编　　号：_____
审定成绩：_____

重庆邮电大学
毕业设计（论文）

设计（论文）题目：基于 Hessian 矩阵的路面裂纹检测

学　院　名　称：计算机科学与技术
学　生　姓　名：刘晓杰
专　　业：计算机科学与技术
班　　级：0491101
学　　号：2011211973
指　导　教　师：秦红星
答辩组　负责人：陈龙

填表时间：2015 年 6 月

重庆邮电大学教务处制

重庆邮电大学计算机科学与技术学院

本科毕业设计（论文）诚信承诺书

1. 本人郑重地承诺所呈交的毕业设计（论文），是在指导教师的指导下严格按照学校和学院有关规定完成的。

2. 本人在毕业设计（论文）中引用他人的观点和参考资料均加以注释和说明。

毕业设计（论文）作者签名：

年　　月　　日

摘　要

路面裂纹存在形状不确定，覆盖面广等特征，所以成了一个让道路交通部门头疼的问题。及时发现并修补这些裂纹可以减少由此引发的问题。但是，路面裂纹往往又不易发现，于是就成了检测的难点。

Hessian 矩阵在图形图像，尤其是裂纹，血管等条纹状图像中具有重要的应用。Hessian 矩阵的特征值和特征向量可以反应图像特征。其中，幅值最大的特征值对应的特征向量代表二维曲面曲率最大的方向，即裂纹或血管的切向方向，幅值最小的特征值对应的特征向量代表曲率最小的方向，也就是裂纹或血管方向。利用 Hessian 矩阵的这个特征可以进行裂纹的检测。

本文以路面裂纹图像为研究对象分析了裂纹图像的特征，根据 Hessian 矩阵的几何意义改进了 Hessian 矩阵的计算方法。以此为基础，本文提出一种基于 Hessian 矩阵的裂纹检测算法。算法内容如下：

首先对图像进行高斯滤波，然后计算多尺度下的 Hessian 矩阵，并利用特征向量进行分类及提取，获取裂纹初始点集。然后对 Hessian 矩阵的计算方法进行优化，并和原方法进行比较，确认最快捷有效的方法。最后受区域增长启发，对初始点集进行杂点去除，获取裂纹图像并分类。实验表明简化了的 Hessian 矩阵算法依然能有效检测路面裂纹。

【关键词】路面裂纹　Hessian 矩阵　特征向量　区域增长

ABSTRACT

Cracks exist features such as uncertainty and wide coverage, so become a problem of road traffic department. Timely detection and repair of these cracks can reduce the resulting problems. However, cracks are often not easy to find, so become difficult to detect.

Hessian matrix, especially to the crack, such as vascular stripes , has important application in the image. Hessian matrix eigenvalue and eigenvector can reflect the features of the image. Among them, eigenvector corresponding the amplitude of the largest eigenvalue represents the direction of the maximum curvature, the two-dimensional curved surface crack or blood vessels in the tangential direction. And eigenvector corresponding the amplitude of the smallest eigenvalues represents the direction of the minimum curvature, is the direction of crack or blood vessels. Cracks can be detected by applying the characteristics of the Hessian matrix.

This paper analyzes the characteristics of the images of crack by using pavement crack image as the research object.It improves the calculation method of the Hessian matrix according to the geometric meaning of Hessian matrix. On this basis, this paper proposes a crack detection algorithm based on the Hessian matrix.Algorithm content is as follows:First, using Gaussian filtering to filter the image. Then calculate the multi-scale Hessian matrix and use classification and feature vectors to extract the crack pixel to obtain crack initial points. Next, optimize the Hessian matrix calculation method, and compare with the original method, confirm the most fast and effective method. Finally, inspired by region growing, remove the impurity of the initial point set to obtain the crack of image and classify. Experiments show that simplified the Hessian matrix algorithm can effectively detect cracks.

【Key words】Roadcrack Hessian matrix eigenvector region growing

目　录

前　言

第一节　研究的背景和意义

近几十年来，随着全球各国公路建设迅速发展，全球经济得到了飞跃的发展。由此可见，道路交通在日常生活和宏观经济中占有举足轻重的的地位。然而由于自然环境和使用情况等因素的影响，路面都不可避免地受到破坏，严重降低道路寿命。通常破损的路面会给日常出行带来影响，比如行车舒适度，交通安全性。同时路面的承重性、耐久性也会受到严重的影响。如果不能及时地发现这些问题，对于交通安全将会是一个很大的隐患。由此产生了一个很重要的问题：如何对路面进行保养维护。对路面进行维护必须首先获取路面信息，比如是否有裂纹以及裂纹的大小。但是，图像质量在采集过程中会受到噪声等的干扰，导致图像原本的信息失真，再加上受到形状各异的路面裂纹的影响，要想仅仅使用某种特定的方法来检测裂纹不仅从方法上行不通，从效果上来看也很难令人满意。因此在裂纹的识别检测中会遇到如下问题。

1.图像质量在采集过程中会受到噪声等的干扰，导致图像原本的信息失真，出现图像细节模糊等不利于检测的问题。图像的滤波处理就是为了改善图像质量，但噪声种类多，滤波算法也多种多样，选取一种匹配的滤波算法意义重大。

2.路面裂纹形态各异，常见的有横向、纵向、龟裂、块状等。由于每一种裂纹都有其各自的几何特征，使得选择的算法很难满足各种裂纹检测的需要。

3.图像中除了裂纹骨架外，还存在很多的细小的裂纹以及众多孤立点，它们被称为杂质。因此，处理时要分开考虑，并针对不同类型的杂质的几何特征选择合适的除杂算法。

4.有些裂纹极难检测，比如一些非常细小的裂纹。它们的特征不是很明显，为了便于检测，通常需要特征增强算法。如何对细小裂纹进行特征增强而不影响大裂纹的检测也是一个问题。

解决这问题如果仅仅依靠人力来完成，那么会是一件相当痛苦而又繁杂的事情。人工检测存在以下不足：

(1) 人工检测存在主观性，检测结果会受到检测标准的不同而导致测量结果差异大。

(2) 耗时费力又不安全。人工检测需要专门的人员实地测量，会影响正常交通，也给自己的安全带来隐患。

因此，如何对整段路面裂纹做便捷有效的检测，成为解决裂纹识别问题的燃眉之急。

第二节　相关研究

目前的检测方法有以下几种：

第一种是基于FCM（Fuzzy C-Means clustering）分割和形态学的沥青路面图像裂缝提取算法[1]。首先分析了裂缝相对于路面的亮度、几何纹理等特征差异，然后根据这些差异再进行去噪、亮度非均匀校正和对比度增强等一系列预处理。之后采用FCM聚类方法分割图像就可以得到裂缝图像的粗糙版本。再利用形态学操作将粗糙版本精细化，最后对裂缝连通区域进行阈值判决，从而精确地提取了裂缝。实验结果证明该方法能有效地提取细小裂缝和弱对比度的裂缝。但公式过于复杂，计算量很大，而且阈值不好确定，因此难以大规模运用。

