CHAPTER 3

BOUNDARY DETECTION BY CONTEXTUAL NONLINEAR SMOOTHING

In this chapter we present a two-step boundary detection algorithm. The first step is a nonlinear smoothing algorithm which is based on an orientation-sensitive probability measure. By incorporating geometrical constraints through the coupling structure, we obtain a robust nonlinear smoothing algorithm, where many nonlinear algorithms can be derived as special cases. Even when noise is substantial, the proposed smoothing algorithm can still preserve salient boundaries. Compared with anisotropic diffusion approaches, the proposed nonlinear algorithm not only performs better in preserving boundaries but also has a non-uniform stable state, whereby reliable results are available within a fixed number of iterations independent of images. The second step is simply a Sobel edge detection algorithm without non-maximum suppression and hysteresis tracking. Due to the proposed nonlinear smoothing, salient boundaries are extracted effectively. Experimental results using synthetic and real images are provided. The results presented in this chapter appeared in [79] [80].

3.1 Introduction

One of the fundamental tasks in low-level machine vision is to locate discontinuities in images corresponding to physical boundaries between a number of regions. A common practice is to identify local maxima in local gradients of images - collectively known as edge detection algorithms. The Sobel edge detector [14] consists of two 3 x 3 convolution kernels, which respond maximally to vertical and horizontal edges respectively. Local gradients are estimated by convolving the images with the two kernels, and thresholding is then applied to get rid of noisy responses. The Sobel edge detector is computationally efficient but sensitive to noise. To make the estimation of gradients more reliable, the image can be convolved with a low-pass filter before estimation and two influential methods are due to Marr and Hildreth [89] and Canny [13]. By convolving the image with a Laplacian of Gaussian kernel, the resulting local maxima, which are assumed to correspond to meaningful edge points, are zero-crossings in the filtered image [89]. Canny [13] derived an optimal step edge detector using variational techniques starting from some optimal criteria and used the first derivative of a Gaussian as a good approximation of the derived detector. Edge points are then identified using a non-maximum suppression and hysteresis thresholding for better continuity of edges. As noticed by Marr and Hildreth [89], edges detected at a fixed scale are not sufficient and multiple scales are essentially needed in order to obtain good results. By formalizing the multiple scale approach, Witkin [142] and Koenderink [66] proposed Gaussian scale space. The original image is embedded in a family of gradually smoothing images controlled by a single parameter, which is equivalent to solving a heat equation with input as the initial condition [66]. While Gaussian scale space has nice properties and is widely used in machine vision

[75], a major limitation is that Gaussian smoothing inevitably blurs edges and other important features due to its low-pass nature. To overcome the limitation, anisotropic diffusion, which was proposed by Cohen and Grossberg [19] in modeling the primary visual cortex, was formulated by Perona and Malik [105]:

$$\frac{\partial I}{\partial t} = div(g(\|\nabla\|)\nabla I) \tag{3.1}$$

Here *div* is the divergence operator, and g is a nonlinear monotonically decreasing function and ∇I denotes the gradient. By making the diffusion conductance dependent explicitly on local gradients, anisotropic diffusion prefers intra-region smoothing over inter-region smoothing, resulting immediate localization while noise is reduced [105]. Because it produces visually impressive results, anisotropic diffusion generates much theoretical as well as practical interest (see reference [138] for a recent review). While many improvements have been proposed, including spatial regularization [137] and edge-enhancing anisotropic diffusion [15], the general framework remains the same. As shown by You et al. [145], anisotropic diffusion given by (3.1) is the steepest gradient descent minimizer of the following energy function:

$$E(I) = \int_{\Omega} f(\|\nabla\|) d\Omega \tag{3.2}$$

with

$$g(\|\nabla\|) = \tfrac{f'(\|\nabla\|)}{\|\nabla\|}$$

Under some general conditions, the energy function given by (3.2) has a unique and trivial global minimum, where the image is constant everywhere, and thus interesting results exist only within a certain period of diffusion. An immediate problem is how to determine the termination time, which we refer to as the termination problem. While there are some heuristic rules on how to choose the stop time [137] [15], in general it corresponds to the open problem of automatic scale selection. As in Gaussian scale space, a fixed time would not be sufficient to obtain good results. Another problem of anisotropic diffusion is that diffusion conductance is a deterministic function of local gradients, which, similar to non-maximum suppression in edge detection algorithms, makes an implicit assumption that larger gradients are due to true boundaries. When noise is substantial and gradients due to noise and boundaries cannot be distinguished based on magnitudes, the approach tends to fail to preserve meaningful region boundaries.

To illustrate the problems, Figure 3.1(a) shows a noise-free image, where the gradient magnitudes along the central square change considerably. Figure 3.1(b) shows a noisy version of Figure 3.1(a) by adding Gaussian noise with zero mean and $\sigma = 40$, and Figure 3.1(c) shows its local gradient magnitude obtained using Sobel operators [14]. While the three major regions in Figure 3.1(b) may be perceived, Figure 3.1(c) is very noisy and the strong boundary fragment is barely visible. Figures 3.1 (d)-(f) show the smoothed images by an anisotropic diffusion algorithm [106] with specified numbers of iterations. Figures 3.1 (g)-(i) show the edge maps of Figure 3.1 (d)-(f), respectively, using the Sobel edge detection algorithm. While at the 50th iteration the result is still noisy, the result becomes meaningless at the 1000th iteration. Even though the result at the 100th iteration is visually good, the boundaries are still fragmented and it is not clear how to identify a "good" number of iterations automatically.

These problems to a large extent are due to the assumption that local maxima in gradient images are good edge points. In other words, due to noise, responses from true boundaries and those from noise are not distinguishable based on magnitude.

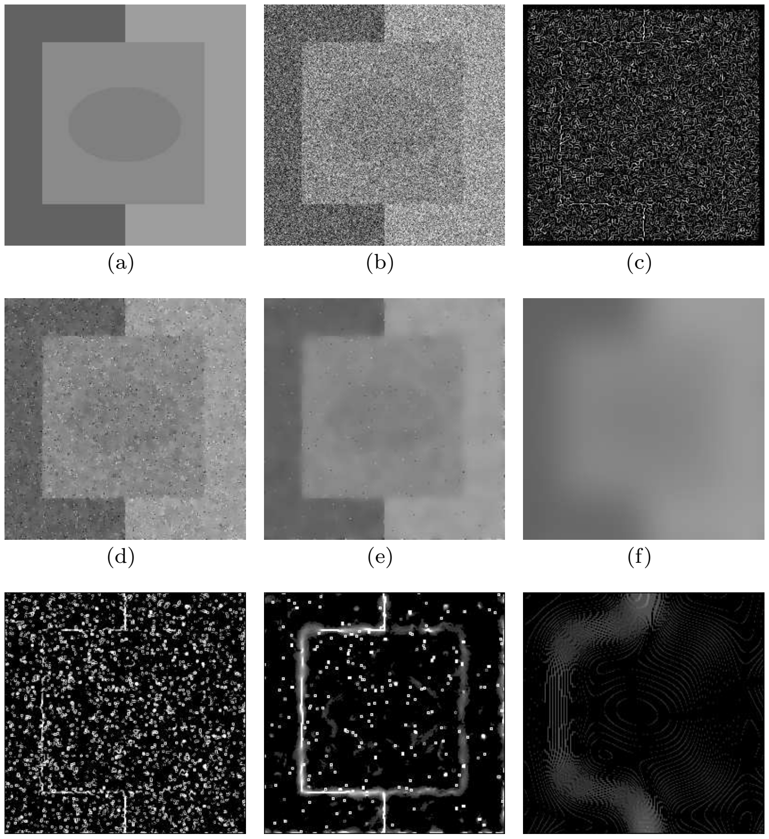


Figure 3.1: An example with non-uniform boundary gradients and substantial noise. (a) A noise-free synthetic image. Gray values in the image: 98 for the left '[' region, 138 for the square, 128 for the central oval, and 158 for the right ']' region. (b) A noisy version of (a) with Gaussian noise of $\sigma = 40$. (c) Local gradient map of (b) using the Sobel operators. (d)-(f) Smoothed images from an anisotropic diffusion algorithm [106] at 50, 100, and 1000 iterations. (g)-(i) Corresponding edge maps of (d)-(f) respectively using the Sobel edge detector.

To overcome these problems, contextual information, i.e., responses from neighboring pixels, should be incorporated in order to reduce ambiguity as in relaxation labeling and related methods [112] [55] [72] [42]. In general, relaxation labeling methods use pair-wise compatibility measure, which is determined based on a priori models associated with labels, and convergence is not known and often very slow in numerical simulations[88]. In this chapter, by using an orientation-sensitive probability measure, we incorporate contextual information through the geometrical constraints on the coupling structure. Numerical simulations show that the resulting nonlinear algorithm has a non-uniform stable state and good results can be obtained within a fixed number of iterations independent of input images. Also, the oriented probability measure is defined on input data, and thus no a priori models need to be assumed.

In Section 3.2, we formalize our contextual nonlinear smoothing algorithm and show that many nonlinear smoothing algorithms can be treated as special cases. Section 3.3 gives some theoretical results as well as numerical simulations regarding to the stability and convergence of the algorithm. Section 3.4 provides experimental results using synthetic and real images. Section 3.5 concludes the chapter with further discussions.

