数字图像的边缘检测

本科毕业设计（2003）

西安交通大学
理学院

指导教师：程正兴
作　　者：王郑耀

http://www.quzhi.net

QQ:16748251

摘 要

边缘检测是数字图像处理中的重要内容。本文首先对图像的边缘检测的各种算法和算子做了总结和分析。Canny 最早提出了边缘检测的三条连续准则:最优检测结果、最优定位和低重复响应,并在这些准则的基础上得到了“最优线性滤波器”－高斯函数的一阶导数。经过十几年的发展,目前已经有了对这个准则的很多改进,本文也对这个方面的工作做了小结。Demigny 在理论分析和实践的基础上给出了边缘检测的离散准则,并且证明在离散准则中 Canny 提出的第三个准则可以被阀值操作所取代。本文利用了数值方法求出了 Demigny 离散准则下阶梯形边缘检测的最优线性滤波器和对应着它的平滑算子。利用这个算子和 Canny 边缘检测方法得到了一个完整的边缘检测算法并用 VC++实现了这种算法.从算法对大量图像边缘检测的结果来看,这种算法虽然简单但是效果很好,是边缘检测的一种很好的实用方法。

关键词：边缘检测，线性滤波器，Canny 准则，离散准则，图像处理

ABSTRACT

Edge detection is important in image procession. This paper made a summary and analysis of edge detecting algorithm and edge detector. Canny has proposed three continuous criteria to compare the performance of different filters: good detection, good localization and low-responses. Based on these criteria he got optimal filter for edge detection: derivative of Gaussian function. After more than ten years research, Canny's theory has been ameliorated in many aspects, this paper also made a review of it. Based on the practice and theory. Demigny gave three discrete criteria for edge detection like Canny's criteria and he has proofed that the third criterion can be replaced by an appropriate thresholding operation. This paper used numerical method to get the optimal filter and smooth operator under the Demigny's criteria. Then I combine these filters and Canny's edge detecting technique to get an integrated edge detecting algorithm. I have implemented the algorithm using VC++. From the results of the program for many images this filter and algorithm are effective.

KEY WORDS: edge detection, linear filter, canny's cratria, discrete criteria, image processing

目 录

前 言

边缘检测是图像处理中的重要内容。边缘是图像的最基本特征。所谓边缘,是指周围像素灰度有阶跃变化或屋顶变化的那些像素的集合。 Poggio 等在[42]中说:“边缘或许对应着图像中物体(的边界)或许并没有对应着图像中物体(的边界),但是边缘具有十分令人满意的性质,它能大大地减少所要处理的信息但是又保留了图像中物体的形状信息。”并定义边缘检测为“主要是(图像的)灰度变化的度量、检测和定位”。边缘与图像中物体的边界有关但又是不同的。边缘反映的是图像灰度的不连续性[40]。

边缘在边界检测、图像分割、模式识别、机器视觉等中有很重要的作用。

边缘是边界检测的重要基础，也是外形检测的基础 [12]。边缘广泛存在于物体与背景之间、物体与物体之间,基元与基元之间，因此它也是图像分割所依赖的重要特征。

边缘检测对于物体的识别也是很重要的。主要有以下几个理由:首先,人眼通过追踪未知物体的轮廓(轮廓是由一段段的边缘片段组成的)而扫视一个未知的物体。第二,经验告诉我们:如果我们能成功地得到图像的边缘,那么图像分析就会大大简化,图像识别就会容易得多。第三,很多图像并没有具体的物体,对这些图像的理解取决于它们的纹理性质,而提取这些纹理性质与边缘检测有极其密切的关系[18]。

计算机视觉处理可以看作是为了实现某一任务从包含有大量的不相关的变量中抽取不变量，总之就是简化信息。这就意味着要扔掉一些不必要的信息而

尽可能利用物体的不变性质。而边缘就是最重要的不变性质:光线的变化显著地影响了一个区域的外观,但是不会改变它的边缘。更重要的是人的视觉系统也是对边缘很敏感的。

研究边缘检测的文章有十分多。1959 年,文献上最早提到边缘检测[17]。1965 年 L. G. Roberts 最早开始系统研究边缘检测 [23]。从那以后每年都会出现很多关于边缘检测的文章。边缘检测的重要的文章大都发表在 IEEE Trans. On Pattern Analysis and Machine Intelligence, CVGIP: Image Processing, IEEE Trans. On Image Processing(1990 年创刊), Journal of the ACM 等上。

边缘检测的方法主要有以下几种:

第一种检测梯度的最大值。由于边缘发生在图像灰度值变化比较大的地方,对应连续情形就是说是函数梯度较大的地方,所以研究比较好的求导算子就成为一种思路。Roberts 算子、Prewitt 算子和 Sobel 算子等就是比较简单而常用的例子。还有一种比较直观的方法就是利用当前像素邻域中的一些像素值拟合一个曲面,然后求这个连续曲面在当前像素处梯度。从统计角度来说,我们可以通过回归分析得到一个曲面,然后也可以做类似的处理。

第二种是检测二阶导数的零交叉点。这是因为缘处的梯度取得最大值(正的或者负的),也就是灰度图像的拐点是边缘。从分析学上我们知道,拐点处函数的二阶导数是 0。

第三种,统计型方法。比如说利用假设检验来检测边缘[17],D.H.Marimont 在[6]中利用对二阶零交叉点的统计分析得到了图像中各个像素是边缘的概率,并进而得到边缘检测的方案。

第四种，小波多尺度边缘检测。九十年代,随着小波分析的迅速发展，小波开始用于边缘检测。作为研究非平稳信号的利器,小波在边缘检测方面具有得天独厚的优势[50][51]，Mallat 在这一方面做了不少工作。

当然还有其它一些方法,比如说模糊数学的方法,最近提出来的利用边缘流(Edgeflow)来检测边缘[49]，基于积分变换的边缘检测方法[55]，还有基于张量的边缘检测方法[56]等等。

那种方法是比较好呢?一般地说这个问题是没有意义的,因为方法好不好依赖于具体的应用领域。但是这些不同的领域之间还是存在着一些共同要求。1986 年 Canny[14]总结了以往理论和实践的成果,提出边缘检测 Canny 三准则:好的检测结果,好的定位还有对单一边缘低重复响应，并给出了他们的数学表达式。这是一个连续准则。利用这些准则的数学表达式,可以求出特定边缘的最优线性滤波器,也可以对不同的算子的性能进行比较。

在 Canny 三准则的基础上,人们进行了深入研究,得到了很多结果[11][22][28][33]。但是由于 Canny 准则是连续准则,得到的最优滤波器也是一个连续函数,这就需要抽样得到离散模板。可是理论和实践[3]告诉我们,在连续情形下好的滤波器实际上并不好。

Didier Demigny 通过长期的研究[5][4][3][1],最终提出了边缘检测质量的离散准则 [1]，他发现在这种离散准则下第三个准则的作用可以被阀值操作代替，并在此基础上得到了这个离散 Canny 准则下的最优线性滤波器。

我自己做的工作有以下几个方面：首先,阅读了大量的有关边缘检测的论文,对边缘检测和 Canny 准则的发展分别做了综述。第二，Didier Demigny 的论文

[1]中只给出了离散 Canny 准则和准则下的最优滤波器的计算思路,但是没有给出最终的结果。我用 Mathematica 计算出了宽度为 1-20 的 Didier Demigny 所说的最优线性滤波器及其对应的平滑算子。第三,使用这个滤波器和 Canny 边缘检测技术得到了一个比较好的边缘检测方法,并用 Vc++实现了它。

1. 背景：数字图像处理与边缘检测

1.1　数字图像及数字图像处理

每天我们都是在报纸、杂志、书籍、电视、各种小册子等大量的图像信息包围中度过的。这些图像包括文字、照片、图表、插图等，它使我们感到安适和生活情趣。

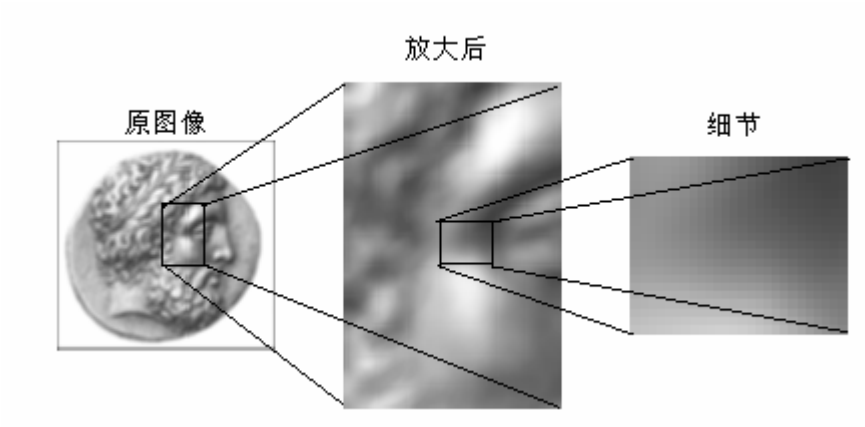
视觉是人类从大自然获取信息的最主要的来源。据统计在人类获取的信息当中，视觉信息约占 60%，听觉信息约占 20%，其它的如味觉信息，触觉信息等加起来约占 20%。由此可见视觉信息对人们的重要性。而图像正是人类获取视觉信息的主要途径。图像是用各种观测系统以不同形式和手段观测客观世界而获得的,可以直接或者间接作用于人眼并进而产生视知觉的实体[53]。

为了能严格地用数学来研究图像的边缘检测（我们只研究灰度图像的边缘检测），我们有必要对数字图像做理论假设：

A. 图像是一个二元连续函 $f(x,y):(x,y)\in D$。 函数的定义域设为 D, (x,y)表示二维空间中某个点的坐标， $f(x,y)$表示(x,y)点的灰度值，值域为 V;

B. 数字图像是对函数 $f(x,y),(x,y)\in D$ 的离散表示。在空间域 D 上进行抽样:用有限个像素(Pixel)来表示定义域 D, 每一个像素表示对应区域的平均灰度值);在值域空间 V 进行量化：用有限个值代表 V;

C.由于机器设备等的原因这些像素值都是有误差的或者是带有随机噪声的;



147 150 152 153 153 151 148 144 139 133 127 120
147 149 151 152 151 149 145 141 136 130 124 118
146 148 149 149 148 146 142 138 133 127 121 115
145 147 148 147 146 143 140 136 130 124 119 113
144 146 146 145 143 141 137 132 127 121 116 110
143 145 145 143 142 139 135 130 124 119 114 109
143 145 144 143 140 136 132 128 123 118 113 108
144 145 144 142 139 135 131 127 123 118 113 109
145 146 145 142 138 135 131 127 123 119 115 111
146 146 145 142 139 136 133 129 126 122 117 114
147 148 146 144 142 139 136 133 129 125 121 118
149 150 149 147 145 142 140 137 133 130 126 123
152 152 152 150 149 147 145 142 139 135 133 130
156 156 156 155 153 152 150 148 145 142 139 137
160 161 161 160 159 158 157 155 153 150 148 146

图 1.1　数字图像及其矩阵表示

D. 显然我们得到的是一个矩阵, 矩阵中每一个元代表一个像素, 像素的取值代表这个像素的灰度值。因此在图像的离散模型中我们也常用 M 表示图像, 使用 $m(i, j)$ 代表图像的第 (i, j) 元。

所谓数字图像处理就是利用计算机对图像信息进行加工以满足人的视觉心理或者应用需求的行为。利用计算机进行图像处理有两目的:一是产生更适合人

观察和识别的图像,而是希望计算机能自动识别和处理图像。无论为了那一种目的,图像处理中关键的一步就是对含有大量各式各样景物信息的图像进行分解,分解的最终结果是图像被分解成一些具有某种特征的最小成分:称为图像的基元。相对于整幅图像来说,这种基元更容易被快速处理。

图像的特征指图像场中可用做标志的属性。他可以分为图像的统计特征和图像的视觉特征。图像的统计特征指的是一些人为定义的特征,通过变换才能得到,如图像的直方图、距、频谱等等;图像的视觉特征指的是人的视觉可以直接感受到的自然特征,如区域的亮度,纹理或轮廓等。利用这两类特征把图像分解为一系列有意义的目标或区域的过程称为图像的分割。

1.2 边缘及边缘检测

边缘是图像的最重要的特征。

边缘是指周围像素灰度有阶跃变化或屋顶变化的那些像素的集合。Poggio等在[42]中说:“边缘或许对应着图像中物体(的边界)或许并没有对应着图像中物体(的边界),但是边缘具有十分令人满意的性质,它能大大地减少所要处理的信息但是又保留了图像中物体的形状信息”。

常见的边缘点有三种。第一种是阶梯形边缘(Step-edge),即从一个灰度到比它高好多的另一个灰度。第二种是屋顶型边缘(Roof-edge),它的灰度是慢慢增加到一定程度然后慢慢减小。还有一种是线性边缘(Line-edge),它的灰度从一级别跳到另一个灰度级别之后然后回来。

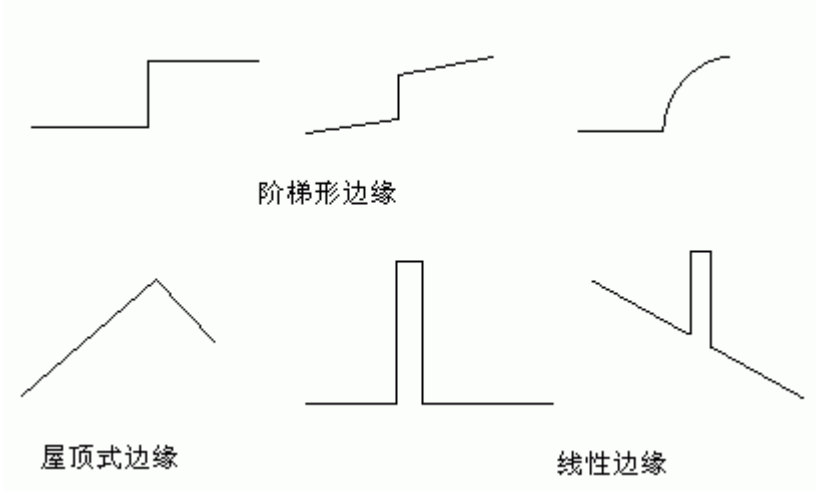


图 1.2 边缘的类型

各种不同的边缘有不同的特征。 Nalwa 和 Binford[44]认为图像中的边缘可以由许多的短直线段(他称之为 edgel,我们下面称之为边缘元)来逼近,每一个边缘元都由一个位置和一个角度确定。边缘元对应着图像上灰度曲面阶数的不连续性。如果灰度曲面在一个点的 N 阶导数是一个 Delta 函数,那么我们就定义灰度曲面在这个点是 N 阶不连续的。那么线性边缘是 0 阶不连续的,阶梯形边缘的一阶不连续的,而屋顶形边缘是 2 阶不连续的。

在大部分情况下,我们都是把图像的边缘全部看作是阶梯形边缘,然后求得检测这种边缘的各种最优滤波器，然后用在实践中。

边缘检测“主要是(图像的)灰度变化的度量、检测和定位”[42]。有很多种不同的边缘检测方法,同一种方法使用的滤波器也不尽相同。边缘检测就是研究更好的边缘检测方法和检测算子。

1.3 论文总体结构

第二章我们将对边缘检测各种算法和算子进行综合介绍。

第三章主要是 Canny 提出的边缘检测质量的连续准则,以及后来的发展。

第四章介绍 Didier Demigny 最新给出的更好的边缘检测的离散准则,并给出最优线性算子的计算及其程序。

第五章综合上述结论得到了一种完整的边缘检测算法。

第六章介绍算法的 VC++实现和算法的结果。

第七章是结论。

2. 边缘检测综述

在本章中我们将对目前出现的边缘检测做一个比较全面的介绍[40][41]。我基本上是按照这些算法提出的时间顺序给出的。1986 年 Canny 提出了边缘检测质量的三条准则，对边缘检测的发展有很大的作用，由于这部分内容我们将在后面几章中详细给出，所以这里就不再涉及。

2.1 Roberts 算子

边缘，是指周围像素灰度有阶跃变化或屋顶等变化的那些像素的集合。图像的边缘对应着图像灰度的不连续性。显然图像的边缘很少是从一个灰度跳到另一个灰度这样的理想状况。真实图像的边缘通常都具有有限的宽度呈现出陡峭的斜坡状。

边缘的锐利程度由图像灰度的梯度决定。梯度是一个向量，∇f 指出灰度变化的最快的方向和数量。

$$\nabla f = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}) \tag{2.1}$$

梯度的大小和方向是由

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \tag{2.2}$$

$$\theta = \tan^{-1}\left(\frac{\partial f / \partial y}{\partial f / \partial x}\right) \tag{2.3}$$

因此最简单的边缘检测算子是用图像的垂直和水平差分来逼近梯度算子：

$$\nabla f = (f(x,y) - f(x-1,y), f(x,y) - f(x,y-1)) \tag{2.4}$$

因此当我想寻找边缘的时候，最简单的方法是对每一个像素计算出（2.4）的向量,然后求出他的绝对值，然后进行阀值操作就可以了。利用这种思想就得到了 Roberts 算子：

$$R(i,j) = \sqrt{(f(i,j) - f(i+1,j+1))^2 + (f(i,j+1) - f(i+1,j))^2} \tag{2.5}$$

它是一个两个 2×2 模板作用的结果(标注 $\bullet$ 的是当前像素的位置)：

$$\begin{pmatrix} 1_{\bullet} & 0 \\ 0 & -1 \end{pmatrix} \qquad \begin{pmatrix} 0_{\bullet} & 1 \\ -1 & 0 \end{pmatrix}$$

图 2.1 是图 2.0 被 Roberts 算子检测的结果。

图 2.0 我的书桌 原图像

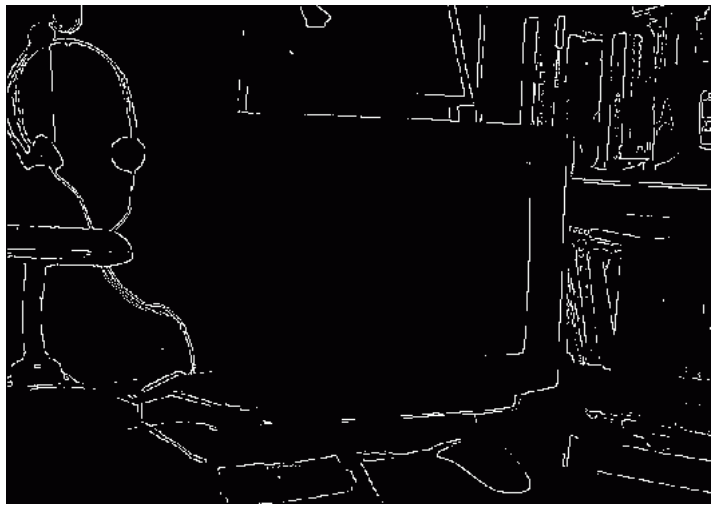


图 2.1　　我的书桌: Roberts 算子处理的结果(Matlab6.5)

能查到的最早的有关边缘检测的文献就是 1959 年 B. Jule 的[17]，这算是最早的提出边缘检测和边缘检测算子的文章了。

1963 年 Roberts[23]提出了边缘检测和边缘检测的这个简单算子。Machine Perception of 3-D Solids 是 Roberts 在 1963 年写的(MIT)博士毕业论文。这是一篇大家都应该阅读的论文。它是最早分析图像中的边缘、线、模型和图形学的文章。作者提出的系统是第一个 3D 视觉系统,其中有许多后来被大家常用的算子。它使用了三维物体的多边形模型,通过计算图像中的灰度数据寻找图像中的“块”然后对他最适当的表示，最后通过匹配来寻找物体。复杂的物体是由很多“块”组成的，而“块”是由边缘组成的。作者使用他的简单的2×2算子得到了边缘。相邻的边缘点组成线，如果线比较短，那么这条线就被忽略，如果线比较长，那么我们就把它延长直到它与其它的边缘线相交与一点。然后通过对这些线的匹配来识别物体。一旦一个物体识别出来，表示这个物体的边缘就从图像的边缘线表示上删除，然后再对下一个物体进行匹配。

2.2 Prewitt 算子和 Sobel 算子

Roberts 算子是直观的也是简单的，但是显然效果不好。实践中人们做了大量的实践，总结出了一些经验。

1970 年左右 Prewitt[16]和 Sobel[25]分别提出了一个算子，这就是 Prewitt 算子和 Sobel 算子。

Prewitt 边缘检测算子使用两个有向算子(一个水平的，一个是垂直的，一般称为模板)，每一个逼近一个偏导数：

$$P_V=\begin{pmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ 1 & 1 & 1\end{pmatrix} \qquad P_H=\begin{pmatrix}-1 & 0 & 1\\ -1 & 0 & 1\\ -1 & 0 & 1\end{pmatrix} \tag{2.6}$$

如果我们用 Prewitt 算子检测图像 M 的边缘的话，我们可以先分别用水平算子和垂直算子对图像进行卷积，得到的是两个矩阵，在不考虑边界的情形下也是和原图像同样大小的 M1，M2，他们分别表示图像 M 中相同位置处的两个偏导数。然后把 M1，M2 对应位置的两个数平方后相加得到一个新的矩阵 G,G 表示 M 中各个像素的灰度的梯度值(一个逼近)。然后就可以通过阀值处理得到边缘图像。总的过程是：

$$E=((M\otimes P_V)^2+(M\otimes P_H)^2)>Thresh^2 \tag{2.7}$$

Sobel 算子和 Prewitt 算子的不同就在于使用的模板不一样：

$$s_1=\begin{pmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{pmatrix} \qquad s_2=\begin{pmatrix}-1 & 0 & -1\\ -2 & 0 & -2\\ -1 & 0 & -1\end{pmatrix} \tag{2.8}$$

这些模板是怎么来的呢?

我们假设图像的灰度满足下面这个关系：

$$M_{x,y} = \alpha x + \beta y + \gamma \tag{2.9}$$

则梯度是(α, β)。

显然,当前像素 3×3 邻域内像素值为

$$\begin{pmatrix} -\alpha-\beta+\gamma & -\alpha+\gamma & -\alpha+\beta+\gamma \\ -\beta+\gamma & \gamma & \beta+\gamma \\ \alpha-\beta+\gamma & \alpha+\gamma & \alpha+\beta+\gamma \end{pmatrix} \tag{2.10}$$

定义垂直算子和水平算子形如：

$$\begin{pmatrix} -a & -b & -a \\ 0 & 0 & 0 \\ a & b & a \end{pmatrix} \qquad \begin{pmatrix} -a & 0 & a \\ -b & 0 & b \\ -a & 0 & a \end{pmatrix} \tag{2.11}$$

之所以这样定义是为了满足对称性和电路设计的需要。

利用这两个模板对当前像素进行卷积，得到的方向导数为

$$\begin{aligned} g_x &= 2\beta(2a+b) \\ g_y &= 2\alpha(2a+b) \end{aligned} \tag{2.12}$$

因此当前像素处的梯度的大小为

$$G = 2(2a+b)\sqrt{\alpha^2+\beta^2} \tag{2.13}$$

显然要有：

$$2(2a+b) = 1 \tag{2.14}$$

如果我们取 a=b=1/6 则得到的模板就是 1/6 乘 Prewitt 算子;

如果我们取 a=1/8，b=1/4 则得到的就是 1/8 乘 Sobel 算子。

那一个算子是好呢？这个问题的答案取决于图像的噪声，如果在每个点噪声都是相同的，那么 Prewitt 算子是比较好的；如果靠近边缘的噪声是沿着边缘的 2 倍，那么 Sobel 算子是比较好的。也就是算子的好坏取决于噪声的结构。

事实上,它们存在一些共同的问题：

A.他们的结果对噪声很敏感，图像的离散差分对噪声比对原图像更敏感；

B.可以通过先对图像做平滑以改善结果，但是又会产生一个问题：会把一些靠在一起的边缘平滑掉，而且会影响对边缘的定位；

图 2.2 我的书桌：Prewitt 算子处理结果(Matlab6.5)

C.用这些模板卷积后得到的边缘可能是跨跃好几个点而不是一个点；为了改善这个问题，还要做一些改进：边缘像素不只是要大于阀值，而且在梯度方向上梯度的大小要大于它的前者和它的后者，这个方法称之为非极值抑制；

图 2.3 我的书桌：Sobel 算子处理结果(Matlab6.5)

2.3 Kirsch 算子

1971 年，R.Kirsch[34]提出了一种边缘检测的新方法：它使用了 8 个模板来确定梯度和梯度的方向。假设原来的 3×3 子图像的如下：

$$\begin{pmatrix} a_3 & a_2 & a_1 \\ a_4 & (i,j) & a_0 \\ a_5 & a_6 & a_7 \end{pmatrix} \tag{2.15}$$

则边缘的梯度大小为

$$m(i,j)=\max\{1,\max\{|5s_k-4t_k|:k=0,1,\ldots 7\}\} \tag{2.16}$$

其中

$$\begin{aligned} s_k &= a_k + a_{k+1} + a_{k+2} \\ t_k &= a_{k+3} + a_{k+4} + \ldots + a_{k+7} \end{aligned} \tag{2.17}$$

上面的下标如果超过 7 就用 8 去除取余数。.