第二种是基于图像显著性的路面裂缝检测算法[2]。该算法先根据裂缝中心的灰度是局部最小值等特点对路面裂纹图像进行分块灰度校正，然后为了避免面临线性特征阈值和灰度阈值的选取问题转而计算粗尺度下的裂缝显著值，再根据人类视觉对于高对比度区域更感兴趣对图像进行不断扩张的细尺度的局部邻域显著性增强操作，然后又由于部分裂纹虽然偶尔出现断裂但依然整体连续的特点进行空间显著性加强操作，最后通过自适应阈值分割就能将裂缝很好地检测和提取出来。实验表明，该算法能正确有效地检测出裂缝整体区域，即使裂缝整体程度较为复杂该算法依然有效，抗噪声能力强，漏检率和误检率很低，但它的缺点和第一种一样，太复杂。

第三种是基于 BP 神经元提取方法[3]。该方法的核心思想是把输出层的误差从输出层反向向输入层逐层传递，并在传递的过程中降低误差。它有两个过程：数据流的正向传播和误差的反向传播。当正向传播时，输入信息从输入层经隐含层计算各个单元的输出值，如果得到的数据不是所期望的值，则对数据进行反向传播。通过反向传播，逐层计算各个单元的误差，并根据误差修正前一层权值。通过两个过程的交替进行，使误差在梯度方向以最快速度下降。该方法最大的优点就是有一定的容错能力，即使其中某些数据丢失或者被改动，误差依然能通过多次的修正而被纠正。该方法效果可以，缺陷就是太复杂，误差修正次数太多。

第四种是无人监督的裂纹检测——基于块的图像分析方法[4]。首先对图像进行分块，块中包含明显裂缝点的标记为“1”, 剩下的标记为“0”，由此产生一个二值化矩阵，记为 Mm。（这里要介绍两个图像集 TrS 和 TstS。TrS 集是一个训练集，它是根据图像的 Mm 矩阵分析得来的，它应当包含相关的路面裂缝的点。TstS 集就是所有的图像减去 TrS 集剩下来的）接下来就是裂纹的检测。对于每一张 TstS 集中的图像，计算它们的二维特征点来识别图像块是否有裂纹像素，没有则是 c1 类，有裂缝像素就是 c2 类。接下来就是进行裂缝类型的分类和严重程度等级的判定。该方法效果好，核心算法也不麻烦。唯一的缺陷就是最初的图像分组部分。该部分有强度归一化和像素饱和法（目标是获得相同的平均像素强度。块标有“0”表示是背景，而剩下的裂纹块保持一个较低的平均强度，标为“1”，表示是相关裂缝像素），特征提取和规范化（可以通过该方法形成一个一个“簇”，它取决于在一个图像块强度的平均值和标准差）以及集群技术（旨在将有类似特征的图像分组）。这些算法的复杂度高，效

率低下，不利于快速检测。

第三节　本文研究的难点

本文的研究对象是路面裂纹图像，将设计并实现一种用 Hessian 矩阵对路面图像中的裂缝进行精确地检测和提取的算法。首先深入研究 Hessian 矩阵在图形图像中的运用以及它的计算过程，然后根据传统的 Hessian 矩阵裂缝检测技术中存在的问题，对其进行改进及化简。最终，本文提出了有效可行的改进算法，其主要解决的难点如下：

第一、横截面的不同。现有的文献中 Hessian 矩阵主要运用在血管上[5][6]，但血管为 U 型，而路面裂纹为 V 型。因此，同样是用 Hessian 矩阵，计算出来的值会有差异，后面的用来提取目标像素的特征函数的阈值就会不一样。

第二、Hessian 矩阵涉及到二阶微分，计算量很大[7]，它会大量消耗计算机的资源，导致运行速度慢。因此，如何减少计算量，加快运行速度也是一个问题。

第三、方向的突变。血管的方向不会突变，最多偏离原方向 20 到 30 度。而路面裂纹的突变可能就是和原方向垂直，突变方向变化大。因此，有些简单而且直观的算法就不适用了。比如说利用脊线的几何方向作为跟踪方向[8][9]，脊线方向会突变，而特征向量的方向在脊线处相同。所以，理论上能使用的只有区域增长法。

第四节　论文的组织结构

本文采用 Hessian 矩阵进行裂纹的检测。因此，本文的章节安排如下：

首先是前言，介绍了裂纹检测的研究背景及意义，简述了使用不同方法进行裂纹检测的的国内外研究现状，分析了用 Hessian 矩阵进行裂纹检测算法研究存在的难题。

第一章是路面裂纹研究的近况。首先分析了路面裂缝的特征，包括灰度特征和种类的分类。然后介绍了近年来裂缝检测算法的核心算法，详细阐述了核心算法的思路和步骤，以及用到的公式，并分析了这些方法的优缺点。

第二章详细阐述基于 Hessian 矩阵的路面裂纹检测算法的整个流程。第一节先介绍了图像的滤波处理。分别介绍了三种不同的滤波方法，包括理论上的分析与预测，然后通过比较实际的运行效果选择最佳的滤波函数。然后第二节介绍了 Hessian 矩阵运用到图形图像的原理。接着介绍了四个版本的 Hessian 矩阵的计算方法和三个特征函数。通过组合不同矩阵的计算方法和特征函数来确定取得最佳效果的组合。第三节介绍非裂纹区和裂纹区杂质点的除杂方法以及裂纹的分类方法。非裂纹区主要是形态学的开闭运算和区域增长，裂纹区主要是针对特征向量方向的归类。通过这些除杂方法来获取最终的裂纹图像。最后，依据裂纹种类在几

何特征上的差异设计分类方法对裂纹分类。

第三章介绍了实验结果和分析。通过对另外的图像的测试，分析了整个流程的每一步产生的结果，找到本文所提出的方法的优点。

接下来是本文的结论。在结论中简述了本次裂纹检测的流程，并且分析了该算法的优点和缺陷，最后提出了进一步的改进意见和对以后工作的展望。

第一章　路面裂纹处理的研究现状

第一节　路面裂纹的特征

一、路面裂纹的灰度特征

我国路面中属沥青和混凝土比较常见，两者作为公路建设的普遍材料，具有耐磨、平整、成本低、方便施工等特点，但在使用的过程中，其性能都会随着使用时间及行车负载的影响而被破坏，出现不同程度的破损，其中最主要的表现形式就是出现裂缝，严重影响出行安全。

尽管路面的材料不同，但是其背景存在大量的噪声。裂缝与背景通常会呈现不同的灰度特征，对于已经成型的裂纹，其像素点的灰度值会高于背景上像素点的灰度值，背景和裂缝有较强的对比度；但是对于刚形成的裂缝，由于此时裂缝不清晰、较细，对比度较低的特点，像素点的灰度值与背景像素值比较接近，即噪声与裂缝有相似的灰度特征，导致裂缝检测过程中误差较大，影响检测的准确率。

二、路面裂纹的分类

裂缝形态各异，成因也多种多样，主要分为横向裂缝、纵向裂缝、龟裂裂缝、块状裂缝四类[11]。

（1）横向裂缝

横向裂缝是指大致垂直于行车方向的裂缝。横向裂缝根据形成原因可分为两大类：载荷性裂缝和非载荷性裂缝。横向裂缝的主要形式是非载荷性裂缝，这种裂纹的成因多种多样，不是人为能改变的。而载荷性裂缝的产生的主要原因是铺路时所使用的材料的质量低劣以及施工过程中偷工减料。

