estimated. Junctions cannot be dealt with at all.

5.2 The SUSAN "Corner" Finder in Detail

The SUSAN "corner" finder is very similar to the edge finder, particularly in the early stages.

As before, all pixels within a circular mask are compared with the nucleus, using exactly the same comparison equation as the edge detector, i.e., Equation 4. The sum of these comparisons is also calculated in the same way, using Equation 2.

Next, n is again compared with the geometric threshold g. Here the "corner" detector is quite different from the edge detector, where g was only necessary in the presence of noise. For a "corner" to be present, n must be less than half of its maximum possible value, $n_{\max}$. This idea was introduced in Section 2, and is equivalent to saying that if the nucleus lies on a corner then the USAN area will be less than half of the mask area, and will be a local minimum. It is safe to set g to be exactly half of $n_{\max}$ because of the nature of a quantized straight edge; an USAN formed from a straight edge will always be larger than half of the mask size, as it includes the nucleus.

In feature extraction algorithms there will always be at least one threshold which needs to be set up. This is so that the response given by feature enhancement can be used to distinguish between features and non-features. This is a restatement of the fact that to reduce the amount of data in the image, there must be at least one place in the algorithm where a "decision" or non-linear mathematical process occurs. Setting the value of the threshold is usually critical if the algorithm is to behave correctly. The dependence of the correct threshold on the data and the capability of setting it without human intervention are two factors which have a large effect on the success of the algorithm. In other words, if a threshold affects the quality of output of an algorithm in such a way that it is not possible to set it automatically to the optimum value appropriate to the input data, then the algorithm is eventually of limited use, although it may appear to give fine results on individual (hand tuned) examples.

There are two main types of threshold; these can be loosely labelled "quality" and "quantity". Most thresholds used in algorithms will fall partly into both categories, but often more into one than the other. The thresholds g and t are good examples of the two categories. The geometric threshold g clearly affects the "quality" of the output. Although it affects the number of "corners" found, much more importantly, it affects the shape of the "corners" detected.[15] For example, if it were reduced, the allowed "corners" would be sharper. Thus this threshold can be fixed (to the value previously explained) and will need no further tuning. Therefore no weakness is introduced into the algorithm by the use of the geometric threshold. The

[15]This assumes that the USAN is a contiguous region. The refinements described later are designed to enforce this contiguity.

brightness difference threshold is very different. It does not affect the quality of the output as such, but does affect the number of "corners" reported. Because it determines the allowed variation in brightness within the USAN, a reduction in this threshold picks up more subtle variations in the image and gives a correspondingly greater number of reported "corners". This threshold can therefore be used to control the "quantity" of the output without affecting the "quality". This can be seen as a negative or a positive point. On the one hand, it means that this threshold must be set according to the contrast, noise etc. in the image, and on the other, this threshold can be easily tuned to give the required density of "corners". In practice, there is no problem. A value of 25 is suitable for most real images, and if low contrast images need to be catered for, the threshold can be varied automatically, to give a final number of reported "corners" which is appropriate to the higher level uses of the "corners". When SUSAN was tested on an extremely low contrast image this threshold was reduced to 7, to give a "normal" number of corners. Even at this low value, the distribution was still good (i.e. not over-clustered) and the "corners" found were still quite stable.

Note also that if thresholds are used in an intelligent way (compare Equation 4 with Equation 1), the introduction of instability to the overall algorithm can be minimized, and thus one of the major disadvantages of using intermediate non-linearities can be largely eliminated.

Thus Equation 3 is used to create an initial response image as before, but using a different value for g. (Note that in a combined "corner" and edge detector it would be trivial to form an initial response independent of the value of g, by setting g equal to $n_{\max}$. The edge and "corner" detector could then threshold the initial response at a later stage thus saving a large amount of effort.) The final stage in the SUSAN two dimensional feature detector is to search the initial response over square 5 by 5 pixel regions for local maxima (above zero). However, before this non-maximum suppression is carried out, two small procedures are run to reduce false positive responses from edges and from noise.

As described thus far, SUSAN will give false positives in some circumstances. This can occur with real data where blurring of boundaries between regions occurs. See, for example, Figure 6, case (c), where there is a thin line with a brightness approximately half way between the two surrounding regions. A thin line such as this will usually be broken up, and, being usually only one pixel thick, it may cause corners to be wrongly reported.

This problem has been eliminated by the following method. The centre of gravity of the USAN is found. (This is only done if the main test was passed, so that the algorithm does not suffer much of a slow down.) Then the distance of the centre of gravity from the nucleus is found. Clearly an USAN corresponding to a proper corner will have a centre of gravity that is not near the nucleus, whilst the false positives can be rejected here because a thin line passing through the nucleus will have a short corresponding distance from the centre of gravity to the nucleus. This simple addition to SUSAN is effective both in theory and when tried on real images.

The final addition to SUSAN is a simple rule which enforces contiguity in the USAN. This is occasionally necessary in real images where there is a lot of noise or fine complicated structure. The step removes false positives. All of the pixels within the mask, lying in a straight line pointing outwards from the nucleus in the direction of the centre of gravity of the USAN must be part of the USAN, for a corner to be detected. This is effective in forcing the USAN to have a degree of uniformity, and reduces false positives in the reported corners.

The final stage of non-maximum suppression results in a list of features being reported. In summary then, the algorithm performs the following steps at each image pixel:

1. Place a circular mask around the pixel in question (the nucleus).
2. Using Equation 4 calculate the number of pixels within the circular mask which have similar brightness to the nucleus. (These pixels define the USAN.)
3. Using Equation 3 subtract the USAN size from the geometric threshold (which is set lower than when finding edges) to produce a corner strength image.
4. Test for false positives by finding the USAN's centroid and its contiguity.

5. Use non-maximum suppression to find corners.

5.3 Analysis of the SUSAN "Corner" Detector

In edge detection the exact position of an edge is fairly well defined. However, as explained earlier, the exact position of a two dimensional feature is more subjective. Thus there is no strong reason why mathematical understanding of how the SUSAN "corner" detector works should give any more validity to the algorithm than the "intuitively picturesque" descriptions already given, or an analysis of results. However, a brief mathematical analysis of the algorithm is now given.

For a corner to be detected, a local minimum in n must be found;

$$\frac{dn(\vec{r_0})}{d\vec{r_0}} = 0, \tag{27}$$

that is, differentiating with respect to both components of $\vec{r_0}$ must give zero. (n must also be below the geometric threshold. This is accounted for later.) Thus, using Equations 4 and 2 we have

$$\frac{d}{d\vec{r_0}} \iint_{\text{over mask}} e^{-\left(\frac{I(\vec{r})-I(\vec{r_0})}{t}\right)^6} dx\, dy = 0, \tag{28}$$

where $dx\, dy$ is the elemental area within the mask. (The summation has been replaced by a double integral to clarify the analytical approach later.) This gives on differentiation

$$\iint_{\text{over mask}} B(\vec{r},\vec{r_0}) \left(\frac{dI}{d\vec{r_0}}(\vec{r}) - \frac{dI}{d\vec{r_0}}(\vec{r_0})\right) dx\, dy = 0, \tag{29}$$

with B given by

$$B(\vec{r},\vec{r_0}) = \left(\frac{I(\vec{r}) - I(\vec{r_0})}{t}\right)^5 e^{-\left(\frac{I(\vec{r})-I(\vec{r_0})}{t}\right)^6}. \tag{30}$$

This function B has the property of picking out the narrow bands that lie on the boundaries between the univalue regions; it is zero everywhere apart from fairly sharp peaks centred at $\mathrm{I}(\vec{r}) = I(\vec{r_0}) \pm t$. See Figure 5(c). In other words, in the context of this algorithm, it is acting as a region boundary detector. Figure 19 shows a simplified example of the area picked out by B.

Thus the integral is effectively

$$\iint_{\text{over edges}} \left(\frac{dI}{d\vec{r_0}}(\vec{r}) - \frac{dI}{d\vec{r_0}}(\vec{r_0})\right) dx\, dy = 0, \tag{31}$$

giving, on integration,

$$A\, \frac{dI}{d\vec{r_0}}(\vec{r_0}) = \iint_{\text{over edges}} \frac{dI}{d\vec{r_0}}(\vec{r})\, dx\, dy, \tag{32}$$

where A is the effective total area of the narrow bands that make up the edges between the univalue regions (the bands picked out by B). Dividing by A, we have

$$\begin{aligned} \frac{dI}{d\vec{r_0}}(\vec{r_0}) &= \frac{1}{A} \iint_{\text{over edges}} \frac{dI}{d\vec{r_0}}(\vec{r})\, dx\, dy \\ &= < \frac{dI}{d\vec{r_0}}(\vec{r}) >_{\text{over edges}} . \end{aligned} \tag{33}$$

This equation expresses the fact that the intensity differential at the nucleus must be the same as the intensity differential averaged over all of the boundaries. This must be the case for both the x and y

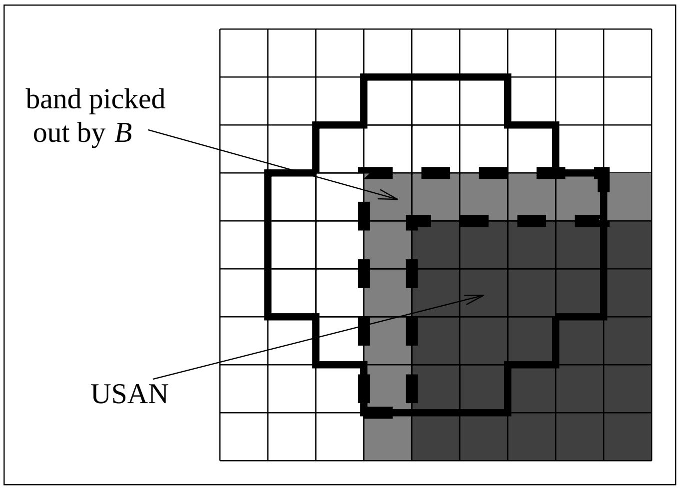

Figure 19: Part of a typical real scene, including a 90° corner, with edges slightly blurred. The area picked out by function B is surrounded by the thick dotted line.

differentials. For this to hold, the nucleus must be placed on a line of reflective symmetry of the boundary pattern within the mask area. This is so that the contributions to the average differential on either side of the nucleus may balance out. This is equivalent to saying that the nucleus must lie on a local maximum in the edge curvature; consider a Taylor's expansion of the edge curvature. At the smallest local level, the nucleus will only lie on a centre of symmetry of the curvature if it lies at a maximum (or minimum) of curvature.

The conditions derived ensure that the nucleus lies not only on an edge, but also that it is placed on a sharp point on the edge. It will as it stands give a false positive result on a straight edge; this is countered by not just finding local minima in n, but by forcing n to be below the geometric threshold g.

Finally, an explanation is needed of why the SUSAN "corner" detector is a successful two dimensional feature detector, and not just a simple corner detector. As will be seen shortly, multi-region junctions are correctly reported by the feature detector with no loss of accuracy. This is expected, as the corner finder will look for the vertex of each region individually; the presence of more than two regions near the nucleus will not cause any confusion. (This is a strong advantage over derivative based detectors, where the two dimensional derivative model does not hold at multi-region junctions.) The local non-maximum suppression will simply choose the pixel at the vertex of the region having the sharpest corner for the exact location of the final marker. Thus the different types of two dimensional junction are not in fact seen as special cases at all, and are all treated correctly.

5.4 Testing of the SUSAN "Corner" Detector

The results of testing this corner finder are now shown and discussed.

Firstly, the output of SUSAN given the test image (shown earlier in Figure 10) is shown, in Figure 20. Compare this with Figure 21, the output from the Plessey corner finder, described earlier. The spatial smoothing scale σ and the sensitivity were both adjusted to give the best possible output from the Plessey algorithm. The accurate localization and reliability of the SUSAN algorithm is apparent; it does all it is expected to do. The inaccuracies and false positives of the Plessey corner finder, even at simple two region corners, are visible. This is fully explained in [62]. With respect to speed, SUSAN took 0.3 seconds to process this picture on a single Sun SPARC-2 processor; the Plessey corner finder took 3.5 seconds.

