Survey of Edge Detection

Ting-jen Yen
Courant Institute of Mathematical Sciences
New York University

September 9, 2002

1 Introduction

Edge detection is one of the fundamental tasks in computer vision. Many high level image processing, such as object recognition, 3-dimensional reconstruction, motion detection, depend on the accuracy of edge detection.

(a) Ramp Profile (b) Ridge Profile

Figure 1: (a) Ramp Profile (b) Ridge Profile

Edge point detection could be considered as the first step of edge detection. We define **edge points** as the pixels in an image with sharp change of colors and/or intensity, and **edges** as a one-dimensional curve on an image that passes through nearby edge points. A simple monochromatic edge has constant **profile**, some examples are the **ramp** and the **ridge** (see Figure 1). It is important to keep the above terminology clear, as the literature do not always make the kinds of distinction we propose here. Most edge point detection techniques are based on finding the maxima of the first derivative of the image, or the zero crossings of the second derivative of the image. As seen in Figure 2.

Edge detection is a very complex subject and most text books in computer vision and image processing such as [GW92, RW00, Par96] and many papers on

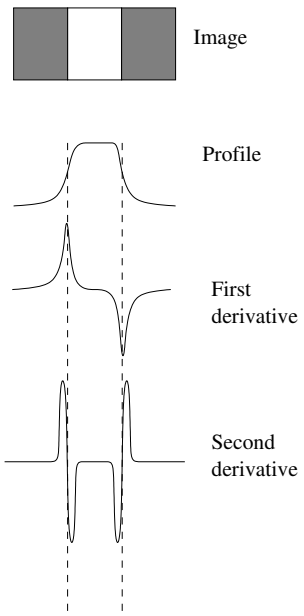


Figure 2: Edge detection by derivative operators

image feature extraction have overviewed and compared different edge detectors. Some surveys can be found in [Dav75, Kos95, ZT97]. This paper surveys some methods of edge point detection and edge detection methods for raster images. The mathematical developments in Canny's theory [Can86] and active contours [KWT88] are reviewed in detail. Some necessary mathematical techniques will be reviewed in appendices.

2 Edge Point Detection

Most of the work on edge point detection are based on gray-scale images. In a gray-scale image, an edge is defined by intensity discontinuity. For a color image, it could get more complicated, especially in the area with low contrast [Kos95]. Though generally we could apply the same techniques on gray-scale images and color image, with proper adjustments. We will discuss color images in the Section 4.

Traditional edge point detection is usually composed of three operations: smoothing, differentiation, and labeling. The purpose of smoothing is to reduce noise. The purpose of differentiation is to evaluate the desired derivatives of the image. Labeling amounts to locating the edge points and suppressing false edge points.

2.1 Smoothing

All images are subject to noise of some type, except for some simple synthetic images, so there is no point in ignore it.

There are two types of noise in respect of image analysis. **Signal-independent noise** is a set of grey levels, statistically independent to the image data, added to the the pixels in the image to form the result noisy data. That is, the noisy image $I_N = I_0 + N$, where I_0 is the noise-free image, and N is the noise function. In other word, this type of noise is additive. The noise could have any kind of statistic properties, though it generally has mean of zero, and some presume standard deviation. The noise comes from optical device is generally this type of noise.

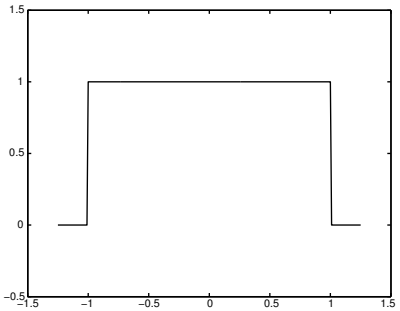
The second type of noise is called **signal-dependent noise**. The noise value of each pixel is a function of the pixel value. The image degradation in television lines and grain in photographs are examples of this kind of noise. It is generally harder to deal with. Two kind of the most common signal-independent noise are multiplicative noise, which results in a noisy image $I_N = I_0 + kI_0n$, and film-grain noise which yields a noisy image $I_N = I_0 + kI_0^y n$, where $I_)$ the noise-free image, k and y constant parameters, and n the noise function. Homomorphic filtering has been used for the removal of multiplicative [AO68] and film-grain noise [HA83]. The basic idea is to perform a logarithmic operation over the image to make noise additive, so we could decouple the noise from the image as if it is signal-independent.

In general, if the source of the noise of an image is not specified, it is usually assumed that the image is corrupted by signal-independent noise, with zero-mean Gaussian distribution. Such noise is called **Gaussian noise.**

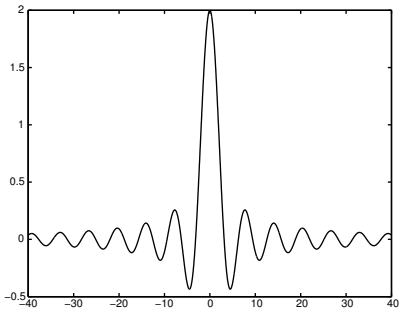
Smoothing is used to reduce noise in edge point detection and to ensure robust detection. Generally, smoothing is achieved by filtering the image with a low-pass filter. It could reduce the signal-independent noise because smoothing is generally taking some form of averaging locally for every pixel. By taking average, the noise will be damped because of the nature of zero mean of noise. In the view point of frequency domain, because the noise is usually high frequency signal, smoothing filter, as a low-pass filter, would efficiently remove the noise by filtering away the high frequency data. On the other hand, the edge information is also considered as high frequency signal, and would be removed as well. This is a trade-off between loss of information and noise reduction.

Another purpose of smoothing, is to provide a scaling factor in multi-scale approach. We will discuss about multi-scale approach of edge point detection later in section 3.6.

A common technique for noise smoothing is **linear filtering**, which consists in convolving the image with a constant matrix called **mask** or **kernel**. For an $N \times M$ image I, and a kernel of a linear filter A, which is a $m \times m$ matrix, the



(a) an 1-D averaging filter with width 2



(b) sinc function: the Fourier transform of 1-D averaging filter with width 2

Figure 3: Spatial domain and frequency domain of an 1-D averaging filter

filtered image I_A will be defined as

$$I_A(i,j) = I * A = \sum_{h=-\frac{m}{2}}^{\frac{m}{2}} \sum_{k=-\frac{m}{2}}^{\frac{m}{2}} A(h,k)I(i-h, j-k).$$

The simplest filter is the average smoothing filter $f_{avg}(x)$ which in 1D is shown in figure 3(a). What it does is replacing each pixel with the average of its neighborhood. This method works well for Gaussian noise, which is noise with Gaussian distribution. Though it has some problems. First, it would not only filter out the signal, but dull the sharp edges making them harder to detect. Furthermore, the accuracy of the feature localization would be affected. In the frequency domain, the Fourier transform of a one-dimensional filter kernel will be the "sinc" function which is

$$\mathrm{sinc}\,\omega = \frac{2\sin\omega W}{\omega}.$$

As shown in Fig 3(b), the sinc function has secondary lobes which will let noise into the filtered image, assuming that noise is generally in the high frequency zone.

Gaussian smoothing filter $f_{Gauss}(x) = e^{-\frac{x^2}{\sigma^2}}$, is another commonly chosen filter. Since the Fourier transform of a Gaussian is $F\omega = \sqrt{2\pi}\sigma e^{-\sigma^2\omega^2/2}$, which is also a Gaussian, there are no secondary lobes, therefore, it makes a better choice of smoothing filter in the view of frequency domain.

In comparison to **linear filter**, there is also **nonlinear filter**. A most common example of **nonlinear filter** is median smoothing filter. The idea is to replace the current point in the image by the median of the brightnesses in its neighborhood. The median of the brightnesses in the neighborhood is not

affected by individual noise spike, so median smoothing eliminates impulsive noise quite well. Furthermore, it does not blur edges much, and can be applied repeatedly. The disadvantage of median filtering is that it will not preserve the location of thin lines and sharp corners. This can be avoid if we limit the neighborhood to the pixels on the same row or same column of the current point.

Median smoothing filter is actually a special case of **order statistics filtering** [ACB83]. Values in the neighborhood are ordered into sequence, and a new value is given as a linear combination of the values of this sequence. Other special case for **order statistics filtering** is minimum filter or maximum filter which defines generalizations of dilation and erosion operators in gray-scale images.

Another approach of smoothing is using diffusion equation. The basic idea is that if one creates a series of images $I_t(x, y, t)$ from original image $I_0(x, y)$ by convolving $I_0(x, y)$ with a Gaussian kernel $G(x, y; t)$ with variance t, as pointed out by Koenderink [Koe84] and Hummel [Hum87], this family of images can be viewed as the solution of the isotropic diffusion equation

$$\frac{dI}{dt} = \Delta I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

with the initial condition $I(x, y, 0) = I_0(x, y)$, the original image. One disadvantage of linear isotropic diffusion is that the diffusion would blur features as well as smooth noise.

In [PM90], Perona and Malik introduced a diffusion method which they describe as "anisotropic". It uses a nonlinear diffusion equation:

$$\frac{dI}{dt} = \nabla \cdot (c(x, y, t)\nabla I) = c(x, y, t)\Delta I + \nabla c \cdot \nabla I \tag{1}$$

where $c(x, y, t)$ is chosen as

$$c(x, y, t) = g(\nabla I(x, y, t)) \tag{2}$$

and g is a nonnegative monotonically decreasing function with $g(0) = 1$. Different function g would generate different scale-spaces, though they turn out to be perceptually similar. This way, the points with strong features, i.e., with high $|\nabla I|$ values, would have less diffusion effect than other points, thus retain most of the features in the image. Note that in order to be anisotropic, c(x,y,t) should be calculated separately in all four directions for a digital image. The nonlinear diffusion methods process an image with different smoothing parameters such that it could blur noise while keep the edges sharp.

2.2 Differentiation

Since edge point detection locates points of large intensity changes in the intensity of the image, differentiation, the computation of the derivatives of an image

is needed. The most commonly used operators are the gradient, the Laplacian, and the second-order directional derivative. Generally, like smoothing, they involve the calculation of each point and its neighborhood of specific shape. And unlike smoothing, differentiation operators are considered as high-pass filters in frequency domain.