注意到 k=0，1，...7，其实就是使用了 8 个模板了，参见[18]。

2.4 Hueckel 算法

1971、1973 年, Hueckel 两次在 ACM 上发表文章[26][27]，提出了一个新的算法。 Hueckel 算子的基本思想是：(以一维为例，假设图像是 $M(x)$) 我们利用图像数据拟和一个理想的边缘模型

$$s(x)=\begin{cases} b & if \quad x<x_0 \\ b+h & if \quad x\geq x_0 \end{cases} \tag{2.18}$$

如果这种拟和是比较精确的，那么在这个位置就应该存在一个与这个理想边缘有相同参数的边缘。也就是说如果图像 $M(x)$ 与 s(x)的平均平方误差 E 低于阀值,那么他就是边缘。

$$E=\int_{x_0-l}^{x_0+l}[M(x)-s(x)]^2dx \tag{2.19}$$

二维情况更复杂，拟合曲面定义为

$$s(x,y)=\begin{cases} b & if \quad x\cos(\theta)+y\sin(\theta)<\rho \\ b+h & if \quad x\cos(\theta)+y\sin(\theta)\geq\rho \end{cases} \tag{2.20}$$

拟合误差 E 定义为

$$E=\iint_D[M(x,y)-s(x,y)]^2dxdy \tag{2.21}$$

算法的实现请参考[26][27]。

2.5 Frei 和 Chen 算法

1977 年，Frei 和 Chen[45]又提出了一种新的边缘算子，这种算子有 9 个模板。原图像的一个 3×3 子图像可以表示为一个 9 维向量。

$$\begin{pmatrix} b_4 & b_3 & b_2 \\ b_5 & b_0 & b_1 \\ b_6 & b_7 & b_8 \end{pmatrix} \tag{2.22}$$

就可以用向量 b 来表示：

$$b = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_8 \end{pmatrix} \tag{2.23}$$

用 V 表示 3×3 子图像的向量空间， B_V 表示 V 的正交基，

$$\begin{pmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & -1 & \sqrt{2} \\ 1 & 0 & -1 \\ -\sqrt{2} & 1 & 0 \end{pmatrix} \begin{pmatrix} \sqrt{2} & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & -\sqrt{2} \end{pmatrix}$$

V_1 V_2 V_3 V_4

边缘子空间基

$$\begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{pmatrix} \quad \begin{pmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{pmatrix} \quad \begin{pmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{pmatrix}$$

V_5 V_6 V_7 V_8

直线空间基

图 2.4 Frei-Chen 中使用的 B_V 基（待续）

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$V_9$$

图 2.4 Frei-Chen 中使用的 B_V 基（续）

令 θ_E 是向量 b 和 b 在边缘空间上的投影，则有

$$\theta_E = \cos^{-1}\sqrt{\frac{\sum_{i=1}^{4}(V_i \bullet b)^2}{\sum_{i=1}^{9}(V_i \bullet b)^2}} \tag{2.24}$$

这里的 $\bullet$ 表示：

$$b \bullet c = \sum_{i=0}^{8} b_i c_i \tag{2.25}$$

对每一个点都算出这个 θ_E 后，就需要进行阀值操作：给定一个 T>0，如果 $\theta_E < T$ 就认为是边缘，否则就是不是边缘。

2.6 Wallis 对数算子[48]

如果一个像素它的灰度的对数值超过其四邻接像素对数值的平均值 T(T 是一个固定的阀值) 以上，则这个像素就是边缘。像素的四邻接像素：

$$\begin{pmatrix} & a_1 & \\ a_2 & (i,j) & a_0 \\ & a_3 & \end{pmatrix} \tag{2.26}$$

则

$$b(i,j)=\log_b(a(i,j))-\frac{1}{4}(\log_b(a_0)+\log_b(a_1)+\log_b(a_2)+\log_b(a_3)) \quad (2.27)$$

到 1980 年的时候已经有了很多的边缘检测的算子，在实践中人们也积累了很多的经验。大家发现尺度选择和噪声处理的问题是边缘检测的最重要的问题。

[10]中给出了一些 1980 年前有关边缘检测的参考文献。

2.7　Marr 和 Hildreth 的零交叉点算子

1980 年，Marr 和 Hildreth[9]提出了一种新的边缘检测理论和技术。

前面我们都是利用边缘处的梯度取得最大值(正的或者负的)，也就是灰度图像的拐点位置是边缘。我们知道，在拐点位置二阶导数是 0。如图 2.5 所示，

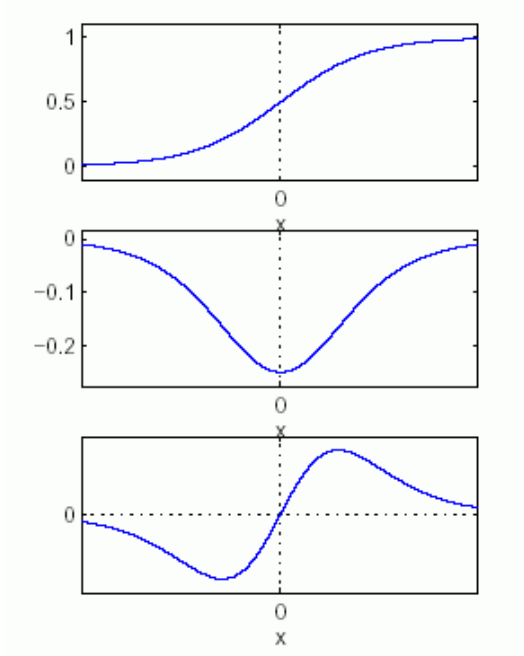


图 2.5 阶梯边缘、局部极值与零交叉算子

所以我们也可以通过寻找二阶导数的零交叉点来寻找边缘。二元函数 f(x,y) 的 Laplacian 变换定义为：

$$\nabla^2 f = (\frac{\partial}{\partial x^2} + \frac{\partial}{\partial y^2}) f(x,y) = \frac{\partial f(x,y)}{\partial x^2} + \frac{\partial f(x,y)}{\partial y^2} \qquad (2.28)$$

也就是二阶偏导数的和。

和前面的梯度算子一样，我们这里也是在给定的噪声假设下我们选择一个模板，然后利用模板进行卷积求离散 Laplacian 变换，然后通过寻找零交叉点寻找边缘点。

最常用的模板是

$$L = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \qquad L = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix} \qquad (2.29)$$

显然，既然一阶导数对噪声是敏感的(因而是不稳定的)；那么二阶导数对噪声就会更敏感，因而更不稳定。所以在做 L 变换之前做平滑是必需的了。

又因为卷积是可结合可交换的，所以先做高斯卷积然后用 L 算子做卷积就等价于对原图像用高斯函数的 Laplacian 变换后的滤波器做卷积是一样的。 这就是一个新的滤波器 LoG 滤波器(是墨西哥帽函数)：

$$\begin{aligned} LoG(x,y) &= \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\frac{1}{2\pi\sigma^2}\exp\left(-\frac{1}{2}\left(\frac{x^2+y^2}{\sigma^2}\right)\right) \\ &= \frac{-1}{2\pi\sigma^4}\left(2-\left(\frac{x^2+y^2}{\sigma^2}\right)\right)\exp\left(-\frac{1}{2}\left(\frac{x^2+y^2}{\sigma^2}\right)\right) \end{aligned} \qquad (2.30)$$

图 2.6 是这个函数的图像。

LoG 滤波器由一个中心负区域(半径为$\sqrt{2}\sigma$)，然后有一个正区域(总的半径为最小是$3\sqrt{2}\sigma$)围绕着他。

在实际中先使用 LoG 模板做卷积，然后寻找那些零交叉像素：如果一个像素处值小于$-\theta_0$。而且其它周围的各个邻接 8 个像素都是大于θ_0，θ_0是一个正数,那么这个像素就是零交叉点。

LoG 算子有一个线性变化条件：沿着和平行于"Open"边缘的灰度变化是局部线性的。当这个条件满足的时候，LoG 算子的性能比较好；而如果这个条件不满足，则检测结果就不好。但是，对于一些小尺度边缘、边缘角落和 T 形边缘来说这些条件无法满足，所以效果就不好。LoG 算子还有一个缺点就是：它对噪声比较敏感，它检测出的是梯度图像具有局部最大值和最小值的点，但实际上边缘只是那些梯度为局部最大的点。所以说 LoG 算子多算了一些点。

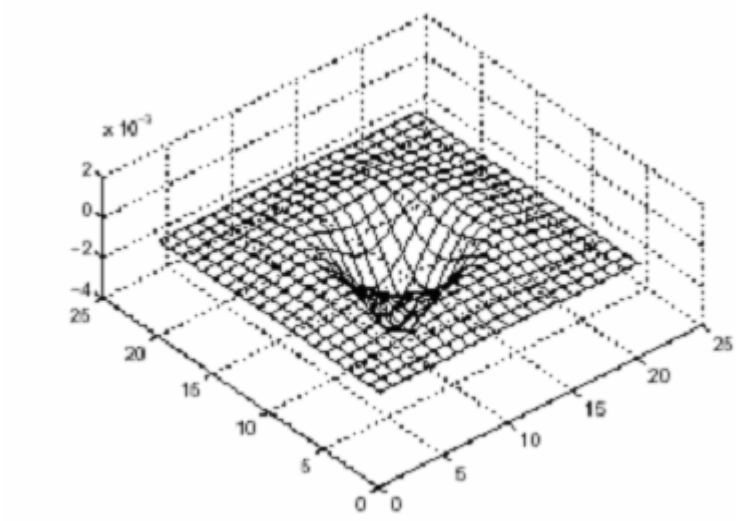


图 2.6　LoG 算子的图像

为了处理更多的范围的灰度变化，Marr 和 Hildreth 使用了 4 个不同尺寸的高斯函数，称为不同的通道。将这四个不同尺度的 LoG 滤波器同时作用于图像，然后在空间符合假设的基础上综合处理得到结论。空间符合假设指的是：导致图像上灰度变化的原因是空间局部性。所以零交叉点出现在几个不同的通道中，那么这个点就是边缘点。

有趣的是，人的视觉系统中一些早期的神经元就具有类似 LoG 的性质[8]，并且[21][43]说他们是最优的。

这个算子引起了人们的很大兴趣，1986 年 W. H. Lunscher 和 M. P. Beddoes 在[46][47]中对这个算子做了经一步的讨论。

他指出 LoG 算子的优点：

A. 和其它那些算子一样，LoG 算子也是先对边缘作出假设，然后在这个假设下寻找边缘像素。但是，LoG 算子对边缘的假设条件最少，因此它的应用范围更广；

B. 其它一些算子的检测得到的边缘是不连续的也是不规则的，还需要连接这些边缘，而 LoG 算子的结果就没有这个缺点；

C.可以通过一个参数(也就是高斯函数标准偏差)来调整 LoG 算子的结果。

但是到底如何选择这个参数当时并没有。于是 W. H. Lunscher 和 M. P. Beddoes 在[46]中对 LoG 算子的参数选择做了深入的讨论，并研究了在高斯噪声下的 LoG 算子的性能。由于前面得到的是一个连续的滤波器，因此还需要抽样和对系数进行量化，在[47]中作者又讨论了这个问题。这两篇论文使得 LoG 算法更加实用化。图 2.7 是 LoG 算子处理的结果

图 2.7 我的书桌 LoG 算子处理的结果(Matlab6.5)

虽然这个算子具有人的视觉系统的某些特点，但是它并不好。[18,pp.338]中作者曾十分风趣的说：“问题是人类可以制造一个类似于动物的图像感知器吗？我们制造的飞机完全不同于鸟：我们需要大马力的引擎,它们要满足的是物理定律而不是生物定律。”

2.8 Facet 模型

1984 年 IEEE 的资深会员 Robert M. Haralick[35] 对前面的工作做了一个小结,提出了 Facet 模型。Facet 模型的本质是：任何利用一个像素的邻域中像素的灰度值分析边缘的方法都可以最终解释为关于这个像素邻域中一个灰度值曲面的分析，而这个邻域中的像素值无非就是这个曲面的一个有误差的抽样。

Haralick 使用离散 Chebyshew 正交多项式作为基底,利用当前像素邻域中的灰度值拟合出一个二元三次多项式。不妨记为

$$f(r,c)=k_1+k_2r+k_3c+k_4r^2 \\ +k_5rc+k_6c^2+k_7r^3+k_8r^2c+k_9rc^2+k_{10}c^3 \tag{2.31}$$

这里的 r,c 分别表示的是行和列。

那么沿着α (表示从r轴开始沿顺时针方向的方向角)方向的方向导数定义为

$$f_\alpha^{'}(r,c)=\lim_{h\to 0}\frac{f(r+h\sin\alpha,c+h\cos\alpha)-f(r,c)}{h} \tag{2.32}$$

这个式子等价于

$$f_\alpha^{'}(r,c)=\frac{\partial f}{\partial r}(r,c)\sin\alpha+\frac{\partial f}{\partial c}(r,c)\cos\alpha \tag{2.33}$$

同理有

$$f_\alpha''=\frac{\partial^2 f}{\partial r^2}\sin^2\alpha+\frac{\partial^2 f}{\partial r\partial c}\sin\alpha\cos\alpha+\frac{\partial^2 f}{\partial c^2}\cos^2\alpha \tag{2.34}$$

从而有梯度方向角α 由下式决定：

$$\sin\alpha=\frac{k_2}{\sqrt{k_2^2+k_3^2}} \tag{2.35}$$

$$\cos\alpha=\frac{k_3}{\sqrt{k_2^2+k_3^2}} \tag{2.36}$$

因为我们只希望得到沿着α 方向的导数，所以

令

$$r=\rho\sin\alpha \\ c=\rho\cos\alpha \tag{2.37}$$

则

$$f''_\alpha(\rho)=6[k_7\sin^3\alpha+k_8\sin^2\alpha\cos\alpha+k_9\sin\alpha\cos^2\alpha+k_{10}\cos^3\alpha]\rho+2[k_4\sin^2\alpha+k_5\sin\alpha\cos\alpha+k_6\cos^2\alpha] \tag{2.38}$$

记为

$$f''_\alpha(\rho)=A\rho+B \tag{2.39}$$

给定一个阀值 ρ_0 (它比像素边长稍微小一点)，如果对于一些$|\rho|<\rho_0$有

$$f''_\alpha(\rho)<0\text{或者}f''_\alpha(\rho)=0\text{并且}f'_\alpha(\rho)\neq 0 \tag{2.40}$$

那么当前像素就是边缘点。

作者花了很大的篇幅从统计学的角度讨论了这种方法的有效性，并和其它一些算子尤其是 Marr 和 Hildreth[9]的零交叉点算子做了比较。

作者发现：随着模板变宽检测的结果会变好(噪声就变少)，可是边缘变的比较粗，也就是对边缘的定位就会变差，对一个边缘点会出现多个响应。

2.9 Nalwa 算法

1986 年, Vishvjit S. Nalwa 和 Thomas O. Binford 在[44]中提出了一种边缘(Step-edges)检测的新方法。他用的也是曲面拟和的方法，与其它曲面拟合的方法不同的是。它采用了很多个不同的基底去拟合以检测边缘。

Nalwa 和 Binford 认为图像中的边缘可以由许多的短直线段(他称之为 edgel, 我们下面称之为边缘元)来逼近，每一个边缘元都由一个位置和一个角度确定。边缘元对应着图像上灰度曲面阶数的不连续性。如果灰度曲面在一个点的 N 阶导数是一个 Delta 函数，那么我们就定义灰度曲面在这个点是 N 阶不连续的。

那么如图 2.8 所示，线性边缘是 0 阶不连续的，阶梯形边缘的一阶不连续的，而屋顶形边缘是 2 阶不连续的。我们的目的就是寻找这些边缘元，然后利用边缘连接技术把这些边缘元连接起来即可。

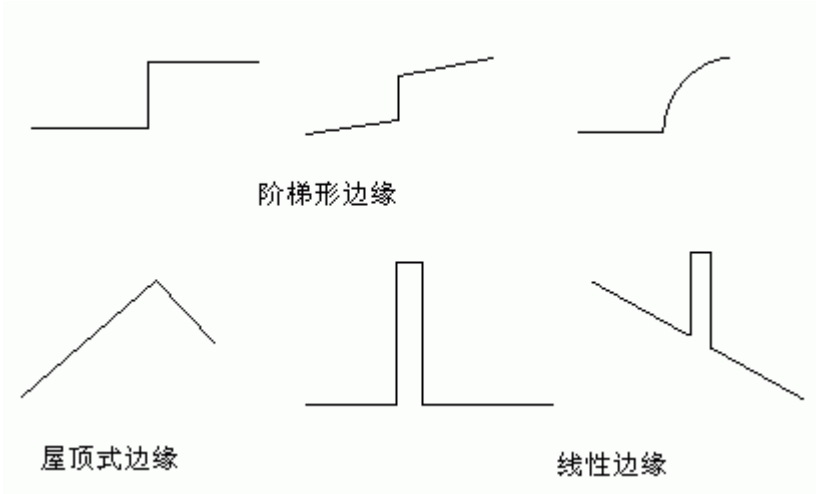


图 2.8　边缘的类型与边缘的不连续性阶数

他还认为当灰度曲面被设备转化为数字图像时，都要经过一个“模糊(Bluring)”的过程。这个过程可以用高斯滤波来逼近。尽管“模糊”过程限制了分辨率，可是并不是说它完全是不好的，因为这种“模糊”过程同时也在抽样前限制了信号的频带。如果没有这种模糊化就会带来剧烈的 Aliasing 现象。这种模糊化的结果就是图像上没有了灰度的不连续性。

他们把一般的阶梯形边缘定义为：

$$E(x)=\begin{cases}k_1 x & x<0\\ k_2 x+S & x>0\end{cases} \qquad (2.41)$$

也就是边缘两边都是一定斜率 k1，k2 的直线，阶梯高度为 S。理想的阶梯形边缘是它的特殊情况 k1=k2=0。经过图像系统后，图像就变为 E(x)*G(x),注意这

里的*表示卷积，G(x)表示的是一定参数的高斯函数

$$G(x) = ne^{-x^2/2\sigma_{blur}^2} \tag{2.43}$$

于是,

$$\begin{aligned}(E(x)*G(x))'' &= E(x)*G(x)'' \\ &= \int_x^{+\infty} k_1(x-u)G''(u)du + \int_{-\infty}^{x}[k_2(x-u)+S]G''(u)du \\ &= \int_{-\infty}^{+\infty} k_1(x-u)G''(u)du + \int_{-\infty}^{x}[(k_2-k_1)(x-u)+S]G''(u)du \\ &= [S-(k_2-k_1)x]G'(x) + (k_2-k_1)[xG'(x)-G(x)] \\ &= SG'(x) - (k_2-k_1)G(x)\end{aligned} \tag{2.44}$$

令（2.44）等于 0，计算得到

$$x = \Delta_{slope}\sigma_{blur}^2 / s, \qquad \Delta_{slope} = k_1 - k_2 \tag{2.45}$$

显然这里的 x 就是零交叉点算法的偏差。

很多的人都说高斯预滤波处理可以减少噪声，于是[9]中用 LoG 算子做卷积(等价于先用高斯函数滤波,然后用 Lalacian 算子求二阶导数)，通过检测零

128	128	128	124	108
128	127	118	97	75
124	110	86	69	64
99	76	66	64	64
70	64	64	64	64

原图像(不含噪声)

135	132	132	127	118
121	133	110	101	88
111	110	72	75	61
108	78	61	69	71
71	63	53	61	66

图像(含有噪声)

斜直线分别代表的真正的边缘和检测出的边缘

图 2.9 一个图像块的例子

交叉点来检测边缘。事实上由于两个高斯函数的积是可以看作是一个高斯函数

（新的高斯函数的参数将大于原来任何一个的参数）；利用上述计算过程可以看到：这只能更加破坏边缘的定位。这就说明了 LoG 算子的对边缘的定位很不好。

作者提出了一种方法(以图 2.9 中这个5×5的图像块为例)：

使用5×5的模板记为 Image[x,y]，以步长为 1 像素沿着 x 和 y 方向移动,遍历整个图像。

A. 首先用最小二乘方法拟合出一个平面

$$\underset{a_0,a_x,a_y}{Min}\ \xi_p = \sum_{x,y=0}^{4} (Image[x,y]-(a_0+a_x x+a_y y))^2 \qquad (2.46)$$

求得的拟合平面为

$$I[x,y]=74.73-7.34x+16.52y \qquad (2.47)$$

最小二乘拟合的残差为 2683；得到梯度的方向为（这个作为下面非线性拟合的初值用）：

$$\theta_0 = tan^{-1}(\frac{16.52}{-7.34}) = 115^\circ \qquad (2.48)$$

B. 用最小二乘方法拟合出一个 3 次曲面(这个曲面沿着与梯度方向正交的方向是常数) ：

$$\underset{\substack{z=x\cos(\theta)+y\sin(\theta)\\ a_0,a_1,a_2,a_3,\theta}}{Min}\ \xi_C = \sum_{x,y=0}^{4} (Image[x,y]-(a_0+a_1 z+a_2 z^2+a_3 z^3))^2 \qquad (2.49)$$

由于这是一个非线性问题，需要用高斯迭代法求解，因此要调用 A 中方向角作为初值。

求得的曲面是

$$\begin{aligned} &I[x,y]=71.74+21.83z+6.19z^2-2.16z^3 \\ &z=x\cos(\theta)+y\sin(\theta) \\ &\theta=120^\circ \end{aligned} \qquad (2.50)$$

最小二乘拟合的残差为 1295。这里求出来的梯度方向作为最终的梯度方向。

C. (可选项) 计算 10*(2683-1295)/1295=10.7；因为 10.7>1.47(阀值,具体计算方法参见论文)。因此继续计算，否则当前检测没有检测到边缘元。

D. 再利用 B 中求出的沿着上述用最小二乘来拟合下面这个函数

$$\begin{aligned} &\underset{s,p,k}{Min}\,\xi_p=\sum_{x,y=0}^{4}(Image[x,y]-(sTanh(\frac{0.85}{0.6}(z+p)+k))^2 \\ &z=x\cos(\theta)+y\sin(\theta) \end{aligned} \qquad (2.51)$$

这里的 0.85 和 0.6 的来源请参见论文，0.6 决定了检测的尺度。

求得的拟合函数是

$$\begin{aligned} &I[x,y]=95.57+32.52Tanh((\frac{0.85}{0.6})(z-p)) \\ &p=0.9 \\ &z=x\cos(120^\circ)+y\sin(120^\circ) \end{aligned} \qquad (2.52)$$

最小二乘拟合的残差为 1203，这里的 p 就是沿着 Z 轴边缘元的位置

E. 再沿着这个方向拟合一个函数

$$\begin{aligned} &\underset{a_0,a_1,a_2}{Min}\,\xi_Q=\sum_{x,y=0}^{4}(Image[x,y]-(a_0+a_1z+a_2z^2))^2 \\ &z=x\cos(120^\circ)+y\sin(120^\circ) \end{aligned} \qquad (2.53)$$

得到的结果是

$$I[x,y]=77.80+15.86z+1.45z^2$$
$$z=x\cos(120^\circ)+y\sin(120^\circ) \tag{2.54}$$

最小二乘拟合的残差为 2615

检查 D、E 中最小二乘拟合的残差，因为 2615＞1203 所以本次检测检测到了边缘元。

F. 下面求边缘元的位置和方向：

由于梯度的方向角是 120 度,而边缘元的方向和它正交,所以边缘元的方向是 30 度。D 中的 p 指的是边缘位于离当前的原点(5X5 矩阵的左下角)沿着 Z(120 度角)方向 0.9 像素处。

他特别指出他没有说这种方法是最优的。他说前面有很多人在论文中说自己的方法是最优的,事实上他们所说的最优只有他们所假定的条件满足的时候才使最优的。

但是,这种方法实在是太麻烦了！要遍历所有像素，每一次都要至少求两个拟合多项式(而且大部分像素位置都要计算 4 个拟合多项式)。而且第二个多项式还得通过搜索法求角，其复杂性可想而知。