The SUSAN algorithm has also been tested with respect to its sensitivity to noise. The results are excellent; the quality of its output (both the reliability and localization) degrades far less quickly than other

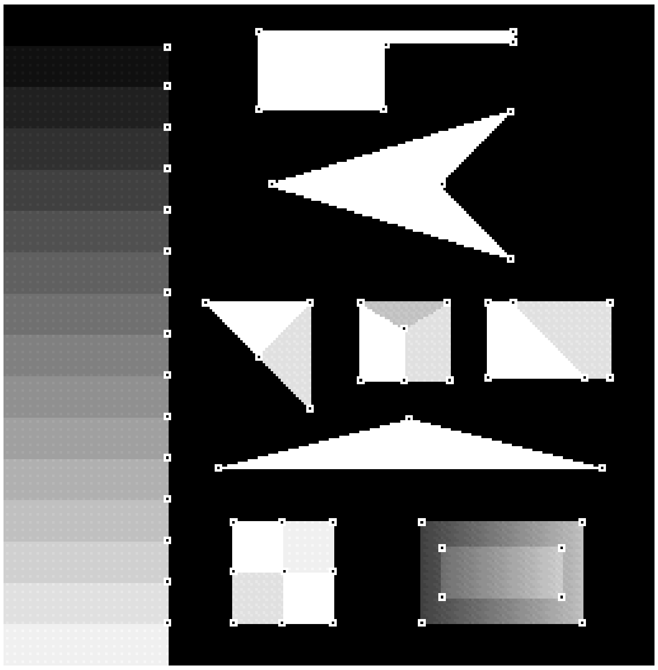

Figure 20: Output of the SUSAN "corner" finder (t=10) given the test image.

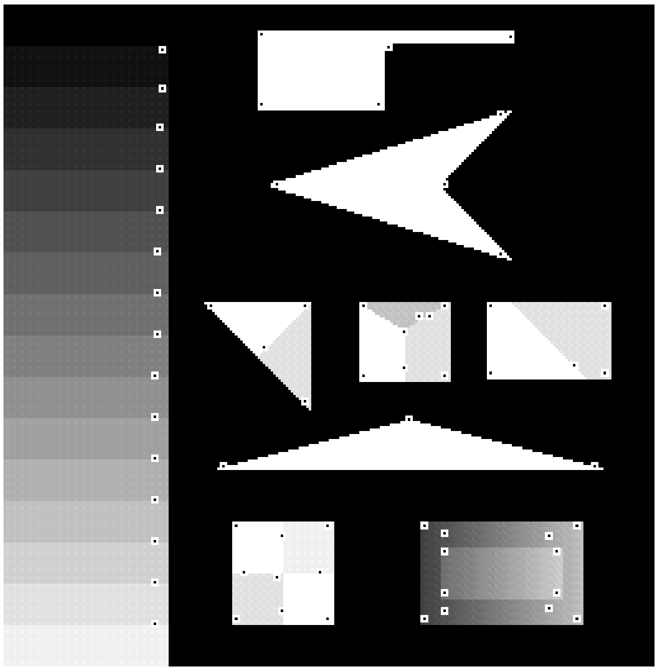

Figure 21: Output of the Plessey "corner" finder (σ=2.0) given the test image.

algorithms tested as noise in the image is increased. The test image used was the same as the noisy image previously used to test edge finders. The outputs of SUSAN and the Plessey corner finder are shown in Figures 22 and 23 respectively.

To test the SUSAN corner detector's sensitivity to both corner orientation and noise, a test described by Haralick in [24] has been used. Here 9 squares of brightness 175 greylevels and size 20 by 20 pixels are placed on a background of brightness 75 greylevels. The squares are drawn with orientations ranging from $0°$ to $80°$ in $10°$ increments. Then Gaussian noise of standard deviation 10 is added to the image. The outputs of SUSAN and the Plessey corner finder given this image are shown in Figures 24 and 25 respectively.[16] The parameters of the Plessey corner finder were adjusted to give the best results. In [24] this image is used to test the Kitchen-Rosenfeld and Dreschler-Nagel corner detectors, and the best facet model-based detector of Zuniga and Haralick. The results show that both the SUSAN and Plessey detectors perform better than the three investigated in [24], which either do not detect all of the corners, or produce a large number of false positives. It is not surprising that the Plessey detector performs well, as one of the known strengths of the Plessey algorithm is its detection reliability. Indeed, it was designed to have this particular quality. However, as already noted, the SUSAN detector finds the corners with much greater accuracy.

Two examples of the output of SUSAN given real images are shown in Figures 26 and 27.

Finally, the stability of SUSAN has also been tested. Initially this was done by running it on several (typically 10) images taken in succession from a moving video camera. The output can be viewed (in a short video created by running the 10 frames, with corners marked, forwards and backwards on the computer screen) by eye to judge how much "flicker" there is amongst the corners. The corners should be seen to travel smoothly with the scene and not appear or disappear, except at the edges of the screen or at occluding boundaries. There will always be some flicker of the corners, mostly due to the quantization of the image. The Plessey corner finder has excellent stability. This makes it appropriate for use in systems where high accuracy is not necessary. Unfortunately, the good stability is basically won at the cost of its speed and localization accuracy. The SUSAN corner finder gave good results; the stability looked similar to the Plessey algorithm, and was as good or better than the other corner finders tested.

For a more quantitative test of stability, the output from 10 consecutive frames was used as the first stage of the DROID three dimensional vision system developed by Plessey (see [25] and [10]). This program tracks corners through time in order to reconstruct a three dimensional description of the world. The results obtained when the Plessey corner finder was used were compared with those obtained when SUSAN was used. Several different sequences were tested. Some sequences gave slightly better three dimensional output data when using the SUSAN algorithm, and the rest gave similar results with both algorithms. For example, Figures 28 and 29 show scene 9 of a short sequence in which a small box (5 cm high) is placed on a fairly flat floor. DROID has been run on the 9 frames, and the output data has been used by a program designed to find the unobstructed ground plane. This program fits a least squares plane to the list of three dimensional points given by DROID, and then recomputes the plane, ignoring upper outliers, until convergence. From these pictures it can be seen that in both cases the small box was correctly identified as an obstacle, but the ground plane found when using the SUSAN algorithm did not break up as it did when using the Plessey algorithm. Note also that the driveable region created when using the Plessey corner finder goes further up the end wall than the driveable region created by using the SUSAN detector. These observations suggest that in this example the quality of the data was slightly better when using the SUSAN "corner" finder. This investigation was originally fully reported in [61], where the spatial "resolution" of DROID was found using the Plessey corner finder. Since then, as indicated above, the resolution has been improved by a factor of approximately two, by using the SUSAN "corner" detector as a front end.

In these tests, SUSAN ran on average 10 times faster than the Plessey algorithm. (This factor is quite important, as the Plessey corner detector takes about 85% of the total running time of the complete DROID program.)

Finally, as another test of its stability and general suitability for real applications, SUSAN has been used in the ASSET motion segmentation system, described in [65], with excellent results.

[16]These images are only 128 by 128 in size.

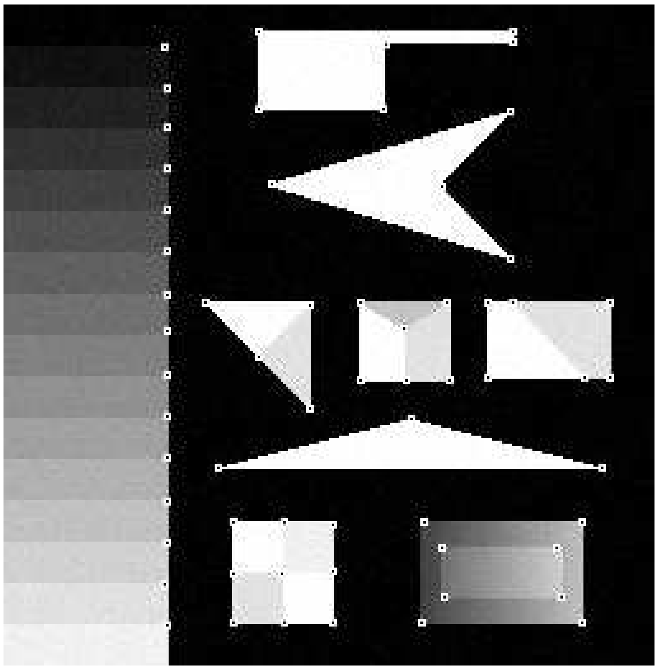

Figure 22: Output of the SUSAN "corner" finder (t=10) given the test image with Gaussian noise of minimum SNR=5 added.

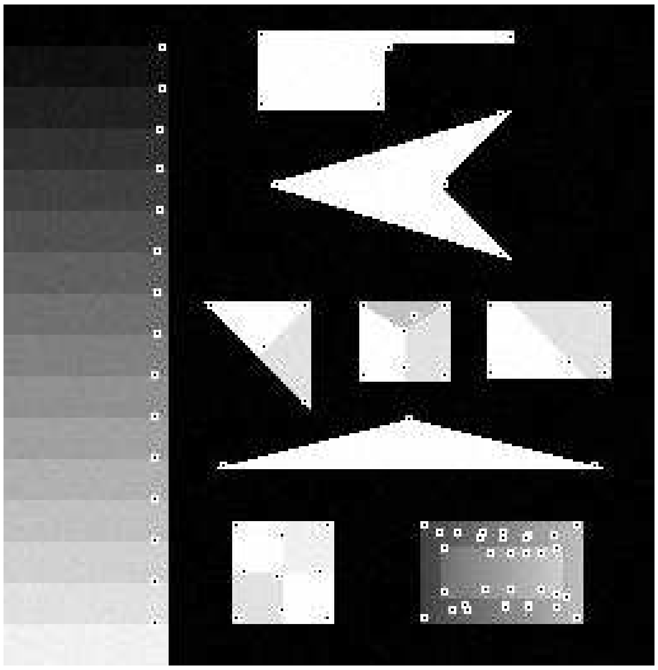

Figure 23: Output of the Plessey "corner" finder (σ=2.0) given the test image with Gaussian noise of minimum SNR=5 added.

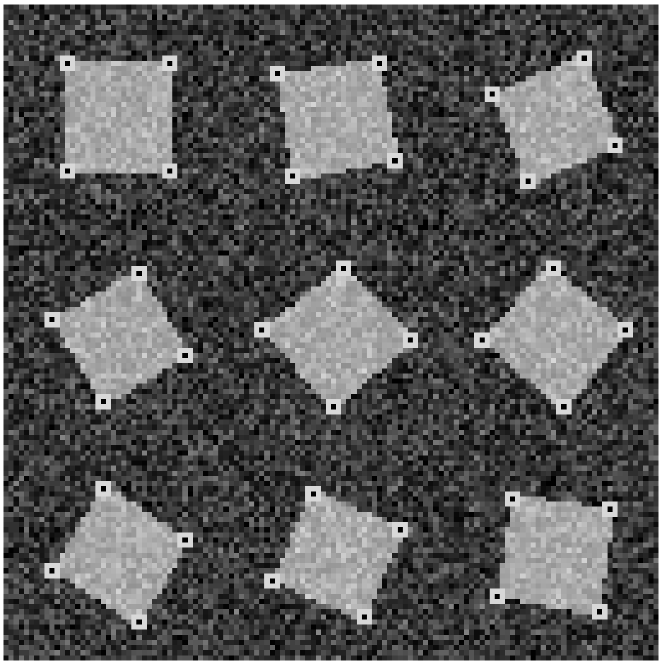

Figure 24: Output of the SUSAN “corner” finder (t=60) given the orientation test image with Gaussian noise of standard deviation 10 added.

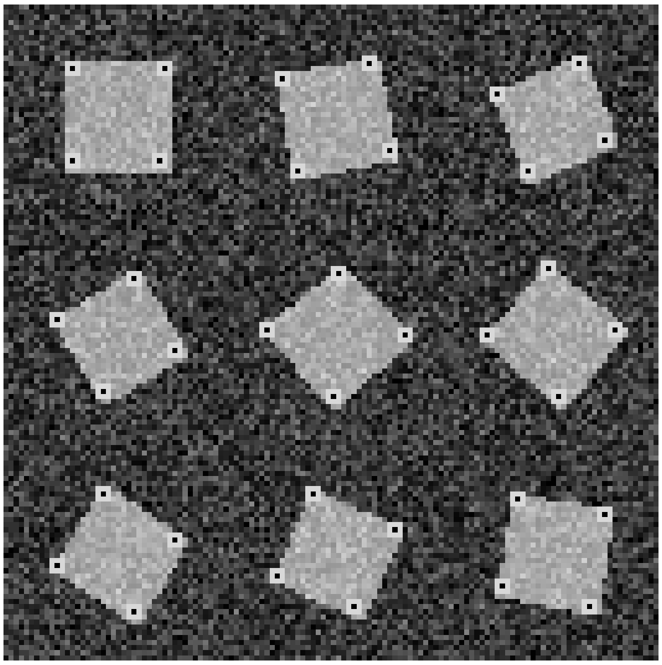

Figure 25: Output of the Plessey “corner” finder (σ=2.0) given the orientation test image with Gaussian noise of standard deviation 10 added.