2.2.1 First Derivative Operators

These operators are based on the gradient $(\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ at each point of images. For digital images, one could estimate the gradient by convolve the rows and columns with the mask $[10-1]$. Because this gradient expression neglect the impact of the point itself, sometimes, asymmetric expression of the gradient $[1-1]$ is used instead.

2.2.2 Second Derivative Operator

The second derivatives are obtained by filtering the image by the Laplacian operator Δ which is

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}.$$

The basic form of the numerical Laplacian operator in matrix mask form is

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix},$$

though some other matrices could be used as long as the center coefficient is positive, and the sum of all coefficients is zero, and it is rotationally symmetric. For example, the following two matrices are two different approximations of the Laplacian operator.

One disadvantage of second-derivative operators is that they are more sensitive to noise.

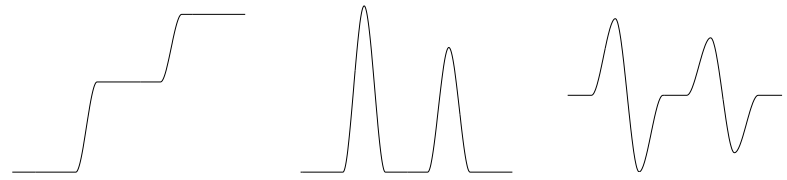
$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}, \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}.$$

Edge points correspond to position P s.t. $\Delta(P) = 0$. Such points are usually called the zero-crossings.

2.3 Edge Labeling

The goal of edge labeling is to localize edge points and suppress false edge points.

The localization depends on the differentiation operator used. The earlier gradient detectors located the edge points by thresholding the magnitude of the



(a) Staircase edge profile

(b) Two maxima and one minimum of the first derivative

(c) Three zero-crossings of the second derivative

Figure 4: Example of phantom edge

gradients. This could not achieve good result because it is not easy to find a single reasonable threshold value for the whole image. There is no reason why there should be a single threshold value of a whole image. A improvement is achieved by the use of the non-maximum suppression proposed in [Can86]. The basic idea is to find the local maxima along the direction of the gradient vector.

For second-derivative detectors, we are looking for zero-crossings. To locate the zero-crossings, we can check the a pixel with the pixels to the left and below it. If the three pixels do not have the same signs, there is a zero-crossing. In other word, second-derivative detectors are similar to gradient operators with non-maximum suppression. Though second-derivative detectors are usually more sensitive to noise.

The elimination of false edge points increases the signal-to-noise ratio of the differentiation and smoothing operations. Few works have been done on this subject. While more robust smoothing operators have been introduced, and may reduce the impact of noise in an image, however, there are other phenomena that produce false edges other than noise. For example, certain edge profiles would generate false edge points called "phantom edge points". A example is shown in 4. A most commonly used technique to eliminate false edges is thresholding. Only an edge point with a magnitude which is above a given threshold is accept as a true edge point. While thresholding works generally because the magnitudes of false edge points is usually much smaller than those of true ones, the resulting edge points from thresholding tend to be broken due to fluctuation of the measure. A improvement of thresholding technique uses hysteresis algorithm [Can86], which takes the edge continuity into account to avoid fluctuation. Two thresholds are used; only the edge points which is above a low threshold can link into edges, and each edge most have at least one point above a high threshold in order to be accepted as a true edge.

Another aspect of edge labeling is threshold computation. A threshold value is usually found by a trial-and-error process. The threshold depends on the edge characteristics, properties of the smoothing filter and the properties of the differentiation operator, therefore is very hard to find a single threshold value for a given image.

3 Survey of Edge Point Detectors

3.1 Differential-Only Detectors

The first detectors based on gradient and Laplacian operators were proposed in 1960's. These detectors are limited to the differentiation operators only.

The following are some operators that can derived directly from the gradients of the image,

Roberts Operator [Rob65] For a given image $I : Z_2 \Rightarrow Z$, the Roberts edge detection is done by filtering the image with the masks

$$M_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, M_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

to obtain two images I_1 and I_2, where

$$I_1 = I * M_1$$

$$I_2 = I * M_2.$$

Then one can get the gradient magnitude at each pixel (i, j) as

$$G(i, j) = \sqrt{I_1^2(i, j) + I_2^2(i, j)}.$$

Sobel Operator The Sobel edge detection is done by filtering every pixel with its eight neighbors with the following two 3×3 convolution masks

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

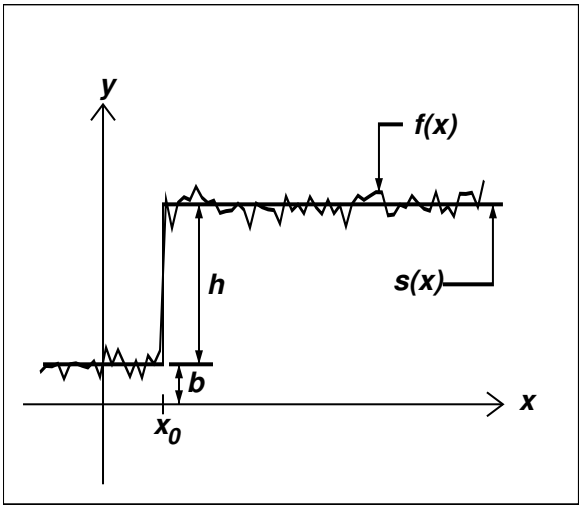
These two masks would generate two numbers for each pixel for both column and row directions. We could obtain the magnitude and orientation of the edge from the two values.

Prewitt Operator The Prewitt is similar to the Sobel, with different mask coefficients. The masks are defined as follows:

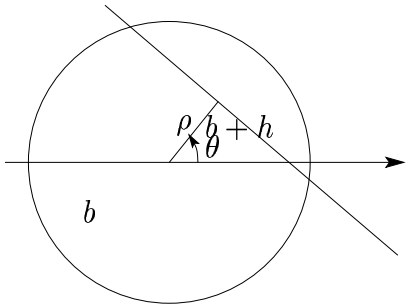
$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}.$$

Like the Sobel, this edge detection can also give the orientations of edges besides the magnitudes.

The three operators above are all linear operators, because they can be done by convolution.



(a) One-dimension edge fitting



(b) Two-dimension edge fitting

Figure 5: One- and two-dimensional edge fitting.

Note that since Sobel operator and Prewitt operator average over more pixels than the Roberts operator does, they would give better smoothing effect, and would perform better in the presence of noise.

3.2 Hueckel Edge Fitting Method

The Hueckel edge detection method is to fit image data into an ideal two-dimensional edge model [Hue69, Hue71, Hue73]. In the one dimensional case described in Figure 5(a), the image signal $f(x)$ is fitted into a step function

$$s(x) = \begin{cases} b & \text{if } x < x_0 \\ b+h & \text{if } x \geq x_0. \end{cases}$$

A edge is assumed present if the mean-square error

$$E = \int_{x_0-L}^{x_0+L} (f(x) - s(x))^2 dx$$

is below some threshold value.

In the two-dimensional formulation, the ideal step edge is defined as

$$s(x) = \begin{cases} b & \text{if } x\cos\theta + y\sin\theta < \rho \\ b+h & \text{if } x\cos\theta + y\sin\theta \geq \rho \end{cases}$$

where θ and ρ represent the polar distance from the center of a circle test region D to the normal point of the edge. See figure 5(b). The edge fitting error is

$$E = \int_D (f(x,y) - s(x,y))^2 \, dx \, dy \tag{3}$$

Hueckel has developed a procedure for two-dimensional edge fitting, in which the image points within the circle D are expanded in a set of two-dimensional basis functions by Fourier transform in polar coordinates. Let $H_i(x,y)$ represent these basis functions. Then the coefficients for the image and the ideal step edge become

$$a_i = \int\int_D H_i(x,y) f(x,y) \, dx \, dy$$

$$s_i = \int\int_D H_i(x,y) s(x,y) \, dx \, dy.$$

Note that $s(x,y)$ is defined parametrically in terms of the set (b,d,ρ,θ). In Hueckel's algorithm, the expansion is truncated to eight terms. The minimization of the mean-square difference equation (3) is equivalent to minimization of $\sum_i (a_i - s_i)^2$. Hueckel has performed this minimization, and has formulated a set of equations expressing the edge parameter set (b,h,ρ,θ) in terms of the coefficients a_i.

Hueckel's detector could handle noise well. One explanation is that by using Fourier transform and truncating the higher frequency terms, most of the noise are filtered out. On the other hand, since the ideal step edge model we use is a straight edge, the results for curvature edges might not be correct.

A problem in Hueckel's algorithms is that I cannot confirm the orthonormal property of the H basis he provided. Also, this parametric fitting process assumes that the edge within the region D is straight. If the direction of edge inside the region changes too much, the result edge parameter set might be dubious.

3.3 Canny's Edge Detection

Canny edge detector [Can86] is a combination of image smoothing, image differential, and edge labeling.

We begin by analyzing the one-dimensional case. But what is an edge? The simplest definition may be the following: a function $G(x)$ is an **edge function** if it is non-constant such that for some $c \in \mathbb{R}$, $g(x) := G(x-c) - G(c)$ is either symmetric or antisymmetric function. That is, $g(x) = g(-x)$ or $g(x) = -g(-x)$ for all x. For instance, $G(x) = 1$ for $x \geq c$ and $G(x) = 0$ else. Another example is $G(x) = |x|$ for $|x| \leq 1$ and $G(x) = 0$ else. These two examples are called the **step edge function** and **roof edge function**, respectively. We call c the **center** of the edge function. Without loss of generality, we may henceforth assume $c = 0$.

Let $S(x)$ be a one-dimensional signal and $G(x)$ be any edge function centered at $c = 0$. Fix some $W > 0$. We say that $G(x)$ **occurs in** $S(x)$ **at position** p **within a** W**-width window** if $G(x) = S(x+p)$ for all $x \in [-W, +W]$. The problem of "edge detection" is to find all occurrences of $G(x)$ in $S(x)$. In practice, W should be chosen for the particular $G(x)$ we want to "detect". In the following, we suppress the explicit reference to W.