3.2 Contextual Nonlinear Smoothing Algorithm

3.2.1 Design of the Algorithm

To design a statistical algorithm, with no prior knowledge, we assume a Gaussian distribution within each region. That is, given a pixel (i_0, j_0) and a window $R_{(i_0,j_0)}$ at pixel (i_0, j_0), consisting of a set of pixel locations, we assume that:

$$P(I_{(i_0,j_0)}, R) = \frac{1}{\sqrt{2\pi}\sigma_R} \exp\left\{-\frac{(I_{(i_0,j_0)} - \mu_R)^2}{2\sigma_R^2}\right\} \qquad (3.3)$$

where $I_{(i,j)}$ is the intensity value at pixel location (i, j). To simplify notation, without confusion, we use R to stand for $R_{(i_0,j_0)}$. Intuitively, $P(I_{(i_0,j_0)}, R)$ is a measure of compatibility between intensity value at pixel (i_0, j_0) and statistical distribution in window R. To estimate the unknown parameters of μ_R and σ_R, consider the pixels in R as n realizations of (3), where $n = |R|$. The likelihood function of μ_R and σ_R is [119]:

$$L(R, \mu_R, \sigma_R) = \left(\frac{1}{\sqrt{2\pi}\sigma_R}\right)^n \exp\left(-\frac{1}{2\sigma_R^2} \sum_{(i,j)\in R} (I_{(i,j)} - \mu_R)^2\right) \tag{3.4}$$

By maximizing (3.4), we get the maximum likelihood estimators for μ_R and σ_R:

$$\hat{\mu}_R = \frac{1}{n} \sum_{(i,j)\in R} I_{(i,j)} \tag{3.5a}$$

$$\hat{\sigma}_R = \frac{1}{\sqrt{n}} \sqrt{\sum_{(i,j)\in R} (I_{(i,j)} - \hat{\mu}_R)^2} \tag{3.5b}$$

To do a nonlinear smoothing, similar to selective smoothing filters [125] [95], suppose that there are M windows $R^{(m)}$, where $1 \leq m \leq M$, around a central pixel (i_0, j_0). Here these $R^{(m)}$'s can be generated from one or several basis windows through rotation, which are motivated by the experimental findings of orientation selectivity in the visual cortex [53]. Simple examples are elongated rectangular windows (refer to Figure 3.6), which are used throughout this chapter for synthetic and real images. The probability that pixel (i_0, j_0) belongs to $R^{(m)}$ can be estimated from equations (3.3) and (3.5). By assuming that the weight of each $R^{(m)}$ should be proportional to the probability, as in relaxation labeling [112] [55], we obtain an iterative nonlinear smoothing filter:

$$I^{t+1}_{(i_0,j_0)} = \frac{\sum_m P\left(I^t_{(i_0,j_0)}, R^{(m)}\right) \hat{\mu}^t_{R^m}}{\sum_m P\left(I^t_{(i_0,j_0)}, R^{(m)}\right)} \tag{3.6}$$

A problem with this filter is that it is not sensitive to weak edges due to the linear combination. To generate more semantically meaningful results and increase the sensitivity even to weak edges, we apply a nonlinear function on weights, which is essentially same as anisotropic diffusion [105]:

$$I^{t+1}_{(i_0,j_0)} = \frac{\sum_m g\Big(P\Big(I^t_{(i_0,j_0)}, R^{(m)}\Big)\Big)\hat{\mu}^t_{R^m}}{\sum_m g\Big(P\Big(I^t_{(i_0,j_0)}, R^{(m)}\Big)\Big)} \tag{3.7}$$

Here g[2] is a nonlinear monotonically increasing function. A good choice for g is an exponential function, which is widely used in nonlinear smoothing anisotropic diffusion approaches:

$$g(x) = exp(x^2/K) \tag{3.8}$$

Here parameter K controls the sensitivity to edges [113]. Equation (3.7) provides a generic model for a wide range of nonlinear algorithms, the behavior of which largely depends on the sensitivity parameter K. When K is large, (3.7) reduces to the equally weighted average smoothing filter. When K is around 0.3, g is close to a linear function in $[0, 1]$ and (3.7) then reduces to (3.6). When K is a small positive number, (3.7) will be sensitive to all discontinuities. No matter how small the weight of one window can be, theoretically speaking, if it is nonzero, when $t \to \infty$, the system will reach a uniform stable state. Similar to anisotropic diffusion approaches, the desired results will be time-dependent and the termination problem becomes a critical issue for autonomous solutions. To overcome this limitation, we restrict smoothing only within the window with the highest probability similar to selective smoothing [125] [95]:

$$m^* = \max_{1 \le m \le M}\Big(P\Big(I^t_{(i_0,j_0)}, R^{(m)}\Big)\Big) \tag{3.9}$$

[2]Because the probability measure given by (3.1) is inversely related to gradient measure used in most nonlinear smoothing algorithms, (3.8) is an increasing function instead of a decreasing function in our method.

The nonlinear smoothing through (3.9) is desirable in regions that are close to edges. By using appropriate $R^{(m)}$'s, (3.9) encodes discontinuities implicitly. But in homogeneous regions, (3.9) may produce artificial block effects due to intensity variations. Under the proposed statistical formulation, there is an adaptive method to detect homogeneity. Based on the assumption that there are M $R^{(m)}$ windows around a central pixel (i_0, j_0), where each window has a Gaussian distribution, consider the mean in each window as a new random variable:

$$\mu^{(m)} = \frac{1}{|R^{(m)}|} \sum_{(i,j)\in R^{(m)}} I_{(i,j)} \tag{3.10}$$

Because $\mu^{(m)}$ is a linear combination of random variables with a Gaussian distribution, $\mu^{(m)}$ has also a Gaussian distribution with the same mean and a standard deviation given by:

$$\sigma_{\mu^{(m)}} = \frac{1}{\sqrt{|R^{(m)}|}} \sigma_{R^{(m)}} \tag{3.11}$$

This provides a probability measure of how likely that the M windows are sampled from one homogeneous region. Given a confidence level α, for each pair of windows $R^{(m1)}$ and $R^{(m2)}$, we have:

$$|\mu^{(m1)} - \mu^{(m2)}| \leq \min\Big(\sqrt{\frac{\log(1/\alpha)}{|R^{(m1)}|}}\hat{\sigma}_{R^{(m1)}}, \sqrt{\frac{\log(1/\alpha)}{|R^{(m2)}|}}\hat{\sigma}_{R^{(m2)}}\Big) \tag{3.12}$$

If all the pairs satisfy (3.12), the M windows are likely from one homogeneous region with confidence α. Intuitively, under the assumption of a Gaussian distribution, when we have more samples, i.e., the window $R^{(m)}$ is larger, the estimation of the mean is more precise and so the threshold should be smaller. In a region with a larger standard deviation, the threshold should be larger because larger variations are allowed.

The nonlinear smoothing algorithm outlined above works well when noise is not very large. In cases when signal to noise ratio is very low, the probability measure given in (3.3) would be unreliable because pixel values change considerably. This problem can be alleviated by using the mean value of pixels sampled from R which are close to the central pixel (i_0, j_0), or along a certain direction to make the algorithm more orientation sensitive.

To summarize, we obtain a nonlinear smoothing algorithm. We define M oriented windows which can be obtained by rotating one or more basis windows. At each pixel, we estimate parameters using (3.5). If all the M windows belong to a homogeneous region according to (3.12), we do the smoothing within all the M windows. Otherwise, the smoothing is done only within the most compatible window given by (3.9).

3.2.2 A Generic Nonlinear Smoothing Framework

In this section we will show how to derive several widely used nonlinear algorithms from the statistical nonlinear algorithm outlined above. Several early nonlinear filters [125] [95] do the smoothing in a window where the standard deviation is the smallest. These filters can be obtained by simplifying (3.3) to:

$$P(I_{(i_0,j_0)}, \hat{\mu}, \hat{\sigma}) = \frac{1}{\sqrt{2\pi}\hat{\sigma}} C \tag{3.13}$$

where C is a constant. Then the solution to (3.9) is the window with the smallest deviation. Recently, Higgins and Hsu [47] extended the principle of choosing the window with the smallest deviation for edge detection.

Another nonlinear smoothing filter is the gradient-inverse filter [131]. Suppose that there is one window, i.e., $M = 1$, consisting of the central pixel (i_0, j_0) itself only, the estimated deviation for a given pixel (i, j) in (3.5b) now becomes:

$$\hat{\sigma} = |I_{(i,j)} - I_{(i_0,j_0)}| \tag{3.14}$$

Equation (3.14) is a popular way to estimate local gradients. Using (13) as the probability measure, (3.6) becomes exactly the gradient inverse nonlinear smoothing filter [131].