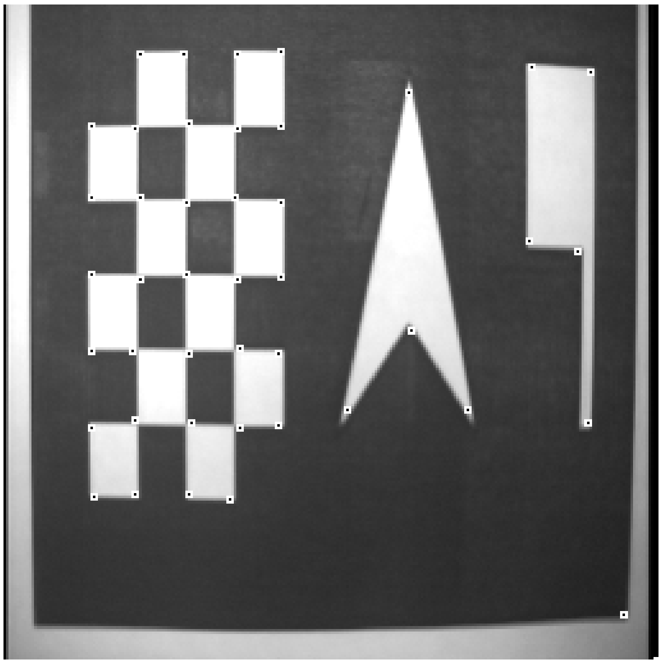

Figure 26: Output of the SUSAN "corner" finder (t=25) given a real test image. The image was created by laser printing a synthetic test image and then digitizing it using a video camera.

Figure 27: Output of the SUSAN "corner" finder (t=25) given a real image captured by video camera.

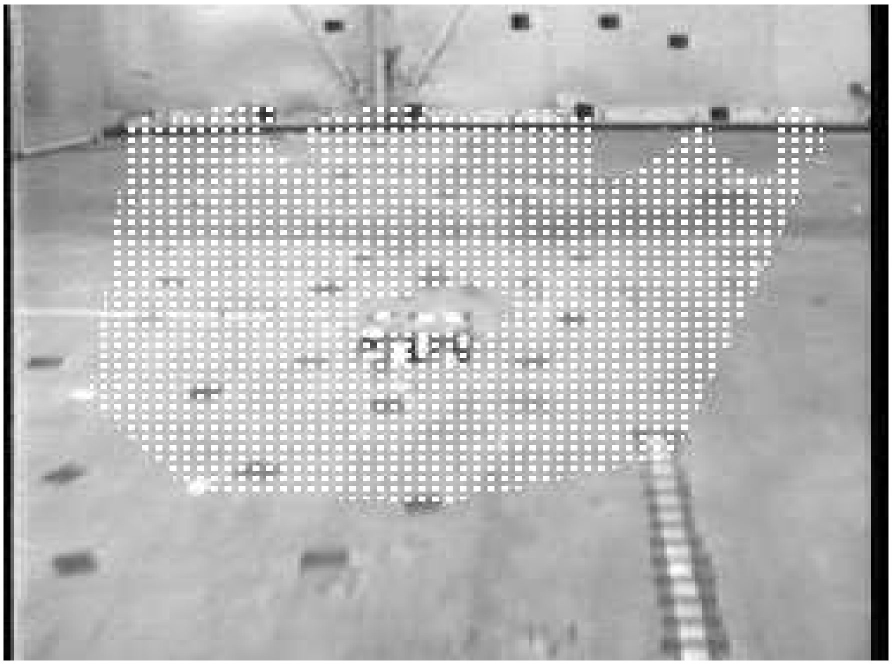

Figure 28: A least squares plane fitted to the data output from DROID using the SUSAN “corner” finder as the “front end” – the shaded area corresponds to parts of the scene which lie on (or very near) the plane.

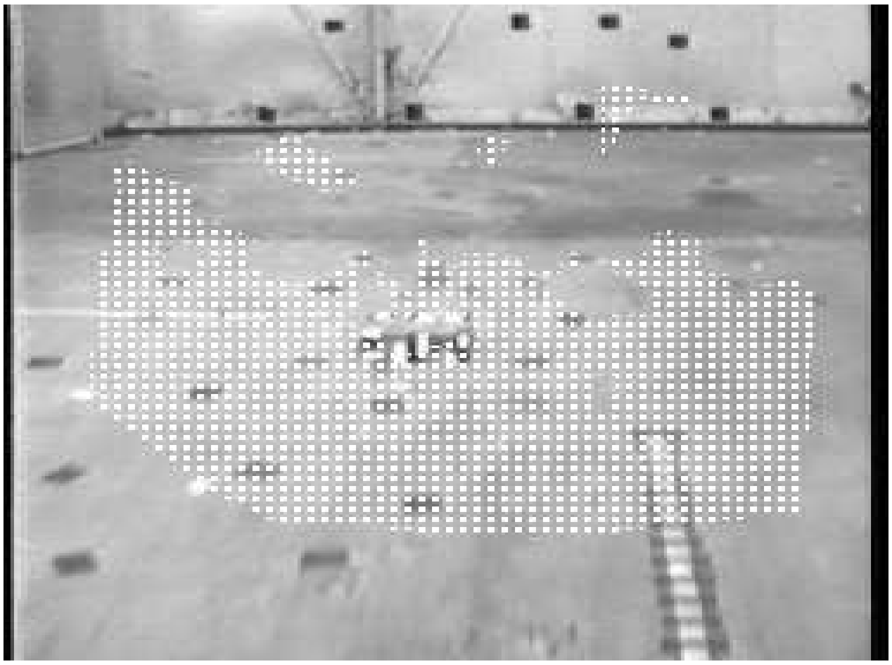

Figure 29: A least squares plane fitted to the data output from DROID using the original (Plessey) “corner” finder as the “front end”.

5.5 Conclusions

This section has described application of the SUSAN principle to the detection of two dimensional features accurately, stably and efficiently. The localization of the features is independent of the mask size used, and noise suppression is shown to be good. Complex junctions are detected correctly by the "corner" finder.

6 SUSAN Structure Preserving Noise Reduction

A brief introduction to the problem of image noise and existing attempts to remove it is now given. (For a longer review, see [63].) This is followed by details of the SUSAN noise reduction algorithm. Finally, results of testing the SUSAN algorithm are presented and discussed.

6.1 Introduction and Review

"Noise" can be introduced into a digitized image in many ways, starting with the lens of the imaging hardware[17] and ending at the digitization of the captured image.[18] The reduction of noise without degradation of the underlying image has attracted much attention in the past. However, whilst many "structure preserving" filters have achieved some degree of success at preserving one dimensional image structure, very few have successfully preserved two dimensional image brightness structure, such as corners and junctions.

This section describes the SUSAN noise filtering algorithm.[19] This uses local measurements to obtain noise reduction whilst preserving both one and two dimensional image structure. The method is based on image region determination using a non-rigid (in fact, spatially undetermined) region model.

The simplest noise reduction method is the *sliding mean* or *box filter* (see [35]). Each pixel value is replaced by the mean of its local neighbours. The *Gaussian filter* is similar to the box filter, except that the values of the neighbouring pixels are given different weighting, that being defined by a spatial Gaussian distribution. The Gaussian filter is probably the most widely used noise reducing filter. It has been incorporated into the design of many algorithms such as the Canny edge detector (see [9]). The Gaussian filter is pleasing because Fourier analysis shows that the Gaussian spatial distribution gives a Gaussian frequency distribution, leading to a good (often described as "optimal") tradeoff between localization in the spatial and in the frequency domains. The sharp cutoff of the box filter causes it to give a noisier output than the Gaussian (see the discussion in [9]) corresponding to a less well localized frequency response.

More complicated "optimal" statistical estimators have been derived which use local means and autocorrelation functions to calculate the underlying signal (for example *Wiener filtering* – see [49]). However, these methods are usually very slow.

Linear filters such as the box and the Gaussian filters tend to create more blurring of image details than do most non-linear filters. An increase in their ability to smooth noise corresponds to an increase in the blurring effect.

Many non-linear filters fall into the category of *order statistic neighbour operators*. This means that the local neighbours are sorted into ascending order (according to their value) and this list is processed to give an estimate of the underlying image brightness. The simplest order statistic operator is the *median* (see [69]), where the central value in the ordered list is used for the new value of the brightness. The median filter is much better at preserving straight edge structure than Gaussian smoothing, but if the edge is curved then image degradation occurs. At corners, other two dimensional features and thin lines the median does not perform well with regard to structure preservation. The median is good at reducing impulse noise (for example "salt and pepper" noise) where the noisy pixels contain no information about their original values.

There have been several variations on the median filter, for example the *weighted median filter* (see [7]) selectively gives the neighbouring pixels multiple entries to the ordered list, usually with the centre pixels

[17] Sometimes even some structure in the imaged world is treated as unwanted "noise".

[18] This is assuming digital storage and not analog.

[19] The acronym is carried over from the closely related SUSAN feature detection algorithms. For naming accuracy, the acronym could now read Smoothing over Univalue Segment Assimilating Nucleus.

of the neighbourhood contributing more entries. The higher the weighting given to the central pixel, the better the filter is at preserving corners, but the less the smoothing effect is. At corners, other pixels within the convex image region including the centre pixel are not given any higher weighting than those outside, so this is an ineffective way of preserving the corner structure. Other methods include the *contrast-dependent thresholded median* (see [55]), *adaptive decimated median filtering* (see [1]) and the *midrange estimator* (see [2]), which places the brightness estimate halfway between the maximum and minimum local brightness values.

Three similar filters, the first two of which are typical of more complicated order statistic operators, are the *K-nearest neighbour operator* (see [13]), the *α-trimmed mean* (see [5]) and the *sigma filter* (see [30]). The K-nearest neighbour operator takes the mean of the K nearest neighbours from within the ordered list. The value which K is usually set to (5 or 6 when a 3 by 3 mask is used) means that corners and thin lines are badly corrupted. The trimmed mean operator takes the mean of the remaining entries in the ordered list once the first and last αN entries in the list (originally containing N entries) have been thrown away. The sigma filter takes an average of only those neighbouring pixels whose values lie within 2σ of the central pixel value, where σ is found once for the whole image. This attempts to average a pixel with only those neighbours which have values "close" to it, compared with the image noise standard deviation.

The *peak noise filter* is specifically designed to remove impulse or peak noise. A pixel's value is compared with the local mean. If it is close then the value remains unchanged. If it is very different then the pixel's value is replaced by the local mean. A more advanced version of this takes a weighted mean of the central value and the local mean, the weighting depending on the difference between these two quantities and also on the local brightness variance. This is similar to the contrast-dependent thresholded median filter, and again, is not very good at structure preservation.

In [47] Perona and Malik use local image gradient to control *anisotropic diffusion*; smoothing is prevented from crossing edges. This is, in a sense, an opposite approach to the K-nearest neighbour, sigma and SUSAN filters, which work out which neighbours to include in smoothing; anisotropic diffusion works out which neighbours to exclude. In [54] Saint-Marc *et al.* "refine and generalize" the approach of Perona and Malik, using many iterations of a 3 by 3 mask which weights a mean of the neighbours' values according to the edge gradient found at each neighbours' position. This method shall be referred to as the SMCM filter. Because pixels both sides of an edge have high gradient associated with them, thin lines and corners are degraded by this process. This problem is also true with Perona and Malik's method. *Biased Anisotropic Diffusion* (see [44]) uses physical diffusion models to inhibit smoothing across boundaries, according to a cost functional which attempts to minimize a combination of desired components. These individual costs are image smoothness and quality of fit between original and smoothed image. This type of approach is similar to the *weak membrane* model (see [6]) which fits a membrane to the image surface, allowing it to "break" at significant discontinuities, incurring a cost, which is balanced against the "badness of fit" cost. This method tends to be very computationally expensive.

Specific surface model fitting techniques have been developed; for example, in [59] a two stage surface fitting method is used. In the first stage a moving least median of squares of errors of a fit to planar facets is used. This is similar to a least mean squares fit to a plane, but uses the median of the errors instead of the mean so that discontinuities are correctly fitted. However, corners are rounded, as with the standard median smoothing; this is expected, and can be clearly seen in the results presented. The second stage fits a weighted bicubic spline to the output of the first stage. As with this approach in general, the success is limited by the model used; image structure not conforming to the model will not be well preserved.

Morphological noise suppression, using mathematical morphology, is related to order statistic neighbour operators. In [43] this type of operator is discussed and compared with other filters. Noble reports that morphological filters perform worse than the moving average and median filters for Gaussian type noise and slightly better than these for "salt and pepper" noise.

The *hysteresis smoothing* approach (see [17]) follows positive and negative gradients in the image, only allowing changes in the gradient when the change is significant, i.e., the new gradient continues further on in the image. The results of using this approach, tested in [24], were not very encouraging, showing a *decrease* in the signal to noise ratio with each of four different types of noise.

The *selected-neighbourhood averaging* (e.g., see [40]) method chooses a local neighbourhood to smooth over based on a variety of criteria. Each pixel can be part of several different "windows"; the one chosen to smooth over is the most homogeneous window which contains the pixel in question as part of its homogeneity. Various window shapes have been tried. These methods are limited to succeeding only on those parts of the image which obey the model used. Sharp corners, for example, will not be correctly dealt with by a square window. A variation on these algorithms is the use of a thin window which is perpendicular to the direction of maximum gradient. Clearly the structure preserving success of this is limited to relatively straight edges.