To detect the occurrences of $G(x)$ in $S(x)$ within some W-width window, we introduce "filter functions". A function f is said to be a **filter for** $G(x)$ in a W-width window if the function

$$H_G(y) = \int_{-W}^{+W} G(y-x) f(x) dx$$

has a unique maxima at 0. Clearly, if $G(x)$ occurs in $S(x)$ at position p then $H_S(y)$ has a local maxima at p.

We must also discuss occurrences which are very close together. E.g., two occurrences cannot be closer than W apart?

In case, $G(x)$ occurs "noisily" in $S(x)$, That is, we think of the signal $S(x)$ as the sum of some pure signal $S_0(x)$ plus some random noise $n(x)$. Typically, $n(x)$ is assumed to be Gaussian. This means that $S(x)$ has a probability distribution. John Canny defines three performance criteria for a good edge detecting filter:

- Good detection. There should be a low probability of failing to mark real edge points, or falsely marking non-edge points.
- Good localization. The points marked as edge points should be as close as possible to the real edge.
- Only one response to a single edge.

3.3.1 Detection and Localization criteria

Let the edge be $G(x)$ which centers at $x = 0$. Our goal is to find the filter function $f(x)$. Then we have the response as the convolution integral:

$$H_G = \int_{-W}^{+W} G(-x) f(x) dx,$$

assuming the filter has a finite response bounded by $[-W, W]$.

The root-mean-squared response to the Gaussian noise $n(x)$ only, will be

$$H_n = n_0 \left[\int_{-W}^{+W} f^2(x) dx \right]^{1/2},$$

where n_0^2 is the mean-squared noise amplitude.

Canny defines the output signal-to-noise ratio as

$$SNR = \frac{\left| \int_{-W}^{+W} G(-x) f(x) dx \right|}{n_0 \left[\int_{-W}^{+W} f^2(x) dx \right]^{1/2}}, \tag{4}$$

which is used as the indication of how good the detection is made. The higher the SNR is, the better the detection we get.

For the second criterion, since the edge is centered at $x = 0$, there should be a local maximum in the response at $x = 0$ if there is no noise.

We have H_n as the response to the filter to noise only, and H_G its response to the edge. If there is a maximum in the total response at the point $x = x_0$. Then we have

$$H_n'(x_0) + H_G'(x_0) = 0.$$

Taylor expansion of of $H_G'(x_0)$ about the origin will give

$$H_G'(x_0) = H_G'(0) + H_G''(0) x_0 + O(x_0^2).$$

By assumption $H_G'(0) = 0$, so the first term could be ignore. Then we assume that the displacement x_0 is very small so we could ignore the higher term. So we have

$$H_G''(0) x_0 \approx -H_n'(x_0).$$

Since $H_n(x)$ is the response of the Gaussian noise, $H_n'(x_0)$ is a Gaussian random quantity whose variance is the mean-squared value of $H_n'(x_0)$, so the expectation value of $H_n'(x_0)^2$ is

$$E[H_n'(x_0)^2] = n_0^2 \int_{-W}^{+W} f'^2(x) dx.$$

Therefore,

$$\begin{aligned} E[x_0^2] &\approx -\frac{H_n'(x_0)}{H_G''(0)} \\ &= \frac{n_0^2 \int_{-W}^{+W} f'^2(x)}{\left[\int_{-W}^{+W} G'(-x) f'(x) dx \right]^2}. \end{aligned}$$

The localization is then defined as the reciprocal of $E[x]$.

$$Localization = \frac{\left|\int_{-W}^{+W} G'(-x)f'(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)dx}}. \tag{5}$$

Since we want to maximize both the SNR and the localization, one way to achieve this is by maximizing the product of both. There are some other forms to combine both arguments, but the use of the product will simplify the analysis. Therefore we seek to maximize

$$\frac{\left|\int_{-W}^{+W} G(-x)f(x)dx\right|}{n_0\left[\int_{-W}^{+W} f^2(x)dx\right]^{1/2}} \frac{\left|\int_{-W}^{+W} G'(-x)f'(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)}} \tag{6}$$

3.3.2 Eliminating Multiple Responses

If we do not add any more constraints other than the criteria of SNR and localization, then we could maximize these two terms easily by getting the upper bound of the two arguments. From Schwarz inequality for integrals, we can have the SNR bounded above by

$$n_0^{-1}\sqrt{\int_{-W}^{+W} G^2(x)dx}$$

and localization by

$$n_0^{-1}\sqrt{\int_{-W}^{+W} G'^2(x)dx}$$

We can get both maximized by having $f(x) = G(-x)$ in $[-W, W]$.

So the optimal detector for step edges is a truncated step, or difference operator. The problem is, the response of this detector to a noisy edge will have many local maxima.

Therefore, there is a third criterion, which is a constraint, that limits the number of peaks in the response. Actually, we would like to make distance between peaks.

To make this into a functional constraint on f, we need to obtain an expression for distance between adjacent peaks. First, we note that the mean distance between adjacent maxima in the output is twice the distance between adjacent zero-crossings in the first derivative of the operator output. The according to Rice[Ric44], the average distance between zero-crossings of the response of a function g to Gaussian noise is

$$x_{avg} = \pi\left(\frac{-R(0)}{R''(0)}\right)^{1/2}$$

where $R(t)$ is the autocorrelation function of g. That is,

$$R(t) = \int_{-\infty}^{+\infty} g(x)g(x-t)dx$$

Now since

$$R(0) = \int_{-\infty}^{+\infty} g^2(x)dx$$

and

$$R''(0) = -\int_{-\infty}^{+\infty} g'^2(x)dx,$$

the mean distance between zero-crossings of f' will be

$$x_{avg}(f) = \pi \left(\frac{\int_{-\infty}^{+\infty} f'^2(x)dx}{\int_{-\infty}^{+\infty} f''^2(x)dx} \right)_{1/2} \tag{7}$$

The average distance between adjacent maxima in the noise response of f, x_{max}, will be twice x_{avg}. We set this distance to be some fraction k to the operator width W.

$$x_{max}(f) = 2x_{avg}(f) = kW.$$

Because the response of the filter will be concentrated in a region of width 2W, the expected number of noise maxima in the region will be

$$\frac{2W}{x_{max}} = \frac{2}{k}.$$

Now we add a scaling factor w to f, s.t. $f_w(x) = f(x/w)$. We will note that $x_{avg}(f_w) = wx_{avg}(f)$. Therefore if function f satisfies the multiple response constraint for fixed k, any scaling of f will, assume W scales with w.

3.3.3 A Detector for Step Edges

In previous sections, we did not define the shape of the edge function $G(x)$. However, it is very difficult to find a closed form of the function f which maximizes (6) while satisfying the multiple response constraint. We will now specialize the case where the input $G(x)$ is a step edge. That is,

$$G(x) = \begin{cases} 0, & \text{for } x < 0; \\ A, & \text{for } x \geq 0; \end{cases}$$

and substituting for $G(x)$ in (4) and (5) gives

$$SNR = \frac{A\left|\int_{-W}^{0} f(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f^2(x)dx}},$$

$$Localization = \frac{A\,|f'(0)|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)dx}}.$$

We now remove the constant A/n_0 and define two performance measures Σ and Λ which depend on the filter f only:

$$SNR = \frac{A}{n_0}\Sigma(f) \text{ where } \Sigma(f) = \frac{\left|\int_{-W}^{0} f(x)dx\right|}{\sqrt{\int_{-W}^{+W} f^2(x)dx}}; \tag{8}$$

$$Localization = \frac{A}{n_0}\Lambda(f') \text{ where } \Lambda(f') = \frac{|f'(0)|}{\sqrt{\int_{-W}^{+W} f'^2(x)dx}}.. \tag{9}$$

Suppose we form a spatially scaled filter f_w from f, where $f_w(x) = f(x/w)$. When we substitute f_w into (8) and (9), we get

$$\Sigma(f_w) = \sqrt{w}\Sigma(f) \text{ and } \Lambda(f'_w) = \frac{1}{\sqrt{w}}\Lambda(f'). \tag{10}$$

The first equation means a filter with broad response would have better signal-to-noise ratio. The second equation means a narrow filter would give better localization. Through spatial scaling, we trade off detection performance against localization. This suggests that the product of (8) and (9) is a pretty good choice for the measure of the first two criteria.

$$\Sigma(f)\Lambda(f) = \frac{\left|\int_{-W}^{0} f(x)dx\right|}{\sqrt{\int_{-W}^{+W} f^2(x)dx}} \frac{|f'(0)|}{\sqrt{\int_{-W}^{+W} f'^2(x)dx}}. \tag{11}$$

According (10), we note that the solutions to (11) will be a class of functions related by spatial scaling.

Assuming the function f is antisymmetric, since the step function is, too. we can change all the limits of the two integrals in the denominator of (11) from $-W$ and $+W$ to $-W$ and 0. The result value would be half the one from the full range. This will make all the integrals isoperimetric. It also makes the term $f'(0)$ as a boundary condition instead of the term $\int_{-W}^{0} f''(x)dx$. Now we have isoperimetric constraint condition[CH53], we could use Lagrange multipliers to combine the all the functionals together by making all but one of the integrals constant. For details about Lagrange multipliers for constraint optimization, see Appendix A.2.

We seek some function f which will minimizes the integral

$$\int_{-W}^{0} f^2(x)dx$$

under the condition

$$\int_{-W}^{0} f(x)dx = c_1$$

$$\int_{-W}^{0} f'^2(x)dx = c_2$$

$$\int_{-W}^{0} f''^2(x)dx = c_3$$

$$f'(0) = c_4. \tag{12}$$

Note that the f'' term is here because of the multiple response constraint in equation (7). The other terms are from (11). We have some boundary conditions. $f(0) = 0$ because of its antisymmetric property. $f(-W) = 0$ because by definition f has finite response bounded by [-W,W] and f has to be continuous.