SUSAN nonlinear smoothing filter [117] is proposed based on SUSAN (Smallest Univalue Segment Assimilating Nucleus) principle. It is formulated as:

$$I^{t+1}_{(i_0,j_0)} = \frac{\sum_{(\delta i,\delta j)\neq(0,0)} I^t_{(i_0+\delta i, j_0+\delta j)} W(i_0, j_0, \delta i, \delta j)}{\sum_{(\delta i,\delta j)\neq(0,0)} W(i_0, j_0, \delta i, \delta j)} \tag{3.15}$$

where

$$W(i_0, j_0, \delta i, \delta j) = \exp\Bigg(-\frac{1}{2\sigma^2} - \frac{\left(I^t_{(i_0+\delta i, j_0+\delta j)} - I^t_{(i_0,j_0)}\right)^2}{T^2}\Bigg)$$

Here (i_0, j_0) is the central pixel under consideration, and $(\delta i, \delta j)$ defines a local neighborhood. Essentially, it integrates Gaussian smoothing in spatial and brightness domains. The parameter T is a threshold for intensity values. It is easy to see from (3.15) that the weights are derived based on pair-wise intensity value differences. It would be expected that the SUSAN filter performs well when images consists of relatively homogeneous regions and within each region noise is smaller than T. When noise is substantial, it fails to preserve structures due to the pair-wise difference calculation, where no geometrical constraints are incorporated. This is consistent with the experimental results, which will be discussed later. To get the SUSAN filter, we define one window including the central pixel itself only. For a given pixel (i, j) in its neighborhood, (3.3) can be simplified to:

$$P(I_{(i,j)}, R) = C \exp\Big(-\frac{(I_{(i,j)}, -\hat{\mu}_R)^2}{T^2}\Big) \tag{3.16}$$

where C is a scaling factor. Because now $\hat{\mu}_R$ is $I_{(i_0,j_0)}$, (3.6) with the probability measure given by (3.16) is equivalent to Gaussian smoothing in the brightness domain in (3.15).

Now consider anisotropic diffusion given by (3.1). By discretizing (3.1) in image domain with four nearest neighbor coupling [106] and rearranging terms, we have:

$$I_{(i,j)}^{t+1} = \eta_{(i,j)}^{t} I_{(i,j)}^{t} + \lambda \sum_{m} g(P(I_{(i,j)}^{t}, R^{(m)}))\hat{\mu}_{R^{(m)}} \tag{3.17}$$

If we have four singleton regions, (3.17) is essentially a simplified version of (3.7) with an adaptive learning rate.

3.3 Analysis

3.3.1 Theoretical Results

One of the distinctive characteristics of the proposed algorithm is that it requires spatial constraints among responses from neighboring locations through coupling structure as opposed to pair-wise coupling structure. Figure 3.2 illustrates the concept using a manually constructed example. Figure 3.2(a) shows the oriented windows in a 3 x 3 neighborhood, and Figure 3.2(c) shows the coupling structure if we apply the proposed algorithm to a small image patch shown in Figure 3.2(b). The directed graph is constructed as follows. There is a directed edge from (i_1, j_1) to (i_0, j_0) if and only if (i_1, j_1) contributes to the smoothing of (i_0, j_0) according to equations (3.12) and (3.9). By doing so, the coupling structure is represented as a directed graph as shown in Figure 3.2(c). Connected components and strongly connected components [20] of the directed graph can be used to analyze the temporal behavior of the proposed algorithm. A strongly connected component is a set of vertices, or pixels here, where there is a directed path from any vertex to all the other vertices in the set. We obtain a connected component if we do not consider the direction of edges along a path. In the example shown in Figure 3.2(c), all the black pixels form a strongly connected component and so do all the white pixels. Also, there are obviously two connected components.

Essentially our nonlinear smoothing algorithm can be viewed as a discrete dynamic system, the behavior of which is complex due to spatial constraints imposed by coupling windows and adaptive coupling structure by probabilistic grouping. We now prove that a constant region satisfying certain geometrical constraints is a stable state of the smoothing algorithm.

Theorem 1. If a region S of a given image I satisfies:

$$(i_1, j_1) \in S \text{ and } (i_2, j_2) \in S \Rightarrow I_{(i_1,j_1)} = I_{(i_2,j_2)} \tag{3.18a}$$

$$\forall (i,j) \in S \Rightarrow \exists m \, R^{(m)}_{(i,j)} \subseteq S \tag{3.18b}$$

Then S is stable with respect to the proposed algorithm.

Proof. Condition (3.18a) states that S is a constant region and the standard deviation is zero if $R^{(m)}$ is within S according to equation (3.5b). Consider a pixel (i_0, j_0) in S. Inequality (3.12) is satisfied only when all $R^{(m)}$'s are within S. In this case, the smoothing algorithm does not change the intensity value at (i_0, j_0). Otherwise, according to equation (3.9) must be within S because there exists at least one such window according to (3.18b) and thus the smoothing algorithm does not change the intensity value at (i_0, j_0) also. So S is stable. Q.E.D.

A maximum connected component of the constructed graph is stable when its pixels are constant and thus maximum connected components of the constructed graph are a piecewise constant stable solution of the proposed algorithm. For the image patch given in Figure 3.2(b), for example, a stable solution is that pixels in each of the two connected components are constant. The noise-free image in Figure

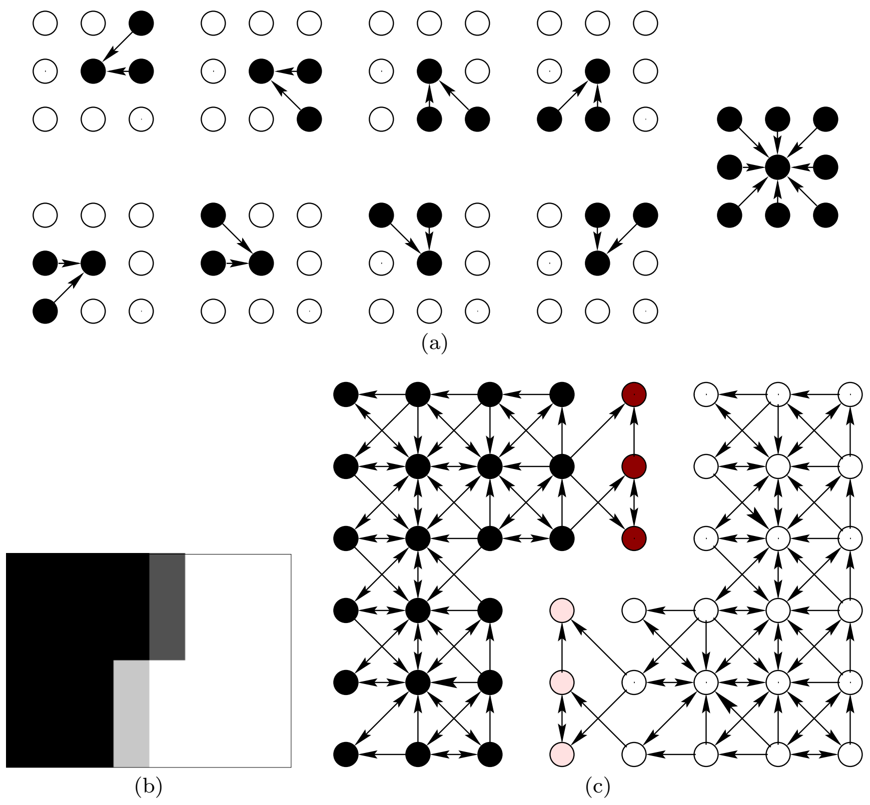


Figure 3.2: Illustration of the coupling structure of the proposed algorithm. (a) Eight oriented windows and a fully connected window defined on a 3 x 3 neighborhood. (b) A small synthetic image patch of 6 x 8 in pixels. (c) The resulting coupling structure for (b). There is a directed edge from (i_1, j_1) to a neighbor (i_0, j_0) if and only if (i_1, j_1) contributes to the smoothing of (i_0, j_0) according to equations (3.12) and (3.9). Each circle represents a pixel, where the inside color is proportional to the gray value of the corresponding pixel. Ties in (3.9) are broken according to left-right and top-down preference of the oriented windows in (a).

3.1(a) is also a stable solution by itself, as we will demonstrate through numerical simulations later on. It is easy to see from the proof that any region which satisfies conditions (3.18a) and (3.18b) during temporal evolution will stay unchanged. In addition, due to the smoothing nature of the algorithm, a local maximum at iteration t cannot increase according to the smoothing kernel by equation (3.12) or (3.9), and similarly, a local minimum cannot decrease. We conjecture that any given image approaches an image that is almost covered by homogeneous regions. Due to the spatial constraints given by (3.18b), it is not clear if the entire image converges to a piece-wise constant stable state. Within each resulting homogeneous region, (3.18b) is satisfied and thus the region becomes stable. For pixels near boundaries, corners, and junctions, it is possible that (18b) is not uniquely satisfied within one constant region, and small changes may persist. The whole image in this case attains a quasi-equilibrium state. This is supported by the following numerical simulations using synthetic and real images. While there are pixels which do not converge within 1000 iterations, the smoothed image as a whole does not change noticeably at all. The two maximum strongly connected components in Figure 3.2(b) satisfy condition (3.18b). Both of them are actually uniform regions and thus are stable. Gray pixels would be grouped into one of the two stable regions according to pixel value similarity and spatial constraints.