2.10 Torre 和 Poggio:图像数值导数是病态问题

1986 年 Vincent Torre 和 Tomaso A. Poggio[43]指出：数字图像的边缘检测本质上是一个数值导数问题，但是数值导数是一个病态问题。所谓病态问题是 20 世纪初 Hadamard 定义：如果一个问题的解满足:

A. 存在;

B. 唯一;

C. 连续依赖于初始数据。

那么这个问题就是良态问题,否则就是病态问题。

因此再求数值导数前先要做滤波处理以使得这个问题良态化。根据数学上已经成熟的理论,, Torre 和 Poggio 找到了几种方法，也可以参考[50]。因此,边缘检测由两步组成：先对数字图像滤波使它变为良态问题，然后再用导数算子求梯度。作者对具有最小不确定性(Hermite 函数和 Gabor 函数)，具有带通属性(Sinc 函数)和有限支撑的滤波函数进行了讨论。 综合起来考虑计算复杂性和把病态问题装化为良态问题的性能，具有最小不确定性的滤波器是最好的选择。从实践的角度来讲高斯函数是不错的滤波器。在二维时，高斯函数滤波可以分解为两个一维滤波器的合成，这就更加简化了计算的复杂性。

Torre 和 Poggi 证明利用旋转对称的滤波器(如高斯滤波器)滤波可以保证零交叉点算法中边缘的封闭性的高概率。利用方向滤波器滤波无法保证这一点，但是定位更准确。

Torre 和 Poggi 证明在一般情况下要用多个不同尺度的各种不同类型的求导算子来检测灰度的变化。求利用梯度算子求梯度的时候，只需要求两方向的导数即可。当采用旋转不变算子的时候，$\partial^2/\partial\vec{n}^2$ 要比 ∇^2 的定位要好，可是前者由于没有与卷积的交换性，所以计算复杂性太大。

在这篇论文中， Torre 和 Poggi 使用了很多数学方法证明了很多结论，提出了很多解决问题的方法，因为本文受到十分广泛的关注。

2.11 统计变点算法

1988年 Jun S. Huang和 Dong H. Tseng在[18]中从概率统计的角度对边缘检测进行了讨论。他们指出,边缘检测中最大的问题是噪声对检测结果的影响。比如说,在边缘检测中一个很大的问题是阀值的选择，要想有好的检测结果就要十分恰当地选择阀值，但是没有一般的好方法。虽然有些人提出了一些方法，但是事实上也不怎么好：Frei 和 Chen[45]提出利用图像的几何性质得到一种避免了调节阀值的方法，但这种方法还是对噪声十分敏感的；J.Prager([18]的引文4)提出的松弛方法，但是何时停止这个问题没有解决并且时间也太慢；Haralick提出[35]了 Slope Facet 模型，本质上说他是实用量三个参数的回归分析,可是其中的回归变点问题(regressional change-point)还没有解决，作者通过分析各种边缘，发现这个问题的本质是多决策理论：一个简单的 F 检验是无法解决问题的。为了解决这个问题， Jun S. Huang 和 Dong H. Tseng 提出来统计变点理论。

把图像分成很多小块，每一块都是一个$n\times n$的正方形，使得这个正方形中最多只有一个边缘。显然这里的 n 是由图像的分辨率决定的，而且还是比较小的。考察这些小块就不难发现有四种类型, 如图 3.10 所示。

我们发现，对于(b)，(c)和(d)这些有边缘的情形,都是在这个小块中存在灰度值相差较大的两块区域。我们用$X_i, i=1,2, n^2=N$表示这个$n\times n$小块的灰度值。由于各种噪声的作用，它是随机的。一般地说，图像上全局噪声不是一致的,但是在局部却是相关的，在这个小块上我们假设噪声是独立正态分布。

下面我们重新表示一下上述灰度值。在第一块区域中像素灰度值为

$$\{X_i, i=1,2,m\} \tag{2.55}$$

在第二块区域中像素灰度值为

$$\{X_i', i=1,2,m'\} \tag{2.56}$$

这里有 $m+m'=N$

12	11	10	12	10
9	10	9	11	10
11	9	11	9	12
10	11	10	12	11
11	11	10	11	10

(a)一致区域

22	101	100	99	102
21	100	100	101	20
23	99	102	100	21
20	102	100	20	23
21	21	20	21	20

(b)阶梯形边缘

由两块组成,每一块至少有三个像素

1	2	91	93	0
1	0	94	92	1
0	91	93	1	2
2	92	90	2	0
1	94	91	0	1

(c)线

有一条线通过一致区域

0	1	1	2	0
1	2	81	82	1
1	83	80	85	1
2	0	1	0	2
0	1	2	1	1

(d) 点

在一致区域上由一个点,至少由三个

像素组成

图 3.10　$n\times n$ 子块的几种情形

(注意:上述对像素数目的要求主要是为了使得统计规律能满足。)

再令

$$X_i \square N(\mu_i, \sigma^2) \tag{2.57}$$

$$X_i' \square N(\mu_i', \sigma^2) \tag{2.58}$$

那么就是说在给定分法(P)的时候存在着下面的这样一个条件假设：

$$\begin{aligned} &H_0 : \mu_1 = \mu_2 = \cdots = \mu_m = \mu_1' = \cdots \mu_m' = \mu_0 \\ &against \\ &H_a : \mu_1 = \mu_2 = \cdots = \mu_m \neq \mu_1' = \cdots \mu_m' \end{aligned} \tag{2.59}$$

因为 n 比较小,所以上述两个假设至少有一个是真的(当然也有可能都不对，可是这样概率太小，所以忽略不计了)如果前者是真的，这就是一个一致区域，不存在边缘；如果后者是真的,那么这种划分就对应着边缘。通过统计分析构造了统计量

$$W^2 = \max_p \frac{mm}{N} \frac{\left|\bar{X} - \bar{X}'\right|^2}{S_p^2} \tag{2.60}$$

其中

$$S_p^2 = \left[\sum_{i=1}^{m} (X_i - \bar{X})^2 + \sum_{i=1}^{m'} (X_i' - \bar{X}')^2 \right] / (N-2) \tag{2.61}$$

P 代表一种对这块分为两块的分法。

如果$W^2 > \xi_a$则拒绝H_0，也就是说这不是一个一致区域；那么这个时候的使得W^2最大的分组就对应着边缘。在论文中作者给出了ξ_a经过数值计算结果。

2.12 边缘检测的 Green 函数方法

Torreao 和 Amaral 利用 Green 函数的性质得到了一种边缘检测的新方法。下面我们给出一维时的公式，这些公式都可以简单地推到二维。假设信号是$I(x)$，我们的任务就是要计算它的导数。

$$I'(x)=\lim_{u\to 0}\frac{I(x+u)-I(x-u)}{2u} \tag{2.62}$$

令

$$\tilde{I}(x)=I(x-u) \tag{2.63}$$

则有

$$\tilde{I}(x+u)=I(x) \tag{2.64}$$

把 $\tilde{I}(x)$ 泰勒展开

$$\frac{u^2}{2}\tilde{I}''+u\tilde{I}'+\tilde{I}=I \tag{2.65}$$

显然 $\tilde{I}(x)$ 可以通过求解（2.65）这个微分方程得到。假设边界条件为

$$\lim_{x\to\pm\infty}\tilde{I}(x)=0 \tag{2.66}$$

则有

$$\tilde{I}(x)\equiv I(x-u)=\int_{-\infty}^{+\infty}G_2(\xi)I(x-\xi)d\xi \tag{2.67}$$

其中 G_2 为（2.68）定义的 Green 函数

$$G_2(\xi)=\begin{cases}0, & \xi<0\\ \dfrac{2}{u}\sin\left(\dfrac{\xi}{u}\right)\exp\left\{-\dfrac{\xi}{u}\right\}, & \xi>0\end{cases} \tag{2.68}$$

它是（2.69）的解

$$\frac{u^2}{2}\tilde{I}''+u\tilde{I}'+\tilde{I}=\delta(x-x') \tag{2.69}$$

同理可以得到

$$\hat{I}(x)\equiv I(x+u)=\int_{-\infty}^{+\infty}G_2(-\xi)I(x-\xi)d\xi \tag{2.70}$$

从而

$$I'(x)=\lim_{u\to 0}\frac{I(x+u)-I(x-u)}{2u}=\lim_{u\to 0}\frac{\hat{I}(x)-\tilde{I}(x)}{2u}$$
$$=\lim_{u\to 0}\frac{1}{2u}\int_{-\infty}^{+\infty}[G_2(-\xi)-G_2(\xi)]I(x-\xi)d\xi \qquad (2.71)$$

于是，如果我们定义

$$D_2(x)\triangleq\frac{1}{2u}\left[[G_2(-\xi)-G_2(\xi)\right] \qquad (2.72)$$

则当$u\to 0$的时候，$D_2(x)$就是一个微分系统的冲击响应。

同样的道理，（2.65）可以换为（2.73）

$$\frac{u^3}{6}\tilde{I}'''+\frac{u^2}{2}\tilde{I}''+u\tilde{I}'+\tilde{I}=I \qquad (2.73)$$

则得到一个新的滤波器

$$D_3(x)\triangleq\frac{1}{2u}[G_3(-x)-G_3(x)] \qquad (2.74)$$

其中

$$G_3(x)=\begin{cases}0, & \xi<0\\ A\left[\cos\phi\exp\{-\alpha\xi\}-\cos(\gamma\xi+\phi)\exp\{-\beta\xi\}\right], & \xi>0\end{cases} \qquad (2.75)$$

$$\alpha=\frac{1.596}{u},\beta=\frac{0.702}{u},\gamma=\frac{1.807}{u}$$

$$A=\frac{2}{u}\left\{\cos\phi[(\beta-\alpha)+(\alpha^2-\beta^2+\gamma^2)/3]+\gamma\sin\phi[1-2\beta/3]\right\}^{-1} \qquad (2.76)$$

还可以把 $D_2(x)$ 和 $D_3(x)$ 结合起来得到一个更好的滤波器。图 2.11 给出了这两个滤波器的图像。

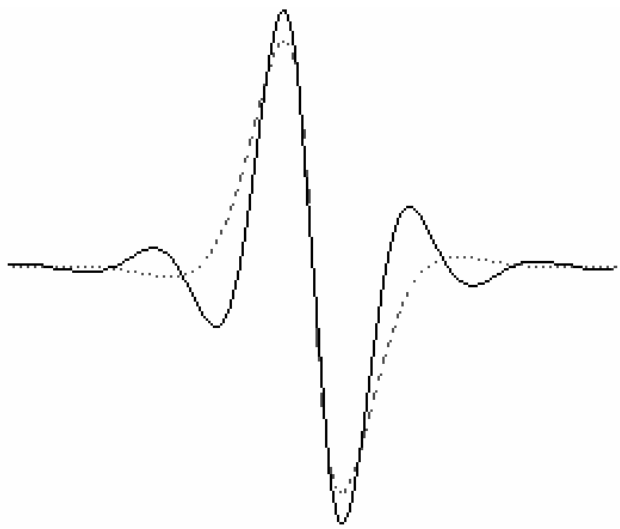


图 2.11　$D_2(x)$（虚线）和 $D_3(x)$（实线）两个滤波器的图像

（在相同的 u 下）

2.13 小波多尺度边缘检测方法［51］［52］

多尺度边缘检测方法是先磨光原信号，再由磨光后信号的一阶或者二阶导数检测出原信号的剧变点（也就是边缘了）。

假设原信号是 $I(x)$，$\theta(x)$ 是磨光函数，满足

$$\int_0^1 \theta(x)dx = 1 \tag{2.77}$$

$$\lim_{x\to\pm\infty} \theta(x) = 0 \tag{2.78}$$

这里的 $\theta(x)$ 可以取 Gauss 函数和规范 B-样条.

为了磨光原始信号，运用卷积

$$I*\theta(x)=\int_{-\infty}^{+\infty}I(t)\theta(x-t)dt \tag{2.79}$$

现在分别对其求一阶导数和二阶导数，有

$$\frac{\partial}{\partial x}I*\theta(x)=\int_{-\infty}^{+\infty}I(t)\theta'(x-t)dt \tag{2.80}$$

$$\frac{\partial^2}{\partial x^2}I*\theta(x)=\int_{-\infty}^{+\infty}I(t)\theta''(x-t)dt \tag{2.81}$$

定义

$$\Psi^I(x)=\frac{\partial}{\partial x}\theta(x)\quad \Psi^{II}(x)=\frac{\partial^2}{\partial x^2}\theta(x) \tag{2.82}$$

由（2.77）和（2.78）有

$$\int_{-\infty}^{+\infty}\Psi^I(x)dx=0\quad ,\int_{-\infty}^{+\infty}\Psi^{II}(x)dx=0 \tag{2.83}$$

所以，$\Psi^I(x)$和$\Psi^{II}(x)$是小波.

记 $\theta_s(x)=\frac{1}{s}\theta(x/s)$

则

$$\Psi_s^I(x)=s\frac{\partial}{\partial x}\theta_s(x)=\frac{1}{s}\Psi^I(x/s) \tag{2.84}$$

$$\Psi_s^{II}(x)=s^2\frac{\partial^2}{\partial x^2}\theta_s(x)=\frac{1}{s}\Psi^{II}(x/s) \tag{2.85}$$

那么$I(x)$关于小波$\Psi^I(x)$和$\Psi^{II}(x)$在尺度s和位置x上的规范小波变换定义为

$$W_s^I I(x) = I * \Psi_s^I(x) = I * (s\frac{\partial \theta_s}{\partial x})(x) = s\frac{\partial}{\partial x}(I * \theta_s)(x) \qquad (2.86)$$

$$W_s^{II} I(x) = I * \Psi_s^{II}(x) = I * (s^2\frac{\partial^2 \theta_s}{\partial x^2})(x) = s^2\frac{\partial^2}{\partial x^2}(I * \theta_s)(x) \qquad (2.87)$$

这样$W_s^I I(x)$的局部极值对应着$W_s^{II} I(x)$的零交叉点和$I * \theta_s(x)$的拐点。则求边缘点就可只求$W_s^I I(x)$的局部极值或者$W_s^{II} I(x)$的零交叉点。

理论证明可以知道阶梯形边缘上点的小波变换的值不随尺度变化而变化，其它两种则不然。

也就是第一种边缘有

$$\frac{W_{s_j}^I I(x)}{W_{s_l}^I I(x)} = 1, \quad (j \neq l) \qquad (2.88)$$

由于不一定是理想的第一种边缘或者有噪声及其它情况的影响，只能约等于 1 。因此找一个适当大于 1 的 R，则有

$$\frac{1}{R} \leq \frac{W_{s_j}^I I(x)}{W_{s_l}^I I(x)} \leq R \qquad (2.89)$$

于是算法（一维）是：

A. 取不同的尺度$s_1, \cdots, s_j$和计算$W_{s_j} I(x)$，（$1 \leq j \leq J$）；

B. 设阀值为 T，$\left|W_{s_j} I(x)\right| \geq T$；

C. 找到另一个阀值 R，$\frac{1}{R} \leq \frac{W_{s_j}^I I(x)}{W_{s_l}^I I(x)} \leq R$，检测$x$是不是边缘。

这种方法很容易推广到二维。

2.14 边缘检测的一种概率方法

1998 年 David H. Marimont 和 Yossi Rubner［6］ 提出一种边缘检测的概率方法。

他们认为一般的方法都要选择阀值，它是与噪声有关的。可是那怕是按照噪声自适应的阀值选择算法实际上也不是很好。他们提出一种在考虑噪声的影响下对零交叉点的统计分析得出的算法，这种算法避免了阀值的计算。

我们以一维为例对算法做简单说明。

假设 $I(x)=S(x)+n(x)$，其中 $S(x)$ 为真实信号，$n(x)$ 是噪声。假设噪声是平稳的、可加的、均值为 0 的白噪声过程。如果我们检测出 S_{xx} 是一个零交叉点，那么我们就认为这个点是边缘，也就是说满足（2.90）。

$$\left|S_{xx}\Big/S_{xxx}\right|<\Delta x \tag{2.90}$$

这里的 Δx 表示半个像素长。

为了避免假零交叉点，我们增加一个条件：

$$S_x S_{xxx}<0 \tag{2.91}$$

先使用 Lindeberg 提出的平滑算子（2.92）作平滑。

$$T_\alpha(x)=\Upsilon(x;\alpha\sigma^2)=e^{-\alpha\sigma^2}B_x(\alpha\sigma^2) \tag{2.92}$$

于是就有

$$
\begin{aligned}
&(D_x T_1)(x) = \frac{1}{2}(T_1(x+1) - T_1(x-1)),\\
&(D_{xx} T_1)(x) = T_1(x+1) - 2T_1(x) + T_1(x-1)),\\
&(D_{xxx} T_1)(x) = \frac{1}{2}((D_{xx} T_1)(x+1) - (D_{xx} T_1)(x-1)),
\end{aligned}
\tag{2.93}
$$

令

$$
\vec{n} = \begin{bmatrix} n_x \\ n_{xx} \\ n_{xxx} \end{bmatrix}
$$

则可以计算得$\vec{n}$的自相关矩阵为

$$
K(\vec{n}) = \sigma_n^2 \begin{bmatrix} B & 0 & F \\ 0 & D & 0 \\ F & 0 & E \end{bmatrix}
\tag{2.94}
$$

其中，

$$B = \frac{1}{2}T_2(0) - \frac{1}{2}T_2(2)$$

$$D = 2T_2(2) - 8T_2(1) + 6T_2(0)$$

$$E = -\frac{1}{2}T_2(4) + 2T_2(3) - 2T_2(2) - 2T_2(1) + \frac{5}{2}T_2(0)$$

$$F = -\frac{1}{2}T_2(3) + T_2(2) + \frac{1}{2}T_2(1) - T_2(0)$$

用 zc 表示满足（2.90）的零交叉点，则

$$P\{zc \mid I_{xx}; I_{xxx}\} = P\{\left|S_{xx} / S_{xxx}\right| < \Delta x \mid I_{xx}; I_{xxx}\}$$
$$= \mathrm{G}\left(\frac{\sigma_{xxx}}{\sigma_{xx}}\Delta \mathrm{x}\right) - \mathrm{G}\left(-\frac{\sigma_{xxx}}{\sigma_{xx}}\Delta \mathrm{x}\right) \qquad (2.95)$$

其中

$$G(t) = L\left(\frac{a-bt}{\sqrt{1+t^2}}, -b; \frac{t}{\sqrt{1+t^2}}\right) + L\left(\frac{-a+bt}{\sqrt{1+t^2}}, b; \frac{t}{\sqrt{1+t^2}}\right) \qquad (2.96)$$

$$a = \frac{I_{xx}}{\sigma_{xx}} \quad , \quad \mathrm{b} = \frac{I_{xxx}}{\sigma_{xxx}}$$

$$L(h,k;\rho) = \int_h^{+\infty}\int_k^{+\infty} \psi(x,y;\rho)dydx$$

$\psi(x,y;\rho)$ 是标准的二元正态分布。

由于（2.91）的限制，边缘的概率有下面式子决定

$$P\{edge\} = P\{zc \mid s_{xx}s_{xxx} < 0\} \qquad (2.97)$$

所以

$$P\{edge\} = P\{zc \mid S_{xx}S_{xxx} < 0\}$$
$$= P\{edge \mid S_x > 0\}P\{S_x > 0\} + P\{edge \mid S_x < 0\}P\{S_x < 0\}$$
$$= P\{zc \mid S_x > 0; S_{xxx} < 0\}P\{S_x > 0\} + P\{zc \mid S_x < 0; S_{xxx} < 0\}P\{S_x < 0\} \qquad (2.98)$$
$$= \left[\mathrm{G}^{+}\left(\frac{\sigma_{xxx}}{\sigma_{xx}}\Delta \mathrm{x}\right) - \mathrm{G}^{+}\left(-\frac{\sigma_{xxx}}{\sigma_{xx}}\Delta \mathrm{x}\right)\right]P\{S_x > 0\}$$
$$+ \left[\mathrm{G}^{-}\left(\frac{\sigma_{xxx}}{\sigma_{xx}}\Delta \mathrm{x}\right) - \mathrm{G}^{-}\left(-\frac{\sigma_{xxx}}{\sigma_{xx}}\Delta \mathrm{x}\right)\right]P\{S_x < 0\}$$

由于 S_x 满足高斯分布，所以我们利用（2.98）就可以知道某个点是边缘的概率。

但是只知道这种概率是也可能导致错误的检测结果。因为导致概率较低的原因有两个，一种情况就是这个点本来就不是边缘，但是还有一种情况估计这

个概率的方法噪声很大。所以这个结果又能是不值得信赖的。因此区分这两各种低概率就很重要。它们的区别是：当尺度增加的时候是边缘的像素它的概率就会大幅增加因此，引入了“可信度概率(Confidence Probability)”：

令 $X=\begin{bmatrix} S_{xx} \\ S_{xxx} \end{bmatrix}$，其分布可以计算得到是 $N(\mu,\Sigma)$

$$\mu=\begin{bmatrix} I_{xx} \\ I_{xxx} \end{bmatrix} \quad ,\Sigma=\begin{bmatrix} \sigma_{xx}^2 & 0 \\ 0 & \sigma_{xxx}^2 \end{bmatrix} \tag{2.99}$$

$(X-\mu)^t\Sigma^{-1}(X-\mu)$ 概率分布函数是具有两个自由度的 chi-square 分布，记为 χ_2^2。

做假设检验 $H_0:\mu=0$，如果 $Q=X^t\Sigma^{-1}X>c_m(\alpha)$ 则拒绝 H_0 假设。这里的 $c_m(\alpha)$ 指的是 χ_2^2 分布的上$100\alpha\%$ 点。如果 H_0 不成立，就说明 $Q=X^t\Sigma^{-1}X$ 不是中心 chi-square 分布，而是一个非中心 chi-square 分布，记为 $\chi_2^2(\delta)$，其中 $\delta=\mu^t\Sigma^{-1}\mu$。这个检验的强度是

$$\beta(\delta)=P\{\chi_2^2(\delta)>c_m(\alpha)\} \tag{2.100}$$

这个强度定义为正确决定 $\mu\neq0$并且$Q>c_m(\alpha)$ 的概率。这个概率就是我们所说的“可信度概率(Confidence Probability)”。

这个“可信度概率(Confidence Probability)”随着检测的信噪比的提高而提高，所以如果这个概率比较小，那么当前像素处前面给出的是边缘的概率就不可靠。

这个结果推广到二维情况就得到图像边缘检测的算法。图 2.12 是作者给出的这种算法处理的结果：边缘概率图。

图 2.12　(a)原图像　(b)细节

(c) 边缘概率图：越深表示概率越大 (d)细节

2.15 任意曲面上图像边缘的检测：基于张量的方法

2002 年，王晓明等[56]提出了一种基于张量的边缘检测算法。他们认为现有的算子总是基于直角坐标系的，无法检测任意曲面上图像的边缘：比如说柱面贴图、球面贴图、动态景物的数字化效果和三维医学图像等。

将图像视为欧氏空间中的曲面 $(x, y, f(x, y))$，那么图像滤波可以看作是求图

像的热扩散方程$\frac{\partial f}{\partial t}=c\nabla^2 f$的差分解。这里的$c$是演化速率函数，如果$c>0$，则对应为图像锐化，如果$c<0$则对应为图像平滑。扩散方程的演化相当于加权模板与图像的迭代卷积，该模板的权系数是由对应点信号的连续性，即像素的梯度来决定的。由曲面演化理论和欧拉公式可以知道，演化速率函数是曲面主曲率的增函数时，曲面演化对任意点截形有光顺的效果；反之若是减函数时，则有锐化的效果。利用拉普拉斯算子的曲线坐标形式，可以得到曲面坐标下的热扩散方程，从而实现图像滤波。

2.17 数学形态学方法[57]

数学形态学是研究数字影像形态结构特征与快速并行处理方法的理论，是通过对目标影像的形态变换实现结构分析和特征提取的目的。数学形态学以图像的形态特征为研究对象，它的主要内容是设计一整套概念、变换和算法，用来描述图像的基本特征和基本结构，也就是描述图像中元素与元素、部分与部分间的关系。数学形态学作为一种用于数字图像处理和识别的新理论和新方法，它的理论虽然很复杂，被称为“惊人数学”，但它的基本思想却是简单而完美的。

原始图像 → 伊位、交、并、补等集合运算 → 输出图像
↑
结构元

图 2.13　数学形态学的方法

数学形态学算子的性能主要以几何方式进行刻画，传统的理论却以解析方

式的形式描述算子的性能,而几何描述特点似乎更适合视觉信息的处理和分析。数学形态学的基本思想如图 2.13 所示。

数学形态学的理论包含内容十分广阔。特别地，传统图像处理中的线性算子和非线性算子均是形态学算子的特例！这个令人吃惊的结论说明数学形态学是一个图像处理的统一理论，是对传统理论的推广，在这个统一理论的框架下，经典的方法得以在一个新的、统一的层次上进行分析，从而帮助我们从不同的侧面更深入地了解经典算法的性质并在更广泛的范围、以更灵活的方式对它们进行改进。利用数学形态学进行图像边缘检测就是其中很重要的结果。

数学形态学的主要内容是设计一整套变换，来描述图像的基本特征，或基本结构。最常用的有 7 种基本变换,它们是膨胀、腐蚀、开、闭、击中、薄化、厚化。其中膨胀(Dilation)和腐蚀(Erosion)是两种最基本最重要的变换，其它变换由这两种变换的组合来定义。

灰度图像膨胀和腐蚀定义如下:

设 A、C 等符号表示灰度图像,A(m,n)表示灰值图 A 在(m,n)点的灰度值。B 为各种运算的结构矩阵,B(i,j)为其在(i,j)点的值.