By applying Lagrange multipliers, we form a functional that is a linear combination of the functions that to be minimized. The functional is

$$\Psi(x, f, f', f'') = f^2 + \lambda_1 f_2' + \lambda_2 f_2'' + \lambda_3 f. \tag{13}$$

The Euler equation is

$$\Psi_f - \frac{d}{dx}\Psi_{f'} + \frac{d^2}{d^2x}\Psi_{f''} = 0.$$

See Appendix A.1 for the method of Euler equation for solving variational problems.

We substitute Ψ from (13) in Euler equation and would get:

$$2f(x) - 2\lambda_1 f''(x) + 2\lambda_2 f''''(x) + \lambda_3 = 0. \tag{14}$$

The solution of this differential equation is in the form of $e^{\gamma x} + c$ where c is a constant. Now γ must satisfy

$$2 - 2\lambda_1\gamma^2 + 2\lambda_2\gamma^4 = 0$$

so

$$\gamma^2 = \frac{\lambda_1}{2\lambda_2} \pm \frac{\sqrt{\lambda_1^2 - 4\lambda_2}}{2\lambda_2}. \tag{15}$$

This equation may have roots that are purely imaginary, purely real, or complex depends on the values of λ_1 and λ_2. We can tell that λ_2 should be

positive because f''^2 is to be minimized. We still could determine the roots of the equation.

We next derive the second variation δ^2 of the functional (see [[CH53], p.214]). First let

$$J[f] = \int_{x_0}^{x_1} \Psi(x, f, f', f'')dx.$$

By Taylor's theorem,

$$J[f + \epsilon g] = J[f] + \epsilon J_1[f, g] + \frac{1}{2}\epsilon^2 J_2[f + \sigma g, g]$$

where σ is a number between 0 and ϵ, g is an arbitrary admissible function and is 0 where $x = x_0$ and x_1, and

$$J_1[f, g] = \int_{x_0}^{x_1} \Psi_f g + \Psi_{f'} g' + \Psi_{f''} g'' dx$$

$$\begin{aligned} J_2[f, g] \quad = \quad & \int_{x_0}^{x_1} \Psi_{ff} g^2 + \Psi_{f'f'} g'^2 + \Psi_{f''f''} g''^2 \\ & + 2\Psi_{ff'} gg' + 2\Psi_{f'f''} g'g'' + 2\Psi_{ff''} gg'' dx. \end{aligned} \tag{16}$$

Using the fact g is an admissible function, we could have

$$\int_{x_0}^{x_1} \Psi_{f'} g' dx = \Psi_{f'} g|_{x_0}^{x_1} - \int_{x_0}^{x_1} g \frac{d}{dx} \Psi_{f'} dx = - \int_{x_0}^{x_1} g \frac{d}{dx} \Psi_{f'} dx.$$

Similarly, we have

$$\int_{x_0}^{x_1} \Psi_{f''} g'' dx = \int_{x_0}^{x_1} g \frac{d}{dx} \Psi_{f''} dx.$$

Therefore, J_1 is just the integral of g times the Euler equation of f and will be zero. Finally, define the second variation $\delta^2 J$ as

$$\delta^2 J = \frac{\epsilon^2}{2} J_2[f, g].$$

Since we are looking for minimum, the second derivatives of Ψ

$$J_2[f] \geq 0,$$

which gives

$$\delta^2 J \geq 0.$$

We compute the second derivatives of Ψ from (13):

$$J_2[f, g] = \int_{x_0}^{x_1} 2g^2 + 2\lambda_1 g'^2 + 2\lambda_2 g''^2 dx \geq 0.$$

Given the fact that g is an admissible function, therefore $g(x_1) = 0$, $g(x_2) = 0$. So we can transform the above into

$$J_2[f,g] = 2\int_{x_0}^{x_1} g^2 - \lambda_1 g'^2 + \lambda_2 g''^2 dx \geq 0.$$

The latter can be written as

$$2\int_{x_0}^{x_1} \left(g^2 - \frac{\lambda_1}{2} g''\right)^2 + \left(\lambda_2 - \frac{\lambda_1^2}{4}\right) g''^2 dx \geq 0.$$

The integral is guaranteed to be positive if

$$4\lambda_2 > \lambda_1^2.$$

If we refer back to (15), we will note that this condition will give complex root for γ. We can now assume four complex roots of the form $\gamma = \pm\alpha \pm i\omega$ with α, ω real. Now $\gamma^2 = \alpha^2 - \omega^2 \pm 2i\alpha\omega$ and we have

$$\alpha^2 - \omega^2 = \frac{\lambda_1}{2\lambda_2} \text{ and } 4\alpha^2\omega^2 = \frac{4\lambda_2 - \lambda_1^2}{4\lambda_2^2}.$$

The solution in the range $[-W, 0]$ now could be written as

$$f(x) = a_1 e^{\alpha x}\sin\omega x + a_2 e^{\alpha x}\cos\omega x + a_3 e^{-\alpha x}\sin\omega x + a_4 e^{-\alpha x}\cos\omega x + c \quad (17)$$

with the boundary conditions

$$f(0) = 0,\ f(-W) = 0,\ f'(0) = s,\ f'(-W) = 0, \quad (18)$$

where s is an unknown constant. Since $f(x)$ is antisymmetric, we can extend the above definition to the range $[-W, W]$ by having $f(-x) = -f(x)$.

The four boundary conditions from (18) enable us to solve the four parameters a_1 to a_4 in terms of the unknown constants α, ω, c, and s.

The boundary conditions could be rewritten as

$$a_2 + a_4 + c = 0$$

$$a_1 e^{\alpha}\sin\omega + a_2 e^{\alpha}\cos\omega + a_1 e^{-\alpha}\sin\omega + a_2 e^{-\alpha}\cos\omega + c = 0$$

$$a_1\omega + a_2\alpha + a_3\omega - a_4\alpha = s$$

$$a_1 e^{\alpha}(\alpha\sin\omega + \omega\cos\omega) + a_2 e^{\alpha}(\alpha\cos\omega - \omega\sin\omega)$$

$$+a_3 e^{-\alpha}(-\alpha\sin\omega + \omega\cos\omega) + a_4 e^{-\alpha}(-\alpha\cos\omega - \omega\sin\omega) = 0.$$

Solve these equations would yield

$$
\begin{aligned}
a_1 &= c(\alpha(\beta-\alpha)\sin 2\omega - \alpha\omega\cos 2\omega + (-2\omega^2\sinh\alpha + 2\alpha^2 e^{-\alpha})\sin\omega + \\
&\quad 2\alpha\omega\sinh\alpha\cos\omega + \omega e^{-2\alpha}(\alpha+\beta) - \beta\omega)/4(\omega^2\sinh^2\alpha - \alpha^2\sin^2\omega) \\
a_2 &= c(\alpha(\beta-\alpha)\cos 2\omega + \alpha\omega\sin 2\omega - 2\alpha\omega\cosh\alpha\sin\omega - 2\omega^2\sinh\alpha\cos\omega + \\
&\quad 2\omega^2 e^{-\alpha}\sinh\alpha + \alpha(\alpha-\beta)/4(\omega^2\sinh^2\alpha - \alpha^2\sin^2\omega) \\
a_3 &= c(-\alpha(\beta+\alpha)\sin 2\omega + \alpha\omega\sin 2\omega + (2\omega^2\sinh\alpha + 2\alpha^2 e^{\alpha})\sin\omega + \\
&\quad 2\alpha\omega\sinh\alpha\cos\omega + \omega e^{2\alpha}(\beta-\alpha) - \beta\omega)/4(\omega^2\sinh^2\alpha - \alpha^2\sin^2\omega) \\
a_4 &= c(-\alpha(\beta+\alpha)\cos 2\omega - \alpha\omega\sin 2\omega + 2\alpha\omega\cosh\alpha\sin\omega + 2\omega^2\sinh\alpha\cos\omega \\
&\quad - 2\omega^2 e^{\alpha}\sinh\alpha + \alpha(\alpha-\beta)/4(\omega^2\sinh^2\alpha - \alpha^2\sin^2\omega)
\end{aligned}
$$

where β is the slope s at the origin divided by the constant c. Since c appears in every term in $f(x)$ now, we could consider c as a scaling factor of $f(x)$ which would not affect the criteria.

The next step is to replace the $f(x)$ in the integral in in (12) and then (7) to get the values of the constants α, β, and ω. Unfortunately, it is too complex to be solved analytically, and has to be solved by numerical methods. Therefore, it is impractical to apply this function for real line detection.

A more manageable function is the first derivative of a Gaussian $G'(x)$, where

$$G(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right),$$

since the shape of $G'(x)$ is very close to that of our optimal detector, and would give pretty good result. Also, it is very easy to be extended to a two-dimensional filter which we will discuss next.

3.3.4 Two-Dimensional Canny Detector

In one dimension, we could characterize an edge with only its location. In two dimension, there is another parameter we need to consider, the directions of the edges. If we get the direction of a edge normal, we can apply the one-dimensional filter to locate the edge.

Now we try to filter the image with an operator $G_{\boldsymbol{n}}$ which the first derivative of a two dimensional Gaussian G in a direction $\boldsymbol{n}$, i.e.

$$G = \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

and

$$G_{\boldsymbol{n}} = \frac{\partial G}{\partial \boldsymbol{n}} = \boldsymbol{n}\cdot\nabla G,$$

where $\boldsymbol{n}$ should be the direction of the edge normal.