3.3.2 Numerical Simulations

Because it is difficult to derive the speed of convergence analytically, we use numerical simulations to demonstrate the temporal behavior of the proposed algorithm. Since smoothing is achieved using equally weighted average within selected windows, the algorithm should converge rather quickly in homogeneous regions. To obtain quantitative estimations, we define two measures similar to variance. For synthetic images, where a noise-free image is available, we define the deviation from the ground truth image as:

$$D_{(I)} = \sqrt{\frac{\sum_i \sum_j (I_{(i,j)} - I^{gt}_{(i,j)})^2}{|I|}} \tag{3.19}$$

Here I is the image to be measured and I^{gt} is the ground truth image. The deviation gives an objective measure of how good the smoothed image is with respect to the true image. To measure the convergence, we define relative variance for image I at time t:

$$V^t_{(I)} = \sqrt{\frac{\sum_i \sum_j (I^t_{(i,j)} - I^{t-1}_{(i,j)})^2}{|I|}} \tag{3.20}$$

We have applied the proposed algorithm on the noise-free image shown in Figure 3.1(a) and six noisy images generated from it by adding zero-mean Gaussian noise with σ from 5 to 60. Figure 3.3 shows the deviation from the ground truth image with iterations, and Figure 3.4 shows the relative variance of the noise-free image and four selected noisy images to make the figure more readable. As we can see from Figure 3.3, the noise-free image is a stable solution by itself, where the deviation is always zero. For the noisy images, the deviation from true image is stabilized within a few number of iterations independent of the amount of noise. Figure 3.4 shows that relative variance is bounded with a small upper limit after 10 iterations. This variance is due to the pixels close to boundaries, corners and junctions that do not belong to any resulting constant region. As discussed before, because the spatial constraints cannot be satisfied within one homogeneous region, these pixels have connections from pixels belonging to different homogeneous regions, and thus fluctuate. These pixels are a small fraction of the input image in general, and thus the fluctuations do

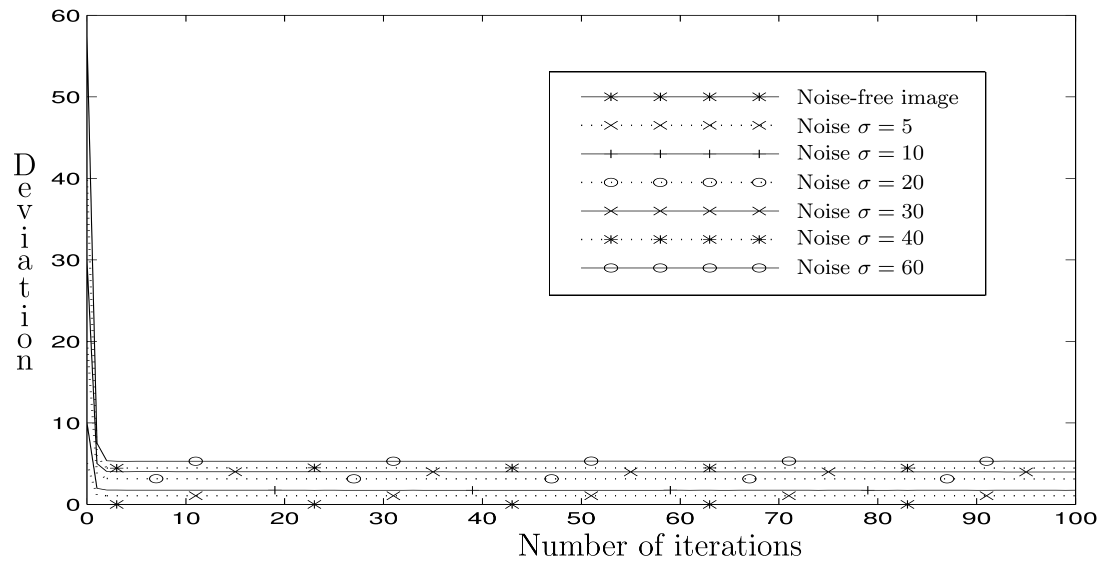


Figure 3.3: Temporal behavior of the proposed algorithm with respect to the amount of noise. Six noisy images are obtained by adding zero-mean Gaussian noise with σ of 5, 10, 20, 30, 40, and 60, respectively, to the noise-free image shown in Figure 3.1(a). The plot shows the deviation from the ground truth image with respect to iterations of the noise-free image and six noisy images.

not affect the quality of the smoothed images noticeably. As shown in Figure 3.3, the deviation is stabilized quickly.

Real images are generally more complicated than synthetic images statistically and structurally, and we have also applied our algorithm to the four real images shown in Figure 3.9-3.12 which include a texture image. Figure 3.5 shows the relative variance in 100 iterations, where the variance is bounded after 10 iterations independent of images. This indicates that the proposed algorithm behaves similarly for synthetic and real images.

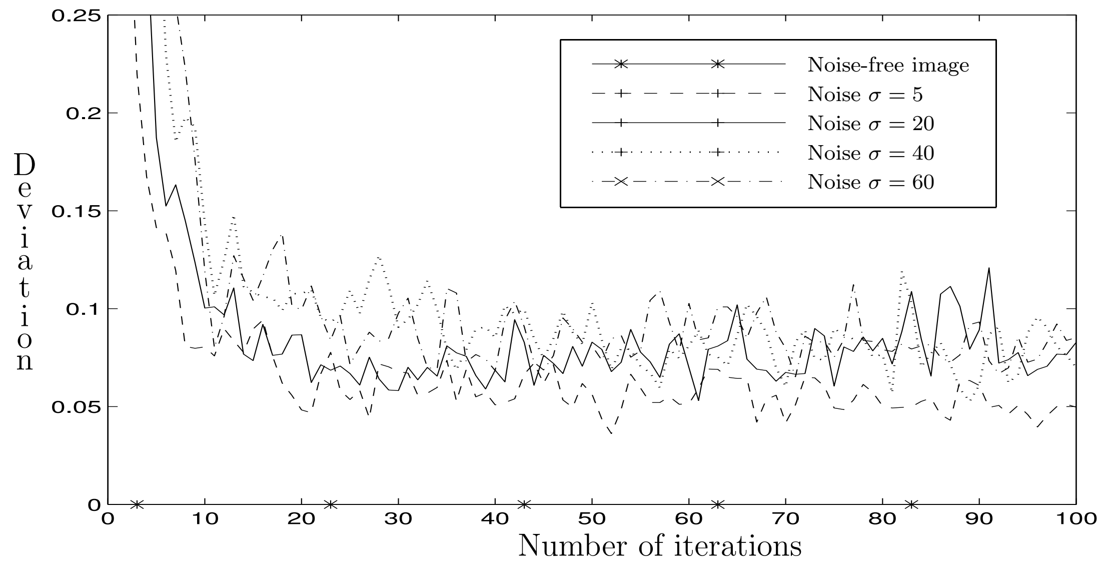


Figure 3.4: Relative variance of the proposed algorithm for the noise-free image shown in Figure 3.1(a) and four noisy images with Gaussian noise of zero-mean and σ of 5, 20, 40 and 60, respectively.

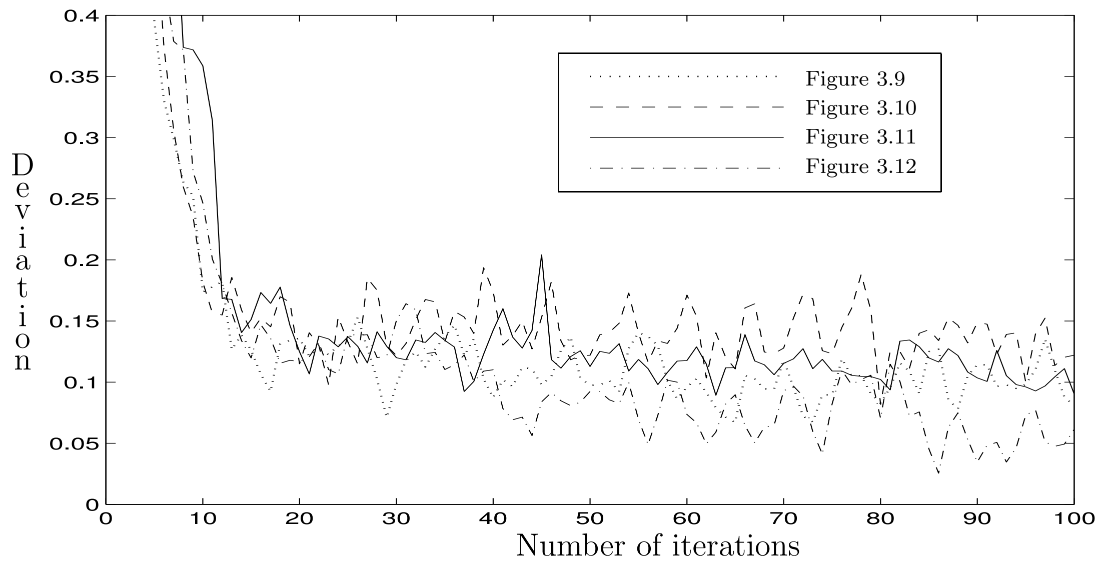


Figure 3.5: Relative variance of the proposed algorithm for real images shown in Figure 3.9-3.12.