定义 1　A 被 B 膨胀,记为$C=A\oplus B$,当且仅当

$$C(m,n)=\bigvee_{b(i,j)}[A(m-i,n-j)+B(i,j)-1] \qquad (2.101)$$

其中$\bigvee_{b(i,j)}[\cdots]$表示对所有的 B(i,j)取$[\cdots]$中最大值.

定义 2　A 被 B 府蚀,记为$C=A\Theta B$,当且仅当

$$C(m,n)=\bigwedge_{b(i,j)}[A(m-i,n-j)+B(i,j)-1] \qquad (2.102)$$

其中 $\bigwedge_{b(i,j)}[\cdots]$ 表示对所有的 B(i, j)取[…]中最小值.

在考察图像各部分之间的关系时,我们需要设计一种收集信息的探针,称为“结构元素”或“结构矩阵”,如上面膨胀和腐蚀变换中的 B。若在图形中不断移动结构元素,便可以考察各个部分之间的关系。在形态学中,结构元素是最重要、最基本的概念,它是这样定义的:

设 n 维欧氏空间 E(n),则结构元素 B 可以定义为是 E(n)或其子空间 E(n)上的一个集合,具有一定的几何形状(如圆、球、有向线段、有向点对等)。结构元素包含原点,其尺寸相对地远小于所考察的物体。$\{B(x),x\in E(n)\}$称为结构元素簇,点 x 为其“中心”。结构元素在形态变换中的作用类似于信号处理中的“滤波窗口”。

边缘反映的是图像的不连续性。对图像的边缘检测正是利用了这种特性。数学形态学中的膨胀和腐蚀运算有着很直观的几何背景,它们可以使被处理的图像在一定方向上增厚或减薄,其方向取决于所选取的结构元素 B.结构元素 B 的选择非常重要,若选择一个点或一条直线来作结构元素 B。显然,一个点的单元素结构矩阵只能使图形发生平移(B 不包含原点)或不变化(B 包括原点);而直线形结构矩阵则适于按照一定方向处理图像,使图形某一端或某一侧面发生变化,而不是在四周进行运算。若选择图 2.14 中的四邻域团形结构矩阵进行膨胀或腐蚀运算时，图形四周被膨胀或腐蚀一圈，其厚度等于结构元素的厚度。原

图像与这两种运算的差则可以用作全方位的边缘检测，即 A-(A ⊕B) 或 A-(A ΘB) 便可检出图像 A 的边缘。

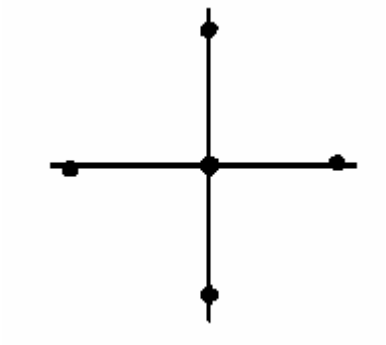


图 2.14　邻域团形结构矩阵

除此之外，边缘检测还有很多种方法，比如说最近提出来的边缘流方法[49]和基于积分变换的边缘检测方法 [55] 等等。

这么多方法，到底哪一种方法好呢？这个问题没有一般的答案。因为任何一种方法都是在一定的假设基础上给出的，效果的好坏要看实际与假设的符合程度。除此之外还要看应用的目的，边缘检测只是一个中间过程，它不是我们的最终目的。

3. Canny 准则及 Canny 算法

1986 年 John Canny 在 IEEE 上发表了划时代意义的文章 A Computational Approach to Edge Detection[14]。这篇文章承前启后，作者对过去的一些方法和应用做了小结[9][15],在此基础上提出了边缘检测的三条准则——这就是著名的 Canny 准则(Canny's Criteria)，并在此基础上得到了一个很不错的实用算法。 本章主要是介绍 Canny 准则及其发展。

Canny 准则的目的就在于:在对信号和滤波器做出一定假设的条件下利用数值计算方法求出最优滤波器并对各种滤波器的性能进行比较。

3.1 Canny 连续准则极及其算法

3.1.1 边缘检测的 Canny 准则

Canny 考察了以往的边缘检测算子和边缘检测的应用，他发现尽管这些应用出现在不同的领域，但是他们都有一些共同的要求：

A. 好的检测结果，或者说对边缘的错误检测率要尽可能低：就是在图像上边缘出现的地方检测结果中不应该没有；另一方面也不要出现虚假的边缘。这是显然的，所有使用边缘检测做更深入工作的系统，它的性能都依赖于边缘检测的误差；

B. 对边缘的定位要准确：也就是我们标记出的边缘位置要和图像上真正边缘的中心位置充分接近；

在实践中我们发现仅仅满足这两条的算子并不好，有的算子会对一个边缘产生多个响应。也就是说图像上本来只有一个边缘点的，可是检测出来就会出现多个边缘点。于是再添加一个要求：

C. 对同一边缘要有低的响应次数。

这就是 Canny 三准则。事实上，在 Canny 之前就有人提出了类似的要求[13][48]，但 Canny 是第一个明确提出这三条并完整解决这个问题的。

更重要的是 Canny 给出了这三条准则的数学表达式(以一维为例)。这就使得寻找给定条件下最优算子的工作转化为一个泛函优化问题。从而为寻找给定条件下最优滤波器开辟了新的也更有效的道路。

假设滤波器的有限冲击响应 $f(x), x \in [-W, W]$，假设要检测边缘的曲线为 $G(x)$，并且我们假设它边缘就在 x=0 处，噪声为 $n(x)$。

A. 好的检测结果(Good Detection)：

就是要把是边缘但是没有检测出来的和不是边缘却检测出来是边缘的这种概率降到最低。由于这两个概率都随着信噪比提高而单调下降，所以这第一个准则就等价于求 $f(x)$ 使得检测后的图像在边缘点的信噪比最大化。

经过 f(x)滤波后，边缘点处的图像信号的响应是：

$$H_G = \int_{-W}^{W} G(-x) f(x) dx \tag{3.1}$$

而噪声的响应的平方根是

$$H_n = n_0 [\int_{-W}^{+W} f^2(x) dx]^{1/2} \tag{3.2}$$

这里的 n_0^2 是单位长度上噪声振幅的均方

于是 Canny 第一个准则的数学表达式就是：

$$SNR(f)=\frac{H_G}{H_n}=\frac{\left|\int_{-W}^{W}G(-x)f(x)dx\right|}{n_0[\int_{-W}^{+W}f^2(x)dx]^{1/2}} \tag{3.3}$$

B.定位准则(Good Localization)：

设检测出的边缘位置在 x_0 (记住：实际的边缘在 $x=0$)，则有

a. $H_G(x)+H_n(x)$ 在 x_0 处取得最大值，所以

$$H_n'(x_0)+H_G'(x_0)=0 \tag{3.4}$$

b. $H_G(x)$ 在 $x=0$ 取得最大值，所以

$$H_G'(0)=0 \tag{3.5}$$

c. 于是就有

$$H_G'(x_0)=H_G'(0)+H_G''(0)x_0+o(x_0^2)\approx H_G''(0)x_0 \tag{3.6}$$

即

$$H_G''(0)x_0=-H_n'(x_0) \tag{3.7}$$

从而

$$E(x_0^2)=\frac{E[(H_n''(x_0))^2]}{(H_G''(0))^2}=\frac{n_0^2\int_{-W}^{+W}f'^2(x)dx}{\left[\int_{-W}^{+W}G'(-x)f'(x)dx\right]^2} \tag{3.8}$$

这里的 $E(x)$ 表示的是 x 的期望。

因为 x_0 越小定位越准确，所以定位准则的数学表达式定义为

$$Loc(f)=\frac{\left|\int_{-W}^{+W}G'(-x)f'(x)dx\right|}{n_0\int_{-W}^{+W}f'^2(x)dx} \tag{3.9}$$

则我们的目标是求一个函数 $f(x)$，使得下面这个式子达到最大值

$$J(f)=\frac{\left|\int_{-W}^{W}G(-x)f(x)dx\right|}{n_0[\int_{-W}^{+W}f^2(x)dx]^{1/2}}\frac{\left|\int_{-W}^{+W}G'(-x)f'(x)dx\right|}{n_0\int_{-W}^{+W}f'^2(x)dx} \tag{3.10}$$

C. 在理想情况下，我们用滤波器对噪声响应的两个峰值间的距离来近似滤波器对一个边缘点响应的长度。因为输出信号中相邻两个极大值点的距离是相邻两个零交叉点距离的 2 倍，而 Rice 在[38]中给出了高斯噪声在函数 g 滤波后输出信号中相邻两个零交叉点的距离

$$x_{ave}=\pi\left(\frac{-R(0)}{R''(0)}\right)^{1/2} \tag{3.11}$$

其中

$$R(0)=\int_{-\infty}^{+\infty}g^2(x)dx \tag{3.12}$$

$$R''(0)=\int_{-\infty}^{+\infty}g'^2(x)dx \tag{3.13}$$

所以噪声在 $f(x)$ 滤波后两个相邻极大值点的距离为

$$x_{ave}=2\cdot\left(\frac{\int_{-\infty}^{+\infty}f'^2(x)dx}{\int_{-\infty}^{+\infty}f''^2(x)dx}\right)^2 \square\ kW \tag{3.14}$$

这里的 W 是滤波器 $f(x)$ 的半宽度。

所以在 2W 长的区域里出现最大值的个数的期望为

$$N_n=\frac{2W}{x_{\max}}=\frac{2W}{kW}=\frac{2}{k} \tag{3.15}$$

显然,只要我们固定了 k，就固定了 $2W$ 长区域中出现最大值的个数。这就是第三个准则。

我们注意到如果 $f(x)$ 满足这个准则，那么由(3.13)式 $f_w = f(x/w)$ 也满足这个准则。假设 W 与 w 成比例,也就是说对给定的 k，第三个准则的结果与 $f(x)$ 的空间尺度无关。

有了这三个准则的数学表达式，寻找最优的滤波器的问题就转化为泛函的约束优化问题了。

3.1.2 Canny 准则下最优边缘检测滤波器的求解

一般情况下，在给定约束(即给定 k)下泛函 $J(f)$ 的最优点不存在紧凑的表达式。不过可以离散化这个表达式然后求 $f(x)$ 的数值解。Canny 利用“罚函数”方法求的了屋顶形边缘和 ridge 边缘的最优算子，你可以在这论文中看到他的结果。事实上，给定任何形状的边缘都可以求出最优的滤波器出来。

下面开始讨论阶梯形边缘。假设这个边缘 $G(x)$ 为是振幅为 A 的阶跃信号)，则其第一个准则和第二个准则的数学表达式可进一步写为：

$$SNR(f) = \frac{A}{n_0} \frac{\left| \int_{-W}^{0} f(x)dx \right|}{[\int_{-W}^{+W} f^2(x)dx]^{1/2}} \tag{3.16}$$

和

$$Loc(f) = \frac{A}{n_0} \frac{|f'(0)|}{\int_{-W}^{+W} f'^2(x)dx} \tag{3.17}$$

对 f 的尺度进行变化，也就是令 $f_w(x) = f(x/w)$，我们发现

$$SNR(f_w) = \sqrt{w} SNR(f) \tag{3.18}$$

$$Loc(f_w) = \sqrt{w} Loc(f) \tag{3.19}$$

也就是说增大滤波器的宽度会提高信号比，但是会降低定位准则，可是他们的

积准则是不会改变的

$$J(f_w)=J(f)=\frac{A}{n_0}\frac{\left|\int_{-W}^{0}f(x)dx\right|}{[\int_{-W}^{+W}f^2(x)dx]^{1/2}}\frac{A}{n_0}\frac{|f'(0)|}{\int_{-W}^{+W}f'^2(x)dx} \tag{3.20}$$

于是我们可以把 J(f)简化为

$$J(f)=\frac{\left|\int_{-W}^{0}f(x)dx\right|}{[\int_{-W}^{+W}f^2(x)dx]^{1/2}}\frac{|f'(0)|}{\int_{-W}^{+W}f'^2(x)dx} \tag{3.21}$$

Canny 在论文中利用变分法求得了这个带有约束的泛函最优化问题(结果参见论文)。而(3.13)，(3.17)，(3.18)告诉我们：这个问题的解是与空间尺度没有关系的。

Canny 发现这个问题的解可以由高斯函数的一阶导数去很好的逼近。尽管高斯函数的性能要差一点(是原来的 80%)，但只这种差别肉眼是分不清楚的，但是使用高斯函数的一阶导数计算就变得比较简单了。

在二维情形下，我们就使用二维高斯函数导函数作为滤波器。由于求导和求卷积是可结合的，所以我们可以先用高斯函数滤波然后再求导：

$$H(x,y)=(\nabla G(x,y))*I(x,y)=\nabla(G(x,y)*I(x,y)) \tag{3.22}$$

显然我们可以看作是先用高斯滤波器做一个平滑，然后再求梯度。

边缘强度——也就是图像在这个像素处的跳跃幅度——可以用“平滑”后的图像在该像素处的梯度的大小来估计。而这个梯度又可以利用二元函数 $G(x,y)*I(x,y)$ 在这个点处的两个正交方向的方向导数来求：

$$\text{边缘强度}=\left|\nabla(G*I)\right|=\left[\left(\frac{\partial}{\partial\vec{n}_1}G*I\right)^2+\left(\frac{\partial}{\partial\vec{n}_2}G*I\right)^2\right]^{1/2} \tag{3.23}$$

而边缘的方向可以（3.24）式计算：

$$\vec{n}=\frac{\nabla(\mathrm{G*I})}{\left|\nabla(\mathrm{G*I})\right|}=\left(\frac{\partial}{\partial\vec{n}_1}G*I\right)\vec{n}_1+\left(\frac{\partial}{\partial\vec{n}_2}G*I\right)\vec{n}_2 \qquad (3.24)$$

然后用阀值操作检测其局部最大值就可以检测出就可以得到结果。

3.1.3 Canny 边缘检测算法

3.1.3.1 双阀值技术

Canny 还提出一种对噪声的估计的实用的方法。假设边缘信号的响应是比较少的而且是比较大的值而噪声的响应是很多的但是值相对较小，那么阀值就可以通过滤波后的图像的统计累积直方图得到(实践数据表明取阀值为这个累积直方图的 0.8(后来的 Matlab 中设定的是 0.70)点处的响应值比较好。

但是，仅仅有这一个阀值是不够的。由于噪声的影响边缘信号响应只有差不多一半是大于这个阀值的，由此造成了斑纹现象(Steaking)，也就是说边缘是断的。如果我们把这个阀值降低，我们往往发现会出现错误的“边缘”。为了解决这个问题，Canny 提出了一种双阀值方法。前面利用累计统计直方图得到一个高阀值T_1，然后再取一个低阀值T_2(Matlab6.5 中使用的是$T_2=0.4T_1$)。如果图像信号的响应大于高阀值，那么他一定是边缘；如果低于低阀值，那么它一定不是边缘；如果在低阀值和高阀值之间，我们就看它的 8 个邻接像素有没有大于高阀值的边缘。

3.1.3.2 多尺度技术

滤波器的尺度选择一直是边缘检测的一大难题。所谓滤波器的尺度在离散情形下就是指模板宽度 W。如果 W 越大，则检测出的边缘的效果就越好,噪声的

影响越少，但是定位就变的越不准确。

因此很多作者提出了尺度空间的概念，也就是利用多个尺度进行边缘检测。这是因为:

A. 不可能做到无穷小的度量,在现实世界中的任何度量都是在一定尺度下进行的;

B. 尺度的大小会影响到度量，我们这里的模板的宽度 W 就是如此;

C. 信息包含在不同尺度中，边缘信息也是如此。因此要想很好地求出边缘就需要在多个尺度下进行检测;

D. 小的“孔径”并不一定就比大的尺度提供更多的信息。

而在连续滤波器中，尺度指的是不同的滤波器一些参数。这些参数决定了它们当$|x| \to +\infty$时的衰减速度，比如说高斯函数的参数等等。我前面得到的最优滤波器是与空间尺度无关的，所以我们可以从它得到各种不同空间尺度的最优滤波器。用多个不同尺度的滤波器检测边缘的时候，对同一边缘检测出的边缘的位置是不同的，这个时候我们就选择那个尺度最小的滤波器的结果，因为理论分析表明尺度小的时候得到的滤波器定位比较好。具体地可以这样做：先用最小的滤波器去检测边缘并把边缘标记出来，然后估计一下一个较大的滤波器检测到的这个边缘的位置(把检测结果和高斯函数做平滑)。然后用一个较大的滤波器和原来的图像做卷积，如果在刚才预测的地方检测的边缘了，那么只有它的振幅远远大于低尺度滤波器时才接受这个边缘。

在此基础上，Canny 设计了一个边缘检测算法，具有很不错的结果：

A. 首先用 2D 高斯滤波模板进行卷积以消除噪声;

$$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

图 3.1　σ=1.4 时的高斯滤波器的逼近模板

B. 利用导数算子(比如 Prewitt 算子、Sobel 算子)找到图像灰度的沿着两个方向的偏导数(G_x, G_y)，并求出梯度的大小；

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{3.25}$$

C. 利用 B 的结果计算出梯度的方向

$$\theta = Arc\tan\left(\frac{G_y}{G_x}\right) \tag{3.26}$$

D. 一旦知道了边缘的方向，我们就可以把边缘的梯度方向大致地分为四种（水平，竖直，45°方向，135°度方向）。也就是把 0~180°分为 5 个部分：0-22.5°以及 157.5-180°算做是水平方向；22.5-67.5°算做 45°方向；67.5-112.5°算是竖直方向；112.5-157.5°记为 135 度方向。需要记住的是记住：这些方向是梯度的方向，也就是可能的边缘方向的正交方向)。通过梯度的方向,我们就可可以找到这个像素梯度方向的邻接像素；

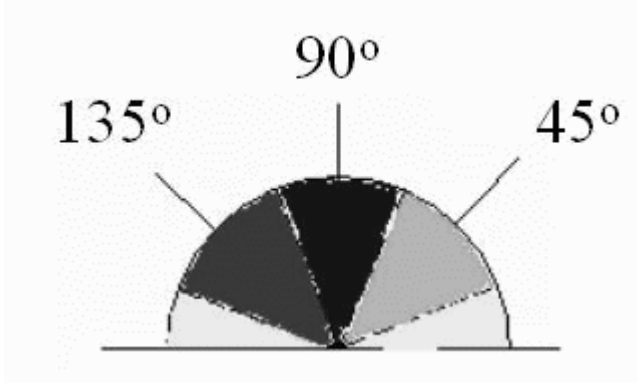


图 3.2 梯度方向的分类

E. 非最大值抑制：遍历图像，若某个像素的灰度值与其梯度方向上前后两个像素的灰度值相比不是最大的，那么这个像素值置为 0，即不是边缘；

F. 使用累计直方图计算两个阀值。凡是大于高阀值的一定是边缘； 凡是小于低阀值的一定不是边缘；如果检测结果大于低阀值但又小于高阀值，那就要看这个像素的邻接像素中有没有超过高阀值的边缘像素：如果有的话那么它就是边缘了，否则他就不是边缘；

G.. 还可以利用多尺度综合技术做的更好。

图 3.3 是 Canny 检测的处理的结果。

3.2 在 Canny 准则基础上边缘检测的发展

Canny 准则开辟了一条寻找最优滤波器的一条道路。在 Canny 论文发表之后陆续出现了很多论文，它们对 Canny 准则及 Canny 的理论进行补充和深入研究的。下面是一些主要的成果：

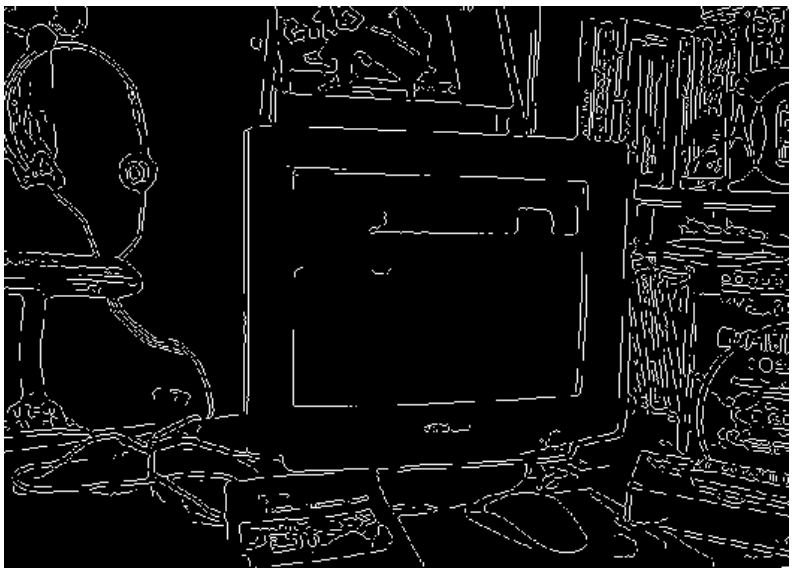


图 3.3　我的书桌--这是用 Matlab6.5 中的 Canny 算法

3.2.1 Spacek 的最优边缘检测算子[22]

1986 年 Spacek 利用 Canny 提出的三个准则，但是采用了与 Canny 不同的优化技术得到了阶梯形边缘的一个含有参数(4 个参数，它们有一些边界条件确定)的最优边缘检测算子。

3.2.2 R. Deriche 边缘检测的迭代算子[33]

1987 年 R.Deriche[33]利用 Canny 得到了有限带宽的滤波器。更重要的是他的这个滤波器可以迭代实现所以更加有效。1990 年 R.Deriche[32]又给出了对图像进行平滑、求一、二阶导数以及求图像 Laplacian 变换的迭代滤波器，利用这种迭代方法可以大大减少计算量。迭代滤波器的产生是这样的：

考虑一维卷积问题

$$y(i)=\sum_{k=0}^{N-1}h(k)x(i-k) \tag{3.27}$$

利用 Z 变换求得它的传输函数是

$$H(z^{-1})=\sum_{n=0}^{N-1}h(n)z^{-n} \tag{3.28}$$

如果我们能找一个有理传输函数

$$H_a(z^{-1})=\frac{\sum_{k=0}^{m-1}b_k z^{-(k-1)}}{1+\sum_{k=0}^{n-1}a_k z^{-k}} \tag{3.29}$$

那么就得到一个 n 阶迭代系统

$$y(i)=\sum_{k=0}^{m-1}b_k x(i-k)-\sum_{k=1}^{n}a_k y(i-k) \tag{3.30}$$

当原来的 N 比较大的时候使用这个系统会大大减少计算量。所以问题归结为如果寻找一个有理传输函数 $H_a(z^{-1})$ 来比较好的逼近 $H(z^{-1})$，然后对具体的滤波器按照这种方法就可以设计各种不同的迭代系统。对二维情形可以用可分离的方式的到类似的结果。

3.2.3 对定位准则的改进

1990 年 Tagare 和 deFigueiredo 在论文[11]中指出很多文献显示高斯函数的一阶导数是边缘检测的一个很不错的算子[9][14][42]。 Lunscher 和 Beddoes 的[46][47]对这个算子进行了深入的研究。可是 Canny 的准则中得出的结果它并不是最优的，而仅仅是最优算子的一个良好逼近。作者经过研究发现 Canny 的定

位准则有一些问题，可以做进一步的改进。

第一个问题是(3.4)式中满足等式的 x_0 的不只是一阶导数的第一个零交叉点，但是实际上还有其它一些零交叉点，它们的影响不能忽略。 噪声信号是一个随机过程，根据随机过程理论(3.8)式中 $E[(H_n''(x_0))^2]$ 的估计是有问题的。