A edge point is defined as the local maximum of the operator $G_{\boldsymbol{n}}$ applied to the image I. That is, at an edge point, we have

$$\frac{\partial}{\partial \boldsymbol{n}} G_{\boldsymbol{n}} * I = 0$$

where $*$ denotes convolution. And since $G_{\boldsymbol{n}} = \frac{\partial G}{\partial \boldsymbol{n}}$, we get

$$\frac{\partial^2}{\partial \boldsymbol{n}^2} G_{\boldsymbol{n}} * I = 0.$$

At such edge point, the edge strength we be

$$s(i,j) = |G_{\boldsymbol{n}} * I| = |\nabla (G * I)|.$$

Because of the associativity of convolution, we can convolve the image with a circular Gaussian G first, then compute the second derivative for zero-crossings to locate edges. Then get the first derivative to estimate the edge strength. And for the direction of the edge point, it will be the direction of the vector from $\nabla(G * I)$ and should be

$$\frac{\nabla (G * I)}{|\nabla (G * I)|}.$$

3.3.5 Nonmaximum Supression

By applying the detector function onto the image, we get a response for each pixel. Since Canny's detector is basically a first dreivative operator, that is, we expect a edge at a local maximum, we have to supress the responses which are not local maximum in order to get the edge points. We can achieve this by examining the responses of two neighbors along the direction of the edge normal of the response of each pixel, if at least one neighbor has bigger response, then the pixel is not a local maxinum, and will be supressed to 0.

3.3.6 Hysteresis Thresholding

After the step of nonmaximum supression, we thin the lines detected in an image into one-pixel wide. However, the result still contains the local maxima caused by noise. We can use a threshold and discard any pixel which response small than that threshold. But the has problem. If the threshold is small in order to get the weak edges, some noise maxima will be accepted, but if the threshold is high, some edge points might be missed, which will break the result lines into unconnected pieces.

A solution of this is hysteresis thresholding. This is done by setting two threaholds. The higher threshold is used to pick some stronger edges points, then we can get the edges by tracking the stronger edge points , using the information of the direction of edge normal. As long as the response of a pixel is greater than the smaller threshold, we can keep the edge growing.

This reduces the possibility of getting false edges caused by noise, since those false edges tend to have small response values and could not pass the higher threshold to start the tracking. And it would keep each edge connected, since we use lower threshold to track the edge.

3.4 The Mumford-Shah Theory

Mumford and Shah [MS85] proposed another form of energy minimizing to detect boundary of noisy images. For a given image $g : R \to \mathbb{R}$, the energy functional E for a boundary B and smoothed image f is

$$E(f,B) = \mu^2 \int_R \underbrace{(f-g)^2}_{\text{Data Fidelity}} + \int_{R-B} \underbrace{\|\nabla f\|^2}_{\substack{\text{Smoothness} \\ \text{Constraint}}} + \underbrace{\mu\nu|B|}_{\text{Edge Penalty}} \tag{19}$$

where R is the region of the image, $f : R \to \mathbb{R}$ is the smoothed ideal image, $B : R \to \{0,1\}$ is the binary edge process, μ and ν are scalar parameters. $|B|$ denotes the volume of B. For two-dimensional images, $|B|$ is the length of the edges and $R - B$ denotes the region R excluding the edges.

The Mumford-Shah energy $E(f,B)$ formalizes a tradeoff between noise removal and edge detection. A disadvantage of the functional is that it lacks a practical mean to solve for the smoothed image f, and the binary edge process B. Many methods have been proposed to simplify the minimization of the functional. In one of them, [AT90], the binary edge process, B, of the energy function (19) is replaced by a continuous edge field. The energy functional would become

$$E(f,v) = \int_R \left\{ \mu \underbrace{(f-g)^2}_{\text{Data Fidelity}} + \underbrace{\alpha(1-v)^2\|\nabla f\|^2}_{\text{Smoothness Constraint}} + \underbrace{\frac{\beta}{2}\|\nabla v\|^2 + \frac{v^2}{2\beta}}_{\text{Edge Penalty}} \right\} \tag{20}$$

which is called the Ambrosio-Tororelli functional. The $g : R \to \mathbb{R}$ denotes the given image, $f : R \to \mathbb{R}$ the piecewise smoothed image, $v : R \to [0,1]$ the corresponding continuous edge strength of the image, R the image domain, and α, β and μ are scalar parameters.

3.5 SUSAN Edge Detector

SUSAN [SMS97] is a new approach to edge detection, as well as some other low-level image processing such as corner detection and noise reduction. The main difference from all the previous works discussed is that it is not based on differential computation. It is based on a voting scheme on the fact that for each pixel in an image, if we look for the number of pixels with similar pixel

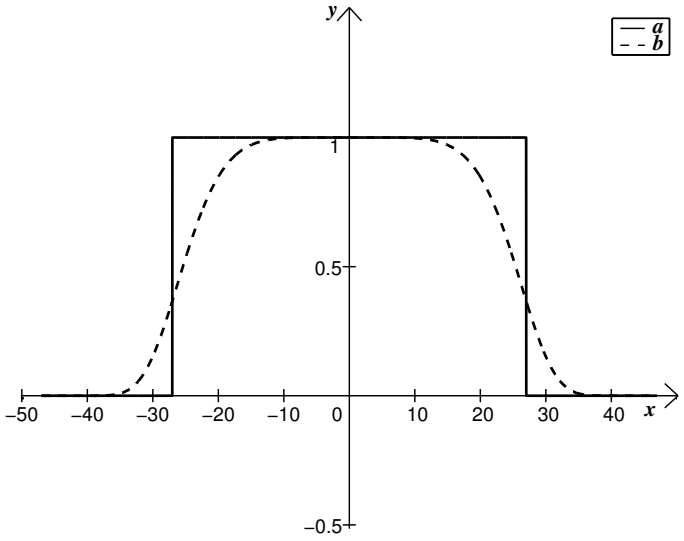


Figure 6: a) The original similarity function (**y** axis) versus pixel brightness difference (**x** axis). The brightness difference threshold is set at 27. b) The more stable function. This figure is adapted from [SMS97]

values around the neighborhood, there will be local minima near the edges since these minima describe the positions of abrupt change of pixel values.

The neighborhood is defined by a pre-determined mask. Circular masks are used in order to give isotropic responses. The usual radius is 3.4 pixels which give a mask of 37 pixels.

The mask is placed at each point, and for each point, the brightness of each pixel within the mask is compared with that of the center pixel, which is called the nucleus. Then the area of the mask which has similar brightness as the nucleus is called "USAN"(Univalue Segment Assimilating Nucleus.)

A simple equation could determine this comparison:

$$c(\mathbf{r}, \mathbf{r}_0) = \begin{cases} 1 & \text{if } |I(\mathbf{r}) - I(\mathbf{r}_0)| \leq t \\ 0 & \text{if } |I(\mathbf{r}) - I(\mathbf{r}_0)| > t, \end{cases} \tag{21}$$

where $\mathbf{r}_0$ is the position of the nucleus in the image, $\mathbf{r}$ is the position of any other point within the mask, $I(\mathbf{r})$ is the brightness of the pixel $\mathbf{r}$, t is the brightness difference threshold and c is the output, see Figure 6(a).

Once all comparisons are done, the total output is defined as:

$$n(\mathbf{r}_0) = \sum_{\mathbf{r}} c(\mathbf{r}, \mathbf{r}_0).$$

This total n is the area of the USAN in pixels. Now, n is compared with the geometric threshold g, which is fixed at $3n_{max}/4$, where n_{max} is the maximum value of n over the whole image. The initial edge response is:

$$R(\mathbf{r}_0) = \begin{cases} g - n(\mathbf{r}_0) & \text{if } n(\mathbf{r}_0) < g \\ 0 & \text{otherwise,} \end{cases}$$

The value $3n_{max}/4$ is calculated from an analysis of the expectation value of the response in the presence of noise only.

To make the result more stable, the equation (21) could be replaced by:

$$c(\mathbf{r}, \mathbf{r}_0) = e^{-(\frac{I(\mathbf{r}) - I(\mathbf{r}_0)}{t})^6}. \tag{22}$$

The equation is plotted in Figure 6(b). This reduces the effect of those pixels with brightness near the threshold value. It was proved in [SMS97] that the use of the sixth power in equation (22) is the theoretical optimum.

The next step is to find the direction of the edge point. In order to get the directions of the edge points we detect, we separate the edge points into to two categories, see Figure 7.

The standard edge points (a) and (b) have USAN shapes expected for an ideal step edge. In this case (which is called "inter-pixel edge"), the vector between the center of gravity $\mathbf{r}_g$ of the USAN and the nucleus of the mask is perpendicular to the local edge direction. The center of the gravity is

$$\mathbf{r}_g(\mathbf{r}_0) = \frac{\sum_{\mathbf{r}} \mathbf{r} c(\mathbf{r}, \mathbf{r}_0)}{\sum_{\mathbf{r}} c(\mathbf{r}, \mathbf{r}_0)}.$$

The point (c) lies on the edge. This usually happens when the edge is very close to the center of a pixel rather than between pixels, or the width of the edge is more than one pixel. In this case (which is called "intra-pixel edge"), the USAN forms a thin line in the direction of the edge. Since the center of gravity is very close to the nucleus, we have to find the edge direction by another method which is instead calculating the longest axis of symmetry. This is estimated by taking the sums

$$\mathbf{r}_x(\mathbf{r}_0) = \sum_{\mathbf{r}} (x - x_0)^2 c(\mathbf{r}, \mathbf{r}_0),$$

$$\mathbf{r}_y(\mathbf{r}_0) = \sum_{\mathbf{r}} (y - y_0)^2 c(\mathbf{r}, \mathbf{r}_0),$$

$$\mathbf{r}_{x,y}(\mathbf{r}_0) = \sum_{\mathbf{r}} (x - x_0)(y - y_0) c(\mathbf{r}, \mathbf{r}_0).$$

The ratio of $\mathbf{r}_x$ to $\mathbf{r}_y$ is used to determine the direction of the edge; the sign of $\mathbf{r}_{x,y}$ is used to determine if a diagonal edge has positive or negative gradient.

With the direction of the edge points, we could apply non-maxima suppression and connect the edge points.