The *symmetric nearest neighbours* (see [27]) approach takes each symmetric pair of neighbouring pixels and uses the closer value to the centre pixel from each pair to form a mean output value. Again this is not well suited to preserving corners.

In [31] three methods of noise reduction using different local image measurements to avoid destroying image structure are presented, all based on the assumption that image regions bounded by edges have *constant* value. In the first method, image edges are found, and for each image position the local edge structure is matched to a list of binary templates which attempt to cover all possible straight edge configurations. Each template has a corresponding weight matrix which determines which elements of the local neighbourhood should be used for smoothing. The system attempts to cope with junctions by taking a convex combination of the different matrix outputs, each weighted according to the quality of the match between the actual local edge configuration and the template. This process is applied iteratively. Although edge structures are well preserved, corners are rounded and ramps are flattened. In the second method the local image gradient is calculated in several different directions, and the results of this are combined to give a smoothing matrix which attempts to smooth only over the bounded constant image region which includes the central pixel. Again, this is applied iteratively. The second method suffers from the same problems as the first method. The third method uses *probabilistic relaxation* where each pixel has a probability distribution amongst the entire set of possible values, thus vastly increasing the data set. The initial distribution is assumed to be Gaussian. Relaxation is then used to allow the pixels' distributions to interact to find a stable best estimate of the underlying signal. The results given suggest that this method does not in general work as well as the first two.

The *gradient inverse weighted operator* (see [73]) forms a weighted mean of the local pixels, with the weights depending on the difference between the central pixel's value and the value of the local pixels;

$$J(x,y) = \frac{\sum \frac{I(x+i,y+j)}{\max\,\{\frac{1}{2},|I(x+i,y+j)-I(x,y)|\}}}{\sum \frac{1}{\max\,\{\frac{1}{2},|I(x+i,y+j)-I(x,y)|\}}}, \tag{34}$$

where $I(x,y)$ is the original image, $J(x,y)$ is the smoothed image, and the summation is taken over a 3 by 3 square neighbourhood. The filter is applied iteratively, typically five times. No preset threshold is needed. This method reduces Gaussian noise well whilst preserving image structure. Impulse noise would be much better dealt with if the central pixel were excluded from the two sums in Equation 34, for obvious reasons, but this was not included in the method.

Finally, image averaging using many images of the same scene (taken quickly one after another) has been used to reduce image noise. For example, see [57]. However, for many image processing applications this approach is impractical; the camera may be moving, the scene may be changing, and it may be necessary to use every image in its own right.

6.2 The SUSAN Noise Filtering Algorithm

The SUSAN noise filtering algorithm, like some of the existing filtering techniques, preserves image structure by only smoothing over those neighbours which form part of the "same region" as the central pixel. Regions are commonly assumed to be constant in value; here they are only assumed to be *roughly* constant; sloping regions are correctly dealt with, as will be seen. The SUSAN filter works by taking an average over all of the pixels in the locality which lie in the USAN. It is obvious that this will give the maximal number of suitable neighbours with which to take an average, whilst not involving any neighbours from unrelated regions. Thus all image structure should be preserved.

The SUSAN filter is most like the gradient inverse weighted method, but instead of using the inverse gradient for each neighbour's weighting (see Equation 34), the weighting is derived from the USAN weighting, shown in Equation 4. Whereas the gradient inverse weighted method uses no "brightness" threshold at all, the brightness similarity equation (4) is strongly threshold dependent. This has the advantages that all neighbours within the USAN are given nearly equal weighting, and that neighbours outside the USAN are given practically zero weighting. Also, the smoothing process does not suffer when the USAN is sloping, as some variation in the USAN is allowed before the smoothing weight is significantly reduced. However, the satisfying sharp cutoff comes at the expense of needing to determine a brightness threshold. Although it is useful to be able to determine a limit to the strength of image noise (compared with the more usual determination of a lower limit to the frequency of the noise) a slightly more smoothed form for the weighting equation is used;

$$c(\vec{r},\vec{r_0}) = e^{-(\frac{I(\vec{r})-I(\vec{r_0})}{t})^2}. \tag{35}$$

Thus a Gaussian in the *brightness* domain is used for smoothing. This means that the SUSAN filter is to the sigma filter (in the brightness and the spatial domains) what the Gaussian filter is to the box filter (in the spatial domain). The dependence of the SUSAN filter on the parameter t is not critical; it is usually not varied, but it is trivial to adjust t if "extreme" images are to be correctly dealt with.

Apart from the different weighting function, the other major difference between the SUSAN filter and the gradient inverse weighted method is that the sums taken over the local neighbourhood do not include the centre pixel itself. As mentioned earlier, this allows much better reduction of impulse noise than otherwise. If the resulting denominator (i.e. the USAN area) is zero (as it can be for impulse noise, due to quantization of the brightness comparison function) then the median of the eight closest neighbours is used to estimate the pixel's correct value.

Thus the complete equation used for the SUSAN filter is;

$$J(x,y) = \frac{\sum_{(i,j)\neq(0,0)} I(x+i,y+j)e^{-\frac{r^2}{2\sigma^2}-\frac{(I(x+i,y+j)-I(x,y))^2}{t^2}}}{\sum_{(i,j)\neq(0,0)} e^{-\frac{r^2}{2\sigma^2}-\frac{(I(x+i,y+j)-I(x,y))^2}{t^2}}}, \tag{36}$$

where $r=\sqrt{i^2+j^2}$, σ controls the scale of the spatial smoothing, t is the brightness threshold (which controls the scale of the "brightness smoothing") and the above mentioned rule is applied when the denominator is zero. Note that this looks quite like the equation used for Gaussian smoothing, except that the Gaussian function is extended into the brightness domain. **The filter clearly integrates the best aspects of the best of the existing noise reducing filters, including edge preserving filters,** described in the previous section, and as a consequence gives very good results, as shown below. (This integration is apparent in the *theory*; the novel SUSAN filter is derived from "first principles", however, and not by any attempt to actually *integrate* previous filters.)

With respect to "deblurring" image edges, it is clear that at a blurred edge, pixels will be pulled towards the neighbouring region to which they are closest in value. Thus, far from destroying image structure, the SUSAN filter can actually improve the image quality.

The SUSAN filter, as pointed out earlier, appears to be very similar to the approach of Saint-Marc *et al.* . In fact, the weighting function used in the SMCM filter has the same form as the brightness comparison part of the SUSAN filter;

$$c(\vec{r},\vec{r_0}) = e^{-\frac{d(\vec{r})^2}{2k^2}}, \tag{37}$$

where $d(\vec{r})$ is the image gradient at $\vec{r}$ and k is a threshold. The very important difference between this and SUSAN is that the numerator inside the exponential is the gradient, found at $\vec{r}$, rather than the difference in brightness between the centre pixel and the pixel at $\vec{r}$, as is the case with SUSAN. The result of this approach with the SMCM filter is that fine structure cannot be correctly dealt with. For example, thin lines (particularly of two pixels width) and sharp corners are badly degraded, as pixels on the edge of regions are wrongly excluded from calculations within those regions. As stated earlier, the same problems exist for the method of Perona and Malik.

6.3 Results

This section contains results of quantitative and qualitative comparisons of the SUSAN noise filter with six other noise reduction methods (including the best existing "structure preserving" filters), as well as some qualitative multi-scale results.

6.3.1 Quantitative Tests of the SUSAN Filter

Two quantitative tests of the structure preserving qualities of seven noise reduction methods (including the SUSAN filter) are now presented. In the first test, noise was added to an image of a straight step edge, and each method was applied to give the same reduction in noise standard deviation. The amount which the edge was blurred was measured for each method. In the second test, noise was added to image corners, and again each method was applied to give the same reduction in noise standard deviation. The amount by which the corners (and their forming edges) were degraded was measured for each method. Both tests were performed with three different noise types at two different noise levels.

The Edge Test

The first test was performed on an image of an ideal step edge with upper and lower brightness values of 150 and 100.[20] Noise was then added to the image and each noise filtering method was applied to give a fixed reduction in the noise standard deviation, so that the comparisons of edge degradation would be performed after each method had smoothed the noisy image by the same amount. The amount that the edge was blurred was quantified by measuring the difference in brightness between neighbouring pixels which were originally across the step edge; this difference was averaged over 150 pixels along the middle of the edge. Therefore a final "edge height" of 50.0 is a perfect score; the lower the "height", the more degraded the edge.

The Corner Test

The second test was performed on an image of 15 squares of size 17 by 17 pixels and brightness 150 on a background of brightness 100. Again noise was added to the image and each filtering method was applied to give a fixed reduction in the noise standard deviation. The degradation of the corners was quantified by taking the sum (over every pixel in all of the squares) of the absolute difference between the estimated brightness and the correct original brightness. Thus a "corner error" of zero is a perfect score; the higher the error, the more degraded the corners.

The Noise Types Added

The three noise types used to corrupt the images were additive Gaussian noise, additive uniform noise and impulse, or "salt and pepper" noise. The "salt and pepper" noise was created by choosing a fraction of the image to corrupt (at each pixel a random number in the range 0 to 1 was compared with this fraction to determine whether or not to corrupt the pixel) and then setting the pixels to be corrupted to random brightnesses in the range 0 to 255. Two noise levels were used for each noise type; a standard deviation of 15.0 (corresponding to an enormous amount of image noise, almost never found in "real" images) and a standard deviation of 2.5 (a standard deviation which is is still more than found in most "real" images). These standard deviations were reduced by the filters to standard deviations of 2.0 and 0.5 respectively, if possible. For each of the twelve combinations of original image, noise type and noise amount, only one image was generated, and used to test each of the filtering methods.

The Algorithms Tested

[20] All brightness values mentioned are within an image brightness scale of 0 to 255.

In [11] Chin and Yeh perform testing of the mean, K-nearest neighbour method, one of Lev's approaches (see [31]), the gradient inverse weighted method, selective neighbourhood weighting, Haralick's facet approach and median filtering. The scope of the testing was somewhat limited; no corner/junction degradation was investigated and only Gaussian noise was considered. The best "all around results" were those obtained by using the median and the K-nearest neighbour methods.

The noise reduction methods chosen for testing here were Gaussian filtering (mainly for the sake of reference), median filtering, midrange filtering, the gradient inverse weighted method (GIW), the K-nearest neighbour method, the SMCM method and the SUSAN filter. The Gaussian filter has variable σ, so that only one iteration is necessary to obtain the desired smoothing effect. The SUSAN filter has variable σ (spatial smoothing) and also variable brightness threshold, which is normally left at 12 greyscale values. The filter can be iterated to achieve increased smoothing. The K-nearest neighbour method has K set to 6, as this is the value found to be optimal in [13]. It is iterated to increase smoothing. The SMCM has a brightness parameter to set. As there is no mention in [54] of the optimal value for this parameter k the best value was chosen for each individual experiment. The method is iterated to increase smoothing. The other three filters have fixed mask sizes and are therefore iterated to achieve the desired smoothing effects. Where consecutive iterations give output standard deviations which lie either side of the target standard deviation, linear interpolation was applied to find the structure degradation measurement. In the case of the corner image corrupted with "salt and pepper noise" the median filter achieved near zero output variance after just one application, so a mask size of three by three pixels was used as well as the default (five by five pixels) to reduce the rounding of the corners, and so present the filter in the best possible light.

Test Results

The results of the twelve individual tests are shown in Tables 1–6. These results clearly show the SUSAN filter to be the one which degrades image structure the least. The following points should be noted:

- The Gaussian filter performed poorly throughout the tests. However, its performance was not very much worse with the large amount of noise than with the smaller amount.
- The median filter performed quite well with the lower amount of noise and quite badly with the higher amount, with the exception of the "salt and pepper" noise, on which it performed very well in both cases. However, the median filter does not preserve corners, particularly when compared with the SUSAN filter, a fact which is reflected clearly in the results.
- The midrange filter performed consistently badly. In two cases, it did not succeed in reducing the standard deviation to the target level.
- The gradient inverse weighted method performed quite well except on the "salt and pepper" noise, which it performed badly on, as expected. It never achieved as good results as the SUSAN filter.
- The K-nearest neighbour method did not perform particularly well.
- The SMCM method did not perform as well as SUSAN in the presence of normal amounts of noise, and performed similarly to SUSAN in the presence of large amounts of noise. It rounded corners, and failed completely to reduce either small or large amounts of "salt and pepper" noise.
- The SUSAN filter performed consistently well. It achieved the best result in nearly all cases. In the tests reported in the first two tables the SUSAN filter did not degrade the edge and the corners at all when it was iterated. (It is normally expected that image structure degradation will increase as more smoothing is performed.) Instead, the filter continued to sharpen the structures, resulting in a final standard deviation of 0.00 (not just the target standard deviation of 0.5) whilst still giving "perfect" scores in the edge heights and corner errors.