（2）纵向裂缝

纵向裂缝的形成原因有两种情况：（1）当路面分块铺路时，相邻两块在连接处未处理好，在车辆重力作用下，易形成纵向裂缝（2）当路面在铺路时，很难保证所用材料是均匀的，所以尽管压出来的路面一样平整，但是内部材料密度不均匀，就容易出现纵向裂纹。在半填半挖处的裂缝处极容易出现这样的裂纹。

（3）龟裂

龟裂也叫网状裂缝。路面经常要暴露在阳光下，会因为长期受热而导致结构软化、稳定

性不良，在这种情况下又受到猛烈撞击极容易产生网状裂纹。这种类型的裂纹通常每段都有较大的宽度。

（4）块状裂缝

块状裂缝是一种严重程度介于龟状裂缝与横纵裂缝的裂缝形态，由较多的横纵裂缝交错构成，裂缝间距大小相近，一般呈现近似直角的多边形大块。

可见，裂缝形态多样，不同形态之间的特征差异很大，因此提出一种能适应形态多变的裂缝检测算法是现代裂纹检测领域中的热点。

第二节　当前裂纹检测技术

裂缝检测是通过运用某种算法对输入的路面图像进行处理，最终达到获取精确的裂缝图像的目的，便于后续的分析工作，同时也是进一步提取裂缝信息并用于诊断路面病害的前提。因此，检测算法的有效性将直接影响裂缝检测的精确度和裂缝信息的真实性。

裂缝检测技术的研究近年来取得了很大的进步，获得了快速发展。通过对近几年国内外文献的查阅，对所查阅到的文献所采用的裂缝检测算法的理论基础进行如下详细介绍。

一、基于 FCM 分割和形态学的沥青路面图像裂缝提取

该算法主要包括图像预处理、FCM 图像分割、基于形态学的裂缝提取等三个部分。核心的 FCM 图像分割过程如下：

设特征空间 R^n 上的有一个有限数据集合集合 $Z=\{z_1,z_2,...,z_n\}$，再把 Z 划分为 c 类 $(2\le c\le n)$，设存在 c 个聚类中心 $V=\{v_1,v_2,...,v_c\}$，那么 FCM 聚类算法的目标函数如下：

$$J_{FCM}(U,V)=\sum_{n}\sum_{c}u_{i,j}^{m}\|z_i-v_j\|^2\ (m\ge 1) \tag{1.1}$$

上式中 c 是指所形成聚类的个数，n 是为数据样本个数，$\sum_{c}u_{i,j}=1,u_{i,j}\in[0,1],U=\{u_{i,j}\}_{i\le c,j\le n}$ 为隶属度矩阵，m 为模糊加权指数。FCM 聚类算法利用迭代法获得该目标函数最小时的聚类中心以及各个样本对于每一类的隶属度值，从而实现聚类的划分。

由于裂缝的像素灰度值较小，因此以表现图像亮度的主要指标灰度值作为聚类特征 z_i。记聚类中心 $v_j(1\le j\le c)$ 最小值对应的索引 $j=J$，如果 u_{ij} 在 $\{u_{i,j}\,|\,1\le j\le c\}$ 中为最大值，则标记 z_i 为裂缝。

实验结果证明该方法充分利用了裂缝与背景在几何上的差异，可以有效地提取细小裂缝和弱对比度的裂缝。但问题主要有两个：（1）公式过于复杂，计算量很大。除了核心算法，预处理中有一个光照非均匀校正步骤，需要计算均值和方差，计算量也不小（2）阈值不好确

定，不具备很好的自适应性。因此难以大规模运用，需要研究更智能的算法，降低人工干预。

二、基于图像显著性的路面裂缝检测

该算法主要包括灰度校正，显著性检测，细尺度显著性增强，显著性分割四大部分。显著性检测步骤内容如下：

为了避免由于线性特征阈值和灰度阈值的选取冲突问题，将原本采用的线性几何特征的方法改为计算显著值来滤除路面杂质。校正时，预设目标区域是每块灰度较暗的前l_r级像素。这样设置由局部分块中裂缝灰度的分布特点决定的，因为大多数分块中并不存在裂缝，没有裂纹的分块的预设目标区域的平均灰度值会比有裂纹的大。

先计算各分块r 前l_r灰度级直方图H_r，然后求所有分块的平均直方图:(N是块的个数)

$$H_l(i)=\begin{cases}\dfrac{1}{N}\sum_{r=1}^{N}H_{r,l_r}(i)...i\le l=\dfrac{1}{N}\sum_{r=1}^{N}l_r\\ 0...其他\end{cases} \tag{1.2}$$

前l_r灰度级直方图H_r中，大于平均区分阈值的点，可认为是伪裂缝，在平均直方图中取值为0，显著值也为0。而裂缝在其中依旧稀疏，其灰度与平均灰度I_{avg}距离大，裂缝局部显著性$S_l(x,y)$定义为公式(1.3)：

$$\begin{aligned}&S_l(x,y)=e^{(-H_l(I(x,y)))}(I_{avg}-I(x,y))\\&其中I_{avg}=\sum_{i=1}^{l}H_l(i)*\frac{i}{\sum_{i=1}^{l}H_l(i)}\end{aligned} \tag{1.3}$$

对灰度校正后的图像高斯模糊，得到I_g:

$$\begin{aligned}&G(x,y,\rho)=\frac{1}{2\pi\sigma^2}e^{\frac{-(x^2+y^2)}{2\sigma^2}}\\&I_g(x,y)=I(x,y)*G(x,y,\rho)\end{aligned} \tag{1.4}$$

然后计算图像的灰度均值I_μ：

$$I_\mu=\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}I(i,j)}{M*N} \tag{1.5}$$

将I_g中每个像素点与I_μ的灰度级的欧氏距离作为该点全局显著值$S_g(x,y)$：

$$S_g(x,y)=(I_g(x,y)-I_\mu)^2 \tag{1.6}$$

高斯模糊核函数标准差σ 反映像素分布密度的散布特征，较大时，周围像素影响增强，中心像素影响减弱，图像越模糊；较小时表示像素影响值集中于分布中心。

综合局部显著性、全局显著性，分别取Sl和Sg中大于各自平均显著值的点，利用公式(1.7)

计算裂缝最终显著值：

$$S_z(x,y)=\sqrt{S_l(x,y)^2+S_g(x,y)^2} \tag{1.7}$$

这是显著性检测步骤中用到的公式，可以看到，公式复杂。实际上另外三个步骤用到的公式和这个步骤差不多，因此计算量相当复杂。实验表明，该方法在较复杂环境下的依然能有效检测裂缝，误检率和漏检率很低。

三、基于 BP 神经元提取方法

该算法的流程如下：先用中值滤波进行去噪，然后对裂纹图像主要目标特征用扩展sobel算子进行增强。这样分割后的二值图像依然存在孤立点，所以用数学形态学的相关操作即可。最后根据裂纹的几何特征提取其特征信息，并设计了BP神经网络对路面裂缝进行分类。