3.4 Experimental Results

3.4.1 Results of the Proposed Algorithm

The nonlinear smoothing algorithm formalized in this chapter integrates discontinuity and homogeneity through the orientation-sensitive probability framework. Equation (3.9) represents discontinuity implicitly and (3.12) encodes homogeneity explicitly. Because of the probability measure, the initial errors for choosing smoothing windows due to noise can be overcome by the coupling structure. Essentially only when majority of the pixels in one window make a wrong decision, the final result would be affected. As illustrated in Figure 3.2, the coupling structure is robust.

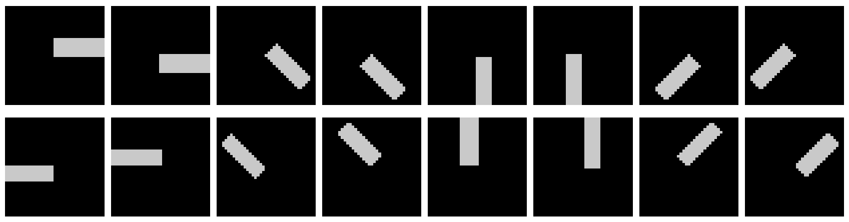


Figure 3.6: The oriented bar-like windows used throughout this chapter for synthetic and real images. The size of each kernel is approximately 3 x 10 in pixels.

To achieve optimal performance, the size and shape of the oriented windows are application dependent. However, due to the underlying coupling structure, the proposed algorithm gives good results for a wide range of parameter values. For example, the same oriented windows are used throughout the experiments in this chapter. As shown in Figure 3.6, these oriented windows are generated by rotating two rectangular basis windows with size of 3 x 10 in pixels. The preferred orientation of each window is consistent with orientation sensitivity of cell responses in the visual cortex [53]. Asymmetric window shapes are used so that 2-D features such as corners and junctions can be preserved.

As is evident from numerous simulations, the proposed algorithm generates stable results around 10 iterations regardless of input images. Thus, all the boundaries from the proposed algorithm are generated using smoothed images at the 11th iteration. As stated above, boundaries are detected using the Sobel edge detector due to its efficiency.

Figure 3.7 shows the results by applying the proposed algorithm on a set of noisy images obtained from the noise-free image shown in Figure 3.1(a) by adding Gaussian noise with σ of 10, 40, and 60 respectively. Same parameters for smoothing are used for the three images. When noise is relative small, the proposed algorithm preserves boundaries accurately as well as corners and junctions, as shown in Figure 3.7(a). When noise is substantial, due to the coupling structure, the proposed algorithm is robust to noise and salient boundaries are well preserved. Because only local information is used in the system, it would be expected that the boundaries are less accurate when noise is larger. This uncertainty is an intrinsic property of the proposed algorithm because reliable estimation gets more difficult when noise gets larger as shown in Figure 3.7(b) and (c). The results seem consistent with our perceptual experience.

Figure 3.8 shows the result for another synthetic image, which was extensively used by Sarkar and Boyer [114]. As shown in Figure 3.8(b), noise is reduced greatly and boundaries as well as corners are well preserved. Even using the simple Sobel edge detector, the result is better than the best result from the optimal infinite impulse responses filters [114] obtained using several parameter combinations with hysteresis thresholding. This is because their edge detector does not consider the responses from neighboring pixels, but rather assumes the local maxima as good edge points.

Figure 3.9 shows an image of a grocery store advertisement which was used throughout the book by Nitzberg, Mumford, and Shiota [98]. In order to get good boundaries, they first applied an edge detector and then several heuristic algorithms to close gaps and delete noise edges. In our system, the details and noise are smoothed out due to the coupling structure and the salient boundaries, corners and junctions are preserved. The result shown in Figure 3.9(c) is comparable with the result after several post-processing steps shown on page 43 of the book.

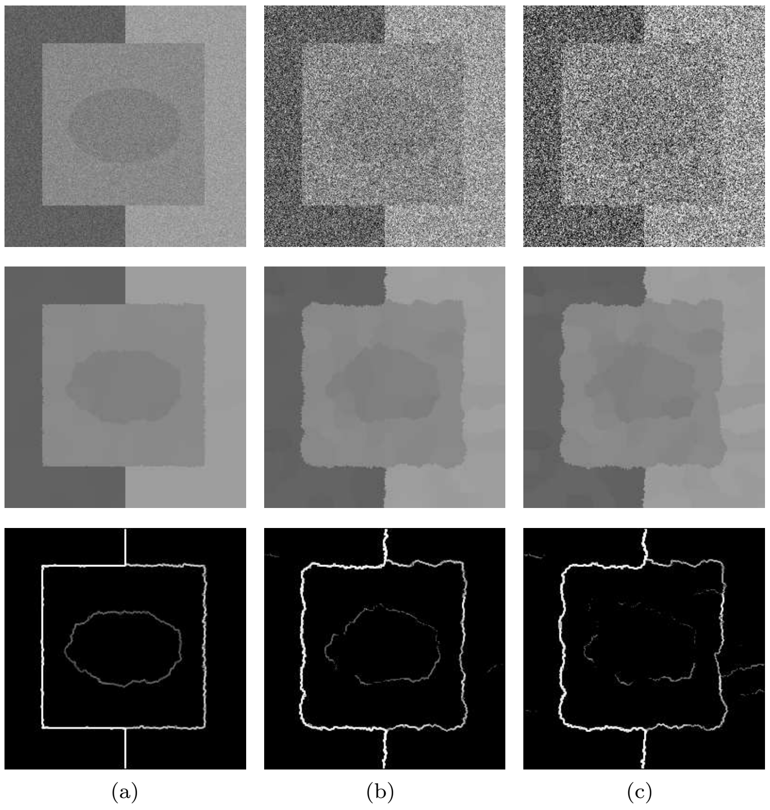


Figure 3.7: The smoothed images at the 11th iteration and detected boundaries for three synthetic images by adding specified Gaussian noise to the noise-free image shown in Figure 3.1(a). Top row shows the input images, middle the smoothed image at the 11th iteration, and bottom the detected boundaries using the Sobel edge detector. (a) Gaussian noise with $\sigma = 10$. (b) Gaussian noise with $\sigma = 40$. (c) Gaussian noise with $\sigma = 60$.

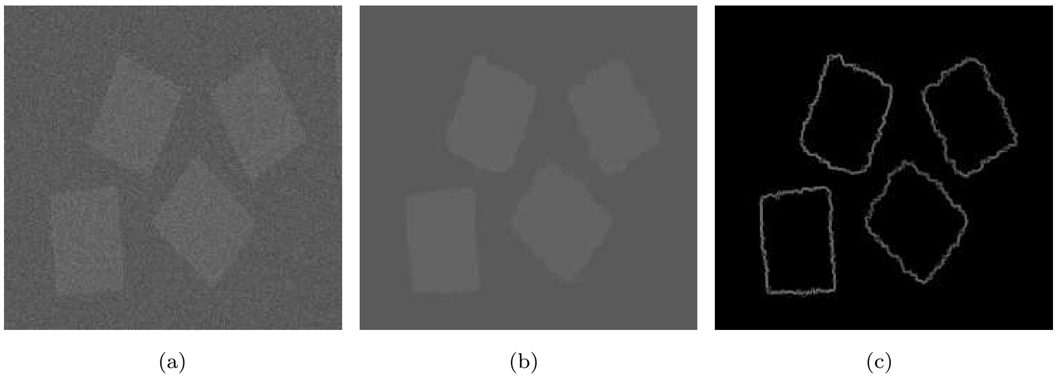


Figure 3.8: The smoothed image at the 11th iteration and detected boundaries for a synthetic image with corners. (a) Input image. (b) Smoothed image. (c) Boundaries detected.

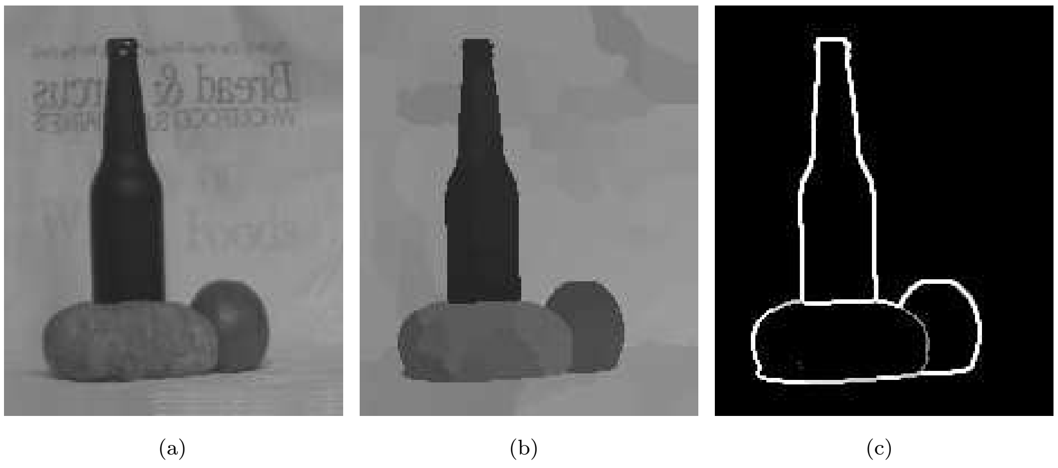
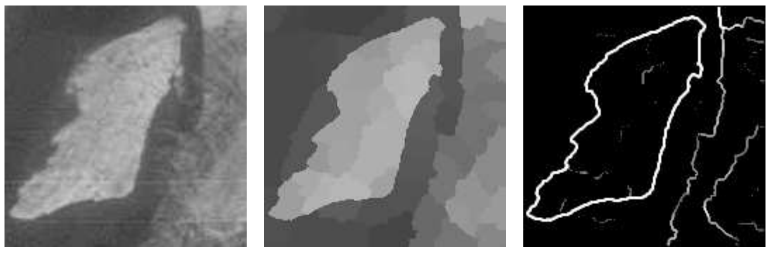


Figure 3.9: The smoothed image at the 11th iteration and detected boundaries for a grocery store advertisement. Details are smoothed out while major boundaries and junctions are preserved accurately. (a) Input image. (b) Smoothed image. (c) Boundaries detected.