Filter Method	Edge Height	Comments	Corner Error	Comments
Gaussian	12.31	σ=1.6	22652	σ=1.7
Median (5x5)	44.90	2 iters	10734	3 iters
Midrange (3x3)	12.27	10 iters	30745	7 iters
GIW	46.65	8 iters	3362	7 iters
KNN	26.02	9 iters	21138	11 iters
SMCM	49.00	5 iters, k=30	2130	4 iters, k=20
SUSAN	50.00	σ=7.1	0	σ=7.1, variance reduced to zero after 2 iters

Table 1: The results of testing the noise filters when Gaussian noise of standard deviation 2.5 is added.

Filter Method	Edge Height	Comments	Corner Error	Comments
Gaussian	12.67	σ=1.53	26626	σ=1.95
Median (5x5)	44.69	2 iters	8666	2 iters
Midrange (3x3)	22.07	4 iters	21811	3 iters
GIW	46.75	8 iters	5341	10 iters
KNN	25.38	11 iters	29506	16 iters
SMCM	48.62	4 iters, k=40	2004	5 iters, k=20
SUSAN	50.00	σ=7.1	0	σ=7.1, variance reduced to zero after 2 iters

Table 2: The results of testing the noise filters when uniform noise of standard deviation 2.5 is added.

Filter Method	Edge Height	Comments	Corner Error	Comments
Gaussian	12.00	σ=1.50	13785	σ=1.17
Median (5x5)	50.00	1 iter	3750	1 iter
(3x3)			750	1 iter
Midrange (3x3)	failed	variance increasing after 100 iters	failed	variance increasing after 100 iters
GIW	19.93	67 iters	failed	variance increasing after 100 iters
KNN	33.00	1 iter	4527	1 iter
SMCM	failed	variance increasing after 100 iters	failed	variance increasing after 100 iters
SUSAN	50.00	σ=1.4	150	σ=1.4

Table 3: The results of testing the noise filters when “salt and pepper” noise of standard deviation 2.5 is added.

Filter Method	Edge Height	Comments	Corner Error	Comments
Gaussian	10.09	σ=2.0	30498	σ=2.1
Median (5x5)	20.19	4 iters	29471	5 iters
Midrange (3x3)	8.85	14 iters	38910	10 iters
GIW	33.88	26 iters	15596	29 iters
KNN	23.47	16 iters	23882	17 iters
SMCM	50.00	10 iters, k=60	9804	17 iters, k=45
SUSAN	46.59	σ=1.4, 6 iters	8930	σ=1.4, 9 iters

Table 4: The results of testing the noise filters when Gaussian noise of standard deviation 15.0 is added.

Filter Method	Edge Height	Comments	Corner Error	Comments
Gaussian	9.73	σ=2.0	36981	σ=2.5
Median (5x5)	17.53	11 iters	39012	9 iters
Midrange (3x3)	19.86	4 iters	26481	5 iters
GIW	36.15	36 iters	17838	37 iters
KNN	21.91	25 iters	38093	31 iters
SMCM	49.13	13 iters, k=35	9710	15 iters, k=45
SUSAN	46.02	σ=1.4, 10 iters	8852	σ=1.4, 10 iters

Table 5: The results of testing the noise filters when uniform noise of standard deviation 15.0 is added.

Filter Method	Edge Height	Comments	Corner Error	Comments
Gaussian	9.42	σ=2.00	27814	σ=2.00
Median (5x5)	50.00	1 iter	3281	1 iter
(3x3)			738	1 iter
Midrange (3x3)	0.54	82 iters	71694	74 iters
GIW	38.69	36 iters	13137	34 iters
KNN	32.26	1 iter	4970	1 iter
SMCM	failed	variance increasing after 100 iters	failed	variance increasing after 100 iters
SUSAN	49.40	σ=0.35	730	σ=0.35

Table 6: The results of testing the noise filters when "salt and pepper" noise of standard deviation 15.0 is added.

6.3.2 Qualitative Tests of the SUSAN Filter

Some visual results of testing the SUSAN filter are now presented. Comparisons of the results are made with those obtained by using the other filters tested quantitatively.

The first results presented show the effect of attempting to reduce Gaussian noise added to the two dimensional test image introduced in Section 4.4. The noise was reduced by the filters from having a standard deviation of 3.0 to a standard deviation of 1.0. The noisy input image is shown in Figure 30. The outputs of the filters are shown in Figures 31–37. These figures are fairly self explanatory, with the weaknesses of the Gaussian, median and midrange filters clearly shown, and the good structure preserving qualities of the SUSAN filter evident. The GIW and SUSAN filters did not noticeably degrade the image structure when the standard deviation was reduced to 1.0, so they were re-applied. SUSAN was re-applied until the noise was barely noticeable, and GIW was re-applied until image degradation became apparent. In Figure 30 three sections of interest are marked; these areas of the image are shown as surfaces in Figures 38, 39 and 40, where the noisy input image and the outputs of the the seven filters are plotted. The noise level visible in the input image surface appears to vary from figure to figure; this is because of variation in the contrast of the local image function. Figure 38 shows the end of a sharp corner of high contrast, Figure 39 shows a "T-junction" in the brightness function and Figure 40 shows a corner superimposed on a brightness ramp. The KNN and SMCM filters show noticeable image degradation for the given amount of noise reduction (see particularly Figure 38). The SUSAN filter is shown to have the best structure preserving qualities of the seven filters. In the case of the third section of interest, the SUSAN filter is shown to work correctly even when the regions making up the local image structure are far from horizontal, a difficult situation for many of the noise reducing filters based on strict image modelling. Note a small error in the estimation of the lower edge of the raised corner; at this edge the brightness curves slightly, rather than continuing in a tilted plane. This is due to the lack of lower neighbours to balance the upward pull of the higher neighbours; the pixels in the lower region are too low (as one would hope) to create this balance. Thus a "second order" inaccuracy in the surface estimation occurs.

Finally, in Figures 41 and 42, results are shown of testing the seven filters on parts of more complicated images. In example 1, part of the Mandelbrot set is degraded with Gaussian noise of standard deviation 10.0. In example 2, a shield from a photograph of a building is degraded with uniform noise of standard deviation 10.0. In example 3, part of a real image of some printed letters (shown in full, with edges superimposed, in Figure 15) is degraded with "salt and pepper" noise, as described earlier, with a "corruption fraction" of 0.05. The three images were processed by Gaussian filtering (using $\sigma = 1.0$), median filtering (using a three by three mask to give the best image preservation possible), midrange filtering, the GIW filter, K-nearest neighbour filtering, the SMCM method (with k set to just 1 to give the least blurring possible) and the SUSAN filter. The input images, the degraded images and the outputs of the filters are shown in Figure 41 and 42. Again, the SUSAN filter appears to give the best overall results.

6.3.3 Brief Multi-scale Analysis

Finally, the SUSAN algorithm and the six other methods mentioned above were looked at with respect to their multi-scale properties. As a filter's main smoothing parameter is varied from no smoothing to a large amount of smoothing, the number of image features (e.g., edges) should gradually be reduced. Ideally, before a feature's disappearance, it should not shift its position at all. (See [75] for an excellent introduction to scale-space filtering.) However, scale-space analysis of most noise filters and feature detectors shows different feature positions at different scales.

To produce the results shown in Figure 43, the seven noise filters were run on a 1D test image taken from a single line of a real image. The filters were either run at a variety of scales, or iterated, to give increased smoothing of the original image; each horizontal line in each final image represents a filter's output at a particular scale/iteration number, with least filtering at the bottom of each image. To give added clarity to the results an edge finder (the SUSAN edge finder, running with a 3 by 3 mask, with brightness threshold set to 5) is run on each output image, showing how image features evolve over different scales.

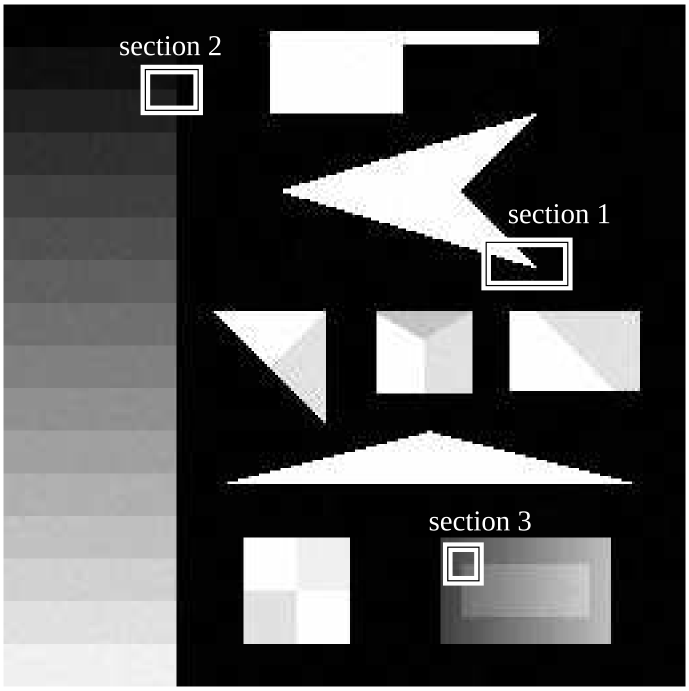

Figure 30: The noisy image used to test noise reduction filters with three particular sections of interest marked.

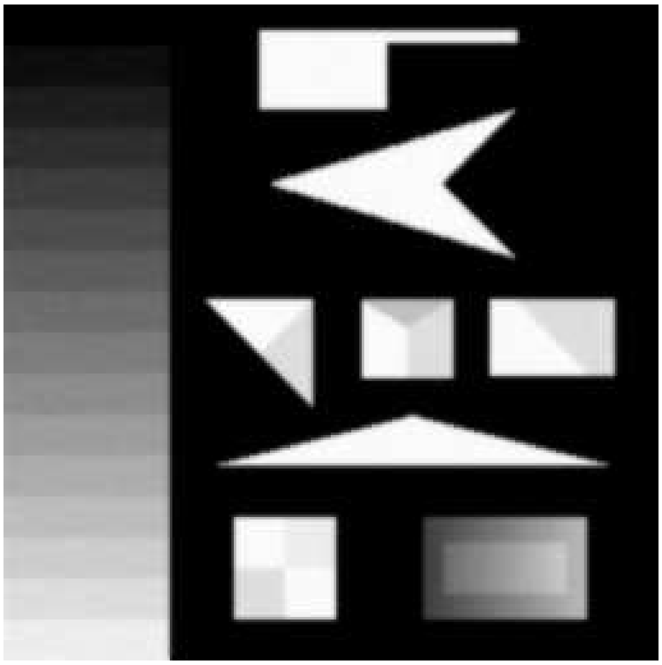

Figure 31: The result of applying Gaussian filtering with $\sigma = 1.0$ to the noisy image.

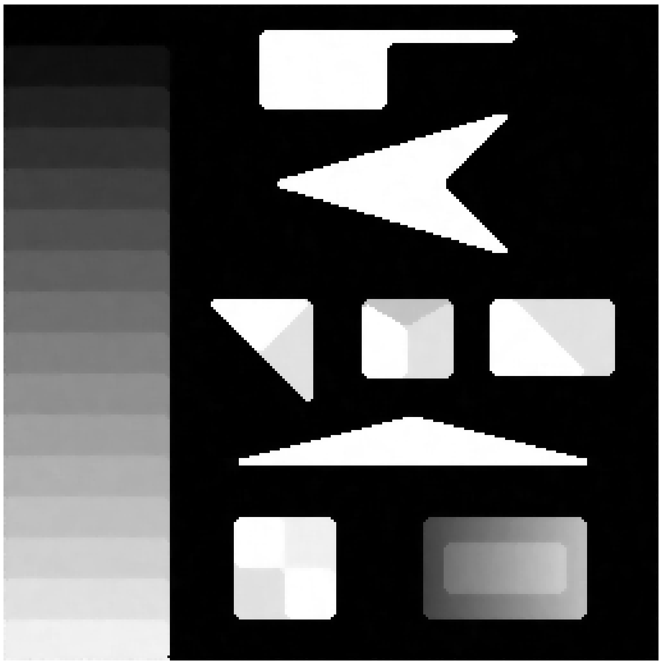

Figure 32: The result of applying median filtering to the noisy image.

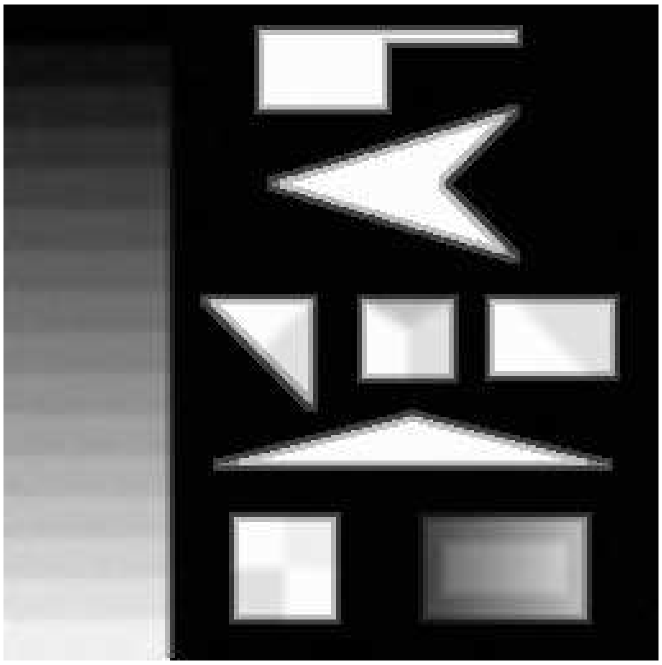

Figure 33: The result of applying midrange filtering to the noisy image.