BP网络的算法的实现如下：

神经元是构成BP网络的主要元素，这些神经元所用的激活函数必须是处处可导的。因此，大多使用S型函数。本BP网络的神经元的网络输入为：

$$net=x_1w_1+x_2w_2+...+x_nw_n \tag{1.8}$$

其中，$x_1,x_2...x_n$ 是基本神经元输入，$w_1,w_2...w_n$ 是对应的链接权值。因此，基本神经元的输出为

$$O=f(net) \tag{1.9}$$

当输入某个样本时，正向算法计算方法如公式(1.10)：

$$net_j=\sum w_{jk}O_i \tag{1.10}$$

其中：

$$O_i=f(neti_i) \tag{1.11}$$

输出层的实际输出值是 $\overline{y_j}=O_j$，Yj是期望输出值，此样本的误差用平方型的误差函数表示如下：

$$E=\frac{1}{2}\sum_j(y_j-\overline{y_j})^2 \tag{1.12}$$

误差信号的定义如下：

$$\delta_j=\frac{\partial E}{\partial net_j} \tag{1.13}$$

由上面两式可得：

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial net_j}*\frac{\partial net_j}{\partial w_{ij}}=\delta_jO_i \tag{1.14}$$

权值的修正的原则就是以误差函数下降最快的方向作为修正方向，修正值由公式(1.15)列出：

$$\begin{aligned}\Delta w_{ij}&=-\eta\frac{\partial E}{\partial net_j}-\eta\delta_jO_j\\ w_{ij}(k+1)&=w_{ij}(k)+\Delta w_{ij}(kk)\end{aligned} \tag{1.15}$$

这时，可以分两种情况来讨论：

如果节点 j 是输出单元，则：

$$O_j = \overline{y_j}$$

$$\delta_j = \frac{\partial E}{\partial \overline{y_j}} * \frac{\partial \overline{y_j}}{\partial net_j} = -(y_j - \overline{y_j}) f'(net_j) \tag{1.16}$$

如果节点 j 不是输出单元，则：

$$\begin{aligned} \delta_j &= \frac{\partial E}{\partial net_j} = \frac{\partial E}{\partial O_j} * \frac{\partial O_j}{\partial net_j} = \frac{\partial E}{\partial O_j} f'(net_j) \\ \frac{\partial E}{\partial O_j} &= \sum_k \frac{\partial E}{\partial net_k} * \frac{\partial net_k}{\partial O_j} = \sum \delta_k w_{jk} \\ \therefore \delta_j &= \{\sum \delta_k w_{jk}\} f'(net_j) \end{aligned} \tag{1.17}$$

在实际计算过程中，有一种加快收敛速度的方法是通过在权值修正量中加入前一次的权值修正量，即：

$$\Delta w_{ij}(k) = \eta \delta_j O_j + \alpha \Delta w_{ij}(k-1) \tag{1.18}$$

η 表示学习速率，η 较小时，学习速率较慢，相对平稳；η 较大时，权值修改量很大，学习速率较快，但可能会产生振荡。α 是一个确定过去学习效率的常数。

综上所述，BP 算法步骤如下：

(1)选定学习速率 η 和训练总误差精度 E，设定权系数初始值。

(2)输入 n 个学习样本以及期望输出，以及 t 个测试样本。

(3)依次计算各层各单元输出。

(4)分别计算各输出层和隐层的局部误差梯度以及修正各层各单元权值。

(5)如果结果达到初始误差精度则训练结束，得到 t 个实际输出。

通过对 BP 神经网络的计算过程观察可以发现，该方法为了修正误差，需要做大量的计算，而且误差的修正取决于收敛速度。复杂度高。但是该方法最大的优点就是有一定的容错能力，即使其中的部分数据丢失或者被改动，依然可以通过多次的误差修正而得到改正。

第三节　本章小结

本章首先分析了路面裂缝的特征，包括灰度特征和种类的分类。然后介绍了近年来裂缝检测算法的核心算法，详细阐述了核心算法的思路和步骤，以及用到的公式，并分析了这些方法的优缺点。通过对近年来的裂纹检测算法的分析，为下文提出利用 Hessian 矩阵提取裂纹的算法做铺垫。

第二章　基于 Hessian 矩阵的裂纹检测

第一章阐述了近年来裂纹检测的核心算法思路和步骤，以及用到的公式。通过对近年来的裂纹检测算法的分析可知，它们的效果好，但是都存在算法流程复杂，效率不高等特点。因此本章提出了基于 Hessian 矩阵的路面裂纹检测算法。该算法流程如下：第一节主要介绍了滤波函数。先介绍了三种滤波函数的实现机制，然后通过实验结果选择最佳滤波函数。第二节阐述了基于 Hessian 矩阵的裂纹检测算法的流程。在这一节中，先介绍了 Hessian 矩阵的原理，然后介绍了四种 Hessian 矩阵的计算方法以及三种特征函数。第三节主要介绍了裂纹的除杂，包括非裂纹区和裂纹区。第四节是本章小结。综合整个流程来看，该算法流程简单，但是效率不高。为了提高效率需要化简 Hessian 矩阵的计算，其中库函数化简版就是为了简化计算而产生的。该算法提高了整个算法的效率，同时也获得了较好的效果。

第一节　图像的滤波处理

通常所说的图像是指的模拟图像。这类图像不能直接被计算机处理，只能间接经过离散化处理转变为数字信号才可以被计算机接受。通常将计算机图像处理称作数字图像处理。数字图像处理的主要内容包括图像信息的描述、分析以及编码和显示。数字图像处理的目的是通过增强图像中的某些成分并抑制某些成分来提高图像质量，滤去图像噪声，使计算机能够很方便快捷地分析和处理图像的频域，纹理特征等。

数字图像处理的优点有再现性好、处理精度高以及易于控制处理效果。因此，近年来数字图像处理已经被广泛的应用于医疗、通信、智能机器人等领域[12]。数字图像处理的方法也可以用在裂纹图像的滤波处理上。对采集的图像进行预处理是为了消除噪声并进一步地获取裂纹信息。在图像上，噪声表现为极大或极小的像素值，这些极大极小值给图像造成亮、暗点干扰，造成裂纹信息不突出，图像质量被大大降低，同时又会由于各种算法运用的原因，影响图像二值化的结果，并进一步影响图像的分割、特征提取、识别等后继工作。判断滤波函数能否有效地抑制噪声的两个标准是：（1）能否有效地去除背景中的噪声，这是最基本的标准（2）能否很好地保护感兴趣区域的大小、形状以及裂纹的几何结构特征，这是滤波函数的深层次要求。在数字图像处理中，滤波处理其实是一种邻域运算。整个滤波过程就是采用一定的算法，将输入图像中的像素及其周围像素值通过某种运算来替代原来的像素值。因此，算法的有效性和是否有针对性会决定滤波算法是否合理。