Tagare 和 deFigueiredo 根据随机过程中的理论估计出输出信号中最大值的密度为

$$\begin{aligned}\mu(x) &= \frac{\mathrm{Prob}\{\text{Maximum Between x and x+dx}\}}{\mathrm{dx}} \\ &= \frac{1}{2\pi}\sqrt{-\frac{R''_{H'_n}(0)}{R_{H'_n}(0)}}\exp\left(-f^2(x)/2\sigma^2_{H'_n}\right)\end{aligned} \tag{2.31}$$

显然 $\mu(x)$ 的极值在 0 点求得为

$$\mu_{\max} = \frac{1}{2\pi}\sqrt{-\frac{R''_{H'_n}(0)}{R_{H'_n}(0)}} \tag{2.32}$$

则相对密度为

$$\mu_{rel}(x) = \frac{\mu(x)}{\mu_{\max}} = \exp\left(-f^2(x)/2\sigma^2_{H'_n}\right) \tag{2.33}$$

我们可以用(3.34)作为定位准则。在这个准则下作者证明高斯函数的导数是边缘检测的最优滤波器。

$$\begin{aligned}J = \frac{n_0 Q}{\pi} &= \frac{n_0}{\pi}\int_{-\infty}^{+\infty} x^2(1-\mu_{rel}(x))dx \\ &\approx \frac{\int_{-\infty}^{+\infty} x^2 f^2(x)dx}{\int_{-\infty}^{+\infty} w^2 \left\| f_w(w)\right\|^2 dw}\end{aligned} \tag{3.34}$$

3.2.4 基于 Canny 准则的斜坡型边缘的最优检测算子

1991 年，Maria Petrou 和 Josef Kittler[28]中指出：前面的算法都是假设图像中的边缘都是带有高斯白噪声的阶梯形边缘，但是实际上图像中的边缘并不是这种理想的情形。正如 Nalwa 在[44]所说的那样，图像在数字化的时候有一个模糊的过程，这种模糊过程会把阶梯形边缘转化为斜坡型边缘。因此，Maria Petrou 和 Josef Kittler 假设边缘是具有如下函数表达式(一维情形)：

$$c(x)=\begin{cases}1-e^{-sx}/2, & x\geq 0\\ e^{sx}/2, & x<0\end{cases} \tag{3.35}$$

其中，s 是一个正的常数。

图 3.4 是我用 Mathematica 作出的当 s=1 时的 c(x)的图像。

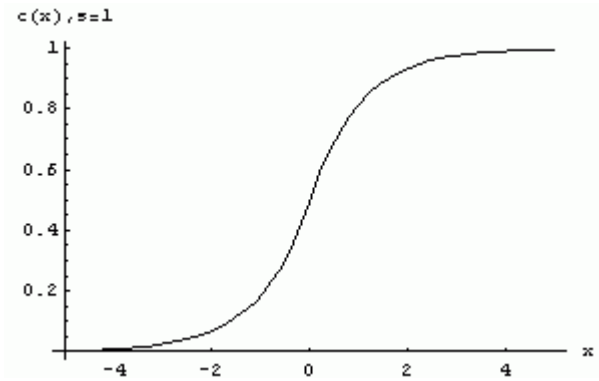


图 3.4 Petrou 和 Kittler 所假设的边缘外观, s=1

然后利用和 Spacek 相同的方法求得了边缘是这种函数表达式下的最优滤波器，并且证明了理想阶梯形边缘不过是其中的极限情形罢了。从而也得到了最优阶梯形边缘的最优检测算子。

3.2.5 边缘检测的最优无限冲击响应滤波器

1991 年 Sudeep Sarkar 和 Kim L.Boyer[40]对前面的工作做了一个小结，并对 Canny 的准则做了一些修改，然后在修改后的准则的基础上利用变分法和数值优化方法求出了阶梯形边缘的最优线性滤波器。这个滤波器可以被一个 IIR 数字滤波器很好的逼近。这个滤波器不是 Canny 所得出的高斯函数的一阶导数。这个 IIR 滤波器的计算效率高，硬件实现起来也很实际，要比高斯函数的一阶导数要好的多。

3.3 Canny 连续准则存在的问题

Canny 准则是一个连续准则，也就是说是在假设图像和滤波器都是一个连续函数的情形下给出的。但实际上数字图像是离散的，滤波器也应该是离散。在实际中就需要把连续的滤波器离散化以选择合适的模板。这就要问：到底选择一个多大宽度的模板？在连续域所谓的最优的滤波器在离散的数字图像上还是不是最优的？下面我们从几个方面说明这个问题。

首先，这种连续准则虽然可以比较很多滤波器的性能，但是对一些离散滤波器它是无法使用的，比如说 Sobel 等滤波器。

第二，在 2.10 中 Torre 和 Poggio 证明了数字图像的导数是一个病态问题。所以直接从连续域中分析然后再把连续滤波器离散化这样得到的滤波器从理论上是不好的。

第三，连续域和离散域之间一个很大的区别在于离散域中的有频谱重叠现象(Spectrum Overlapping)。这也导致了离散域和连续域的性质有很多不同。1994

年 Didier Demigny[3]从 Canny 连续准则出发直接给出了离散准则。也就是说在连续准则中的表达式直接改为离散的:积分换为求和,求导数换为求差分等。然后计算出了这个离散准则下的最优滤波器,他做了很多实验。最后发现:连续域中的最优滤波器抽样后的结果没有直接从离散准则中得到的滤波器好。

后来 Didier Demigny 又对边缘检测的离散准则经行研究[3][4][5]。最终在给出了真正意义上的离散准则[1]。

我们将在第四章介绍 Didier Demigny 在[1]中的结果。

4. 边缘检测的离散 Canny 准则

上一章中我们考察了边缘检测的 Canny 连续准则，在最后我们指出这种连续准则的局限性和无法解决的几个问题。本章我们给出边缘检测的离散准则。这种离散准则的好处在于：首先，利用它可以直接得到离散域中最优滤波器，而不用像以往那样在连续域中研究后再抽样；第二，这个准则可以用来计算经典滤波器的尺度参数。利用这三个准则，就可以通过分析方法和数值方法得到了各种单个准则以及它们的联合准则下最优滤波器。对这些滤波器各种准则的值的研究，发现前两个准则（Σ 和 Λ）与 $x_{\max}$ 准则之间的矛盾性。Didier Demigny［1］发现在新的离散准则下第三条准则的作用被阀值操作取代了。试验结果证明了这一点。

最优化 ΣΛ 准则得到的滤波器是边缘检测的最佳选择。我用数值方法求出了这个最优线性边缘检测算子。

4.1 基本符号定义

4.1.1 离散导数及卷积

对于一个离散抽样信号 g，它的一阶导数序列 g' 定义为

$$\forall n \quad , \quad g'_n = g_n - g_{n-1} \tag{4.1}$$

以同样的方式，我们可以定义 g 的二阶导数：

$$g''_n = g'_n - g'_{n-1} \tag{4.2}$$

我们可以证明下面这个有关卷积和导数的定理：

如果 $s = h * e$，那么就有

$$s' = h * e' = h' * e \tag{4.3}$$

如果说序列 g 的最大值在 n_0 处取得的话，那么必有

$$\begin{cases} g_{n_0} > g_{n_0-1} \\ g_{n_0} < g_{n_0+1} \end{cases} \tag{4.4}$$

也就是说，必然有

$$\begin{cases} g'_{n_0} > 0 \\ g'_{n_0+1} < 0 \end{cases} \tag{4.5}$$

4.1.2 输入信号

Canny 准则需要一个输入信号。它是我们从图像信号中抽象出来的，是我们建立准则的对象。我们就是要求当图像信号是这个信号时的最优滤波器，然后就把这个滤波器作为我们的算子。所以，输入信号的选择很重要。

以往输入信号选择的都是连续信号，如 (2.18)，(2.31)和(3.35)等等，然后在此基础上建立了相关准则。但是真正的数字图像是离散的，所以还得把这个连续滤波器离散化，从而出现了前面我们所说的问题（参见 3.3 节中的结论）。这里我们选择的是离散的输入信号，在此基础上建立了离散准则。

为了把最近的边缘考虑在内，输入信号定义为带宽为 d 的阶跃信号而不是 Canny 使用的单阶梯型信号。

$$e_n = A(U_n - U_{n-d}) \tag{4.6}$$

这里的 U 是离散的 Heaviside 序列，也就是说

$$\begin{cases} U_n = 1, & n \geq 0 \\ U_n = 0, & other \end{cases} \tag{4.7}$$

图 4.1 给出了这个输入序列的图像。

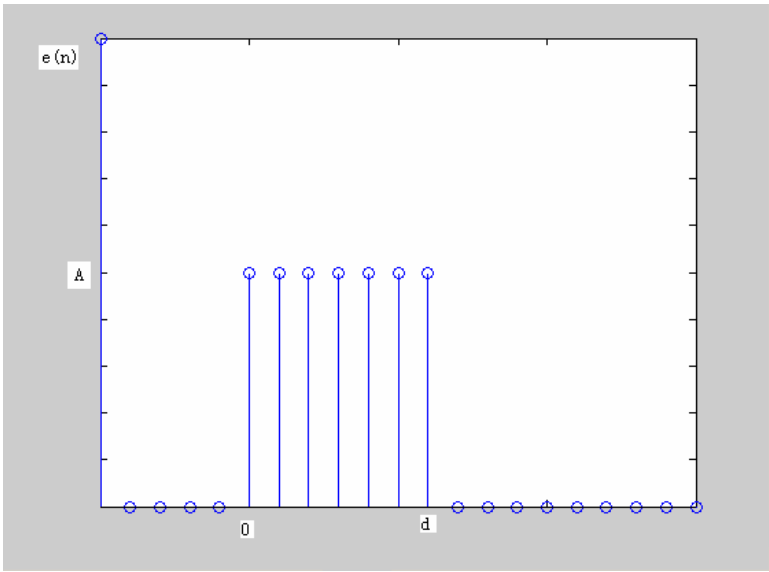


图 4.1 输入信号的

值得注意的时候，这种信号也考虑到了图像的分辨率。有一些滤波器，比如说[36]中的指数滤波器、[32]中的 Deriche 滤波器以及高斯函数的一阶导数做成的滤波器（FDOG）都有一个分辨率，这个分辨率是一个参数的函数(比如说 FDOG 滤波器中的σ参数)。我们这里的参数可以定义为输入的阶跃信号的带宽 d.这是很有道理的：对同一图片在不同的分辨率下扫描，其中的一条线（连续的模拟信号）用不同长度 d 离散数字表示，这个 d 与分辨率成正比。后面给出的Σ准则可以用来计算在给定分辨率下的最优参数。

4.1.3 对噪声的假设

在 Canny 的研究中，在信号中加入了方差为σ_e^2白高斯噪声 N。我们这里假

设噪声为离散的方差为σ_e^2的白高斯过程。

4.1.4 滤波器

离散滤波器具有有限脉冲响应h，对于边缘检测的梯度算法来说h对应一个反对称函数：

$$\forall k>0, \text{有} h_{-k}>0 \text{和} h_{-k}=-h_k \text{成立,并且} h_0=0. \qquad (4.8)$$

$$-W \le k \le W$$

综上,我们的问题就是设计一个h（注意,现在它仅仅是一个具有 W 个参变量，这 W 的量就可以确定这个函数）以在我们假设的噪声下也能有效地检测边缘，也就是要检测卷积信号$s=h*e$的最大值和最小值的位置。

4.2 边缘检测的离散三准则

4.2.1 准则Σ：**好的检测结果**

4.2.1.1 准则Σ：

对于(4.6)给出的信号e_n来说，滤波器的输出结果显示在$n=0$附近取得最大值，而在$n=$d 处取得最小值，而且他们的绝对值相同。不失一般性，我们的准则只考虑检测信号的最大值。在没有噪声的情况下输出信号s_n满足：

$$s_n = A\sum_{k=n-d+1}^{n} h_k \qquad (4.9)$$

仔细检查信号 s 的一阶导数我们发现

$$\begin{cases} s'_n = A(h_n - h_{n-d}) \\ s'_0 = -Ah_{-d} \le 0 \\ s'_{-1} = h_{-1} - h_{-d-1} \end{cases} \tag{4.10}$$

对于一个具有大于或等于 2d+1 个系数的滤波器，因为 $s'_0 < 0$ 所以它的最大值不可能位于 n=0 点。由于当 n 趋于负无穷大的时候 h_n 迅速衰减，因此总是存在着一个小于零的整数 n 使得 s'_n 大于零，所以说 s 的最大值一定落在负无穷大到零之间。令 n_0 为满足 $s'_{n_0} > 0$ 且离 0 最近的整数。那么 n_0 就信号 s 中最大值出现的位置。对于“好的”滤波器， 应该 $n_0 = -1$。

在只考虑噪声的情况下,输出噪声的方差 σ_s^2 可以用下式计算得到：

$$\sigma_s^2 = \sigma_e^2 \sum_{k=-\infty}^{+\infty} h_k^2 \tag{4.11}$$

正如在 Canny 所做的那样，我们定义边缘检测的结果好坏的准则 Σ 为在信号极值点 n_0 处输入信号与输出信号的信噪比的提高。利用(4.9)和(4.11)得到

$$\Sigma = \frac{\sum_{k=n_0-d-1}^{n_0} h_k}{[\sum_{k=-\infty}^{+\infty} h_k^2]^{1/2}} \tag{4.12}$$

4.2.1.2 准则 Σ 下的最优滤波器

根据 h 的反对称性质和在假设下边缘位于 $n_0 = -1$ 的事实，下面这个量必须达到最大化：

$$F = \frac{\left[\sum_{k=-d}^{-1} h_k\right]^2}{\sum_{k=-\infty}^{-1} h_k^2} \tag{4.13}$$

令(4.13)的梯度为 0 我们就可以计算得到h_k的最优值。于是得到：

$$\begin{cases} -d \le k \le -1, h_k = \dfrac{\sum_{i=-d}^{-1} h_i}{F} \\ k < -d, h_k = 0 \end{cases} \tag{4.14}$$

这就证明了在最优检测结果的准则下最优滤波器是冲击响应具有 2d+1 个常系数的滤波器。其实这就是盒式微分滤波器（DOB）：

$$\begin{cases} -d \le k \le -1, h_k = 1 \\ h_0 = 0 \\ 1 \le k \le d, h_k = -1 \end{cases} \tag{4.15}$$

试验结果以及冲击响应最优值每一个系数的小的变化后的结果确认了这个滤波器确实使得这Σ的最大化，即使我们知道当梯度等于 0 的时候只能保证极值的存在性的情况下。

下面我们说明Σ下最优分辨率参数的存在性：当这个滤波器的带宽增大的时候，输出信号（也就是Σ的分子部分）伴随着输入信号的两个灰度跃迁各自的响应的叠加而加速衰减。另一方面，噪声的衰减随着滤波器的带宽的增加而加快。于是，在给定的分辨率下总是存在参数的最优值。

所以利用Σ准则可以计算各种滤波器的分辨率参数的最优值[5]。

4.2.1.3 Σ准则下的最优值和归一化准则

使用表达式（4.12）和（4.15），我们可以计算出Σ的最大值为

$$\Sigma_{Max} = \sqrt{\frac{d}{2}} \tag{4.16}$$

可以定义一个取值在 0 到 1 之间的归一化准则Σ_N

$$\Sigma_N = \frac{\Sigma}{\Sigma_{Max}} \tag{4.17}$$

4.2.2 边缘检测的定位准则

在上一部分，我们已经看到了在另一个方向的第二个跃迁的存在性(存在于离第一个平移的距离为 d 的地方)，它导致了边缘的坏的定位：$n_0 < -1$。这种情况在滤波器的带宽小于 2d+1 的情况下不会发生，但是我们还是的考虑带宽比较大的滤波器。滤波器的带宽参数要按照一定的规则选择以防止上述事情的发生。在这一部分，我们会检查在存在噪声的情况下带来的附加的误差。

4.2.2.1 准则 Λ :噪声对边缘定位的影响

正如我们在上一部分所说那样，我们只是考虑去检测从低到高的跃迁。在n_0的局部看，我们可以看到：$s'_{n_0} > 0$并且$s'_{n_0+1} < 0$.噪声可以引起定位向着 0 或者更小的负数移位。在第一种情况下，定位误差是可以消除的。所以我们只考虑第二种情况，也就是计算$s'_{n_0} > 0$的概率 P。如果方差为σ_e^2的高斯噪声 N 加入到信号中，那么在n_0处的总输出为

$$s_{n_0} = A\sum_{k=n_0-d+1}^{n_0} h_k + \sum_{-\infty}^{+\infty} h_{n_0-k} N_k \tag{4.18}$$

它的一阶离散导数是

$$s'_{n_0} = A(h_{n_0} - h_{n_0-d}) + \sum_{k=-\infty}^{+\infty} h'_{n_0-k} N_k \tag{4.19}$$

得到$s'_{n_0}>0$的概率 P 为

$$P(s'_{n_0}>0)=\frac{1}{2}+erf(\frac{A}{\sigma_e}\Lambda) \tag{4.20}$$

其中,$erf(x)=\frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}}dt$ ， Λ 是定位准则

$$\Lambda=\frac{h_{n_0}-h_{n_0-d}}{\left[\sum_{-\infty}^{+\infty}h_k'^2\right]^{1/2}}=\frac{\sum_{k=n_0-d+1}^{n_0}h'_k}{\left[\sum_{-\infty}^{+\infty}h_k'^2\right]^{1/2}} \tag{4.21}$$

4.2.2.2 Λ 准则下的最优滤波器

根据h的反对称性质以及在$n_0=-1$时要达到好的定位，下面这个量要达到最大化：

$$\Lambda^2=\frac{\left[\sum_{k=n_0-d+1}^{n_0}h'_k\right]^2}{\sum_{-\infty}^{+\infty}h_k'^2} \tag{4.22}$$

注意到下标处于{-d,d}之外的系数仅仅分布在分母上。优化 Λ 含有在 Σ 准则中这些系数等于零的意思。考虑到$h_0=0$且$h_{\pm(d+1)}=0$，所以有

$$\sum_{k=1}^{d+1}h'_k=\sum_{k=-d}^{0}h'_k=0\Rightarrow h'_0=-\sum_{k=-d}^{-1}h'_k \tag{4.23}$$

从而有

$$\Lambda^2=\left(2+2\frac{\sum_{k=-d}^{-1}h_k'^2}{\left(\sum_{k=-d}^{-1}h'_k\right)^2}\right)^{-1} \tag{4.24}$$

所以，下面这个量要最大化

$$F=\frac{\left[\sum_{k=-d}^{-1}h'_k\right]^2}{\sum_{k=-\infty}^{-1}h'^2_k} \tag{4.25}$$

在(4.25)中，如果我们令$h'=g$，不难发现这个式子和(4.13)很相似。

所以有

$$-d\le k\le -1, h'_k=1. \tag{4.26}$$

所以,利用边界条件$h_{-d-1}=0$就产生了在此准则下的最优滤波器的表达式Λ，在表达式中h_k的值随着$|k|$的增加而衰减到0

$$\begin{cases} -d\le k\le -1, h_k=d+1+k \\ h_0=0 \\ 1\le k\le d, h_k=k-d-1 \end{cases} \tag{4.27}$$

和第一个准则一样，这个准则下的滤波器也具有有限$(2d+1)$的冲击响应。

4.2.2.3 Λ准则下的最优值及其归一化准则

综合(4.21)和(4.27)，我们可以计算出Λ的最优值

$$\Lambda_{\max}=\sqrt{\frac{d}{2(d+1)}} \tag{4.28}$$

当d增加的时候到无穷大的时候，定位准则存在极限是$\frac{1}{\sqrt{2}}$。考虑到(4.20)，这就意味着当输入信号的信噪比大于$10dB$的时候，边缘检测 99%的地方都会成功。

我们可以定义一个归一化准则Λ_N

$$\Lambda_N = \frac{\Lambda}{\Lambda_{\max}} \tag{4.29}$$

显然，Λ_N 的取值在 0 到 1 之间。

4.2.3 噪声极值间的距离

4.2.3.1 $x_{\max}$ 准则

我们假设输入信号仅含有噪声，所以我们可以计算出概率来以得到极大值所在的位置 m_0。我们有

$$s'_{m_0} > 0 \quad 和 \quad -s'_{m_0+1} > 0 \tag{4.30}$$

这里

$$\begin{cases} s'_{m_0} = \sum_{k=-\infty}^{+\infty} h'_k N_{m_0-k} \\ -s'_{m_0+1} = \sum_{k=-\infty}^{+\infty} h'_{k+1} N_{m_0-k} \end{cases} \tag{4.31}$$

s'_{m_0} 和 $-s'_{m_0+1}$ 是两个具有相同方差 σ_x 的并且相关系数为 ρ 的随机过程，这里

$$\begin{cases} \sigma_x^2 = \sigma_e^2 \sum_{k=-\infty}^{+\infty} h_k'^2 \\ \rho = -1 + \dfrac{\sum_{k=-\infty}^{+\infty} h_k''^2}{2\sum_{k=-\infty}^{+\infty} h_k'^2} \end{cases} \tag{4.32}$$

于是，极大值点位于 m_0 的概率 P_N 为

$$P_N = \frac{(2\pi\sigma_x^2)^{-1}}{\sqrt{1-\rho^2}} \times \int_0^{+\infty}\int_0^{+\infty} e^{-\frac{x_1^2}{2\sigma_x^2}} e^{-\frac{1}{2(1-\rho^2)}\left(\frac{x_2-\rho x_1}{\sigma_x}\right)^2} dx_1 dx_2 \tag{4.33}$$

于是

$$P_N = \frac{ar\cos(-\rho)}{2\pi} \tag{4.34}$$

于是,极大值间的平均距离 $x_{\max}$ 为 $\frac{1}{P_N}$。这就是 Canny 第三个准则的离散版本：

$$x_{\max} = \frac{2\pi}{ar\cos(-\rho)} \tag{4.35}$$

4.2.3.2. $x_{\max}$ 准则下的最优滤波器

当 $\rho \to -1$ 时, $x_{\max}$ 最大化，所以同时就有下面这个量 G 取得最小值

$$G = \frac{\sum_{k=-\infty}^{+\infty} h_k''^2}{\sum_{k=-\infty}^{+\infty} h_k'^2} \tag{4.36}$$

G 的梯度等于 0，如果对于任意 k

$$h_{k+2} + (G-4)h_{k+1} + (6-2G)h_k + (G-4)h_{k-1} + h_{k-2} = 0 \tag{4.37}$$

为了解这个方程，我们先假设 G 是一个常数，然后我们要证明他的解满足(4.36)。这就导致了一个线性差分方程组，它具有解的形式为 q^k [24]，这里的 q 是特征方程的所有的根。

$$q^4 + (G-4)q^3 + (6-2G)q^2 + (G-4)q + 1 = 0 \tag{4.38}$$

这个方程具有二重根 $q=1$ 以及两个模为 1 的复共轭根

$$q = -\rho \pm i\sqrt{1-\rho^2} \tag{4.39}$$

令 $\cos\theta = -\rho$，则最优滤波器可以用下式得到：

$$h_k = A + Bk + C\cos k\theta + D\sin k\theta \tag{4.40}$$

与其它的几个准则相比，这个结果不依赖于 d(所以，所以第三个准则与输入信

号无关)所以，对于具有任意长冲击响应($2W+1$)的滤波器我们都可以得到最优滤波器。其中的A、B、C、D四个系数中的一个可以任意选取。其他三个常数由边界条件确定：

$$h_0 = h_{W+1} = h_{W+2} = 0$$

通过试验我们发现：利用$x_{\max}$准则，最好的滤波器中$A=C=0$。 而B任意选取。冲击响应长度为($2W+1$)的最优滤波器可以由下式得到：

$$h_k = -(k + D\sin k\theta) \tag{4.41}$$

其中

$$D = (\sin(W+1)\theta - \sin(W+2)\theta)^{-1} \tag{4.42}$$

$$\frac{W+2}{W+1} = \frac{\sin(W+2)\theta}{\sin(W+1)\theta} \tag{4.43}$$

θ取满足(4.43)式的$\frac{3\pi}{2W+3}$的最接近的值。可以证明这个滤波器满足(4.36)式给出的边值条件。

4.2.2.3 $x_{\max}$准则下最优滤波器的数值计算算法、程序及结果

前一节中我们得到了$x_{\max}$准则下最优滤波器的数学表达式（4.41），但是我们并没有得到最终的结果。下面我们通过数值方法利用边界条件求得最优滤波器表达式中的常数并求得最终的表达式。