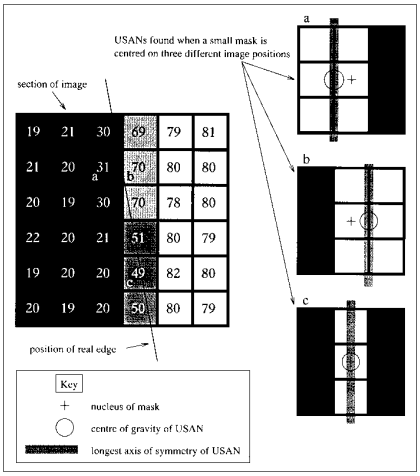


Figure 7: a) The two main edge types, with brightness indicated by numerical text. The USANs for three points of interest are shown as the white regions of a small 3x3 mask. Point (a) and (b) are standard edge points, lying on one side of the edge or the other. Point (c) lies exactly on the edge. This figure is adapted from [SMS97]

3.6 Multi-Scale Approaches

As suggested by Marr and Hildreth [DM80], we can obtain a description of an image at different scales by applying a feature detector at different scales and combining the edge information.

Another approach in multi-scale is to apply a edge detector many times with different smoothing parameters. As mentioned in Section 2.1, all edge point detectors face a tradeoff between localization error and sensitivity to noise. By adjusting the smoothing parameter, we can reduce sensitivity to noise, at the same time, increase localization error, or vise versa. The optimal parameter for one given image, however, may be hard to find, or not even exist. Multi-scale edge point detection is a solution to this problem by operating an edge point detector at multiple scales.

Take of the Gaussian as an example of the smoothing filter. We have the Gaussian

$$G = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where σ is our scaling parameter. For a small σ, this detector will be noise-sensitive, that may result in twisted and broken edges; for a large σ, we could obtain smoother edges, but some edges points would be lost, and some detected edge points would have a large delocalization error. This delocalization could be easily noted in Fig 8 which is a one-dimensional signal processed by a Gaussian

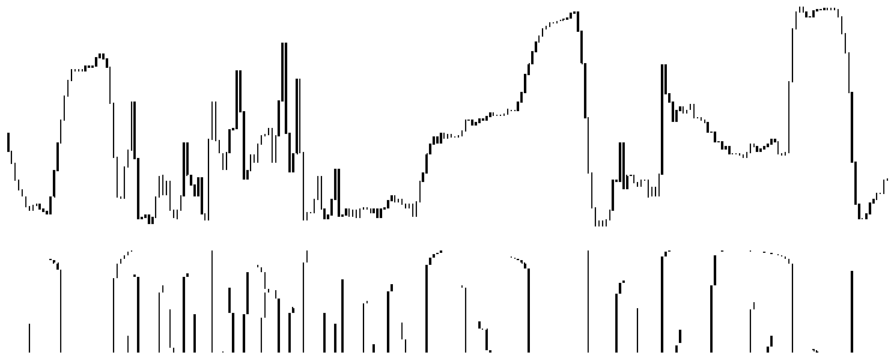


Figure 8: Second derivative of one-dimensional signal smoothed by Gaussian of different scales, and the edge it located (adapted from [Wit83]).

and the second-derivative.

Canny [Can86] has proposed a fine-to-coarse combination process, called feature synthesis, which uses his detector to mark edges at fine scale. From these edges, the coarser-scale gradient output is synthesized and compared to the actual coarser-scale detector output. Additional edges are marked only if the detector has significantly greater response than that predicted from the synthetic response. Bergholm [Ber87] proposes an algorithm called edge focusing, which uses Canny's detector, then combines edge information from a course to a fine scale.

3.7 Other Edge Point Detectors

There are some other edge point detectors. Poggio, Voorhees and Yuille [PVY85] introduced the cubic spline filter. The basic idea is to apply cubic spline interpolation before differentiation. It was shown in that paper that the result of interpolation is very similar to the convolution with a Gaussian. The final filter function is very close to Canny's detector. Deriche [Der87] extended Canny's filter by using an infinite extent filter, and implemented it using recursing filtering. Marr and Hildreth[DM80, Hil83] have proposed the use of zero-crossings of the Laplacian of Gaussian. The image is convolved with the Laplacian of the two-dimensional Gaussian, and the zero-crossings are labeled.

3.8 Subpixel Edge Point Detector

The need of subpixel edge point detection rises because of the aliasing problem of digitized images. No matter how good an edge point detector is, as long as it is pixel based, it will never be able to form smooth edges. However, since the

precision of an digitized image is up to a pixel, we can only get the subpixel positions of edge points through interpolation.

To get subpixel positions of edges, it usually involves parametric fitting methods. By **parametric fitting**, we mean matching an edge model with the selected edge pixels, and finding the parameters with minimal fitting error.

Hueckel edge fitting (see Section 3.2) method produces edge position to subpixel level, by transforming the image to the Hilbert space and interpolating to compute the location.

MacVicar-Whelan and Binford [MWB81] use the gradient operator, and find the subpixel locations of edges by linear-interpolating the positions of zero crossings.

Hyde and Davis [HD83] proposed to use intensity information in neighborhood of a straight edge to determine the location of the edge to subpixel level. They used the concept of **maximum likelihood** of all points on the edge as the criterion of picking the line for the straight edge.

Tabatabai and Mitchell [TM84] proposed a method to find a subpixel level of edge position in one-dimension, with a give n monotonic nondecreasing or monotonic nonincreasing numbers, then extended it to two-dimensional image with noise. The subpixel value locations are calculated using the first three statistic moments. The moments of function $f(x)$ are defined as $m_i = \int (f(x))^i \, dx, i = 1, 2, 3 \ldots$.

In Lyvers, et al. [LMAR89], **spatial moments** are used to calculate edge positions to subpixel level. **Spatial moments** of function $f(x)$ are defined as $M_i = \int x_i f(x) \, dx, i = 0, 1, 2 \ldots$.

4 Color Edge Point Detection

Most of those edge point detection techniques for gray-scale images could be applied on color images as well. These techniques are applied to the three color channels independently, and the results are combined using logical operators. One problem with this approach is that this way, it could not take the correlation among the color channels into account, and might lose some information conveyed by color. One way to avoid this problem is to consider the problem of color edge point detection in vector space. And instead of taking differential of the pixel values, we use the vectors with three color channels and their distances. There are different ways to measure the distance of two vectors. The common used measure is the generalized Minkowski metric(L_p norm). It is defined for two vectors $\mathbf{x}$ and $\mathbf{y}$ as

$$d_M(\mathbf{x}, \mathbf{y}) = \left(\sum_{k=1}^{p} |(x_k - y_k)|^p \right)^{\frac{1}{p}}$$

where p is the dimension of the vectors and x_k is the k-th element of $\mathbf{x}$.

Then, we could apply one of the distance measurements in most of edge point detectors for gray-scale images.

5 Edge Detection

Edge detection is the problem of localizing continuous curves, not just detecting edge points. The usual way to do edge detection is to find some scheme to connect edge points into continuous curves. In fact we could combine the three steps *smoothing*, *differentiation*, and *labeling* in edge point detection and "edge linking" as the four steps of edge detection.

Edge detection is somewhat related to image segmentation, in some special case, even equivalent, since edges are nature boundaries for different regions separated by image segmentation.

The most obvious problems of edge detection are

1. Some edges are broken by noise or occlusion, or by the disappearance of the the intensity gradient, for on reason or another. A very common example of such disappearance is shadow.

2. T-splits tend to confuse the detectors. Though such task generally involves distinguishing background and foreground, therefore does not belong to low-level image processing such as edge detection.

3. When some part of two different edges get too close, it is possible the detector might mix them up.

Since the first three steps are the same as edge point detection, here we will concentrate only on the step of edge linking.

5.1 Hough Transform

The Hough transform is used to detect straight lines in an binary image. For each point $P(x_0, y_0)$ in a $M \times N$ binary image, any line $y = mx + n$ that goes through P would satisfy the condition $n = x_0(-m) + y_0$. Therefore, the parameter pair (m, n) of all the lines through P would form a straight line in the parameter space. In other words, the point $P(x_0, y_0)$ and the line $n = x_0(-m) + y_0$ in the parameter space are equivalent, and for the same reason, a point (m_0, n_0) in parameter space could be presented as a line in the image. If there are exactly N points P_1, P_2, ..., P_N, on the same line $y = m_0x + n_0$, then the N corresponding lines in parameter space would intersect at (m_0, n_0). This gives the following method of line detection: we discretize parameter space by dividing the space into grid of cells, and associate a counter $c(m, n)$ to each cell. Then for each point P_i, we increase all the (m, n) pair on the line corresponding to P_i by one. The final values of $c(m, n)$ would be 0 or 1 except for $c(m_0, n_0)$ which would be N. Therefore, after we do similar process for all points in an

given binary image, each counter value will represent the number of points on the same line. The peaks of the counters represent the possibility of a line composed by the points.

However, since m, n are not bounded and could be any number in $[-\infty, \infty]$, the parameter space is unbounded, too. Therefore, we cannot use a limited number of counters to cover the whole parameter space. So we choose polar representation of lines instead,

$$\rho = x\cos\theta + y\sin\theta, \tag{23}$$

where x, y are the coordinates of a point, ρ is the distance between the origin of the image and the line, and θ is the orientation of the line, $\rho \in [0, \sqrt{M^2 + N^2}]$, $\theta \in [0, \pi)$. The advantage of this representation is that since both ρ and θ are bounded in limited interval, we could cover the whole (ρ, θ) parameter space with limited number of counters if we discretize the parameter space.

The advantage of Hough transform is that we could group all the points in a straight line very easily, even the line of points is not "continuous". Therefore, it would be very noise-resistant. However, it is limited to detect straight lines. Though it could be modified into detect specific curves by replacing the equation (23), it is not easy to be generalized into more than one type of curve. Furthermore, for our edge detection purpose, it could not distinguish between two unrelated edges apart if they are almost on the same line.