如前所述，图像滤波的目的就是消除噪声。常见的噪声有椒盐噪声和高斯噪声。椒盐噪

声是往往由图像切割引起。一般分为两种噪声，一种是盐噪声，另一种是胡椒噪声。当图像上同时存在两种噪声的时候，会呈现黑白杂点散乱分布的情况。高斯噪声是指它的概率密度函数服从高斯分布（即正态分布）的一类噪声。高斯模型有一个很特殊的性质：易处理性。这使得在空间域和频域中这种噪声模型经常被用到。事实上，它经常适用于临界情况下。为了消除这些噪声，均值滤波，中值滤波，高斯滤波等方法是很好的选择。

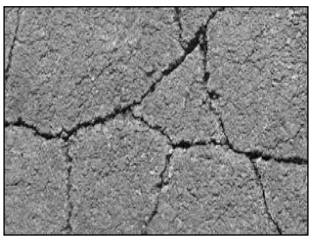
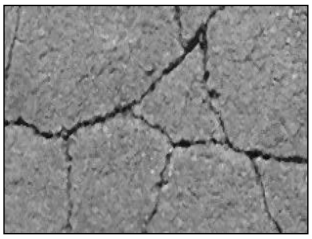


图 2.1 原图

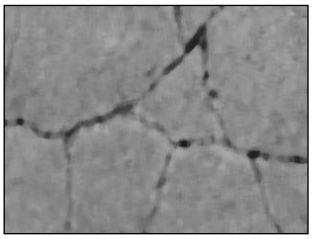
一、均值滤波

邻域平均法是均值滤波的核心方法。其基本原理是用中心像素及其周围像素的均值代替原像素值，即对当前像素点 g(x,y)，选择一个 w*w 个像素的邻域，求模板中所有像素的均值，再把该均值赋予当前像素点 g(x,y)。该算法可用公式(2.1)表示：

$$g(x,y)=\frac{1}{w*w}\sum f(x,y) \tag{2.1}$$



w=3



w=7

图 2.2 不同 w 取值的均值滤波

实验结果证明了模板宽度 w 的大小直接影响了图像滤波的效果。均值滤波是以某个像素点的邻域像素点平均值替代原像素点的像素值，这种算法尽管缩小了图像中各个像素点像素值之间的差距，但是不能很好地保护图像细节，在图像去噪的同时也破坏了图像的细节部分。由于均值滤波会破坏裂纹图像中的细小的裂纹，因此无法应用到本研究中。

二、中值滤波

中值滤波的原理跟均值滤波有点类似，图像内每一个像素值都和邻域内的像素值有关，

不同的是，均值滤波是用模板内像素的平均值来代替中心像素值，而中值滤波则是将模板内的所有像素点排序，再选择中间值代替中心像素值。

中值滤波具体方法如下：

1.选择一个窗口，通常为正方形，而且像素点个数是奇数，如 3*3

2.确定该窗口内的中心像素点

3.将窗口内的所有像素点按照像素值按照某个顺序排序

4.取出排好序的中间的像素值赋给中心像素点

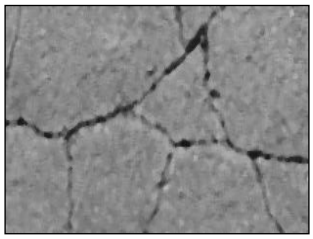


图 2.3 中值滤波图

中值滤波法常用于保护边缘信息，对消除椒盐噪声非常有效，但在条纹中心分析方法中作用不大。通过图2.3和图2.1的比较可以看出，它确实减少了噪声点，保护了边缘，但它有可能改变裂纹的结构。如果裂纹仅1像素宽，那么无论选择多大的邻域，排序以后的目标像素点的位置不是在最后就是在最前，那么原像素就会被替代，从而导致裂纹“消失”。

三、高斯滤波

高斯滤波适用于消除高斯噪声，是一种线性平滑滤波。它的实现方式有两种，一是通过离散化窗口滑窗卷积的方式，另一种通过傅里叶变换实现。第一种比较常见，通俗来讲就是对整幅图像进行加权平均的过程。每一个像素点的值，都由其本身和邻域内的其他像素值经过加权平均后得到。可见第一种方法计算量很大，只有当计算量实在太大才会考虑第二种。

高斯滤波中最重要的就是高斯模板。高斯模板的计算方法如下：（此处计算的是 3*3 的模板）

公式(2.2)表示离散高斯分布：

$$G=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.2}$$

假设以模板的(1,1)为中心，那么高斯分布变成公式(2.3)：

$$g(i,j)=\frac{1}{2\pi\sigma^2}\ell^{-\frac{(i-1)^2+(j-1)^2}{2\sigma^2}} \tag{2.3}$$

计算出来的模板如下：

$$\begin{vmatrix} \ell^{-\frac{2}{2\sigma^2}} & \ell^{-\frac{1}{2\sigma^2}} & \ell^{-\frac{2}{2\sigma^2}} \\ \ell^{-\frac{1}{2\sigma^2}} & \ell^{-\frac{0}{2\sigma^2}} & \ell^{-\frac{1}{2\sigma^2}} \\ \ell^{-\frac{2}{2\sigma^2}} & \ell^{-\frac{1}{2\sigma^2}} & \ell^{-\frac{2}{2\sigma^2}} \end{vmatrix} \overset{\sigma^2=1}{=\!=} \begin{vmatrix} \frac{1}{\ell} & \frac{1}{\sqrt{\ell}} & \frac{1}{\ell} \\ \frac{1}{\sqrt{\ell}} & 1 & \frac{1}{\sqrt{\ell}} \\ \frac{1}{\ell} & \frac{1}{\sqrt{\ell}} & \frac{1}{\ell} \end{vmatrix} \overset{\text{化简}}{=\!=} \begin{vmatrix} 1 & \sqrt{\ell} & 1 \\ \sqrt{\ell} & \ell & \sqrt{\ell} \\ 1 & \sqrt{\ell} & 1 \end{vmatrix} = \begin{vmatrix} 1 & 1.64 & 1 \\ 1.64 & 2.7 & 1.64 \\ 1 & 1.64 & 1 \end{vmatrix} \tag{2.4}$$

以上就是方差为 1 的 3*3 高斯模板。高斯滤波就是将这个模板和像素点本身以及它的邻域内其他像素点经过加权平均后替代原像素值的过程。图 2.4 就是高斯滤波后的效果图。

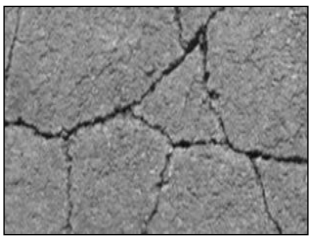


图 2.4 高斯滤波图

通过和前两节滤波效果比较发现：高斯滤波较好地滤除图像中的高斯噪声以及其它因素干扰所产生的噪声，它能够避免中值滤波无法适用条纹中心分析法的缺点，也能避免均值滤波模糊图像的弊端，较好将裂纹的真实信息保留，为后续的 Hessian 矩阵处理提供了方便。

高斯滤波算法有一个缺点，即均值和方差对实验结果的影响很大。如果选取的均值和方差不同，则结果就不同。只有选择合适的均值和方差，才能得到良好的效果。假如选择的值太大就会丢失裂纹信息。