（4.41）是这个最优滤波器的数学表达式，显然只要确定了 D 和θ就可以得出。(4.42)和(4.43)是根据边界条件得到的约束条件，在不同的 W 下(4.43)确定了θ的值，再利用（4.42）就可以得到 D 的值。

经过大量的数值试验，我们以 $\frac{3\pi}{2W+3}$ 初值利用梯度法就可以求的各个尺度下的最优滤波器的参数 θ。然后可以计算得到最优滤波器。

程序和计算结果参看附录一。

4.2.3.4. $x_{\max}$ 准则下的最大值和归一化准则

利用(4.35)和 $\cos\theta = -\rho$，我们可以得到 $x_{\max}$ 的精确值

$$x_{\max} = \frac{2\pi}{\theta} \tag{4.44}$$

利用（4.44）可以计算得到相应于不同 W 的 $x_{\max}$，参看表 4.1

W	1	2	3	4	5	6	7	8	9	10
$x_{\max}$	3.346	4.860	6.266	7.770	9.071	10.41	11.87	13.27	14.67	16.07
W	11	12	13	14	15	16	17	18	19	20
$x_{\max}$	17.47	18.87	20.27	21.67	23.06	24.66	25.86	27.26	28.66	30.06

表 4.1　　不同参数下 $x_{\max}$ 的值

利用最小二乘方法可以得到 $x_{\max}$ 逼近式：2.066522+1.399904□W，逼近式中的 $x_{\max}$ 随着 W 的线性增加

$$x_{\max} = \eta(2W+3) \tag{4.45}$$

其中

$$\eta \approx 0.699$$

它的归一化准则 $x_{\max N}$ (取值在 0,1 之间)可以定义如下：

$$x_{\max N} = \frac{x_{\max}}{0.699(2W+3)} \tag{4.46}$$

4.3 最优检测-最优定位联合准则下的最优滤波器

4.3.1 Σ和Λ的联合优化

对于Σ或者Λ准则中的任何一个来说，最优滤波器的尺度是 $2W+1=2d+1$。下面我们将要寻找在同样的尺度下使得这两个准则同时达到最大化的滤波器。为了简单，我们没有写出下列方程中求和项中的指标限制- 1 和 $-d$：

$$\Sigma^2 = \frac{\left(\sum h_k\right)^2}{2\sum h_k^2} \quad 和 \quad \Lambda^2 = \left(2 + 2\frac{\sum h_k'^2}{h_{-1}^2}\right)^{-1} \tag{4.47}$$

（4.47）中修改了的关于Λ的表达式是从（4.21），（4.23）和 $h_0' = -h_{-1}$ 得到的。我们要记住：最优化并不依赖于滤波器的全局增益（Global Amplification）。这就隐含了我们可以固定 h_k 中的一个，我们选择固定 $h_{-1} = 1$。为了最大化Σ和Λ的组合，也就是要使得下式实现最小化：

$$\frac{\sum h_k^2}{\left(\sum h_k\right)^2} + \mu\sum h_k'^2 \tag{4.48}$$

这里 μ 是一个正的常数。（4.48）取得最小的值的条件是它的梯度为 0，从而可以推出对每一个在 -2 到 $-W$ 中的任何一个整数 k 有：

$$-\mu' h_{k+1} + 2(1+\mu')h_k - \mu' h_{k-1} = 2F \tag{4.49}$$

上面的 μ' 是一个正的常数，F 是一个满足下面条件下的约束条件

$$F=\frac{\sum_{-d}^{-1}h_k^2}{\sum_{-d}^{-1}h_k} \tag{4.50}$$

显然，当$\mu'=0$时这个方程对应着Σ的最优化；当$\mu'\to\infty$时，它对应着Λ最优化。μ'对应着定义了两个准则的相对权重。利用在前面最优化$x_{\max}$的技术，我们找到了当$k\le 0$时的h_k的表达式，其中的正实数q是：

$$q=\frac{1+\mu'\pm\sqrt{1+2\mu'}}{\mu'} \tag{4.51}$$

$$\begin{cases} -d\le k<0, h_k=Aq^k+Bq^{-k}+F \\ k=0, h_0=0 \\ 0<k\le d, h_k=-(Aq^{-k}+Bq^k+F) \end{cases} \tag{4.52}$$

记住对任意$k>0$，$h_k=-h_{-k}$。边界条件$h_{-1}=1$和$h_{W+1}=0$把 A 和 B 定为q和F的函数。接下去对q的每一个值，利用（4.50）确定F。而q自己是两个准则相对权重的函数，由（4.51）确定。

4.3.2 ΣΛ 准则下的优化

Σ 和 Λ 的乘积准则记为ΣΛ 准则，ΣΛ 准则的最优化是前面优化的特殊情形。当我们选择μ'满足

$$\mu'=\frac{\sum_{-d}^{-1}h_k^2}{\sum_{-d}^{0}h_k'^2} \tag{4.53}$$

时，ΣΛ 的梯度为 0。代入（4.51）后我们得到：

$$\frac{\sum_{-d}^{-1}h_k^2}{\sum_{-d}^{0}h_k'^2}=\frac{q}{(1-q)^2} \tag{4.54}$$

当A， B， F 由前一节中那样确定的时候，（4.54）定义了一个乘积ΣΛ下的最优滤波器的参数q。

4.3.3 ΣΛ准则下最优线性滤波器的计算算法、程序及结果

下面我们通过数值方法利用边界条件求得最优滤波器表达式中的常数，并求得ΣΛ准则下对应不同W的最优线性滤波器。

（4.52）给出了滤波器的表达式，但是需要利用边界条件$h_{-1}=1$和$h_{W+1}=0$和约束条件以及（4.50）和（4.54）确定常数。由$h_{-1}=1$和$h_{W+1}=0$把A,B定为q,F的函数

$$\begin{cases} A=-q^{W+1}\dfrac{(1-F)q^{W}+F}{(1-q^{2W})} \\ B=\dfrac{(1-F)+Fq^{W}}{q(1-q^{2W})} \end{cases} \tag{4.55}$$

从而得到关于q,F的方程组

$$\begin{cases} F=\dfrac{\sum_{-d}^{-1}h_k^2}{\sum_{-d}^{-1}h_k} \\ \dfrac{\sum_{-d}^{-1}h_k^2}{\sum_{-d}^{0}h_k'^2}=\dfrac{q}{(1-q)^2} \end{cases} \tag{4.56}$$

其中

$$h_k=Aq^k+Bq^{-k}+F,-d\le k\le -1$$

这是一个非线性方程组，随着W＝d的增加，方程的次数不断增加。方程的解求起来越来越困难，只能选比较好的初值搜索。经过大量的数值试验我们求得了

W=1 时的解；然后我们以 W=1 的解作为初值去搜索求解 W=2 时的解；依此类推。

程序和计算结果参看附录二。

4.4 综合考虑 Σ、Λ、$x_{\max}$ 时的最优滤波器

4.4.1 任意组合

因为对于 Σ 和 Λ 准则来说，最优滤波器的尺度为 $2W+1=2d+1$。下面我们就在这个尺度去寻找在限制 $x_{\max}$ 时的最优滤波器，这里的 $x_{\max}$ 随着滤波器的尺度的增加而变大。正如前面一样，为了简单起见下面我们在写求和的时候我们不写求和的指标限制 -1 和 $-d$。下面这个量就要实现最小化：

$$\frac{\sum \mathrm{h}_\mathrm{k}^2}{\left(\sum \mathrm{h}_\mathrm{k}\right)^2}+\mu\sum h_k'^2+\delta\frac{\sum h_k''^2}{\sum h_k'^2} \qquad (4.57)$$

这里的 μ 和 δ 是连个正的常数，记住 h_{-1} 是可以任意选取的。（4.57）的梯度为 0 则得到：

$$a_2h_{k+2}+a_1h_{k+1}+a_0h_{k+0}+a_1h_{k-1}+a_2h_{k-2}=2F \qquad (4.58)$$

其中

$$a_2=2\delta \quad a_1=-2\mu-8\delta+2G\delta \quad a_0=2+4\mu+12\delta-4G\delta \qquad (4.59)$$

其解必须满足两个条件，

$$F=\frac{\sum_{-d}^{-1}h_k^2}{\sum_{-d}^{-1}h_k} \quad 和 \quad G=\frac{\sum_{-d}^{-1}h_k''^2}{\sum_{-d}^{-1}h_k'^2} \qquad (4.60)$$

根据（4.59）特征方程的根实数或者复数性质，有两类不同的解。大量的试验结果证明当根是复数的时候上述准则达到最大化。当 $k\le 0$ 的时候，h_h 的表达式如下（q 是一个正实数）：

$$h_k = Aq^k \cos k\theta + Bq^{-k} \cos k\theta + Cq^k \sin k\theta + Dq^{-k} \sin k\theta + F \qquad (4.61)$$

我们这里暂且加设q 和θ是已知的。由于在优化中要使用h的 2 阶导数，所以两边各有两个边界条件：

$$h_{-1} = \text{任意值} \qquad h_0 = h_{W+1} = h_{W+2} = 0$$

这里我们不是使h_{-1}强制为某个数，而是定义F 的值为 1。这种方法简化了寻找常数的寻找。使用边界条件，可以计算常数B， C， D。于是，利用（4.60）中 F 式可以决定 A。而约束 G 现在可以用来确定θ作为q的函数。但是，这个很难给出一个解析函数表达式。我们选择数值模仿方法得到三个准则共同作用下的最优滤波器的参数。

4.4.2 Σ．Λ．$x_{\max}$ 共同作用下的最优滤波器的计算程序及结果

这个方程更加复杂，解起来比较困难。Didier Demigny [1]给出了他们计算经过大量数值试验得出的$\Sigma\Lambda x$ 准则下最优滤波器的参数θ, q（仅仅给出了$W = 2,4,6,8,10,20$ 的，见表 4.2）。我们就利用这些参数来计算$W = 2,4,6,8,10,20$ 时的滤波器。我们将在附录三中给出滤波器的计算程序结果。

W	2	4	6	8	10	20
θ	1.2	16.2	11.9	9.38	7.76	4.18
q	.570	.632	.710	.761	.797	.884

表 4.2 三准则共同作用下的

对应不同最优滤波器带宽 W 的参数q和θ的

4.4.3 两个近似最优滤波器

Didier Demigny[1]通过数值模拟发现三个准则同时使用时，θ和q的值很近

似变化不大。例如，对$W=10$，当$q=0.77$时

当$\theta\to 0$时，Σ_N从92%变化到90%；

当$\theta\to 0$时，Λ_N从65%增加到75%；

当$\theta\to 0$时，$x_{\max}$从92%下降到82%；

当$\theta\to 0$时，三个准则的合成从57.2%下降到56.7%；

当$\theta=0$时，$x_{\max}$发生些微的变化（35%）。

这就提示我们：我们可能损失第三个准则以提高定位的准确性，同时保证检测质量和三个准则的积准则的相对稳定。当$\theta\to 0$时，当$k<0$滤波器的表达式为

$$h_k=(A+Ck)q^k+(B+Dk)q^{-k}+F \qquad (4.62)$$

同上，Didier Demigny [1]也给出了他们计算经过大量数值试验得出的$\Sigma\Lambda x$准则下最优滤波器的一个逼近的参数θ,q（仅仅给出了$W=2,4,6,8,10,20$的，见表4.3）。我们就利用这些参数来计算$W=2,4,6,8,10,20$时的滤波器。

W	2	4	6	8	10	20
q	0.37	0.53	0.63	0.69	0.73	0.85

表4.3 强制定位滤波器：

对应不同尺度W的参数q的值

这些值使得$\Sigma\Lambda$取得最大，因为三个准则的积下的值与最优值很接近。我们将在附录四中给出利用这些参数计算最优滤波器的程序及其结果。

试验告诉我们对于（4.62）中的滤波器，当$q=1$的时候$\Sigma\Lambda x_{\max}$取得最大。在这种情况下，（4.58）的根是四个相等的根。答案是一个简单的滤波器：

$$\begin{cases} -d \le k < 0, h_k = -k(k+W+1)(k+W+2) \\ k = 0, h_k = 0 \\ 0 < k \le d, h_k = -h_{-k} \end{cases} \quad (4.63)$$

4.5 各种滤波器性能对比

4.5.1 各种滤波器的冲击响应

我们定义满足准则i和j滤波器为$H_{ij}\cdots$：H_Σ，H_Λ，H_x，$H_{\Sigma\Lambda}$，$H_{\Sigma\Lambda x}$，我们用H_L表示（4.62）中给出的滤波器，用H_P表示（4.63）给出的滤波器。H_Σ在不同 W 下的表达式由（4.15）给出；H_Λ在不同 W 下表达式由(4.27)给出；不同 W 下H_x我们在 4.2.2.3 节计算得到； 不同 W 下$H_{\Sigma\Lambda}$我们在 4.3.2 节计算得到；不同 W 下$H_{\Sigma\Lambda x}$我们在 4.4.2 节计算得到； 不同 W 下H_L我们在 4.4.3 中计算得到；不同 W 下 H_P的表达式由(4.63)。 我们又添加了 Shen 滤波器（又叫指数滤波器），Deriche 滤波器，Gaussian 函数一阶导数滤波器（FDOG）的冲击响应。这些滤波器的尺度参数可以通过最大化$\Sigma\Lambda$得到。在图 4.2 中我们可以看到这些滤波器当$W=20$时的冲击响应。

因为$H_{\Sigma\Lambda x}$， H_L和H_P有相似的响应，所以我们只画出了$H_{\Sigma\Lambda x}$的图形。

有了这些具体的滤波器，我们就可以计算它们在不同准则下的性能指标，从而可以对比它们的性能以挑选最优线性滤波器。

4.5.2 各种滤波器在不同准则下的性能指标比较

我们已经计算得出了所有最优滤波器，有了这些具体的滤波器参数，我们就可以计算它们各种准则下的性能指标，也就它们的Σ，Λ，x_{max}，ΣΛ 以及ΣΛx的值。

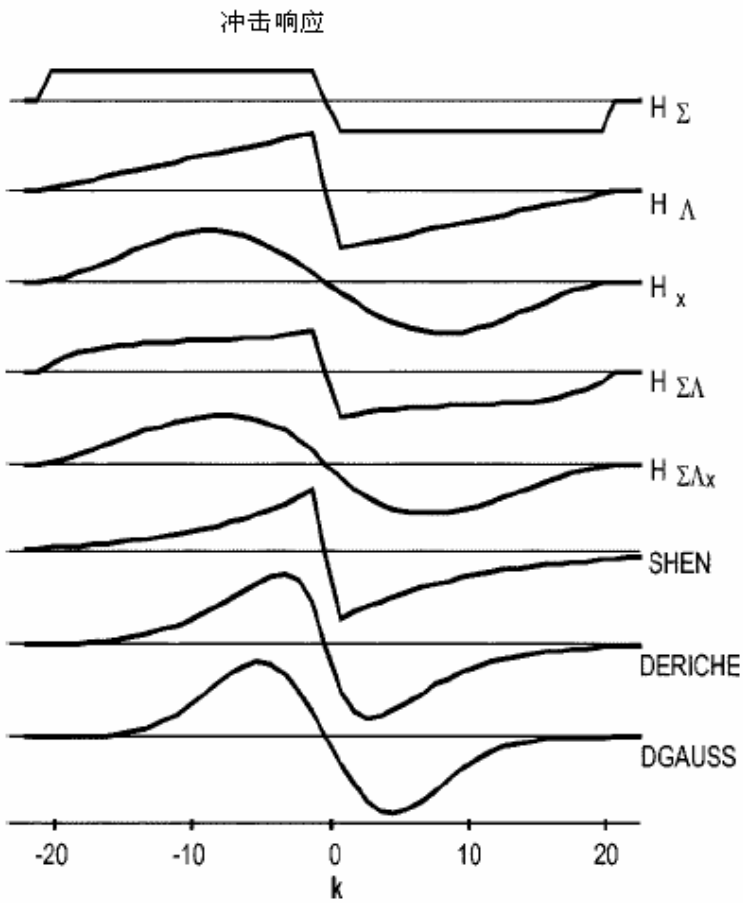


图 4.2　各种标准下的最优滤波器以及一些经典滤波器的冲击响应

图 4.3 给出了所有上述滤波器的最优检测准则Σ的值。我们发现除了 $H_{\Sigma\Lambda}$ 滤波器和 H_{Σ} 滤波器以外其他几个滤波器的最优检测准则的值很接近。而 $H_{\Sigma\Lambda}$ 滤

波器和 H_Σ 滤波器在这个准则下是最优的。

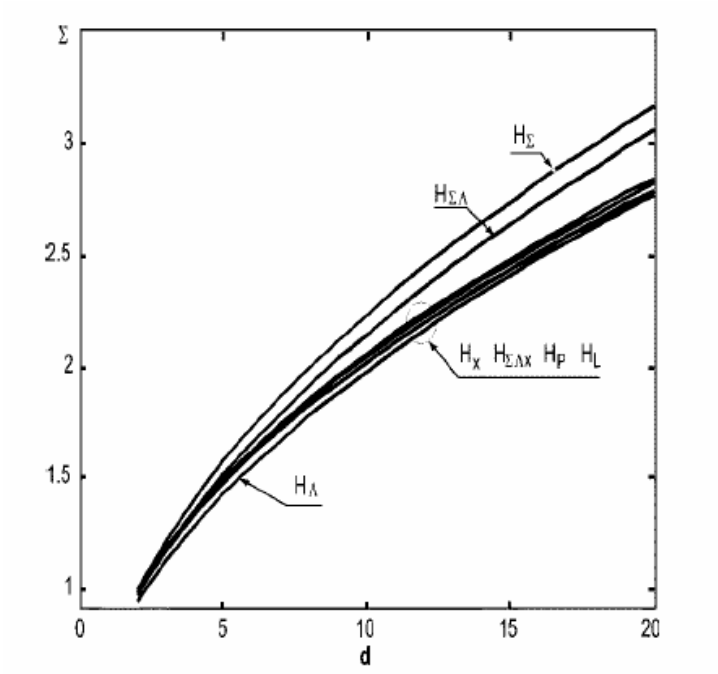


图 4.3　各种最优滤波器在不同 d 时Σ准则值

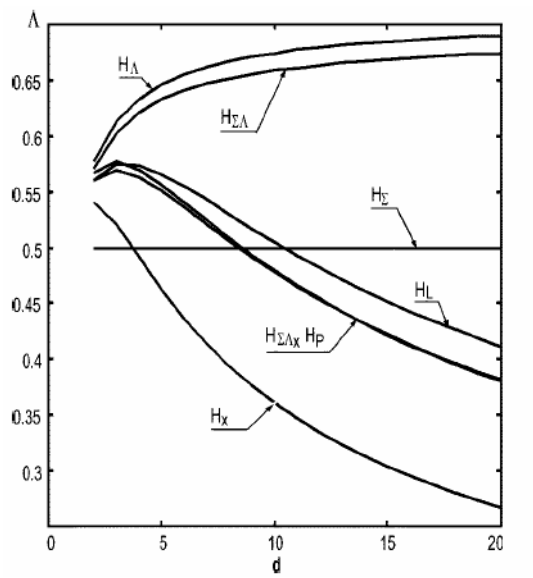


图 4.4　各种最优滤波器在不同 d 时Λ准则值

定位准则的结果在图 4.4 中给出。我们可以看到考虑到 $x_{\max}$ 准则的最优滤波器（比如说 H_x、$H_{\Sigma\Lambda x}$、H_L、H_P）和没有考虑 $x_{\max}$ 准则得到的最优滤波器之间的不同：这些考虑到 $x_{\max}$ 准则的最优滤波器的性能随着带宽 d 的增加而变坏。对于 H_L 来说，因为提高了 ΣΛ 可以使得滤波器的定位准确性得到小的提高，所以可以放松在 $x_{\max}$ 方面的约束。

如图 4.5 所示，与定位性质相似的情形出现在考察 $x_{\max}$ 准则时。我们可以发现，当带宽增加的时候那些冲击响应停留在 0 点的滤波器在 $x_{\max}$ 准则下结果很不好。

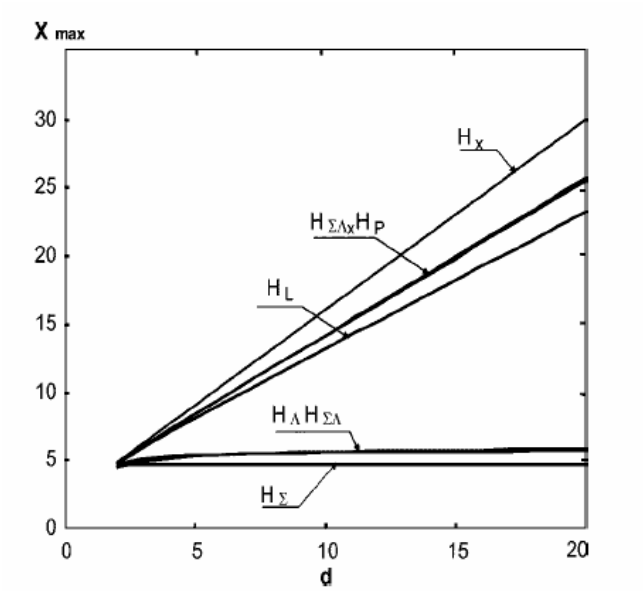


图 4.5　　各种准则下最优滤波器在不同 d 时 $x_{\max}$ 准则的值

图 4.6 给出了 ΣΛ 乘积准则下的结果。这个结果和图 4.3 到图 4.5 给出的独立的准则下的结果告诉我们下面这些滤波器在实际应用中不应该使用。H_x 定位最差，H_Σ 定位也很差，H_Λ 检测结果最差。他们都可以用一个较好的滤波器 $H_{\Sigma\Lambda}$

所代替，在最后一个准则 $x_{\max}$ 下也得到了等价的结果。$H_{\Sigma\Lambda x}$ 滤波器与 H_P 滤波器很相似，但是后者的表示更加简单，参数的计算也变得更简单了。

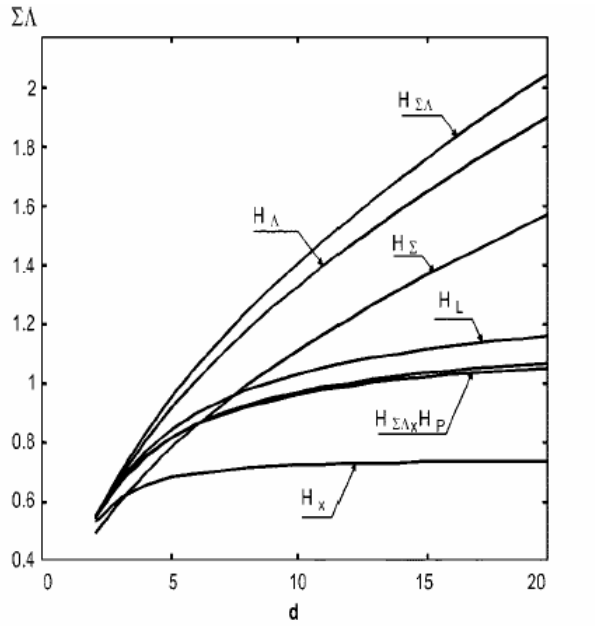


图 4.6　各种准则下最优滤波器在不同 d 时的ΣΛ准则的值

图4.7给出了ΣΛ$x_{\max}$联合准则下的结果。图4.6、图4.7中按照检测性能得到的优劣排序的差异清楚地显示出了ΣΛ准则与 $x_{\max}$ 之间的矛盾。Canny之所以提出 $x_{\max}$ 准则［14］是因为他最大化ΣΛ联合准则时，得到的结果不是令他满意，比如说在有噪声的情况下差分盒子滤波器给出了许错误的响应。Canny是在下面这样情况下发现这个结果的：当一个单阶梯信号作为输入信号时（$d \to \infty$），斜面滤波器（Ramp　Filter，就是Λ准则下的最优滤波器）和差分盒子滤波器有一样的结果，我们的方法可以说明这一点。Deriche等就认为应该在不诉诸于 $x_{\max}$ 准则的情况下寻找有效滤波器。$H_{\Sigma\Lambda}$ 滤波器看起来是一个正确的答案。

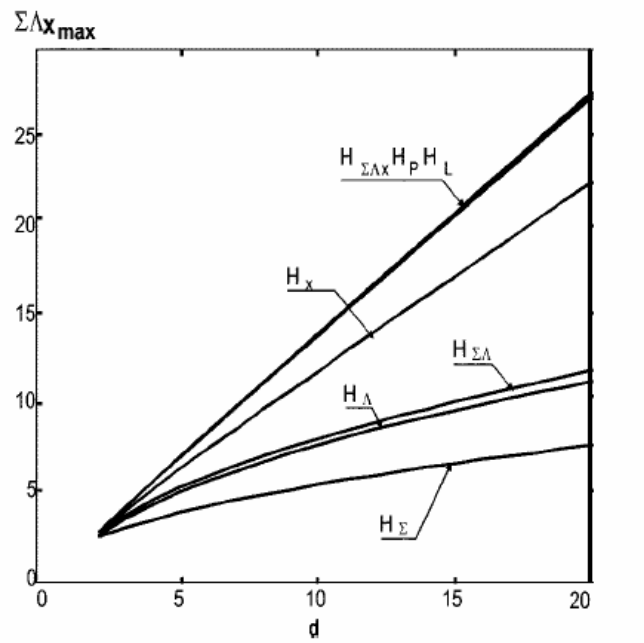


图 4.7　　各种准则下最优滤波器在不同 d 时 $\Sigma\Lambda x_{max}$ 准则的值

Didier Demigny [1]用一些试验也证明了这个一点：只要选择合适的阀值，x_{max} 准则的要求就可以不要，它的作用可以被阀值操作所取代。

4.5.3 ΣΛ 准则下各种滤波器的性能指标

由4.5.2的结论,Canny三准则可以简化为两个准则。也就是只需要最有检测准则Σ和最优定位准则Λ。那么当然由这两个准则决定的最优滤波器是最好的了。我们在前面4.3.2中给出了这个Σ和Λ积准则ΣΛ下的最优滤波器，4.3.3中编程计算出了这些滤波器。

图4.8是各种最优滤波器经典滤波器ΣΛ准则下的性能指标。

所以,我们的最后结果就是：$H_{\Sigma\Lambda}$ 是最优线性滤波器.