5.2 Active Contour

Active contour, also as known as "snake", was introduced by Kass, Witkin and Terzopolous [KWT88]. It uses energy minimization as a technique to shift an initial line segment into a curve that would adhere to an edge of the given image. It is done by assigning three forms of energy to a curve: an *internal energy* that generates a force that tends to keep the curve short and smooth, an *image energy* that tends to move the curve towards edge points on the image, and a *constraint energy* that tends to restrict the curve in some other ways, such as attraction to or expulsion from certain points. More precisely the energy function E^*_{snake} could be written as

$$\begin{aligned} E^*_{snake}(\mathbf{v}) &= \int_0^1 E_{snake}(\mathbf{v}(s))ds \\ &= \int_0^1 E_{int}(\mathbf{v}(s)) + E_{image}(\mathbf{v}(s)) + E_{con}(\mathbf{v}(s))ds \end{aligned} \tag{24}$$

where $\mathbf{v}(s) = (x(s), y(s))$, $0 \le s \le 1$, is the position of the curve, E_{int} is the *internal energy*, E_{image} the *image energy*, and E_{con} the *constraint energy*. Starting from some initial curve $\mathbf{v}_0$, the energy minimization process will continuously transform the curve into lower energy states, until it achieves some

local minimum energy level $E^*_{snake}(\mathbf{v}^*)$ at some position $\mathbf{v}^*$. The three energy forms will be discuss separately.

Internal Energy. The internal energy can be written as

$$E_{int}(\mathbf{v}(s)) = (\alpha(s)|\mathbf{v}_s(s)|^2 + \beta(s)|\mathbf{v}_{ss}(s)|^2)/2 \tag{25}$$

where $\alpha(s)$ and $\beta(s)$ are user-defined parameters with non-negative values. Here $\mathbf{v}_s$ and $\mathbf{v}_{ss}$ are derivatives of the curve $\mathbf{v}$.

The internal energy is composed of a first-order derivative controlled by $\alpha(s)$ and a second-order derivative controlled by $\beta(s)$. The first-order term generates a force that tends to shorten the length of the curve. We will call it **elastic energy**. The smaller the value of $\alpha(s)$ is, the easier for the snake to be stretched. The second-order term generates a force that tends to keep the curve from changing direction sharply. We call it **bending energy**. The smaller the value of $\beta(s)$, the easier for the curve to bend. Thus, the internal energy creates the forces which tend to keep the curve short and smooth.

Image Energy. The image energy derives from an interaction between the snake curve and the image. It pushes the snake curve toward salient image feature like lines and edges. This energy could be categorized into three energy functionals: line energy, edge energy, and termination energy.

Line energy is the functional that is usually relevant to the image intensity itself. This functional makes the snake attracted to pixels of some chosen intensity. Normally, line energy could be expressed as

$$E_{line} = |I(x, y) - I_0|,$$

where I_0 is the expected intensity of line.

Edge energy is the functional that is usually relevant to the gradients of the image. This functional will pull the snake near the the points with high gradients, which are where the edges usually are. A simple example of edge energy is

$$E_{energy} = -|\nabla I(x, y)|^2.$$

Termination energy is the functional that is used to find terminations of line segments and corners. It is done by measuring the curvature of the level contours.

Constraint Energy. The constraint energy is not directly relevant to the to edge detection. It is usually used to generate forces to attract or repel the snake towards or away from some desired positions. For instance if we want the endpoints of the snake to be at some fixed positions, we can introduce a suitable constraint energy to achieve this.

Numerical Methods for Snakes Let $E_{ext} = E_{image} + E_{con}$ denote the **external energy**. When $\alpha(s) = \alpha$, and $\beta(s) = \beta$ are both constants, The equation (24) could be minimized by using Euler equation (As described in Appendix A.1,) and we have

$$-\alpha x_{ss} + \beta x_{ssss} + \frac{\partial E_{ext}}{\partial x} = 0 \tag{26}$$

$$-\alpha y_{ss} + \beta y_{ssss} + \frac{\partial E_{ext}}{\partial y} = 0 \tag{27}$$

When $\alpha(s)$ and $\beta(s)$ are not constants, it would not be easy to solve the equations. So we use a discrete method instead.

Instead of defining the curve continuously, we sample n points from the curve $\mathbf{v}$ such that $\mathbf{v}_i = \mathbf{v}(i/n)$. We now have $\mathbf{v}_i = (x_i, y_i) = (x(i/n), y(i/n))$. And the corresponding α and β will become $\alpha_i = \alpha(i/n)$, and $\beta_i = \beta(i/n)$.

We then expand internal energy $E_{int}(i)$ into

$$E_{int}(i) = -\alpha_i \|\mathbf{v}_i - \mathbf{v}_{i-1}\|^2 n^2 + \beta_i \|\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}\|^2 n^4.$$

The corresponding Euler equations (26) and (27) will be

$$\begin{aligned} -\alpha_i(\mathbf{v}_i - \mathbf{v}_{i-1}) \quad &+ \quad \alpha_{i+1}(\mathbf{v}_{i+1} - \mathbf{v}_i) \\ &+ \quad \beta_{i-1}(\mathbf{v}_{i-2} - 2\mathbf{v}_{i-1} + \mathbf{v}_i) \\ &- \quad 2\beta_i(\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}) \\ &+ \quad \beta_{i+1}(\mathbf{v}_i - 2\mathbf{v}_{i+1} + \mathbf{v}_{i+2}) \\ &+ \quad (f_x(i), f_y(i)) = 0, \end{aligned}$$

where $f_x(i) = \partial E_{ext}/\partial x_i$ and $f_y(i) = \partial E_{ext}/\partial y_i$.

The above equations could be written in matrix form as

$$\mathbf{A}\mathbf{x} + \mathbf{f}_\mathbf{x}(\mathbf{x}, \mathbf{y}) = 0 \tag{28}$$

$$\mathbf{A}\mathbf{y} + \mathbf{f}_\mathbf{y}(\mathbf{x}, \mathbf{y}) = 0 \tag{29}$$

where A is a pentadiagonal banded matrix.

To solve equations (28) and (29), we set the right-hand sides of the equations equal to the product of a step size γ and the negative time derivative of the variables on the left-hand sides. The γ is a damping constant. With some simplification, we have

$$\mathbf{A}\mathbf{x}_t + \mathbf{f}_\mathbf{x}(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}) = -\gamma(\mathbf{x}_t - \mathbf{x}_{t-1}) \tag{30}$$

$$\mathbf{A}\mathbf{y}_t + \mathbf{f}_\mathbf{y}(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}) = -\gamma(\mathbf{y}_t - \mathbf{y}_{t-1}) \tag{31}$$

where γ is the step size. At equilibrium, the time derivative $(\mathbf{x}_t - \mathbf{x}_{t-1}, \mathbf{y}_t - \mathbf{y}_{t-1})$ vanishes and we end up with a solution of equations (28) and (29).

Equation (30) and (31) can be solved:

$$\mathbf{x}_t = (\mathbf{A} + \gamma\mathbf{I})_{-1}(\mathbf{x}_{t-1} - \mathbf{f}_\mathbf{x}(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}) \tag{32}$$

$$\mathbf{y}_t = (\mathbf{A} + \gamma\mathbf{I})_{-1}(\mathbf{y}_{t-1} - \mathbf{f}_\mathbf{y}(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}) \tag{33}$$

Therefore, we can calculate the values of $\mathbf{x}$ and $\mathbf{y}$ until the values reach equilibrium.

It performs well if the initial positions of the control points are well chosen. These points could be the result from some edge points detectors. But we still need to group the edge points into different edges before we apply this method, and such task of grouping is not trivial.

One problem of the active contour is that there is no method to evaluate the result. With any kind of input, we always have a result of minimum energy, even when there is no edge around the initial control points. And to pick right initial control points is not trivial. The results for edge point detectors are not sufficient, since there is no telling how to group edge points into different edges. This technique is widely used in stereopsis and object tracking in motion pictures, since we can use results from one frame as the initial control points of the next.

A Mathematical Background for Edge Detection

A.1 Euler's Equation

Given a real functional $F(x, f_1, f_2, f_3)$, we want to find a function $y = y(x)$ which would minimize the integral

$$I(y) = \int_{x_1}^{x_2} F(x, y, y', y'')dx$$

where y', y'' are derivatives w.r.t. x.

First we define a new function $Y(x)$

$$Y(x, \epsilon) = y(x) + \epsilon\eta(x)$$

where ϵ is a parameter and $\eta(x)$ is an arbitrary function which has continuous third derivative in the interval x_1 to x_2, with the boundary condition

$$\eta(x_1) = \eta(x_2) = 0,$$

$$\eta'(x_1) = \eta'(x_2) = 0.$$

When $\epsilon = 0$, $Y(x, \epsilon) = y(x)$ is supposed to make $I = I(Y)$ minimum. So we want

$$\frac{dI}{d\epsilon} = 0 \text{ when } \epsilon = 0.$$

Since

$$\begin{aligned}\frac{dI}{d\epsilon} &= \int_{x_1}^{x_2} \left(\frac{\partial F}{\partial Y}\frac{\partial Y}{\partial \epsilon} + \frac{\partial F}{\partial Y'}\frac{\partial Y'}{\partial \epsilon} + \frac{\partial F}{\partial Y''}\frac{\partial Y''}{\partial \epsilon} \right) dx \\ &= \int_{x_1}^{x_2} \left(\frac{\partial F}{\partial Y}\eta(x) + \frac{\partial F}{\partial Y'}\eta'(x) + \frac{\partial F}{\partial Y''}\eta''(x) \right) dx\end{aligned}$$

and with the fact that when $\epsilon = 0, Y = y$, we can have

$$\left(\frac{dI}{d\epsilon}\right)_{\epsilon=0} = \int_{x_1}^{x_2} \left(\frac{\partial F}{\partial y}\eta(x) + \frac{\partial F}{\partial y'}\eta'(x) + \frac{\partial F}{\partial y''}\eta''(x) \right) dx = 0.$$

If y'' is continuous, we can integrate the second term by parts:

$$\int_{x_1}^{x_2} \frac{\partial F}{\partial y'}\eta'(x)\,dx = \frac{\partial F}{\partial y'}\eta(x)\Big|_{x_1}^{x_2} - \int_{x_1}^{x_2} \frac{d}{d\,x}\left(\frac{\partial F}{\partial y'}\right)\eta(x)\,dx.$$

Since $\eta(x)$ is 0 at x_1 and x_2, we can have

$$\int_{x_1}^{x_2} \frac{\partial F}{\partial y'}\eta'(x)\,dx = -\int_{x_1}^{x_2} \frac{d}{d\,x}\left(\frac{\partial F}{\partial y'}\right)\eta(x)\,dx.$$

We similarly obtain

$$\int_{x_1}^{x_2} \frac{\partial F}{\partial y''}\eta''(x)\,dx = \int_{x_1}^{x_2} \frac{d^2}{d^2\,x}\left(\frac{\partial F}{\partial y''}\right)\eta(x)\,dx$$

from the third term by applying the same transformation twice.