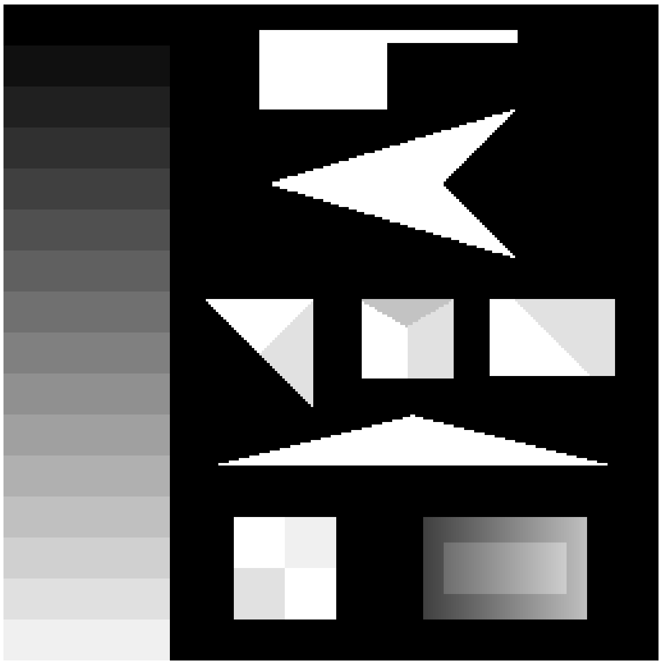

Figure 34: The result of applying GIW filtering to the noisy image.

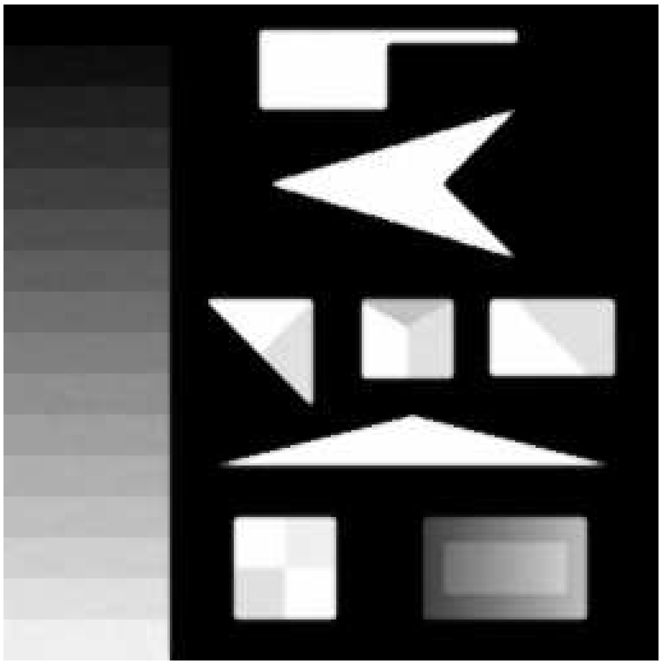

Figure 35: The result of applying K-nearest neighbour filtering to the noisy image.

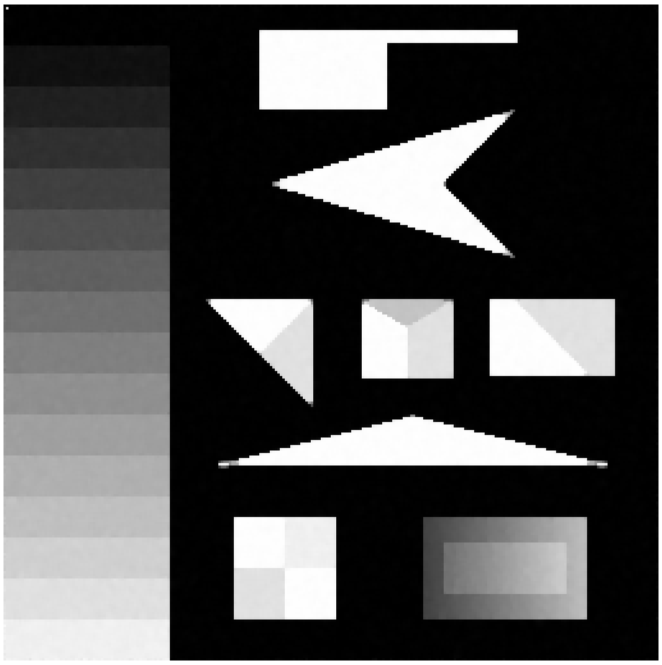

Figure 36: The result of applying SMCM filtering to the noisy image.

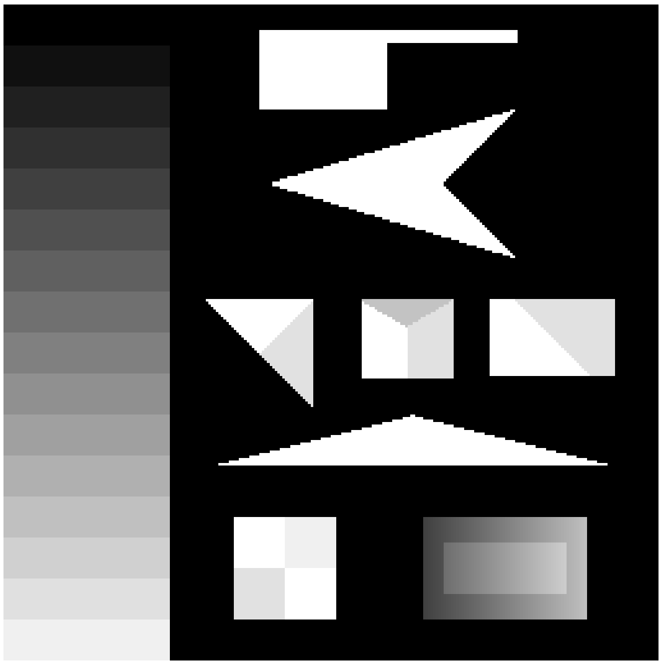

Figure 37: The result of applying SUSAN filtering to the noisy image.

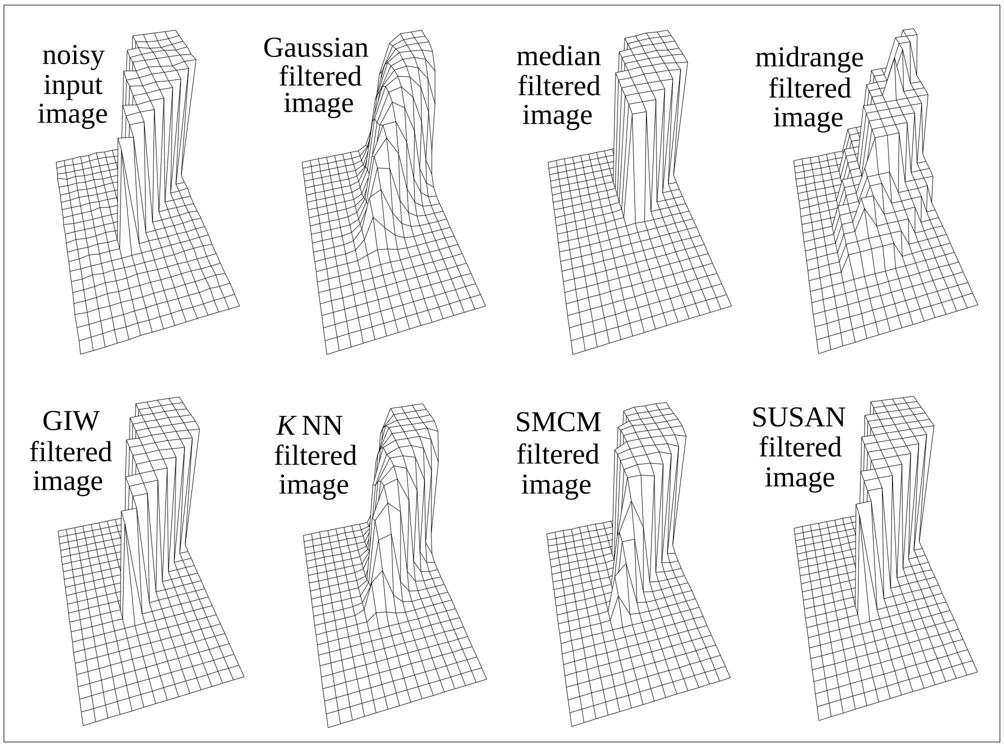

Figure 38: The outputs of the seven filters shown as surfaces for section 1 of the input image.

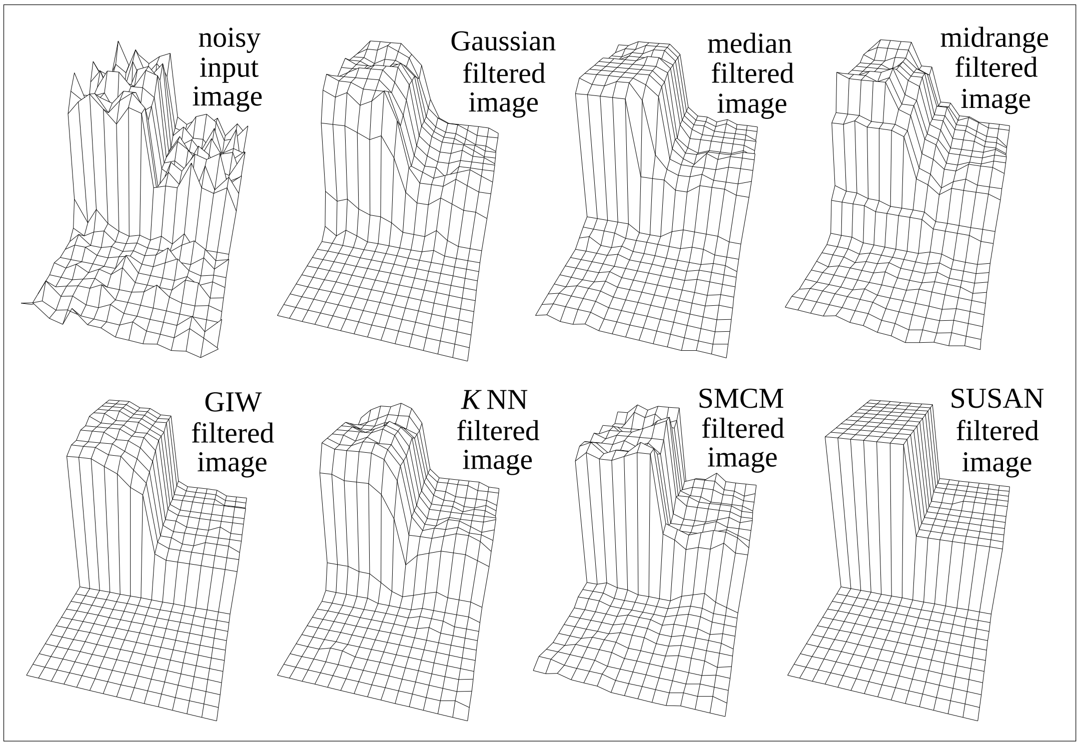

Figure 39: The outputs of the seven filters shown as surfaces for section 2 of the input image.

The median and KNN filters quickly converged so that they were not suitable for scale-space analysis. The other five filters' outputs are shown. For Gaussian filtering, the scale of the smoothing was varied to achieve different scales. In the other four cases iteration was used. The SMCM filter had $k = 20$ and SUSAN had $t = 20$ and $\sigma = 2.1$. Also, the complete Canny edge detector [9] was applied at different smoothing scales.

The results show that Canny (which includes Gaussian filtering), Gaussian filtering and midrange filtering have noticeable scale-space feature drift and that GIW, SMCM and SUSAN filtering have very little scale-space feature drift. Close inspection reveals the least drift with the SUSAN filter.

6.4 Conclusions

In this section a new structure preserving noise filter, closely related to the SUSAN feature detectors, has been described. Without needing to formulate a rigid model of the image, only those sections of the local image structure which are part of the "same region" as each pixel are used to form an estimate of that pixel's original brightness. Quantitative and qualitative results show the SUSAN noise filter to be better at reducing noise (whilst minimizing degradation of the underlying image) than other filters tested.

7 Conclusions

This paper has described a new principle which allows image edges, lines, corners and junctions to be accurately and quickly found, and also a related method for reducing noise whilst preserving image structure. The localization of the features is independent of the mask size used, and noise suppression is shown to be good. Connectivity of edges and lines at junctions is good. SUSAN noise reduction has been shown to be superior to the other methods tested.