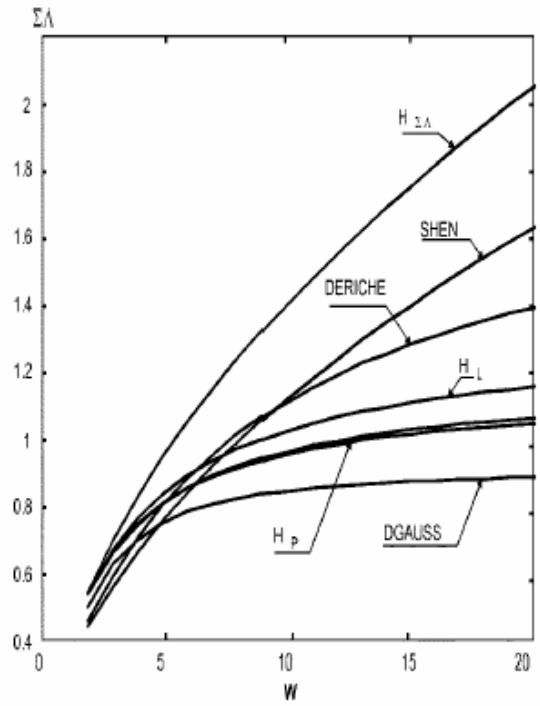


图 4.8　最优滤波器和经典滤波在ΣΛ下的比较

5. 完整的边缘检测方案

5.1 检测算子

我们使用上一章中求出来的 $H_{\Sigma\Lambda}$ 作为我们边缘检测的算子，参见附录二。

5.2 平滑算子的计算

我们在 2.10 中说过，图像的数值导数是一个病态问题。为了把这个病态问题转化为良态问题，我们需要对图像做一个平滑。下面计算平滑算子。

尺度为 W 的平滑算子可以用同尺度的求导算子迭代产生。

假设 d 是尺度为 W 的求导算子，$W=1,2,\cdots,20$。s 为同尺度下的平滑算子。d_i 为 d 的第 i 个数，s_i 为 s 的第 i 个数，$i=-W,-(W-1),\cdots,-1,0,1,\cdots,W$。则有

$$s_i = s_{i-1} + d_i + d_{i-1} \tag{5.1}$$

注意，当 $i \notin \{-W,-(W-1),\cdots,-1,0,1,\cdots W\}$，$d_i$，$s_i$ 按 0 计算。

以 $W=1$ 为例。此时，我们已经有：

$$d = (d_{-1}, d_0, d_1) = (0.5, 0, -0.5)$$

则

$$s_{-1} = s_{-2} + d_{-1} + d_{-2} = d_{-1} = 0.5$$

$$s_0 = s_{-1} + d_0 + d_{-1} = 0.5 + 0 + 0.5 = 1.0$$

$$s_1 = s_0 + d_1 + d_0 = 1.0 - 0.5 + 0 = 0.5$$

于是，

$$s = (s_{-1}, s_0, s_1) = (0.5, 1, 0.5)$$

标准化后得到

$$\overline{s}=\frac{s}{sum(s)}=\frac{(0.5,1,0.5)}{0.5+1+0.5}=(0.25,0.5,0.25)$$

附录五给出了所有 $H_{\Sigma\Lambda}$ 算子的平滑算子。表 5.1 给出了常用的（也就是 $W=1,2,\cdots,6$）求导算子及其对应的平滑算子。

	n	$d_{-n}=-d_n$	$\overline{s}_{-n}=\overline{s}_n$
W=1	0	0	0.5
	1	0.5	0.25
W=2	0	0	0.365469
	1	0.315948	0.25
	2	0.184052	0.067265
W=3	0	0	0.288806
	1	0.2356651	0.220745
	2	0.1630372	0.105597
	3	0.1012976	0.029255
W=4	0	0	0.238481
	1	0.1894026,	0.193312
	2	0.1393324	0.114915
	3	0.1048262	0.056688
	4	0.0664388	0.015844

W=5	0	0	0.202786
	1	0.1589446	0.170554
	2	0.1208371	0.113819
	3	0.0966658	0.069712
	4	0.0755534	0.034788
	5	0.0479992	0.009734
W=6	0	0	0.176132
	1	0.1372384	0.15196
	2	0.1066085	0.10901
	3	0.0876693	0.074792
	4	0.0733448	0.046432
	5	0.0582829	0.023249
	6	0.0368561	0.006492

表 5.1 常用的导算子及其对应的平滑算子

5.3 双阀值的计算

阀值的选取有很多种方法，Canny 算法中一般是用灰度直方图得到的，我们这里也选用这种方法。我们在 3.1 中已经给出了这个方法，下面做更详细的介绍。

A. 求出滤波后的图像的灰度直方图；

B. 然后利用 A 的结果求出累积直方图；

C. 求出累积直方图 0.8 处的值作为高阀值：T_h；

D. 计算低阀值：$T_l = 0.4 \cdot T_h$；

5.4 算法的完整实现

1. 尺度为 W 的平滑算子（s）按行对原图像做卷积运算，然后用相应的最优求导算子（d）对平滑后的算子按列做卷积运算，得到G_y；再用s按列对原图像做卷积运算，然后d对平滑后的算子按行做卷积运算，得到G_x

2. 计算 $G = \sqrt{G_x^2 + G_y^2}$ ； 计算 $\theta = Arc\tan\left(\frac{G_y}{G_x}\right)$

3. 遍历一次灰度矩阵，把梯度方向θ分为四种

   0 （水平方向）： 0-22.5°以及 157.5-180°

   45 （45°方向）： 22.5-67.5°

   90 （竖直方向）： 67.5-112.5°

   135 （135°方向）： 112.5-157.5°

4. 求 G 的累积直方图得到T_h， $T_l = 0.4T_h$；

5. 非最大抑制：遍历 G，若某个像素的灰度值与其梯度方向上前后两个像素的灰度值相比不是最大的，那么这个像素值置为 0;

6. 双阀值操作：遍历 G，大于T_h的设为边缘；小于如果T_l的设为非边缘;小于T_h但是大于T_l的时候看他周围像素处有没有大于T_h的边缘:如果有，这个点就是边缘,否则就不是边缘。

6 算法的 VC++实现和结果.

6.1 算法的 VC++实现的简单说明

将 5.4 中的完整算法用 VC++实现的主要问题是图像的存取。事实上只要得到图像的数据，算法的实现就是一个矩阵操作的简单问题。为了简单，程序只处理 256 色灰度图像。程序中添加了几种其它的边缘检测方法。这些方法在编程的时候没有进行非最大抑制，所以结果比较差。实际上如果进行了非最大抑制，这些方法的检测结果会更好一些的。Matlab 中这些方法都是做过非最大抑制的，可以和他的处理结果做比较。

6.2 算法的结果

图 6.1 是 Didier Demigny 等人多次作为边缘检测算法的测试图像使用过的。我们就用这个图像作为我们的测试图像来检验算法的效果。图 6.2、6.3、6.4 和 6.5 分别是当 W=2、W=4、W=8 和 W=15 时程序处理的结果。当 W=2 和 W=4 的时候检测的结果非常好。可以看出，随着 W 的增加，检测结果的噪声在减少。但是当 W＝8 时结果中丢失了一些细节，当 W=15 时，这一点显得十分明显。事实上，如果这都是符合边缘检测理论的。

边缘检测的滤波方法，其结果与尺度的选择有密切的关系。尺度选择问题一直是边缘检测的难点和热点问题。可以采用 Canny 多尺度技术综合各种尺度下的结果得到更好的结果。事实上，正是在 Canny 的多尺度检测技术的基础上 Mallat 才提出了小波多尺度边缘检测。

图 6.1 测试图像

图 6.2 W=2 时的处理结果

图 6.3 W=4 时的处理结果

图 6.4 W=8 时的处理结果

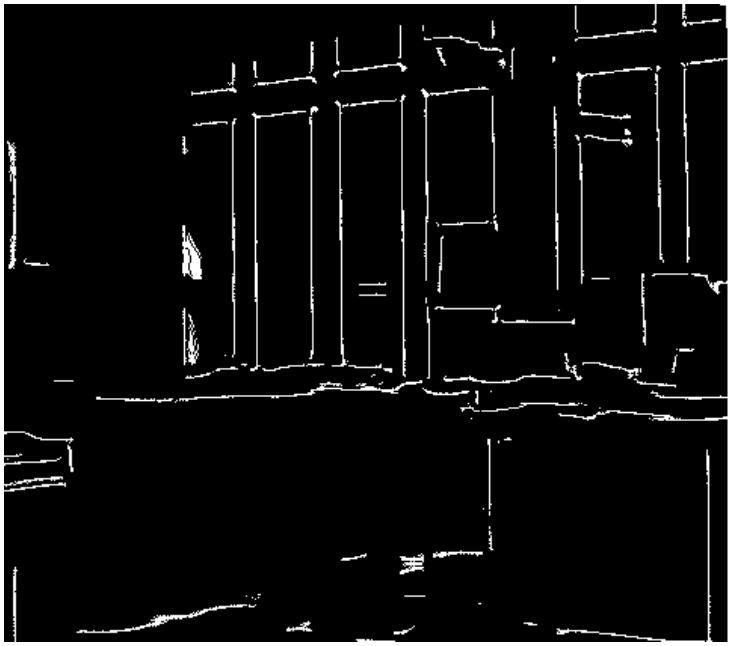


图 6.5　W=15 时的处理结果

我用 Mathematica 将 W 从 1 到 20 变化时程序处理的结果和原图像做成 GIF 动画,从动画可以十分清楚地看到检测结果的变化趋势。而且也可以看出：边缘信息确实藏在不同的尺度下，小尺度并不一定含有更多的信息。

图 6.6 是我自己拍的一张车牌号码的照片和程序的检测结果。

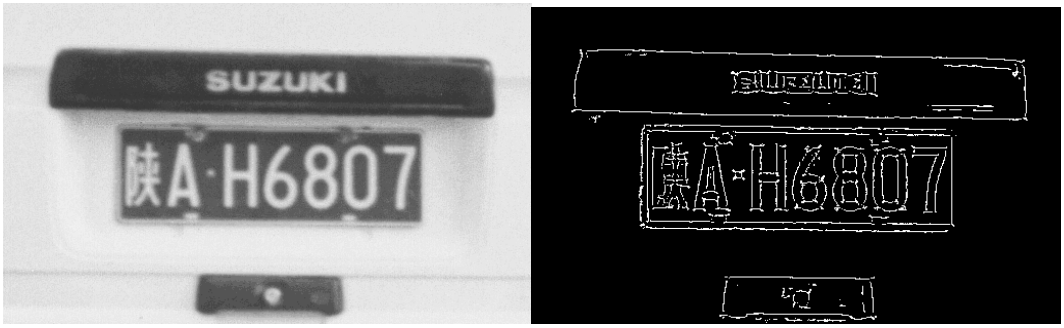


图 6.6 车牌及其检测结果（W=3）

对大量图像的检测试验证明这种算法确实是简单而又实用的。

7. 结 论

本文首先对边缘检测的算法、算子和 Canny 准则的发展分别做了综述。然后根据 Didier Demigny 最新提出来的边缘检测的离散准则，通过大量的数值计算得到了宽度为 1-20 的 Didier Demigny 所说的最优线性滤波器及其对应的平滑算子。然后使用这个滤波器和 Canny 边缘检测技术得到了一个比较好的边缘检测方法，并用 Vc++实现了它。理论和实践证明这个算子是简单而又实用的。

致 谢

本文是在程正兴教授的悉心指导下完成的。整个毕业设计过程中，我要特别感谢程老师，他严谨的治学态度和渊博的知识给了我很多的启示。感谢程老师长期以来在学业上给予我的指导和帮助。

同时，感谢我的朋友郑松峰在我做毕业设计期间所给与我的帮助和启发。

还要为郭萧勇、杨敏等同学和我一起合作讨论以及徐辉同学、赵祥鹏同学和我的网友炮炮 CFSO813 对我 VC 编程提出的建议在此一并表示谢意。

谨此向所有帮助过我的老师和同学们致以由衷的感谢！

参考文献

[1] D.Demigny, “On Optimal Linear Filtering for Edge Detection,” IEEE Trans. Image Processing, Vol.11, NO. 7, pp. 728-737,JULY 2002

[2] D.Demigny, L.Kessal, and J.Pons, “Fast Recursive Implementation of the Gaussian Filter,” in *Proc. 11th IFIP Int. Conf. Very Large Scale Integration*, Montpellier, France, Dec. 2001, pp. 339–346.

[3] D.Demigny and Tawfik Kamlé, “A Discerte Expression of Canny's Criteria for Step Edge Detector Performance Evaluation,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.19, No. 11, pp.1199-1210, Nov. 1997

[4] D.Demigny and M.Karabernou, ”An Effective Resolution Definition or How to Choose an Edge Detectior, It’s Scale Parameter and the Threshold.” Proc. IEEE Int’l Conf. Image Processing, Vol. 1, pp.829-832, Lausanne, Sept. 1996

[5] D.Demigny, F.G. Lorca and L. Kessal, “Evaluation of Edge Detectors Performances with a Discrete Expression of Canny's Criteria,” in Proc. Int. Conf. Image Processing, Washington, MA, pp. 169–172, Oct. 1995

[6] D.H.Marimont and Y.Rubner, ”A Probabilistic Framework for Edge Detection and Scale Selection,” Computer Vision, 1998. Sixth International Conference on , 4-7 pp. 207 –214, Jan 1998

[7] D.Geman and G.Reynolds, “Constrained Restoration and the Recovery of Discontinuities,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 367–383, Jan. 1993.

[8] D.Marr, Vision,Springer-Verlag, New York,1982

[9] D.C.Marr and E.Hildreth, “Theory of Edge Detection,” Proc. Roy. Soc. London, vol. B275, pp.187-217, 1980

[10] Edge Detection, The Early Years, Before 1980 http://iris.usc.edu/Vision-Notes/bibliography/edge217.html#Basic%20Edge%20Detection%20Operations

[11] Hemant D.Tagare and Rui J.P.deFigueiredo, “On the Localization Performance Measure and Optimal Edge Detection,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 12, pp.1186-1190, 1990

[12] Hankyu Moon, Rama Chellappa and Azriel Rosenfeld, ”Optimal Edge-Based Shape Detection,” IEEE Trans. On Image Processing, Vol.11, No.11, pp.1209-1227, Nov. 2002

[13] I.E.Abdou and W.K.Pratt, “Quantitative Design and Evalution of Enhancement and Thrsholding Edge Detectors, Proc. IEEE, Vol.67, No.5, pp. 753-763, May 1979

[14] John Canny, Member, IEEE, “A Computational Approach to Edge Detection,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No. 1, pp.679-697, Novmber 1986

[15] John Canny, “Finding Edges and Lines in Images,” MIT Artif. Intell. Lab., Cambridge, MA, Tech. Rep. AI-TR-720, 1983.

[16] J.Prewitt, “Object Enhancement and Extraction,” *Picture Process. Psychopict.*,

pp.75–149, 1970.

[17] Julez, B., “A Method of Coding TV Signals Based on Edge Detection,” Bell System Tech. (38), No. 4, July 1959, pp. 1001-1020. Compression,Video. Television.

[18] Jun S.Huang and Dong H.Tseng, “Statistical Theory of Edge Detection,” Computer Vision, Graphics, And Image Processing 43,337-346,1988

[19] Jos'e R.A.Torreao and Marcos S.Amaral , “A New Filter for Step-Edge Detection”,www.ic.uff.br/PosGrad/RelatTec/Download/rt_04-01.ps.gz

[20] K.Rao and J.Ben-Arie, “Optimal Edge Detection Using Expansion Matching and Restoration,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 1169–1182, Dec. 1994.

[21] K.S.Shanmugan, F.Dickey and J. Green, “An Optimal Frequency Domain Filter For Edge Detection In Digital Images,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-1, pp.39-47, 1979

[22] L.A.Spacek, “Edge Detection and motion detection,” Image Vision Comput, vol.4, pp.43, 1986

[23] L.G.Roberts, “Machine Perception of Three-Dimension Solids,” in Optical and Electro-Optimal Information Processing, J.t. Tippett, et al., Ed. Cambridge, MA: MIT Press, 1965, 99-159-197

[24] L.Pipes and L.Harvill, “The Calculus of Finite Differences and Linear Difference Equations with Constant Coefficients,” in *Applied Mathematics for*

Engineers and Physicists. New York: McGraw-Hill, 1971, pp. 272–305.

[25] L.Sobel, “Camera Models and Machine Perception,” PhD theses, Stanford University, Standford, CA, 1970

[26] M.Hueckel, “A Local Visual Operator which Recognizes Edges and Lines,” Journal of the ACM, Vol.20, pp. 634-647,Oct. 1973

[27] M.Hueckel, “An Operator which Locates Edges In Digitized Pictures,” Journal of the ACM, Vol.18, pp. 113-125, 1971

[28] Maria Petrou and Josef Kittler, “Optimal Edge Detectors for Ramp Edges,“ IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.13, No. 5,pp.483-491, MAY 1991

[29] Model Based Recognition Systems, http://iris.usc.edu/Vision-Notes/bibliography/match574.html#TT25260

[30] N.Pal and S.Pal, “A Reviewon Image Segmentation Techniques,” *Pattern Recognit.*, vol. 26, no. 9, pp. 1277–1294, 1993.

[31] P.Perona and J.Malik, “Scale-Space and Edge Detection Using Anisotropic Diffusion,” *IEEE Trans. Pattern Anal. Machine Intell.* vol.12, pp. 629–639, July 1990.

[32] R.Deriche, “Fast Algorithm for Low-Level Vision,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 78–87, Jan. 1990.

[33] R.Deriche, “Using Canny's Criteria to Derive a Recursively Implemented Optimal Edge Detector,” Int. J. Comput. Vision, vol.1, no.2, 1987

[34] R.Kirsch, “Computer Determination of the Constituent Structure of Biological Images,” Computer and Biomedical Research, Vol.18, pp.113-125, Jan. 1971

[35] Robert M.Haralick, FELLOW, IEEE, “Digital Step Edges from Zero Crossing of Second Directional Derivatives,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 1, pp.58-68, JANUARY 1984

[36] S.C.J.Shen and J.Zhao, “New Edge Detection Methods Based on Exponential Filter,” in *Proc. ICPR*, vol. 10, 1990.

[37] S.C.J.Shen, “An Optimal Linear Operator for Step Edge Detection,” *CVGIP: Graph. Models Image Process.*, vol. 54, pp. 112–133, Mar. 1992.

[38] S.O.Rice, “Mathematical Analysis of Random Noise,” Bell Syst Tech. J., vol.24, pp.46-156, 1945

[39] S.Sarkar and K.Boyer, “On Optimal Infinite Impulse Response Edge Detection Filters,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 1154–1171, Nov. 1991.

[40] Sudeep Sarkar and Kim L.Boyer, “On Optimal Infinite Impulse Response Edge Detection Filter,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 13, No. 11, pp.1154-1170, Novmber 1991

[41] Sheng Tang, ” Survey of Edge Detection,” http:// www.cs.unr.edu/~tang_s/research/image/surv.ps

[42] T.Poggio, H.Voorhees and A.Yuille, “A Regularized Solution to Edge Detection,” Tech. Rep. MA, Rep. AIM-833, MIT Artificial Intell. Lab., May

1985

[43] Vincent Torre and Tomaso A.Poggio, “On Edge Detection,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No. 2, pp.147-163, March 1986

[44] Vishvjit S. Nalwa and Thomas O.Binford, “On Detecting Edged,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.PAMI-8, No. 6,pp.699-714, Nov. 1986

[45] W.Frei and C.Chen, “Fast Boundary Detection: A Generalization and a New Algorithm,” IEEE Transactions On Electronic Computers, Vol. C-26, Oct. 1977

[46] W.H.Lunscher and M.P.Beddoes, “Optimal Edge Detector Design I: Parameter Selection and Noise Effects,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-8, pp.164-177, 1986

[47] W.H.Lunscher and M.P.Beddoes, “Optimal Edge Detector Design II: Coefficient Quantization,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-8, pp.178-187, 1986

[48] W.K.Pratt, Digital Image Processing, New York: Wiley-Interscience, 1978

[49] Wei-Ying Ma and B. S. Manjunath, ”EdgeFlow: A Technique for Boundary Detection and Image Segmentation,” IEEE Trans. Image Processing, Vol.9, No.8, pp.1375-1388, Aug.2000

[50]Mads Nielesen, Luc Florack ,and Rachid Deriche, “Regularization, Scale-Space, and Edge Detection Filters,”

http://www.it-c.dk/people/malte/pub/eccv96.ps.gz

[51] 程正兴,《小波分析算法与应用》,西安交通常大学出版社,1997

[52] 程正兴，林勇平,《小波分析在图像科学中的应用》，《工程数学学报》第 1 8 卷（小波专刊）pp.57-86.2001

[53] 何斌等,《Visual C++数字图像处理》，人民邮电出版社，2002 年

[54] 章毓晋,《图像处理和分析》，清华大学出版社，1999 年

[55] 王宇生等,《一种基于积分变换的边缘检测方法》,中国图形图像学报, Vol.7(A), No.2, pp.145-149, 2002

[56] 王晓明,顾晓东,刘健,《基于张量的图像边缘检测及滤波》,中国图形图像学报, Vol.7(A), No.8, pp.780-782, 2002

[57]张英琦,何洪涛,《一种基于数学形态学的灰度图象边缘检测方法》,西南交通大学学报，地 31 卷，第五 1 期，1996 年 10 月

附录一 x_{max} 准则下最优滤波器的计算程序及结果

Mathematica 源程序:

```
dd[W_, θ_] := 1/(Sin[(W + 1) * θ] - Sin[(W + 2) * θ]);
h[θ_, d_, k_] := -(k + d * Sin[k * θ]);
s = {};
For[W = 1, W < 21, W++,
 t = FindRoot[(W + 2)/(W + 1) == Sin[(W + 2) * θ]/Sin[(W + 1) * θ], {θ, 3.0 * Pi/(2 * W + 3)}];
 Clear[initialvalue];
 iv = t[[1]][[2]];
 s = Append[s, 2 * Pi/iv];
 d = dd[W, iv];
 η = 2 * Pi/((2 * W + 3) * θ) /. t;
 tab1 = Table[h[iv, d, k], {k, -W, W}];
 pl = Plus @@ Abs[tab1];
 tab = Round[tab1/pl * 1000000]/1000000.0;
 Print["W=", W];
 Print["h=", tab];
 Clear[t, d, η, jd, wt];]
```

输出结果为 W 从 1 到 20 时的滤波器 h。下面我们只给出前 8 个输出结果，如果有兴趣可以在 Mathematica4.0 及以上版本上运行上述程序即可。

W=1 h={0.5,0,-0.5}

W=2 h={0.225708,0.274292,0,-0.274292,-0.225708}

W=3 h={0.116532,0.212784,0.170684,0,-0.170684,-0.212784,-0.116532}

W=4

h={0.067126,0.145363,0.171643,0.115868,0,-0.115868,-0.171643,-0.145363,
-0.067126}