(a) (b) (c)

Figure 3.10: The smoothed image at the 11th iteration and detected boundaries for a natural satellite image with several land use patterns. The boundaries between different regions are formed from noisy segments due to the coupling structure. (a) Input image. (b) Smoothed image. (c) Boundaries detected.

Figure 3.10 shows a high resolution satellite image of a natural scene, consisting of a river, soil land, and a forest. As shown in Figure 3.10(b), the river boundary which is partially occluded by the forest is delineated. The textured forest is smoothed out into a homogeneous region. The major boundaries between different types of features are detected correctly.

Figure 3.11 shows an image of a woman which includes detail features and shading effects, the color version of which was used by Zhu and Yuille [151]. In their region competition algorithm, Zhu and Yuille [151] used a mixture of Gaussian model. A non-convex energy function consisting of several constraint terms was formulated under Bayesian framework. The algorithm, derived using variational principles, is guaranteed to converge to only a local minimum. For our nonlinear algorithm, as shown in Figure 3.11(b), the details are smoothed out while important boundaries are preserved. The final result in Figure 3.11(c) is comparable with the result from the region competition algorithm [151] applied on the color version after 130 iterations. Compared with the region competition algorithm, the main advantage of our approach

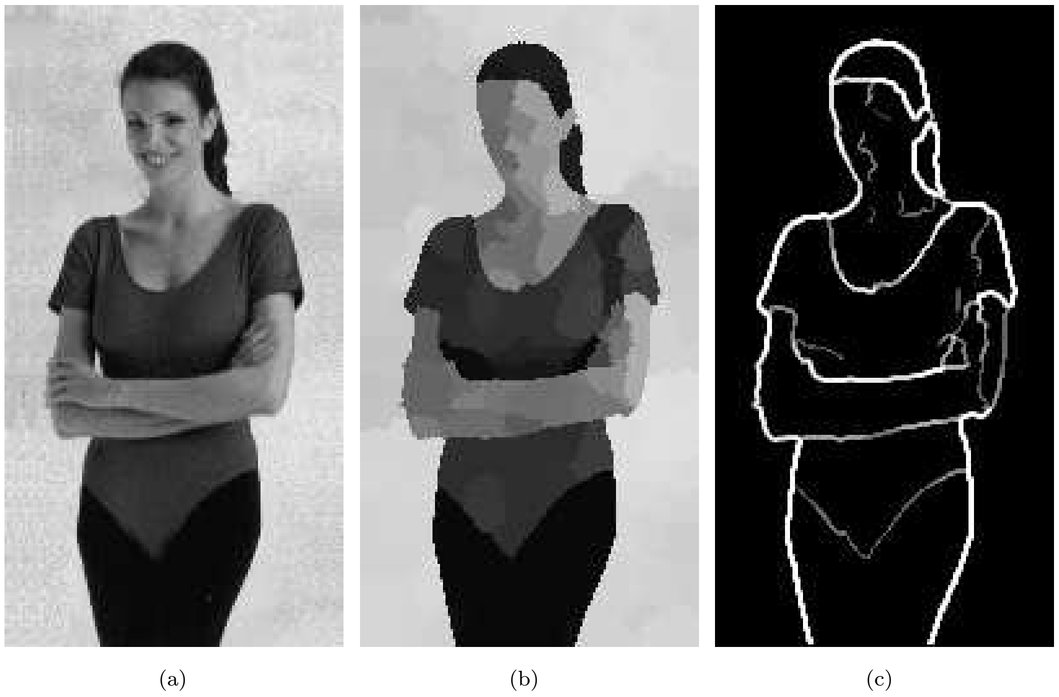


Figure 3.11: The smoothed image at the 11th iteration and detected boundaries for a woman image. While the boundaries between large features are preserved and detected, detail features such as facial features are smoothed out. (a) Input image. (b) Smoothed image. (c) Boundaries detected.

is that local statistical properties are extracted and utilized effectively in the oriented probabilistic framework instead of fitting the image into a global model which, in general, cannot be guaranteed to fit the given data well.

To further demonstrate the effectiveness of the proposed algorithm, we have also applied it to a texture image as shown in Figure 3.12(a). As shown in Figure 3.12(b), the boundaries between different textures are preserved while most of detail features are smoothed out. Figure 3.12(c) shows the detected boundaries by the Sobel edge detector. While there are some noisy responses due to the texture patterns, the main detected boundaries are connected. A simple region growing algorithm would segment

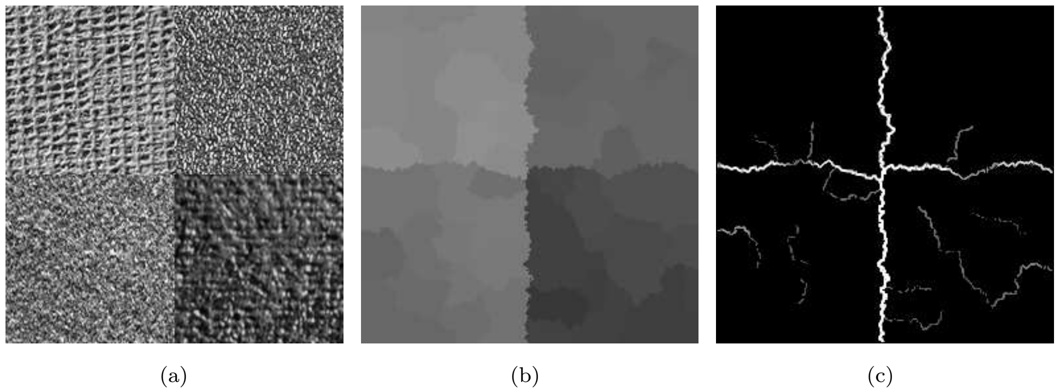


Figure 3.12: The smoothed image at the 11th iteration and detected boundaries for a texture image. The boundaries between different textured regions are formed while details due to textures are smoothed out. (a) Input image. (b) Smoothed image. (c) Boundaries detected.

the smoothed image into four regions. While this example is not intended to show that our algorithm can process texture images, it demonstrates that the proposed algorithm can be generalized to handle distributions that are not Gaussian, which was assumed when formalizing the algorithm.

3.4.2 Comparison with Nonlinear Smoothing Algorithms

In order to evaluate the performance of the proposed algorithm relative to existing nonlinear smoothing methods, we have conducted a comparison with three recent methods. The SUSAN nonlinear filter [117] has been claimed to be the best by integrating smoothing both in spatial and brightness domains. The original anisotropic model by Perona and Malik [105] is still widely used and studied. The edge-enhancing anisotropic diffusion model proposed by Weickert [137] [138] incorporates true anisotropy using a diffusion tensor calculated from a Gaussian kernel, and is probably by far the most sophisticated diffusion-based smoothing algorithm.

To do an objective comparison using real images is difficult because there is no universally accepted ground truth. Here we use synthetic images where the ground truth is known and the deviation calculated by (19) gives an objective measure of the quality of smoothed images. We have also tuned parameters to achieve best possible results for the methods to be compared. For the SUSAN algorithm, we have used several different values for the critical parameter T in (3.15). For the Perona and Malik model, we have tried different nonlinear functions g in (3.1) with different parameters. For the Weickert model, we have chosen a good set of parameters for diffusion tensor estimation. We in addition choose their best results in terms of deviation from the ground truth, which are then used for boundary detection.

Because the three methods and proposed algorithm all can be applied iteratively, first we compare their temporal behavior. We apply each of them to the image shown in Figure fig:context-syn-3-mine(b) for 1000 iterations and calculate the deviation and relative variance with respect to the number of iterations using (3.19) and (3.20). Figure 3.13 shows the deviation from the ground-truth image. The SUSAN filter, which quickly reaches a best state, and converges quickly also to a uniform state due to the Gaussian smoothing term in the filter (see equation (3.15)). The temporal behavior of the Perona-Malik model and the Weickert model is quite similar while the Weickert model converges more rapidly to and stay longer in good results. The proposed algorithm converges and stabilizes quickly to a non-uniform state, and thus the smoothing can be terminated after several iterations.