Then we have

$$\left(\frac{dI}{d\epsilon}\right)_{\epsilon=0} = \int_{x_1}^{x_2} \left(\frac{\partial F}{\partial y}\eta(x) - \frac{d}{d\,x}\left(\frac{\partial F}{\partial y'}\right)\eta(x) + \frac{d^2}{d^2\,x}\left(\frac{\partial F}{\partial y''}\right)\eta(x) \right) dx = 0.$$

Then,

$$\int_{x_1}^{x_2} \left[\frac{\partial F}{\partial y} - \frac{d}{d\,x}\left(\frac{\partial F}{\partial y'}\right) + \frac{d^2}{d^2\,x}\left(\frac{\partial F}{\partial y''}\right) \right] \eta(x) dx = 0.$$

Since η is arbitrary, we have

$$\frac{\partial F}{\partial y} - \frac{d}{d\,x}\left(\frac{\partial F}{\partial y'}\right) + \frac{d^2}{d^2\,x}\left(\frac{\partial F}{\partial y''}\right) = 0 \tag{34}$$

which is usually called Euler-Lagrange equation.

Example from the equations in Active Contours (see Section 5.2) According to equations (24) and (25), we could have

$$E^*_{snake}(\mathbf{v}) = \int_0^1 (\alpha(s)|\mathbf{v}_s(s)|^2 + \beta(s)|\mathbf{v}_{ss}(s)|^2)/2 + E_{image}(\mathbf{v}(s)) + E_{con}(\mathbf{v}(s))ds$$

With the assumption $\alpha(s) = \alpha$, and $\beta(s) = \beta$ are both constants, and let $E_{ext} = E_{image} + E_{con}$, we have

$$E^*_{snake}(\mathbf{v}) = \int_0^1 (\alpha|\mathbf{v}_s(s)|^2 + \beta|\mathbf{v}_{ss}(s)|^2)/2 + E_{ext}(\mathbf{v}(s))ds \tag{35}$$

Since we seek minimizing the energy $E^*_{snake}(\mathbf{v})$, we can apply Euler-Lagrange equation (34) with $F = (\alpha|\mathbf{v}_s(s)|^2 + \beta|\mathbf{v}_{ss}(s)|^2)/2$ and $x = s$, $y = \mathbf{v}$ and get

$$\frac{d}{ds}E_{ext}(\mathbf{v}(s)) - \frac{d}{ds}(\alpha|\mathbf{v}_s(s)|) + \frac{d^2}{d^2s}(\beta|\mathbf{v}_{ss}(s)|) = 0$$

Since $\frac{d}{ds}E_{ext}(\mathbf{v}(s))$ is a first derivative from the direction of the snake, we could replace it with $\nabla E_{ext}(\mathbf{v}(s))$, the gradient.

Therefore, we can get

$$(\alpha|\mathbf{v}_{ss}(s)|) + \frac{d^2}{d^2s}(\beta|\mathbf{v}_{ssss}(s)|) = 0$$

A.2 Lagrange Multipliers

Let $f(x, y)$ be a function where x and y are constrained by an equation $\eta(x, y) = c$, where c is a constant. The problem is to find (x, y) to minimize or maximize $f(x, y)$.

Since $\eta = \text{const}$, we have $d\eta = 0$. We want to find the maximum or minimum of f, so $df = 0$, too.

$$df = \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial y}dy = 0$$

$$d\eta = \frac{\partial \eta}{\partial x}dx + \frac{\partial \eta}{\partial y}dy = 0$$

With these two equations, we could solve them by substitution and elimination. But it may involves complex algebra. So instead, we multiply the $d\eta$ equation by an undetermined value λ, and add it to the df equation. So we get

$$\left(\frac{\partial f}{\partial x} + \lambda\frac{\partial \eta}{\partial x}\right)dx + \left(\frac{\partial f}{\partial y} + \lambda\frac{\partial \eta}{\partial y}\right)dy = 0. \tag{36}$$

Now, we select $\lambda = -(\partial f/\partial y)/(\partial \eta/\partial y)$ such that

$$\frac{\partial f}{\partial y} + \lambda\frac{\partial \eta}{\partial y} = 0. \tag{37}$$

Then from equations (36) and (37) we have

$$\frac{\partial f}{\partial x} + \lambda \frac{\partial \eta}{\partial x} = 0. \tag{38}$$

With equations (37), (38) and the fact $\eta(x, y) = c$, we can now solve for the three unknown x, y, λ.

Now considering the unconstraint function

$$F(x, y) = f(x, y) + \lambda(\eta(x, y) - c) \tag{39}$$

with three independent variables x, y and λ, the equations to solve the function $f(x, y)$ with contraint $\eta(x, y) = c$ (37), (38) and $\eta(x, y) = c$ would be the same equations needed to find the maximum and minimum values of $F(x, y)$. In other word, by introducing λ, we convert a constraint minimizing problem into an unconstraint one. This λ is called a Lagrange multiplier.

References

[ACB83] D. C. Munson A. C. Borik, T. S. Huang. A generalization of median filtering using combination of order statistics. *IEEE Proceedings*, 71(31):1342–1350, 1983.

[AO68] T.G. Stockham A.V. Oppenheim, R.W. Schafer. Nonlinear filtering of multiplied and convolved signals. *Proceedings of IEEE*, 56(2):1264–1291, 1968.

[AT90] L. Ambrosio and V.M. Tororelli. Approximation of functionals depending on jumps by elliptic functional via γ-convergence. *Comm. Pure and Appl. Math*, 43(8), December 1990.

[Ber87] F. Bergholm. Edge focusing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(6), November 1987.

[Can86] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Ananlysys and Machine Intelligence*, 8(6), November 1986.

[CH53] R. Courant and D. Hilbert. *Methods of Mathematical Physics.* Wiley-Interscience, New York, 1953.

[Dav75] Larry S. Davis. A survey of edge detection techniques. *Computer Graphics and Image Processing*, 4, 1975.

[Der87] Rachid Deriche. Using canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, 1987.

[DM80] E. C. Hildreth D. Marr. Theory of edge detection. In *Proceedings of the Royal Society of London*, volume B207, pages 187–217, 1980.

[GW92] R. C. Gonzales and R. E. Wood. *Digital Image Processing.* Addison-Wesley Pub. Co, Reading, Massachusettts, 1992.

[HA83] M. Denis H.H. Arsenault. Image processing in signal-dependent noise. *Canadian Journal of Physics*, 61:309–317, 1983.

[HD83] P.D. Hyde and L.S. Davis. Subpixel edge estimation. *Pattern Recognition*, 16(4):413–420, 1983.

[Hil83] E. C. Hildreth. The detection of intensity changes by computer and biological vision system. *Computer Vision, Graphics, and Image Processing*, 22, 1983.

[Hue69] Manfred H. Hueckel. An operator which locates edges in digitied pictures. Technical Report Memo AIM-105, Artif. Intell. Proj., Stanford University, October 1969.

[Hue71] M. Hueckel. An operator which locates edges in digitized pictures. *Journal of the ACM*, 18:113–125, January 1971.

[Hue73] M. Hueckel. A local visual operator which recognizes edges and lines. *Journal of the ACM*, 20:634–647, October 1973.

[Hum87] A. Hummel. Representations based on zero-crossings in scale-space. In *Proc. IEEE Computer Vision and Pattern Recognition Conf.*, pages 204–209, 1987.

[Koe84] J. Koenderink. The structure of images. *Biological Cybernetics*, 50, 1984.

[Kos95] Andreas Koschan. A comparative study on color edge detection. In *Proc. 2nd Asian Conf. on Computer Vision ACCV'95*, volume III, pages 574–578, 1995.

[KWT88] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes, active contour models. *International Journal of Computer Vision*, 1988.

[LMAR89] E.P. Lyvers, O.R. Michell, M.L. Akey, and A.P. Reeves. Subpixel measurement using a moment-based edge operator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1293–1309, December 1989.

[MS85] D. Mumford and J. Shah. Boundary detection by minimizing functionals. In *IEEE Conf. Computer Vision and Pattern Recognition*, 1985.

[MWB81] P.J. MacVicar-Whelan and T.O. Binford. Line finding with subpixel precision. In *SPIE*, volume 281, 1981.

[Par96] J. R. Parker. *Algorithms for Image Processing and Computer Vision.* John Wiley & Sons, Inc., New York, 1996.

[PM90] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7), July 1990.

[PVY85] T. Poggie, H. Voorhees, and A. Yuille. A regularized solution to edge detection. Technical Report AI Memo 833, Massachusette Institute of Technology, Artifical Intelligence Laboratory, May 1985.

[Ric44] O.S. Rice. Mathematical analysis of random noise. *Bell System Technical Journal*, 23, 1944.

[Rob65] L. G. Roberts. Machine perception of three-dimensional solids. In J. T. Tippet et al., editor, *Optical and Electro-Optical Information Processing*, pages 159–197. MIT Press, Cambridge, Massachusetts, 1965.

[RW00] Gerhard X. Ritter and Joseph N. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra.* CRC Press LLC, Boca Raton, Florida, 2000.

[SMS97] J. Michael Brady Stephen M. Smoth. SUSAN – a new approach to low level image processing. *International Journal of Computer Vision*, 23(1), 1997.

[TM84] A.J. Tabatabai and O.R. Michell. Edge location to suxpixel values in digital imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):188–200, March 1984.

[Wit83] A. P. Witkin. Scale-space filtering. In *Proceedings of the 4th International Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.

[ZT97] D. Ziou and S. Tabbone. Edge detection techniques- an overview. Technical Report 195, Dept Math & Informatique, Universit de Sherbrooke, 1997.