W=5

h={0.041954,0.099532,0.139203,0.135691,0.08362,0,-0.08362,-0.135691,-0.1392
03,-0.099532,-0.041954}

W=6

h={0.027889,0.069939,0.107477,0.123404,0.108173,0.06312,0,-0.06312,-0.108173,
-0.123404,-0.107477,-0.069939,-0.027889}

W=7

h={0.019447,0.050588,0.082473,0.103944,0.106697,0.087547,0.049304,0,-0.049304
,-0.087547,-0.106697,-0.103944,-0.082473,-0.050588,-0.019447}

W=8

h={0.014086,0.037594,0.063787,0.08534,0.095944,0.091703,0.071985,0.039561,0,-
0.039561,-0.071985,-0.091703,-0.095944,-0.08534,-0.063787,-0.037594,-0.014086
}

附录二 ΣΛ 准则下最优线性滤波器的计算程序及结果

Mathematica 源程序:

```
Fpre = 0.9; qpre = 0.3; (*initial value For W=1*)
Hh[k_, q_, F_, f1_, f2_] := f1 * q^k + f2 * q^-k + F;
A[q_, F_, W_] := -(q^(1+W) (-F - q^W + F q^W))/(-1 + q^(2 W));
B[q_, F_, W_] := -(1 - F + F q^W)/(q (-1 + q^(2 W)));
For[W = 2, W < 21, W++,
 f1 = A[q, F, W]; f2 = B[q, F, W];
 h = Flatten[{Table[Hh[k, q, F, f1, f2], {k, -W, -2}], 1}];
 f3 = (Sum_{i=1}^{W} (h[[i]])^2)/(Sum_{i=2}^{W} (h[[i]] - h[[i - 1]])^2 + (h[[1]])^2 + 1);
 f4 = (Sum_{i=1}^{W} (h[[i]])^2)/(Sum_{i=1}^{W} h[[i]]);
 an = FindRoot[{F == f4, f3 == q/(1 - q)^2}, {F, Fpre}, {q, qpre}];
 Fpre = an[[1]][[2]]; qpre = an[[2]][[2]];
 f1 = A[qpre, Fpre, W]; f2 = B[qpre, Fpre, W];
 AA = Table[Hh[k, qpre, Fpre, f1, f2], {k, -W, -1}];
 hh = Plus @@ AA;
 BB = Round[Flatten[{AA, 0, Table[-AA[[W - j]], {j, 0, W - 1}]}]
       / (2 * hh) * 10000000] / 10000000 // N;
 Print["W=", W]; Print["h=", BB];
 Clear[an, AA, BB, hh, f1, f2, f3];]
```

输出结果为 W 从 1 到 20 时的滤波器 h。这是最优线性滤波器，所以我们给出出全部 20 个输出结果，如果有兴趣可以在 Mathematica4.0 及以上版本上运行上述程序即可。

W=1

h={0.5, 0., -0.5}

W=2

h={0.192569, 0.307432, 0., -0.307432, -0.192569}

W=3

h={0.110244, 0.169105, 0.220651, 0., -0.220651, -0.169105, -0.110244}

W=4

h={0.0740478, 0.113594, 0.140923, 0.171436, 0., -0.171436, -0.140923, -0.113594, -0.0740478}

W=5

h={0.0542611, 0.0840673, 0.102832, 0.118998, 0.139842, 0., -0.139842, -0.118998, -0.102832, -0.0840673, -0.0542611}

W=6

h={0.0420167, 0.0658746, 0.0804878, 0.0913162, 0.102406, 0.117899, 0., -0.117899, -0.102406, -0.0913162, -0.0804878, -0.0658746, -0.0420167}

W=7

h={0.0338033, 0.0536158, 0.0657556, 0.074094, 0.0812901, 0.0896388, 0.101802,
0., -0.101802, -0.0896388, -0.0812901, -0.074094, -0.0657556, -0.0536158, -0.0338033}

W=8

h={0.0279715, 0.0448424, 0.0553, 0.0622502, 0.0676237, 0.0729138, 0.0795903, 0.0895081,
0., -0.0895081, -0.0795903, -0.0729138, -0.0676237, -0.0622502, -0.0553, -0.0448424, -0.0279715}

W=9

h={0.0236514, 0.0382843, 0.0474978, 0.0535582, 0.0579561, 0.0617732, 0.0659484, 0.0715087, 0.0

798217, 0. , -0. 0798217, -0. 0715087, -0. 0659484, -0. 0617732, -0. 0579561, -0. 0535582, -0. 0474978, -0. 0382843, -0. 0236514}

W=10

h={0. 0203443, 0. 0332178, 0. 0414597, 0. 0468874, 0. 0506979, 0. 0537315, 0. 0566572, 0. 0601202, 0. 0648842, 0. 0719997, 0. , -0. 0719997, -0. 0648842, -0. 0601202, -0. 0566572, -0. 0537315, -0. 0506979, -0. 0468874, -0. 0414597, -0. 0332178, -0. 0203443}

W=11

h={0. 0177451, 0. 0292007, 0. 0366555, 0. 0415986, 0. 0450176, 0. 0475956, 0. 0498475, 0. 0522231, 0. 0551972, 0. 0593638, 0. 0655553, 0. , -0. 0655553, -0. 0593638, -0. 0551972, -0. 0522231, -0. 0498475, -0. 0475956, -0. 0450176, -0. 0415986, -0. 0366555, -0. 0292007, -0. 0177451}

W=12

h={0. 0156579, 0. 0259479, 0. 0327484, 0. 0373007, 0. 0404358, 0. 0427258, 0. 0445889, 0. 046365, 0. 0483784, 0. 0509966, 0. 0546976, 0. 0601569, 0. , -0. 0601569, -0. 0546976, -0. 0509966, -0. 0483784, -0. 046365, -0. 0445889, -0. 0427258, -0. 0404358, -0. 0373007, -0. 0327484, -0. 0259479, -0. 0156579}

W=13

h={0. 0139515, 0. 0232678, 0. 0295141, 0. 0337397, 0. 0366544, 0. 0387479, 0. 0403719, 0. 0417994, 0. 0432699, 0. 0450307, 0. 0473775, 0. 0507045, 0. 0555707, 0. , -0. 0555707, -0. 0507045, -0. 0473775, -0. 0450307, -0. 0432699, -0. 0417994, -0. 0403719, -0. 0387479, -0. 0366544, -0. 0337397, -0. 0295141, -0. 0232678, -0. 0139515}

W=14

h={0. 0125351, 0. 021027, 0. 0267968, 0. 0307423, 0. 0334771, 0. 0354269, 0. 0368948, 0. 0381092, 0. 03

92592, 0.0405236, 0.0420991, 0.0442308, 0.0472504, 0.0516276, 0. , -0.0516276, -0.0472504, -0.044230
8, -0.0420991, -0.0405236, -0.0392592, -0.0381092, -0.0368948, -0.0354269, -0.0334771, -0.0307423,
-0.0267968, -0.021027, -0.0125351}

W=15

h={0.0113439, 0.0191297, 0.0244853, 0.0281862, 0.0307687, 0.0326066, 0.0339663, 0.0350446, 0.0
359977, 0.0369637, 0.0380825, 0.0395161, 0.0414723, 0.0442343, 0.0482022, 0. , -0.0482022, -0.044234
3, -0.0414723, -0.0395161, -0.0380825, -0.0369637, -0.0359977, -0.0350446, -0.0339663, -0.0326066,
-0.0307687, -0.0281862, -0.0244853, -0.0191297, -0.0113439}

W=16

h={0.0103307, 0.0175058, 0.0224977, 0.0259824, 0.0284321, 0.0301788, 0.0314589, 0.032446, 0.03
32739, 0.0340545, 0.0348938, 0.0359054, 0.0372264, 0.0390357, 0.0415785, 0.0451993, 0. , -0.0451993,
-0.0415785, -0.0390357, -0.0372264, -0.0359054, -0.0348938, -0.0340545, -0.0332739, -0.032446, -0.
0314589, -0.0301788, -0.0284321, -0.0259824, -0.0224977, -0.0175058, -0.0103307}

W=17

h={0.0094602, 0.0161027, 0.0207726, 0.0240641, 0.026396, 0.0280651, 0.0292837, 0.0302069, 0.03
09522, 0.0316143, 0.0322776, 0.0330264, 0.033956, 0.0351847, 0.0368688, 0.0392227, 0.0425458, 0. , -0
. 0425458, -0.0392227, -0.0368688, -0.0351847, -0.033956, -0.0330264, -0.0322776, -0.0316143, -0.03
09522, -0.0302069, -0.0292837, -0.0280651, -0.026396, -0.0240641, -0.0207726, -0.0161027, -0.00946
02}

W=18

h={0.0087059, 0.0148801, 0.0192631, 0.0223805, 0.0246064, 0.0262077, 0.0273763, 0.0282526, 0.0

289415, 0. 0295257, 0. 0300754, 0. 0306563, 0. 0313382, 0. 0322029, 0. 0333541, 0. 0349298, 0. 037119, 0. 04 01844, 0. , -0. 0401844, -0. 037119, -0. 0349298, -0. 0333541, -0. 0322029, -0. 0313382, -0. 0306563, -0. 03 00754, -0. 0295257, -0. 0289415, -0. 0282526, -0. 0273763, -0. 0262077, -0. 0246064, -0. 0223805, -0. 0192 631, -0. 0148801, -0. 0087059}

W=19

h={0. 008047, 0. 0138068, 0. 0179326, 0. 0208923, 0. 0230217, 0. 0245622, 0. 0256887, 0. 0265288, 0. 02 71779, 0. 0277096, 0. 0281842, 0. 0286555, 0. 029177, 0. 0298079, 0. 0306196, 0. 0317042, 0. 0331848, 0. 035 2294, 0. 0380697, 0. , -0. 0380697, -0. 0352294, -0. 0331848, -0. 0317042, -0. 0306196, -0. 0298079, -0. 029 177, -0. 0286555, -0. 0281842, -0. 0277096, -0. 0271779, -0. 0265288, -0. 0256887, -0. 0245622, -0. 023021 7, -0. 0208923, -0. 0179326, -0. 0138068, -0. 008047}

W=20

h={0. 0074675, 0. 0128583, 0. 0167522, 0. 0195681, 0. 021609, 0. 0230944, 0. 0241841, 0. 0249952, 0. 02 56151, 0. 0261104, 0. 0265345, 0. 0269328, 0. 0273483, 0. 0278257, 0. 0284162, 0. 0291836, 0. 0302102, 0. 03 16065, 0. 0335228, 0. 0361651, 0. , -0. 0361651, -0. 0335228, -0. 0316065, -0. 0302102, -0. 0291836, -0. 028 4162, -0. 0278257, -0. 0273483, -0. 0269328, -0. 0265345, -0. 0261104, -0. 0256151, -0. 0249952, -0. 02418 41, -0. 0230944, -0. 021609, -0. 0195681, -0. 0167522, -0. 0128583, -0. 0074675}

附录三 Σ．Λ．x_{max} 共同作用下的最优滤波器的计算程序及结果

Mathematica 源程序:

```
W = {2, 4, 6, 8, 10, 20};
ct = {21.2/180, 16.2/180, 11.9/180, 9.38/180, 7.76/180, 4.18/180} * Pi;
q = {0.570, 0.632, 0.710, 0.761, 0.797, 0.884};
For[i = 1, i < 7, i++,
 Hh[k_, ct_, q_] := A * q^k * Cos[k * ct] + B * q^-k * Cos[k * ct] +
   fC * q^k * Sin[k * ct] + fD * q^-k * Sin[k * ct] + 1;
 f1 = Hh[-W[[i]] - 1, ct[[i]], q[[i]]];
 f2 = Hh[-W[[i]] - 2, ct[[i]], q[[i]]];
 f3 = Hh[0, ct[[i]], q[[i]]];  f4 = Sum[(Hh[k, ct[[i]], q[[i]]])^2, {k, -W[[i]], -1}]
   - (Sum[Hh[k, ct[[i]], q[[i]]], {k, -W[[i]], -1}]);
 ff[k_] = Hh[k, ct[[i]], q[[i]]];
 SS = Solve[{f1 == 0, f2 == 0, f3 == 0, f4 == 0}, {A, B, fC, fD}];
 AA = Table[Table[ff[k] /. SS[[j]], {k, -W[[i]], -1}], {j, 1, 2}];
 CC = Flatten[{AA[[2]], 0, Table[-AA[[2]][[W[[i]] - j]],
     {j, 0, W[[i]] - 1}]}];
 he = Plus @@ (Abs[CC]); BB = CC/he;
 Print["W=", W[[i]]]; Print["h=", BB]; Clear[AA, BB, A, B, fA, fB]]
```

运行后输出结果:

W=2

h={0.198165, 0.301835, 0, -0.301835, -0.198165}

W=4

h={0.0571771, 0.125656, 0.169491, 0.147676, 0, -0.147676, -0.169491, -0.125656, -0.05

71771}

W=6

h={0.0241665,0.0585622,0.0926318,0.116767,0.120015,0.0878573,0,-0.0878573,-0.120015,-0.116767,-0.0926318,-0.0585622,-0.0241665}

W=8

h={0.0124468,0.0316637,0.0532584,0.0735079,0.088783,0.0949944,0.087048,0.0582978,0,-0.0582978,-0.087048,-0.0949944,-0.088783,-0.0735079,-0.0532584,-0.0316637,-0.0124468}

W=10

h={0.00724754,0.0189973,0.0330629,0.0476158,0.0609955,0.071529,0.0773551,0.0762505,0.0654543,0.0414921,0,-0.0414921,-0.0654543,-0.0762505,-0.0773551,-0.071529,-0.0609955,-0.0476158,-0.0330629,-0.0189973,-0.00724754}

W=20

h={0.00120119,0.00335395,0.00624038,0.00966911,0.0134703,0.0174908,0.0215897,0.0256335,0.0294921,0.0330345,0.0361244,0.0386159,0.0403498,0.0411491,0.0408146,0.0391211,0.0358127,0.0305985,0.0231487,0.0130896,0,-0.0130896,-0.0231487,-0.0305985,-0.0358127,-0.0391211,-0.0408146,-0.0411491,-0.0403498,-0.0386159,-0.0361244,-0.0330345,-0.0294921,-0.0256335,-0.0215897,-0.0174908,-0.0134703,-0.00966911,-0.00624038,-0.00335395,-0.00120119}

附录四 Σ．Λ．x_{max} 共同作用下的近似最优滤波器的计算程序及结果

Mathematica 源程序：

```
W = {2, 4, 6, 8, 10, 20}; q = {0.37, 0.53, 0.63, 0.69, 0.73, 0.85};
For[i = 1, i < 7, i++,
 Hh[k_, q_, A_, B_, fC_, fD_] := (A + fC * k) * q^k + (B + fD * k) * q^-k + 1;
 f1 = Hh[-W[[i]] - 1, q[[i]], A, B, fC, fD];
 f2 = Hh[-W[[i]] - 2, q[[i]], A, B, fC, fD];
 f3 = Hh[0, q[[i]], A, B, fC, fD];
 f4 = Sum[(Hh[k, q[[i]], A, B, fC, fD])^2, {k, -W[[i]], -1}]
   - (Sum[Hh[k, q[[i]], A, B, fC, fD], {k, -W[[i]], -1}])^2;
 SS = Solve[{f1 == 0, f2 == 0, f3 == 0, f4 == 0}, {A, B, fC, fD}];
 ff[k_] = Hh[k, q[[i]], A, B, fC, fD];
 AA = Table[Table[ff[k] /. SS[[j]], {k, -W[[i]], -1}], {j, 1, 2}];
 CC = Flatten[{AA[[1]], 0, Table[-AA[[1]][[W[[i]] - j]],
     {j, 0, W[[i]] - 1}]}];
 he = Plus @@ (Abs[CC]); BB = CC/he;
 Print["W=", W[[i]]]; Print["h=", BB];
 Clear[A, B, fC, fD, AA, BB, SS]]
```

运行后输出结果为：

W=2

h={0.191954, 0.308046, 0, -0.308046, -0.191954}

W=4

h={0.056147, 0.120548, 0.166758, 0.156546, 0, -0.156546, -0.166758, -0.120548, -0.056147}

W=6

h={0.0241048,0.0565706,0.0887254,0.113667,0.121745,0.0951876,

0,-0.0951876,-0.121745,-0.113667,-0.0887254,-0.0565706,-0.0241048}

W=8

h={0.012625,0.0310207,0.0511411,0.0702644,0.0858448,0.0944324,0.09041,0.06426
16,0,-0.0642616,-0.09041,-0.0944324,-0.0858448,-0.0702644,-0.0511411,-0.03102
07,-0.012625}

W=10

h={0.00748668,0.0189226,0.0320897,0.0455317,0.0581325,0.068759,0.0759089,0.07
73052,0.0693696,0.046494,0,-0.046494,-0.0693696,-0.0773052,-0.0759089,-0.0687
59,-0.0581325,-0.0455317,-0.0320897,-0.0189226,-0.00748668}

W=20

h={0.00125395,0.00343173,0.00627354,0.00957463,0.0131724,0.0169357,0.020755,0
.0245343,0.0281832,0.0316094,0.0347115,0.037371,0.039444,0.0407516,0.041069,0
.0401128,0.0375257,0.032858,0.0255463,0.014886,0,-0.014886,-0.0255463,-0.0328
58,-0.0375257,-0.0401128,-0.041069,0.0407516,-0.039444,-0.037371,-0.0347115,-
0.0316094,-0.0281832,-0.0245343,-0.020755,-0.0169357,-0.0131724,-0.00957463,-
0.00627354,-0.00343173,-0.00125395}

附录五 对应于最优线性算子的平滑算子的计算程序及结果

Mathematica 源程序:

首先要调用前面计算出来的最优滤波器 $H_{\Sigma\Lambda}$ 的结果，参加附录三。我们假设 dd 是由各种尺度下最优滤波器形成的一个表。则可以用下面简单程序产生各个尺度的平滑算子。

```
s = {}
For[i = 1, i < 21, i++,
 t = {};
 t = Append[t, dd[[i]][[1]]];
 For[j = 2, j < 2 * i + 2, j++, t =
   Append[t, (dd[[i]][[j]] + dd[[i]][[j - 1]] + t[[j - 1]])]];
 sum = Plus @@ t;
 ot = t / sum;
 s = Append[s, ot];
 smooth = Round[s * 1000000] / 1000000 // N;
]
For[k = 1, k < 21, k++, Print["W=", k]; Print[smooth[[k]]]]
```

程序输出结果：

W=1

{0.25, 0.5, 0.25}

W=2

{0.067265, 0.25, 0.365469, 0.25, 0.067265}

W=3

{0.029255, 0.105597, 0.220745, 0.288806, 0.220745, 0.105597, 0.029255}

W=4

{0.015844,0.056688,0.114915,0.193312,0.238481,0.193312,0.114915,0.056688,0.015844}

W=5

{0.009734,0.034788,0.069712,0.113819,0.170554,0.202786,0.170554,0.113819,0.069712,0.034788,0.009734}

W=6

{0.006492,0.023249,0.046432,0.074792,0.10901,0.15196,0.176132,0.15196,0.10901,0.074792,0.046432,0.023249,0.006492}

W=7

{0.004588,0.016484,0.032933,0.052747,0.075807,0.103033,0.136672,0.155473,0.136672,0.103033,0.075807,0.052747,0.032933,0.016484,0.004588}

W=8

{0.003385,0.012208,0.02444,0.039074,0.055769,0.074757,0.096905,0.123961,0.139,0.123961,0.096905,0.074757,0.055769,0.039074,0.02444,0.012208,0.003385}

W=9

{0.002583,0.009349,0.018769,0.030014,0.042705,0.056818,0.072676,0.091036,0.113266,0.125567,0.113266,0.091036,0.072676,0.056818,0.042705,0.030014,0.018769,0.009349,0.002583}

W=10

{0.002024,0.007352,0.014806,0.023708,0.033698,0.04466,0.056689,0.070115,0.085

577,0.104165,0.11441,0.104165,0.085577,0.070115,0.056689,0.04466,0.033698,0.023708,0.014806,0.007352,0.002024}

W=11

{0.001622,0.005909,0.011936,0.019147,0.027221,0.036011,0.045512,0.055863,0.067369,0.080569,0.09634,0.105003,0.09634,0.080569,0.067369,0.055863,0.045512,0.036011,0.027221,0.019147,0.011936,0.005909,0.001622}

W=12

{0.001323,0.004835,0.009796,0.015746,0.022406,0.029626,0.037356,0.045643,0.054633,0.064601,0.076002,0.089549,0.096969,0.089549,0.076002,0.064601,0.054633,0.045643,0.037356,0.029626,0.022406,0.015746,0.009796,0.004835,0.001323}

W=13

{0.001096,0.004016,0.008162,0.013148,0.018732,0.024772,0.031203,0.038025,0.045303,0.05318,0.061898,0.071844,0.083606,0.090031,0.083606,0.071844,0.061898,0.05318,0.045303,0.038025,0.031203,0.024772,0.018732,0.013148,0.008162,0.004016,0.001096}

W=14

{0.00092,0.00338,0.006888,0.01112,0.015866,0.020996,0.026439,0.032173,0.038222,0.044659,0.051616,0.059306,0.068059,0.078366,0.083982,0.078366,0.068059,0.059306,0.051616,0.044659,0.038222,0.032173,0.026439,0.020996,0.015866,0.01112,0.006888,0.00338,0.00092}

W=15

{0.000781, 0.002877, 0.005878, 0.00951, 0.013589, 0.017999, 0.02267, 0.02757, 0.03269
9, 0.038092, 0.043823, 0.050011, 0.056846, 0.064608, 0.073714, 0.078664, 0.073714, 0.0
64608, 0.056846, 0.050011, 0.043823, 0.038092, 0.032699, 0.02757, 0.02267, 0.017999, 0
. 013589, 0.00951, 0.005878, 0.002877, 0.000781}

W=16

{0.00067, 0.002473, 0.005065, 0.008211, 0.011752, 0.015583, 0.019637, 0.023878, 0.028
295, 0.032902, 0.037737, 0.04287, 0.048411, 0.054525, 0.061457, 0.069559, 0.073954, 0.
069559, 0.061457, 0.054525, 0.048411, 0.04287, 0.037737, 0.032902, 0.028295, 0.023878
, 0.019637, 0.015583, 0.011752, 0.008211, 0.005065, 0.002473, 0.00067}

W=17

{0.00058, 0.002145, 0.004403, 0.007152, 0.010252, 0.013609, 0.017161, 0.020872, 0.024
724, 0.028718, 0.032873, 0.03723, 0.041853, 0.046842, 0.052342, 0.058565, 0.065813, 0.
069738, 0.065813, 0.058565, 0.052342, 0.046842, 0.041853, 0.03723, 0.032873, 0.028718
, 0.024724, 0.020872, 0.017161, 0.013609, 0.010252, 0.007152, 0.004403, 0.002145, 0.00
058}

W=18

{0.000506, 0.001874, 0.003855, 0.006273, 0.009007, 0.011971, 0.015108, 0.018382, 0.02
1774, 0.025278, 0.028901, 0.032666, 0.036612, 0.0408, 0.045318, 0.050298, 0.055924, 0.
062458, 0.06599, 0.062458, 0.055924, 0.050298, 0.045318, 0.0408, 0.036612, 0.032666, 0
. 028901, 0.025278, 0.021774, 0.018382, 0.015108, 0.011971, 0.009007, 0.006273, 0.0038
55, 0.001874, 0.000506}

W=19

{0.000444,0.001649,0.003399,0.005541,0.007967,0.010602,0.013392,0.016304,0.019316,0.02242,0.025616,0.028915,0.032341,0.035931,0.039741,0.043852,0.048381,0.053488,0.059403,0.062595,0.059403,0.053488,0.048381,0.043852,0.039741,0.035931,0.032341,0.028915,0.025616,0.02242,0.019316,0.016304,0.013392,0.010602,0.007967,0.005541,0.003399,0.001649,0.000444}

W=20

{0.000393,0.00146,0.003015,0.004924,0.00709,0.009446,0.011943,0.014549,0.017244,0.020015,0.02286,0.025783,0.028798,0.031928,0.035208,0.03869,0.042448,0.046583,0.051241,0.05662,0.05952,0.05662,0.051241,0.046583,0.042448,0.03869,0.035208,0.031928,0.028798,0.025783,0.02286,0.020015,0.017244,0.014549,0.011943,0.009446,0.00709,0.004924,0.003015,0.00146,0.000393}