Figure 3.14 shows the relative variance of the four methods along the iterations. Because the SUSAN algorithm converges to a uniform stable state, the relative variance goes to zero after a number of iterations. The relative variance of Perona-Malik

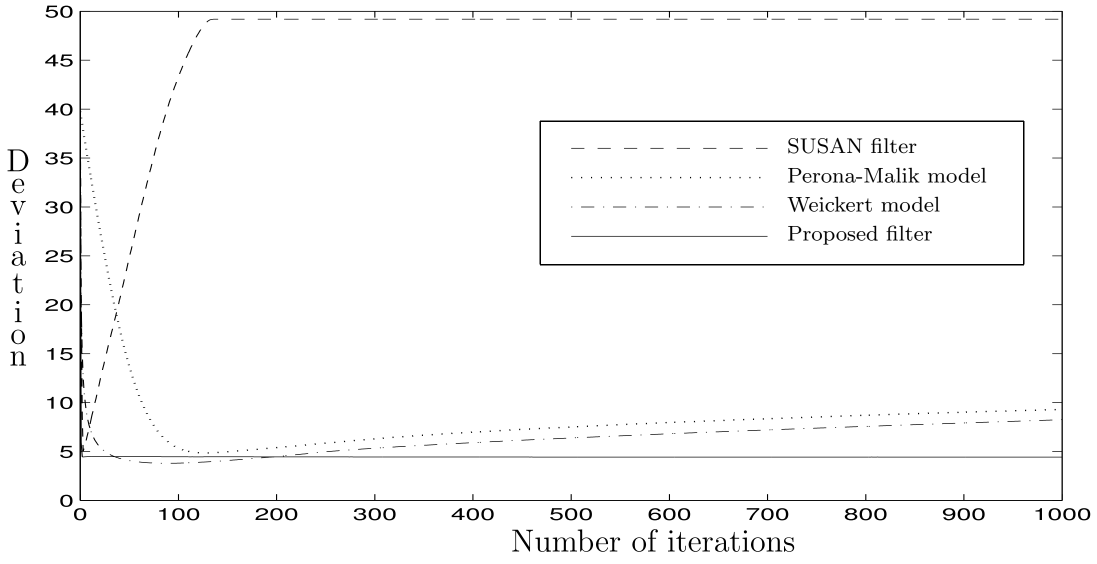


Figure 3.13: Deviations from the ground truth image for the four nonlinear smoothing methods. Dashed line: The SUSAN filter [117]; Dotted line: The Perona-Malik model [105]; Dash-dotted line: The Weickert model of edge enhancing anisotropic diffusion [137]; Solid line: The proposed algorithm.

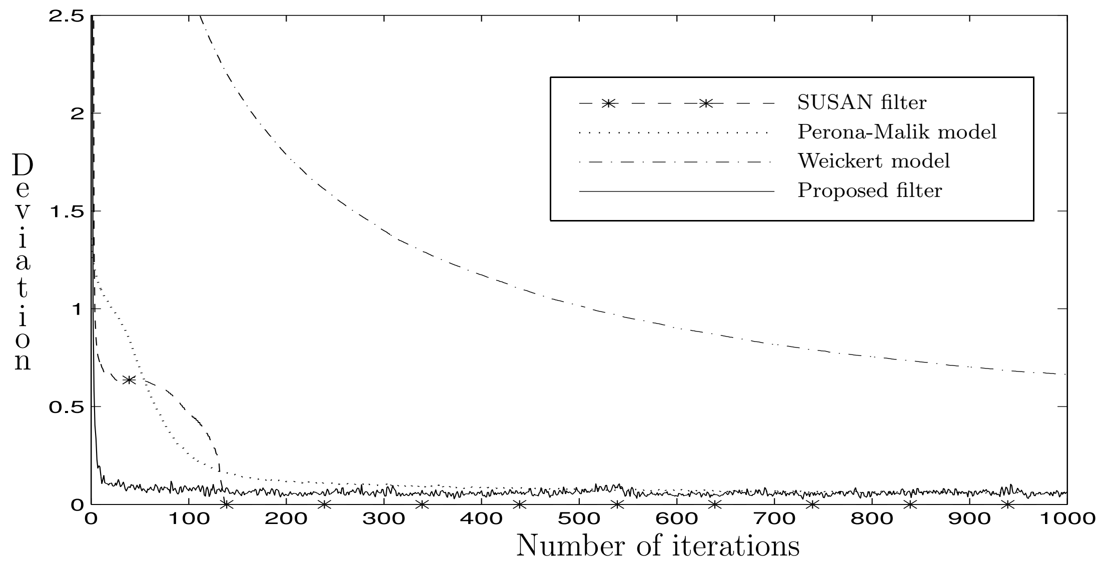


Figure 3.14: Relative variance of the four nonlinear smoothing methods. Dashed line: The SUSAN filter [117]; Dotted line: The Perona-Malik diffusion model [105]; Dash-dotted line: The Weickert model [137]; Solid line: The proposed algorithm.

model is closely related to the g function in (3.1). Due to the spatial regularization using a Gaussian kernel, Weickert model changes continuously and the diffusion lasts much longer, which accounts for the fact why good results exist for a longer period of time than Perona-Malik model. As shown in Figure 3.4 and 3.5, the proposed algorithm generates bounded small ripples in the relative variance measure. Those ripples do not affect smoothing results noticeably as the deviation from the ground truth, shown in Figure 3.13, is stabilized quickly.

Now we compare the effectiveness of the four methods in preserving meaningful boundaries. Following Higgins and Hsu [47], we use two quantitative performance metrics to compare the edge detection results: $P(AE|TE)$, the probability of a true edge pixel being correctly detected by a given method; $P(TE|AE)$, the probability of a detected edge pixel being a true edge pixel. Due to the uncertainty in edge localization, a detected edge pixel is considered to be correct if it is within two pixels from ground-truth edge points using the noise-free image. For each method, the threshold on the gradient magnitude of the Sobel edge detector is adjusted to achieve a best trade-off between detecting true edge points and rejecting false edge points.

For the proposed algorithm, we use the result at the 11th iteration because the proposed algorithm converges within several iterations. As mentioned before, for the other three methods, we tune critical parameters and choose the smoothed images with the smallest deviation. Figure 3.15 shows the smoothed images along with the detected boundaries using the Sobel edge detector, for the image shown in Figure fig:context-syn-3-mine(a), where added noise is Gaussian with zero mean and $\sigma = 10$. Table 3.1 summarizes the quantitative performance metrics. All of the four methods perform well and the proposed method gives the best numerical scores. The boundary of the square is preserved accurately. For the central oval, the proposed algorithm gives a better connected boundary while the other three have gaps. Also the proposed algorithm generated the sharpest edges while edges from Weickert model are blurred most, resulting in the worst numerical metrics among the four methods.

Figure 3.16 shows the result for the image in Figure 3.7(b), where noise is substantial, and Table 3.2 shows the quantitative performance metrics. As shown in Figure 3.16(a), the SUSAN filter tends to fail to preserve boundaries, resulting in noisy boundary fragments. The Perona-Malik model produces good but fragmented boundaries. Due to that only local gradient is used, the Perona-Malik model is noise-sensitive and thus generates more false responses than other methods in this case.

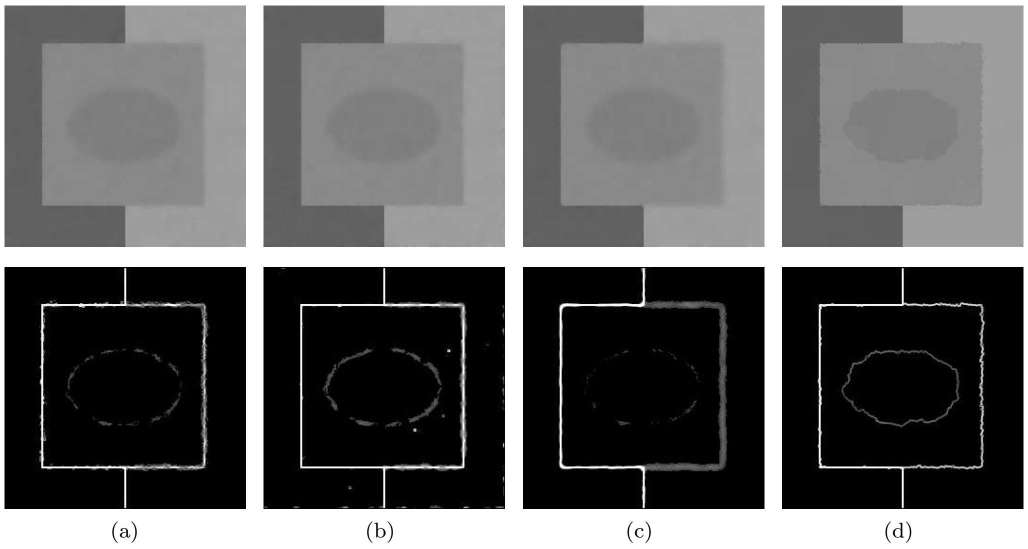


Figure 3.15: Smoothing results and detected boundaries of the four nonlinear methods for a synthetic image shown in Figure 3.7(a). Here noise is not large and all of the methods perform well in preserving boundaries.