第二节　基于 Hessian 矩阵的裂纹检测

Hessian 矩阵在条状目标的提取中有广泛且重要的应用。本章先研究了 Hessian 矩阵的原理。然后通过观察法给出 Hessian 矩阵的第一个应用——观察法 Hessian 矩阵，通过查阅资料给出了 Hessian 矩阵的经典版。然后通过分析发现经典版有可以改进之处，所以就产生了第三个应用——库函数化简版。最后通过查阅资料给出了最后一个版本。介绍完了 Hessian 矩阵的应用，又介绍了三个经常使用到的用于提取裂纹点的特征函数。通过多次的对比实验就得到了一幅较好的裂纹图像。

一、Hessian 矩阵原理

在数学中，海森矩阵（Hessian matrix 或 Hessian）是一个方块矩阵，由自变量为向量的实值函数的二阶偏导数组成。在数字图像处理中，它是一种用高阶微分提取图像特征方向的

方法[13]。假设函数 $f(x)$ 在任一方向上都存在二阶导数，那么海森矩阵即可写成公式(2.5)：

$$H=\begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix} \tag{2.5}$$

可见，海森矩阵其实就是多元函数的二阶导数。同时，在 Hessian 矩阵的两个特征值 λ_1 和 λ_2 中，最大的特征值对应的特征向量代表着该点曲率最大的方向，该方向与裂纹方向垂直，而最小的特征值对应的特征向量代表该点曲率最小的方向，它与裂纹方向平行，由这个较小的特征向量就可以探测裂纹的方向。

二、观察法 Hessian 矩阵

理想的图像裂纹横截面是 V 型，相对应的图像像素值大小也是 V 型。求得的一阶导数就呈线型，再求二阶导数就成了常数，并且大于 0。那么裂纹中心点就是一阶导数为 0，二阶导数大于 0 的点。由于像素上的变化可以直观地通过肉眼看出，所以该方法被称为观察法 Hessian 矩阵。但是，由于现实中图像有噪声以及像素点不连续，所以在选取裂纹点集的时候只需满足一阶导数小于某个值（本文取 30），二阶导数大于 0。（一阶导数，二阶导数的计算方法分别是：

$$\begin{cases} f_x(x,y)=f(x+1,y)-f(x,y) \\ f_{xx}(x,y)=f_x(x+1,y)-f_x(x,y) \\ f_y(x,y)=f(x,y+1)-f(x,y) \\ f_{xy}(x,y)=f_x(x,y+1)-f_x(x,y) \\ f_{yy}(x,y)=f_y(x,y+1)-f_y(x,y) \end{cases} \tag{2.6}$$

$f(x,y)$ 表示图像在 (x,y) 处的像素值）

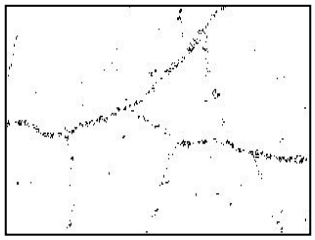


图 2.5 最简处理

该方法将原本大量的乘法全部变成了减法，大大简化了运算。

它的优点是计算简单, 计算量少；但是缺点也很明显，数据不精确。因为会有伪裂纹点的干扰，如地面坑洼之处，而导致计算结果发生变化。和之后的方法比较就可以知道它漏检测了好多像素点。

三、Hessian 矩阵经典版

Hessian 矩阵也可以通过图像与一定尺度的高斯滤波器

$$G=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.7}$$

相应的二阶导数的卷积求得[14][15]：

$$f_{uv}=\sigma^2*G_{uv}*I_{uv}(u,v\in[x,y]) \tag{2.8}$$

那么二阶导数如公式(2.9)：

$$\begin{cases} G_x=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}}*(-\frac{x}{\sigma^2}) \\ G_y=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}}*(-\frac{y}{\sigma^2}) \\ G_{xx}=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}}*(\frac{x^2-\sigma^2}{\sigma^4}) \\ G_{xy}=G_{yx}=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}}*(\frac{xy}{\sigma^4}) \\ G_{yy}=\frac{1}{2\pi\sigma^2}\ell^{-\frac{x^2+y^2}{2\sigma^2}}*(\frac{y^2-\sigma^2}{\sigma^4}) \end{cases} \tag{2.9}$$

最终求得的二阶导数和尺度σ有关，因此采用多尺度更合适，即对每个像素点选取多个不同的σ来计算，并通过比较选择最佳值。然后选择特征函数的最大值作为输出值，这样就得到了一幅原始的裂纹图。特征函数和实验结果都将在本章的第六节讨论。

四、库函数化简版 Hessian 矩阵

通过上一节的公式可以看到，对于某一个像素点(x,y)，要想计算它的 Hessian 矩阵，计算量相当大，主要是它需要大量的乘法。后面还要调用库函数来计算特征向量，它的复杂度很高。不过可以观察到每个二阶导数卷积都可以化简。根据 Hessian 矩阵的表达式：

$$H=\begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix} \tag{2.10}$$

可以计算它的特征多项式：

$$pH(\lambda)=\lambda^2+\alpha_1\lambda+\alpha^2=0 \tag{2.11}$$

其中$\alpha_1=-(f_{xx}+f_{yy}),\alpha_2=f_{xx}*f_{yy}-f_{xy}*f_{yx}$，那么$H(\lambda)$的两个特征值就是

$$\lambda_1 = \frac{-\alpha_1 + \sqrt{\alpha_1^2 - 4\alpha_2}}{2}，\quad \lambda_2 = \frac{-\alpha_1 - \sqrt{\alpha_1^2 - 4\alpha_2}}{2} \tag{2.12}$$

根据方向向量的计算公式

$$\begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.13}$$

可以求得

$$\begin{cases} x * f_{xx} + f_{xy} * y = \lambda x .....(1) \\ x * f_{yx} + f_{yy} * y = \lambda y .....(2) \end{cases}$$

$$(1)\text{式化简得} \ : y = \frac{f_{xx} - \lambda}{f_{xy}} x \tag{2.14}$$

$$(2)\text{式化简得} \ : y = \frac{f_{yx}}{\lambda - f_{yy}} x$$

（(1)式和(2)式虽然表达是不一样，但是一个λ对应一个特征向量，最后两个公式算出的特征向量是一样的）由于较小的特征值对应的特征向量代表该点曲率最小的方向，即与裂纹方向平行，所以λ取较小值。

λ取$\lambda_2 = \frac{-\alpha_1 - \sqrt{\alpha_1^2 - 4\alpha_2}}{2}$, 则$y = \frac{f_{xx} - \lambda_2}{f_{xy}} x$。把$\alpha_1$，$\alpha_2$代入公式(2.12)可得：

$$\begin{aligned} \lambda_2 &= \frac{f_{xx} + f_{yy} - \sqrt{(f_{xx} + f_{yy})^2 - 4(f_{xx} f_{yy} - f_{xy} f_{yx})}}{2} \\ &= \frac{f_{xx} + f_{yy} - \sqrt{(f_{xx} - f_{yy})^2 + 4 f_{xy} f_{yx}}}{2} \end{aligned} \tag{2.15}$$

$$\frac{f_{xx} - \lambda_2}{f_{xy}} = \frac{f_{xx} - f_{yy} + \sqrt{(f_{xx} - f_{yy})^2 + 4 f_{xy} f_{yx}}}{2 f_{xy}} \tag{2.16}$$