8 Source Code and Test Image

The source code for all three algorithms and the test image can be downloaded from:
www.fmrib.ox.ac.uk/~steve

References

[1] L. Akarun and R.A. Haddad. Adaptive decimated median filtering. *Pattern Recognition Letters*, 13:57–62, 1992.

[2] G.R. Arce and S.A. Fontana. On the midrange estimator. *IEEE Trans. on Acoustics, Speech and Signal Processing*, ASSP-36:920–922, 1988.

[3] H. Asada and M. Brady. The curvature primal sketch. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(1):2–14, 1986.

[4] P.R. Beaudet. Rotational invariant image operators. In *Proc. of the Int. Conf. on Pattern Recognition*, pages 579–583, 1978.

[5] J.B. Bedner and T.L. Watt. Alpha-trimmed means and their relationships to median filters. *IEEE Trans. on Acoustics, Speech and Signal Processing*, ASSP-32:145–153, 1984.

[6] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, USA, August 1987.

[7] D.R.K. Brownrigg. The weighted median filter. *Commun. ACM*, 27:807–818, 1984.

[8] J.F. Canny. Finding edges and lines in images. Master's thesis, MIT, Cambridge, USA, 1983.

[9] J.F. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

[10] D. Charnley and R.J. Blissett. Surface reconstruction from outdoor image sequences. *Image and Vision Computing*, 7(1):10–16, 1989.

[11] R.T. Chin and C.-L. Yeh. Quantitative evaluation of some edge-preserving noise-smoothing filters. *Computer Vision, Graphics and Image Processing*, 23:67–91, 1983.

[12] J. Cooper, S. Venkatesh, and L. Kitchen. Early jump-out corner detectors. Technical Report 90/14, University of Western Australia, Department of Computer Science, 1990.

[13] L.S. Davis and A. Rosenfeld. Noise cleaning by iterated local averaging. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-8(9):705–710, 1978.

[14] R. Deriche. Using Canny's criteria to derive a recursively implemented optimal edge detector. *Int. Journal of Computer Vision*, 1(2):167–187, 1987.

[15] R. Deriche and G. Giraudon. Accurate corner detection: An analytical study. In *Proc. 3rd Int. Conf. on Computer Vision*, pages 66–70, 1990.

[16] L. Dreschler and H.-H. Nagel. Volumetric Model and 3D Trajectory of a Moving Car Derived from Monocular TV-frame Sequence of a Street Scene. *Computer Vision, Graphics and Image Processing*, 20(3):199–228, 1981.

[17] R. Ehrich. A symmetric hysteresis smoothing algorithm that preserves principal features. *Computer Graphics and Image Processing*, 8:121–126, 1978.

[18] M.M. Fleck. Spectre: an improved phantom edge finder. In *Proc. 5th Alvey Vision Conference*, pages 127–132, 1989.

[19] W. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *ISPRS Intercommission Workshop, Interlaken*, pages 149–155, June 1987.

[20] H. Freeman and L.S. Davis. A corner finding algorithm for chain code curves. *IEEE Trans. on Computers*, 26:297–303, 1977.

[21] D. Geiger and F. Girosi. Parallel and deterministic algorithms from MRFs: Surface reconstruction and integration. In *Proc. 1st European Conf. on Computer Vision*, pages 89–98, 1990.

[22] R.M. Haralick. Digital step edges from zero crossing of second directional derivatives. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6(1):58–68, January 1984.

[23] R.M. Haralick and J.S.J. Lee. Context dependent edge detection and evaluation. *Pattern Recognition*, 23:1–19, 1990.

[24] R.M. Haralick and L.G. Shapiro. *Computer and Robot Vision*, volume 1. Addison-Wesley, 1992.

[25] C.G. Harris and J.M. Pike. 3D positional integration from image sequences. *Image and Vision Computing*, 6(2):87–90, 1988.

[26] C.G. Harris and M. Stephens. A combined corner and edge detector. In *4th Alvey Vision Conference*, pages 147–151, 1988.

[27] D. Harwood, M. Subbarao, H. Hakalahti, and L.S. Davis. A new class of edge-preserving smoothing filters. *Pattern Recognition Letters*, 6:155–162, 1987.

[28] M.H. Hueckel. An operator which locates edges in digitized pictures. *Journal of the Association for Computing Machinery*, 18(1):113–125, 1971.

[29] L. Kitchen and A. Rosenfeld. Gray-level corner detection. *Pattern Recognition Letters*, 1:95–102, 1982.

[30] J.-S. Lee. Digital image smoothing and the sigma filter. *Computer Vision, Graphics and Image Processing*, 24:255–269, 1983.

[31] A. Lev, S.W. Zucker, and A. Rosenfeld. Iterative enhancement of noisy images. *IEEE Trans. on Systems, Man and Cybernetics*, 7:435–442, 1977.

[32] D. Li, G.D. Sullivan, and K.D. Baker. Edge detection at junctions. In *Proc. 5th Alvey Vision Conference*, pages 121–125, 1989.

[33] S.-T. Liu and W.-H. Tsai. Moment-preserving corner detection. *Pattern Recognition*, 23:441–460, 1990.

[34] D. Marr and E.C. Hildreth. Theory of edge detection. *Proc. Roy. Soc. London.*, B-207:187–217, 1980.

[35] M.J. McDonnell. Box-filtering techniques. *Computer Graphics and Image Processing*, 17:65–70, 1981.

[36] G. Medioni and Y. Yasumoto. Corner detection and curve representation using cubic b-splines. *Computer Vision, Graphics and Image Processing*, 39:267–278, 1987.

[37] R. Mehrotra, S. Nichani, and N. Ranganathan. Corner detection. *Pattern Recognition*, 23:1223–1233, 1990.

[38] H.P. Moravec. Towards automatic visual obstacle avoidance. In *Proc. of the International Joint Conference on Artificial Intelligence*, page 584, 1977.

[39] H.P. Moravec. Visual mapping by a robot rover. In *Proc. of the 6th International Joint Conference on Artificial Intelligence*, pages 598–600, 1979.

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

[41] H.-H. Nagel. Principles of (low level) computer vision. In J.P. Haton, editor, *Fundamentals in computer understanding: speech and vision*, pages 113–139. Cambridge University Press, 1987.

[42] V.S. Nalwa and T.O. Binford. On detecting edges. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):699–714, November 1986.

[43] J.A. Noble. *Descriptions of Image Surfaces.* D.Phil. thesis, Robotics Research Group, Department of Engineering Science, Oxford University, 1989.

[44] N. Nordström. Biased anisotropic diffusion – a unified regularization and diffusion approach to edge detection. In *Proc. 1st European Conf. on Computer Vision*, pages 18–27, 1990.

[45] M. Otte and H.-H. Nagel. Extraction of line drawings from gray value images by non-local analysis of edge element structures. In *Proc. 2nd European Conf. on Computer Vision*, pages 687–695. Springer-Verlag, 1992.

[46] K. Paler, J. Föglein, J. Illingworth, and J. Kittler. Local ordered grey levels as an aid to corner detection. *Pattern Recognition*, 17(5):535–543, 1984.

[47] P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. In *IEEE Workshop on Computer Vision*, pages 16–22, 1987.

[48] P. Perona and J. Malik. Detecting and localizing edges composed of steps, peaks and roofs. In *Proc. 3rd Int. Conf. on Computer Vision*, pages 52–57, 1990.

[49] W.K. Pratt. Generalized Wiener filtering computation techniques. *IEEE Trans. Computers*, C-21:636–692, 1972.

[50] J.M.S. Prewitt. Object enhancement and extraction. In B.S. Lipkin and A. Rosenfeld, editors, *Picture Processing and Psychopictorics*. Academic Press, 1970.

[51] L.G. Roberts. Machine perception of three dimensional solids. In J.T. Tippet, editor, *Optical and Electro-optical Information Processing*, pages 159–197. MIT Press, 1965.

[52] K. Rohr. Modelling and identification of characteristic intensity variations. *Image and Vision Computing*, 10(2):66–76, March 1992.

[53] L. Rosenthaler, F. Heitger, O. Kübler, and R. von der Heydt. Detection of general edges and keypoints. In *Proc. 2nd European Conf. on Computer Vision*, pages 78–86. Springer-Verlag, 1992.

[54] P. Saint-Marc, J.S. Chen, and G. Medioni. Adaptive smoothing: A general tool for early vision. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 618–624, 1989.

[55] I. Scollar, B. Weidner, and T.S. Huang. Image enhancement using the median and the interquartile distance. *Computer Vision, Graphics and Image Processing*, 25:236–251, 1984.

[56] J. Shen and S. Castan. An optimal linear operator for step edge detection. *Computer Vision, Graphics and Image Processing*, 54(2):112–133, March 1992.

[57] Q. Shen. Fuzzy image smoothing. In *Proc. Int. Conf. on Pattern Recognition*, pages 74–78, 1990.

[58] A. Singh and M. Shneier. Grey level corner detection: A generalization and a robust real time implementation. *Computer Vision, Graphics and Image Processing*, 51:54–69, 1990.

[59] S.S. Sinha and B.G. Schunk. A two-stage algorithm for discontinuity-preserving surface reconstruction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(1):36–55, 1992.

[60] S.M. Smith. Method for digitally processing images to determine the position of edges and/or corners therein for guidance of unmanned vehicle. UK Patent 2272285. Proprietor: Secretary of State for Defence, UK. 15 January 1997.

[61] S.M. Smith. A brief quantitative assessment of a passive 3D measurement system. RARDE Memorandum 31/90, DRA Chertsey, Chobham Lane, Chertsey, Surrey, UK, August 1990.

[62] S.M. Smith. Extracting information from images. First year D.Phil. report, Robotics Research Group, Department of Engineering Science, Oxford University, June 1990.

[63] S.M. Smith. *Feature Based Image Sequence Understanding*. D.Phil. thesis, Robotics Research Group, Department of Engineering Science, Oxford University, 1992.

[64] S.M. Smith. Flexible filter neighbourhood designation. In *Proc. 13th Int. Conf. on Pattern Recognition*, volume 1, pages 206–212, 1996.

[65] S.M. Smith and J.M. Brady. A scene segmenter; visual tracking of moving vehicles. *Engineering Applications of Artificial Intelligence*, 7(2):191–204, April 1994.

[66] S.M. Smith and J.M. Brady. SUSAN - a new approach to low level image processing. *Int. Journal of Computer Vision*, 23(1):45–78, May 1997.

[67] I. Sobel. An isotropic 3×3 image gradient operator. In H. Freeman, editor, *Machine Vision for Three-Dimensional Scenes*, pages 376–379. Academic Press, 1990.

[68] H.L. Tan, S.B. Gelfand, and E.J. Delp. A cost minimization approach to edge detection using simulated annealing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(1):3–18, January 1980.

[69] J.W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, Menlo Park, CA, 1971.

[70] S. Venkatesh. *A Study of Energy Based Models for the Detection and Classification of Image Features.* PhD thesis, The University of Western Australia, Department of Computer Science, 1990.

[71] S. Venkatesh and L.J. Kitchen. Edge evaluation using necessary components. *Computer Vision, Graphics and Image Processing*, 54(1):23–30, January 1992.

[72] S. Venkatesh and R. Owens. An energy feature detection scheme. In *Proceedings, IEEE Int. Conf. on Image Processing, Singapore*, pages 553–557, September 1989.

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

[74] H. Wang and J.M. Brady. Corner detection with subpixel accuracy. Technical Report OUEL 1925/92, Dept. Engineering Science, University of Oxford, 1992.

[75] A.P. Witkin. Scale-space filtering. In *Proc. IJCAI 1983*, pages 1019–1021, 1983.

[76] O.A. Zuniga and R.M. Haralick. Corner Detection Using the Facet Model. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 30–37, 1983.