The false responses substantially lower the quantitative metrics of the model, making it the worst among the four methods. The Weickert model produces good boundaries for strong segments but weak segments are blurred considerably. The proposed algorithm preserves the connected boundary of the square and partially fragmented boundaries of the central oval also, yielding the best numerical metrics among the four methods. As shown in Figure 3.13, the smoothed image of the Weickert model has a smaller deviation than the result from our algorithm, but the detected boundaries are fragmented. This is because our algorithm produces sharp boundaries, which induce larger penalties according to (3.19) when not accurately marked.

Comparing Tables 3.1 and 3.2, one can see that our proposed method is most robust in that the average performance is degraded by about 13%. Perona-Malik model is most noise-sensitive, where the performance is degraded by about 35%. For

Models	SUSAN[117]	Perona-Malik[105]	Weickert[137][138]	Our method
$P(TE \mid AE)$	0.960	0.963	0.877	0.988
$P(AE \mid TE)$	0.956	0.964	0.880	0.979
Average	0.958	0.963	0.878	0.983

Table 3.1: Quantitative comparison of boundary detection results shown in Figure 3.15.

Models	SUSAN[117]	Perona-Malik[105]	Weickert[137][138]	Our method
$P(TE \mid AE)$	0.720	0.609	0.692	0.853
$P(AE \mid TE)$	0.713	0.618	0.688	0.854
Average	0.717	0.613	0.690	0.853

Table 3.2: Quantitative comparison of boundary detection results shown in Figure 3.16.

SUSAN and Weickert model, the average performance is degraded by about 24% and 19% respectively.

We have also applied the four methods on the natural satellite image shown in Figure 3.10. The result from the proposed algorithm is at the 11th iteration as already shown in Figure 3.10. The results from the other three methods are picked up manually for best possible results. Due to the termination problem, results from most nonlinear smoothing algorithms have to be chosen manually, making them difficult to be used automatically. The results from the other three methods are similar, and boundaries between different regions are not formed. In contrast, our algorithm generated connected boundaries separating major different regions.

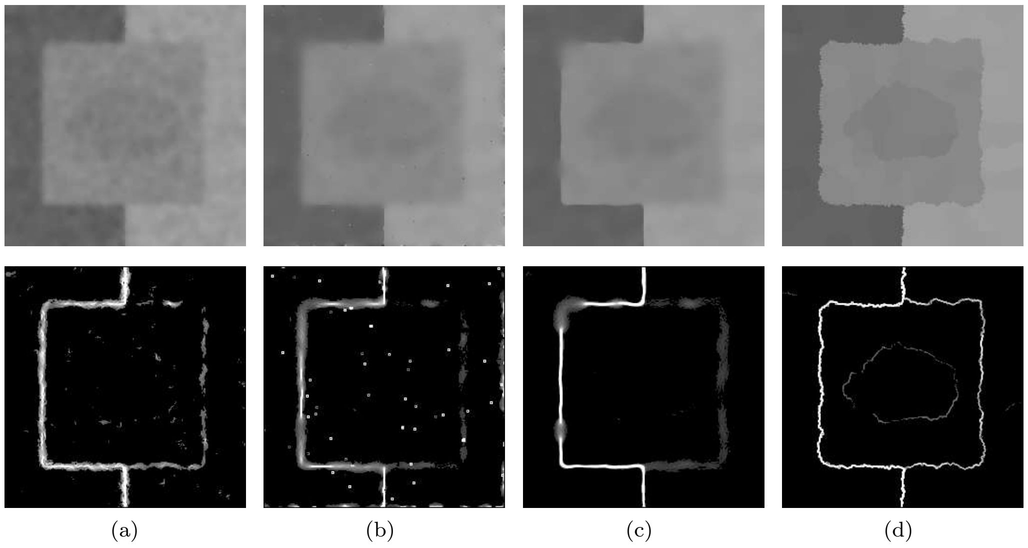

(a) (b) (c) (d)

Figure 3.16: Smoothing results and detected boundaries of the four nonlinear methods for a synthetic image with substantial noise shown in Figure 3.7(b). The proposed algorithm generates sharper and better connected boundaries than the other three methods.

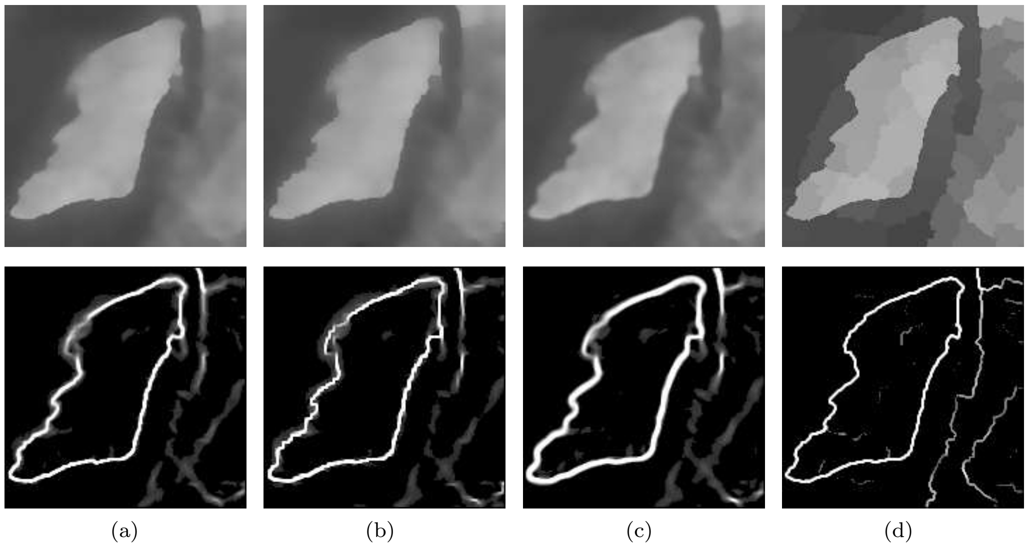

(a) (b) (c) (d)

Figure 3.17: Smoothing results and detected boundaries of a natural scene satellite image shown in Figure 3.10. Smoothed image of the proposed algorithm is at the 11th iteration while smoothed images of the other three methods are chosen manually. While the other three methods generate similar fragmented boundaries, the proposed algorithm forms the boundaries between different regions due to its coupling structure.

3.5 Conclusions

In this chapter we have presented a two-step robust boundary detection algorithm. The first step is a nonlinear smoothing algorithm based on an orientation sensitive probability measure. This algorithm is motivated by the orientation sensitivity of cells in the visual cortex [53]. By incorporating geometrical constraints through the coupling structure, the algorithm is robust to noise while preserving meaningful boundaries. Even though the algorithm was formulated based on Gaussian distribution, it performs well for real and even textured images, showing the generalization capability of the algorithm. It is also easy to see that the formalization of the proposed algorithm would extend to other known distribution by changing equations (3.3)-(3.5) accordingly. One such an extension would be to use a mixture of Gaussian distributions [30] so that the model may be able to describe arbitrary probability distribution.

Compared with recent anisotropic diffusion methods, our algorithm approaches a non-uniform stable state and reliable results can be obtained after a fixed number of iterations. In other words, it provides a solution for the termination problem. When noise is substantial, our algorithm preserves meaningful boundaries better than the diffusion-based methods, because the coupling structure employed is more robust than pair-wise coupling structure.

Scale is an intrinsic parameter in machine vision as interesting features may exist only in a limited range of scales. Scale spaces based on linear and nonlinear smoothing kernels do not represent semantically meaningful structures explicitly [122]. A solution to the problem could be to use parameter K in equation (3.8) as a control parameter [113], which is essentially a threshold in gray values. Under this formalization, (3.12) could offer an adaptive parameter selection. With the robust coupling structure, our algorithm with adaptive parameter selection may be able to provide a robust multiscale boundary detection method.

Another advantage of the probability measure framework is that there is no need to assume a priori knowledge about each region, which is necessary in relaxation labeling [112] [55] and the comparison across windows with different sizes and shapes is feasible. This could lead to an adaptive window selection that preserves small but important features which cannot be handled well by the current implementation.

There is one intrinsic limitation common to many smoothing approaches including our proposed one. After smoothing, the available feature is the average gray value, resulting in loss of information for further processing. One way to overcome this problem is to apply the smoothing in feature spaces derived from input images [139]. Another disadvantage of the proposed algorithm is relatively intensive computation due to the use of oriented windows. Each oriented window takes roughly as long in one iteration as the edge-enhancing diffusion method [137]. On the other hand, because our algorithm is entirely local and parallel, computation time would not be a problem on parallel and distributed hardware. Computation on serial computers could be reduced dramatically by decomposing the oriented filters hierarchically so that oriented windows would be used only around discontinuities rather than in homogeneous regions. The decomposition techniques for steerable and scalable filters [104] could also help to reduce the number of necessary convolution kernels.