把二阶导数代入公式(2.14)的(1)式进行化简，最终得到公式(2.16)。特征函数和实验结果同样都将在本章的第六节讨论。

观察公式 2.4，可以发现对于某个像素点而言，简化版计算方向向量时只需要计算四个二阶导数；而经典版不仅需要计算四个二阶导数，还要调用库函数 cvSVD()。为了可以使用这个库函数，在编码时不仅需要开辟一个 2*2 的空间来存储四个二阶导数，还要开辟一个 2*2 的空间来存储两个方向向量方便取数据。两个 2*2 的空间可以提前开辟，但是数据的存取是需要时间开销的。简化版相较于经典版，节省的时间主要在于库函数的调用和频繁的数据存取。因此，该版本被称为“库函数化简版”。

五、模糊版 Hessian 矩阵

该版本在计算二阶导数方面和经典版一样，但是对像素点的判定标准不同[16]。该版本的判断标准如公式(2.7)：

$$\begin{cases} Det(H) = f_{xx} * f_{yy} - w^2 * f_{xy}{}^2 .....(1) \\ Sig = \begin{cases} 1 ..... f_{xx} + f_{yy} > -0.0001 \\ -1 ..... f_{xx} + f_{yy} < -0.0001 \end{cases} ....(2) \\ V = \begin{cases} 0, Det(H) < 10^{-9} \\ Sig * Det(H), Det(H) > 10^{-9} \end{cases} ....(3) \end{cases} \tag{2.17}$$

公式(1)中 $Det(H)$ 表示 Hessian 矩阵行列式的近似值，为了弥补近似值和原值之间的差值，添加权值 w，本文中 w=1。这里取公式(3)中的 V 表示最后的检测依据。这样的简化把原本复杂的卷积运算分成简单的加减运算，速度大大提升了。但是总体效果和经典版，库函数化简版的方法的效果差距大。后面第六节有分析。由于该方法是建立在近似值上，所以被称为“模糊版”。

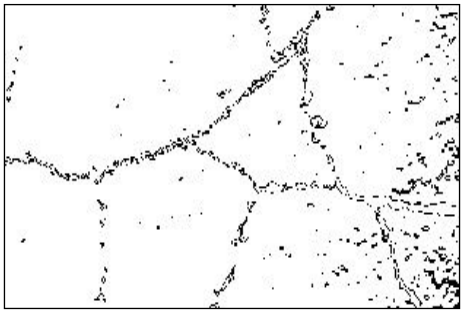


图 2.6 模糊版效果图

六、特征函数的选取和实验对比

在图形图像中，某个特定的特征函数将成为判断该像素点是否满足某个条件的依据。在本课题研究中总共有如下几个特征函数[17][18]：

$$\begin{cases} 1.V = \begin{cases} \ln(|\lambda_1| + 1) ..... \lambda_1 > T \\ 0 ..... \text{...其它} \end{cases}, \ T\text{为阈值} \\ 2.V = \begin{cases} |\lambda_1| - |\lambda_2| ..... \text{...} \lambda_1 < 0 \\ 0 ..... \text{...其它} \end{cases} \\ 3.V = \begin{cases} \dfrac{|\lambda_2|^2}{|\lambda_1|} ..... \text{..} \lambda_1 < 0 \text{且} \lambda_2 < 0 \\ 0 ..... \text{.....其它} \end{cases} \end{cases} \tag{2.18}$$

把经典版记为 A，库函数化简版记为 B，本节的三个特征函数记为 c，d，e。图 2.7 就是所对应的一系列效果图。

先横向看。Ac 组合可以有效的检测裂纹，Ad、Ae 由于每行的数据成线性分布，所以特征函数阈值的取值无论是什么值都无法有效检测裂纹。Bc，Bd，Be 除了右下角的伪裂纹点多以外效果可以，其中，Bc 组合效果最好，因为左下角的伪裂纹点最少。

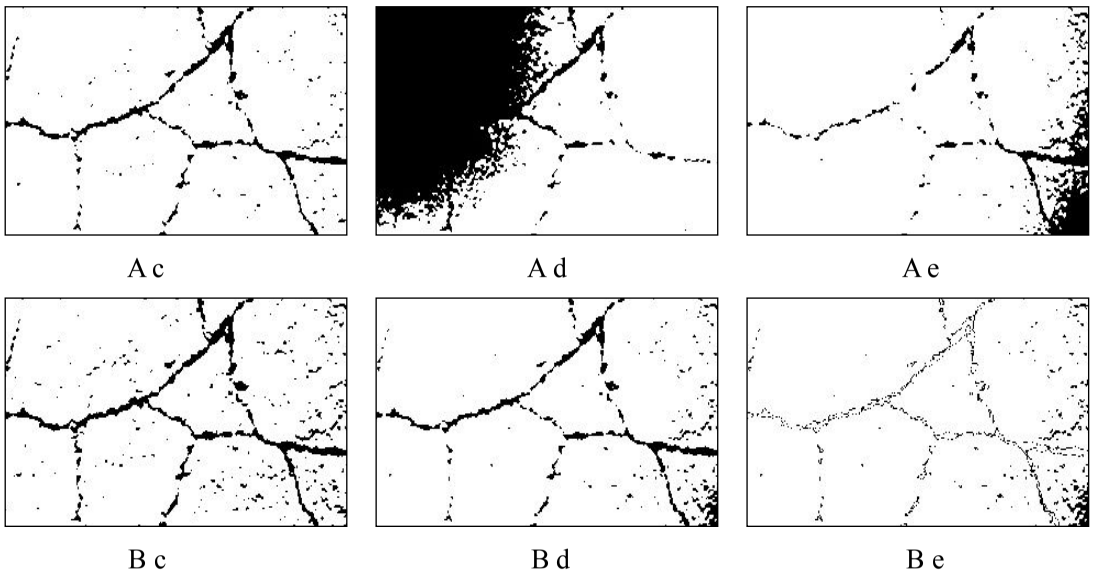


图 2.7　Hessian 矩阵计算效果图

然后纵向比较，主要比较 Ac，Bc。经比较可以得出两个结论。第一，第一个特征函数可以作为本课题裂纹检测的最佳特征函数；第二，库函数简化版 Hessian 矩阵计算方法同样可以有效检测裂纹，因为两图像效果都很好。因此，选 Bc 作为该步骤的最后结果。

第三节　后期处理

通过 Hessian 矩阵处理后的裂纹图像含有大量的杂质点，其中包括非裂纹区和裂纹区。非裂纹区的杂质点主要是由于地面坑洼之处而遗留下来的，可以通过膨胀腐蚀这类形态学处理方法，也可以使用区域增长这种利用非裂纹区杂质点和裂纹点的特征的不同进行除杂。裂纹区的杂质点主要是由特征向量的方向不同引起的，可以通过对特征向量方向的归类进行除杂操作。