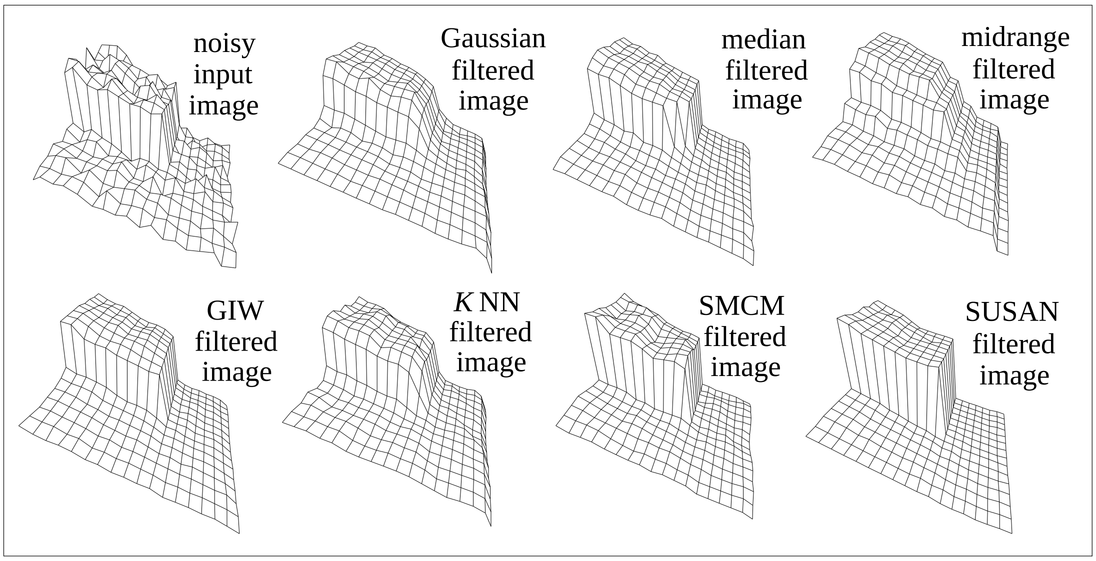

Figure 40: The outputs of the seven filters shown as surfaces for section 3 of the input image.

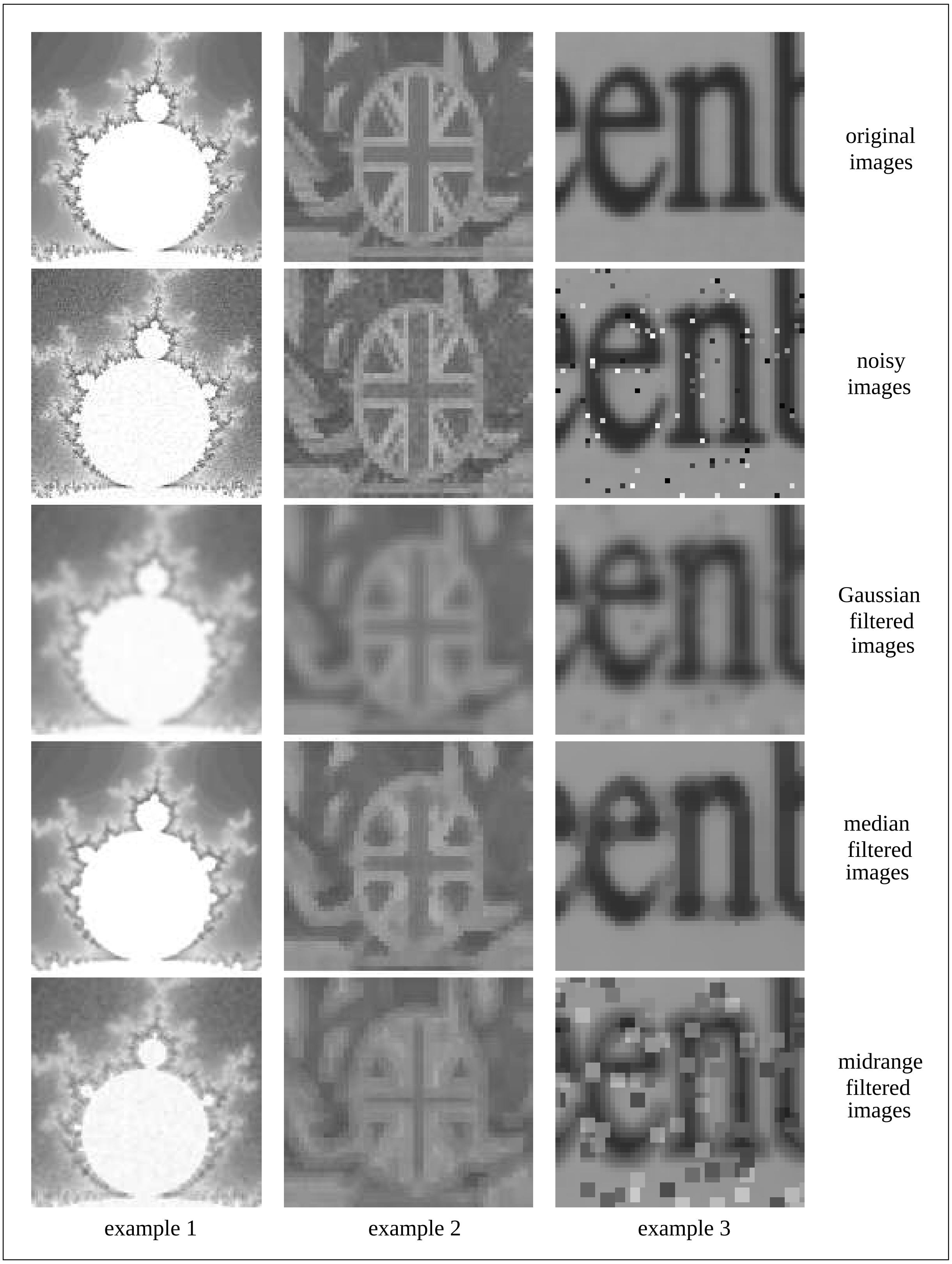

Figure 41: Three example images, each with a different type of added noise, processed by three different filters.

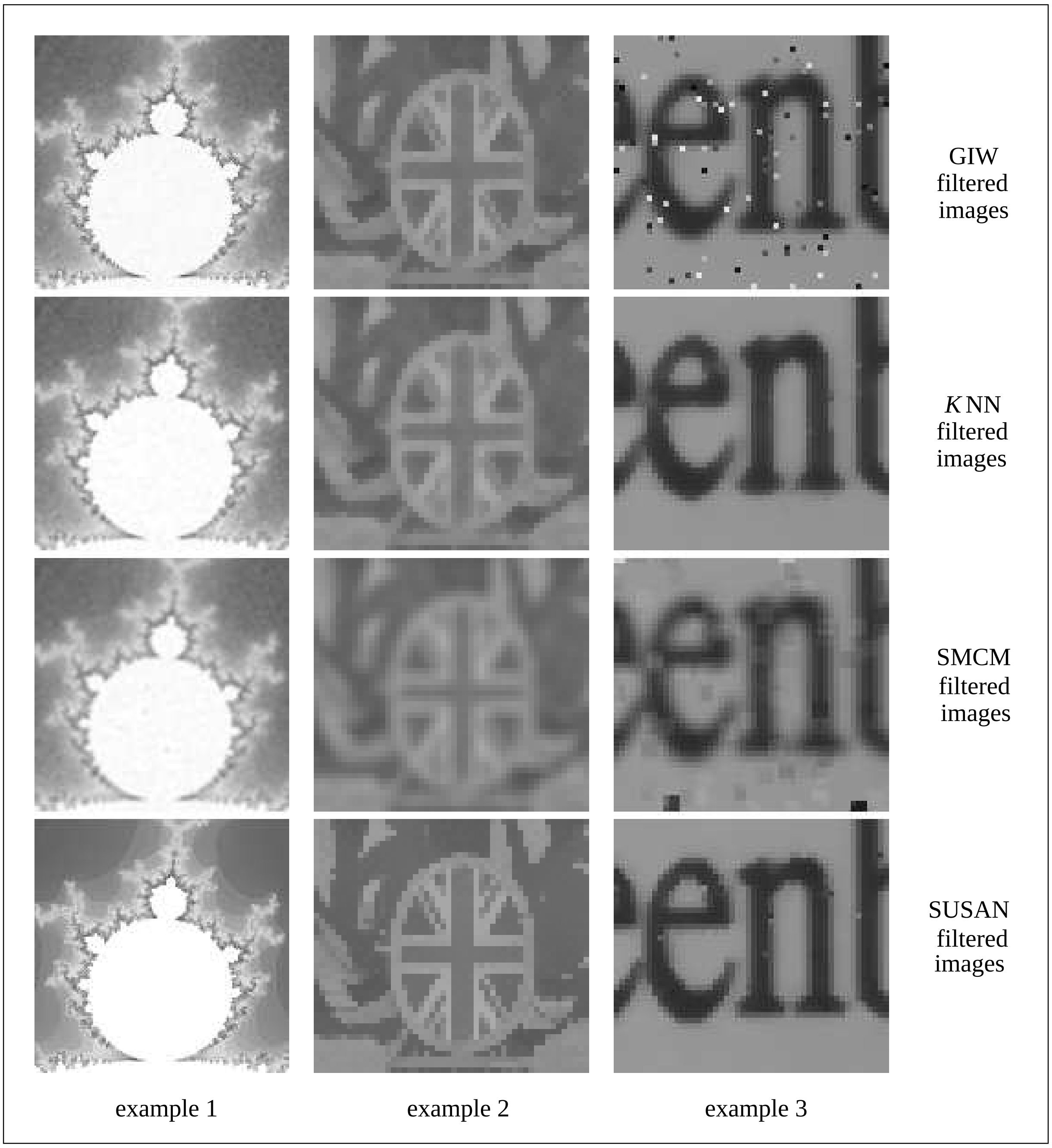

Figure 42: Three example images, each with a different type of added noise, processed by four different filters.

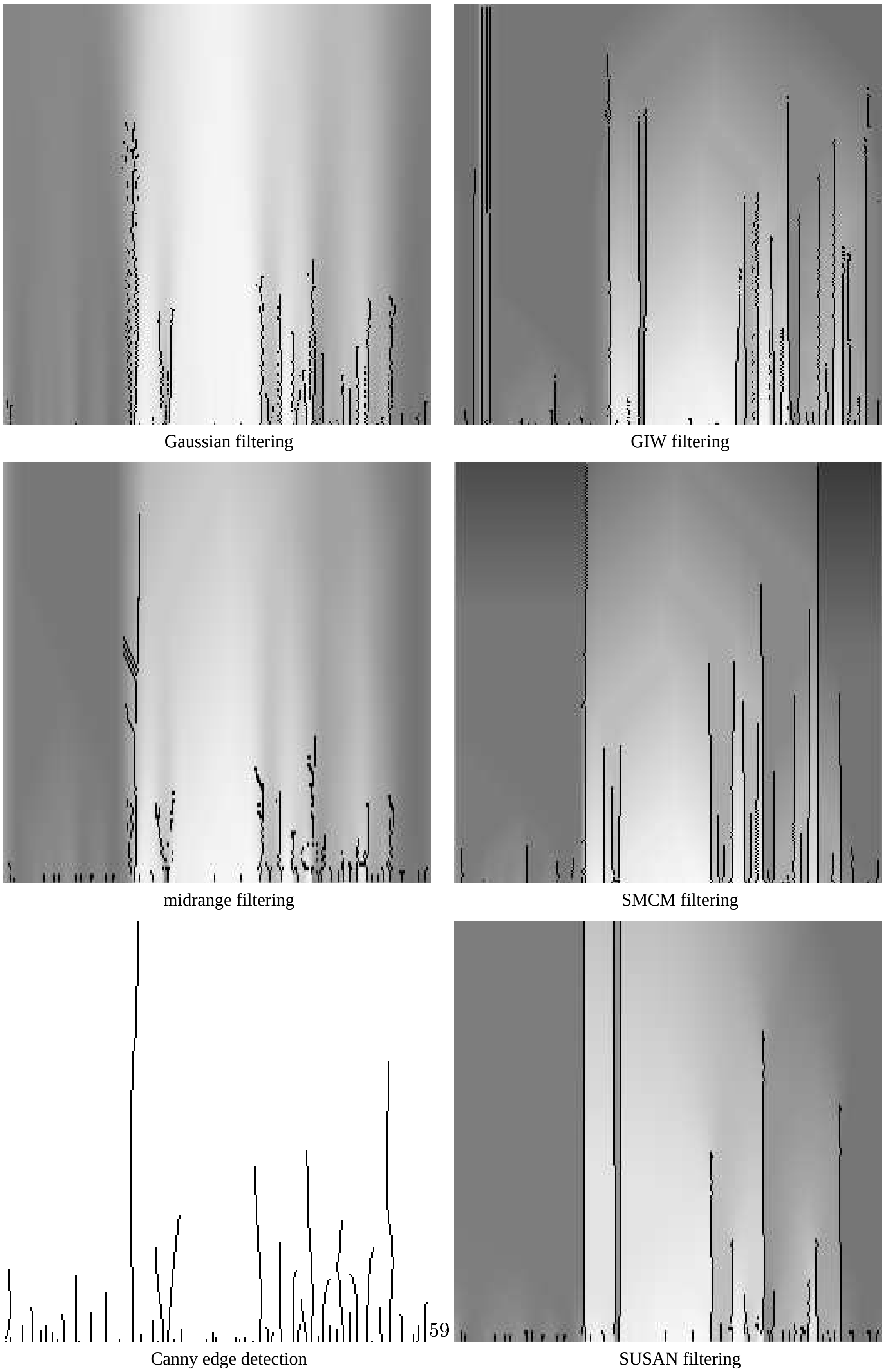

Figure 43: Scale-space analysis of five noise filters and the Canny edge detector on an image formed from a single line of a real test image. The bottom of each image corresponds to the smallest smoothing scale.