一、非裂纹区除杂

1. 形态学

数学形态学的基础是集合论，它是一种可以用来进行几何形态学分析的工具。近年来，

由于其在理论上的不断完善，加上易于用硬件实现，已经引起越来越多的关注。目前，形态学已经在图形学，数据可视化等领域发挥着越来越重要的作用。

形态学中，有两个基本操作：腐蚀和膨胀[19]。在介绍腐蚀膨胀操作前先介绍一个概念：结构元素。结构元素可以是任意形状，但是在图像处理中一般是一个 n 阶矩阵。形态学处理的过程就是将结构元素的中心点和图像中的某个点重合，然后在每个像素位置上与二值图象对应的区域进行特定的逻辑运算，将运算结果赋给重合的像素点。结构元素的形状、大小以及采取何种逻辑运算等对于形态学运算的效果有直接影响。

(1) 腐蚀

通常在裂纹图像中存在这样的杂质点：一些小且孤立的像素块。为了提高图像质量，可以采取腐蚀操作。这是一种用来消除边界点，使边界向内部收缩的操作。腐蚀的概念一般定义为公式(2.19)：

$$E = X \otimes B = \{(x, y) \mid B_{xy} \subseteq X\} \tag{2.19}$$

由公式可知，二值图象 E 是由二值图像 X 用结构元素 B 腐蚀产生的，E 是满足以下条件的点(x,y)的集合：如果 B 的原点平移到点(x,y)，那么 B 将完全包含于 X 中。通俗来讲就是拿 X 上的点和 B 的中心点进行对比，如果 B 上的所有点都在 X 的范围内，则该点保留，否则将该点去掉。最终结果从视觉效果来看就是 E 比原来 X 的范围少了最外面的一层像素点。

(2) 膨胀

裂纹图像中除了存在小且孤立的像素块，还存在这样的情形：一些细但不长的裂纹，如若视为杂质，则会造成漏检测。但是这些裂纹是断开的，为此需要一种可以将它们间接在一起的操作，即膨胀操作。膨胀操作是将与目标区域相邻的所有背景点合并到该物体中，使边界向外部扩张的操作，可以用来连接断裂的目标以及填补目标区域中的空洞。膨胀概念一般定义为公式(2.20)：

$$D = X \oplus B = \{(x, y) \mid B_{xy} \cap X \neq \phi\} \tag{2.20}$$

从公式来看，二值图象 D 是由二值图像 X 用结构元素 B 膨胀产生的，D 是满足以下条件的点(x,y)的集合：如果 B 的原点平移到点(x,y)，那么它与 X 的交集非空。通俗来讲，就是拿 X 上的点和 B 的中心点进行对比，如果 B 中有一个点在 X 的范围内，则该点保留，否则将该点去掉。最终结果就是把图像四周的满足条件的背景点全部合并到区域中，使该区域的面积增大。

(3) 开运算和闭运算

形态学处理的有一个很重要的目的就是保证目标区域面积不变。虽然膨胀处理可以连接

断开的像素块，但是目标面积变大了；虽然腐蚀处理可以分离粘连的像素块，但是目标区域的面积变小了。为了解决这个问题，开闭运算应运而生。

开运算是指先进行腐蚀后进行膨胀的运算。先通过腐蚀消除了孤立的杂质点，但是裂纹的整体区域也少了一层，紧接着的膨胀操作又给裂纹的整体区域也加了一层，由于孤立的点已经滤除，所以膨胀操作不会生成这些孤立点。由此可见，开运算通常用于去除孤立的杂质点，其主要作用与腐蚀相似。但与腐蚀操作相比，最大的优点是可以基本保持目标区域大小不变。

闭运算是指先膨胀后腐蚀的过程。闭运算过程大致如下：通过膨胀可以连接断裂的裂纹，同时也让裂纹区域加了一层，之后用腐蚀操作把加的一层去掉，而已经连接上的不会再次断裂。由此可见，它通常用来填充物体内细小空洞、连接邻近像素点的。

(4) 结果展示

此处的目的是要去除非裂纹区的杂质，应该选择开运算。下图是实验结果。比较图 2.7Bc 和图 2.8，

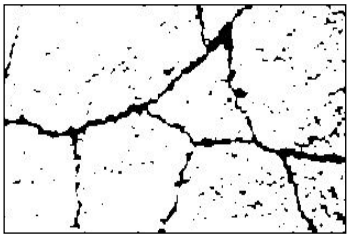


图 2.8 开运算效果图

发现两者变化小，该操作效果差。最有可能的原因是这些伪裂纹点相对于裂纹本身无法被看作“细小”，而且裂纹本身也很细。

2. 区域增长

区域生长（region growing）是指将具有某个属性的种子点归并进某个集合 S，然后遍历所有点集，将每个与种子点有相似属性的像素合并到此 S 集合，最终集合 S 的元素将无法增长，此时区域增长停止[20][21]。

在区域生长中的主要问题有如下两点：1.如何选择初始化种子：在区域生长过程中，选择的种子点要具备合适的属性和特征。2.基于图像的某种属性的生长原则不一定能取得良好的效果。就算选择了合适的种子点，存在的杂质点可能也会影响整个增长过程的效率。如果效率过低，应该选择其他的种子点。

简单地通过形态学处理无法有效的去除非裂纹区与的杂质点，因此可以采取区域增长的方法。非裂纹区域虽然也可能存在方向向量相同的像素点，但它们它们最大的特点是很分散。受上述区域增长的启发，我们将算法改为如下：

先对现在所存在的所有像素点构建集合 S。然后所有 S 集合内的像素点，对于每一个像素点，围绕它构建一个宽度为 w 的窗口。扫描窗口中的像素点总数，如果个数大于 n，则为裂纹点，继续保留在 S 集合内；否则不是裂纹点，把它从 S 集合内剔除。

经过多次测试，算法的最终结果图如图 2.9。将其和比较图 2.7Bc 比较，可以看到效果明显。可见，改进的区域增长方法可以有效去除非裂纹区的杂质点。

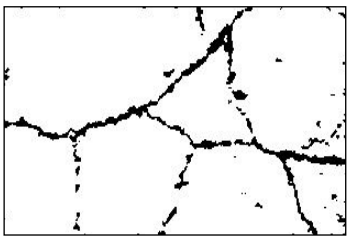


图 2.9 区域增长效果图(w=3,n=7)

二、裂纹区除杂

通过非裂纹区域的除杂已经将大部分杂质点去除，但是在裂纹区域存在着这样的杂质点：它们存在于裂纹区域内，但是由于特征向量的方向和周围的像素点方向不一致，因此也被归类为杂质点[22]。根据 Hessian 矩阵特征向量的方向在裂纹处是相同的[23]，因此分类方法设计如下：

利用 tan 函数在$[-0.5\pi,0.5\pi]$单调递增的属性，求最小特征值对应的特征向量(nx,ny)，算出每个 ny/nx 的值，然后定义两个边界——上限 max 和下限 min，通过比较就可以知道方向处于什么区间内。而这个上下限 max，min 就是要归类的依据。经过测试，max=3，min=-2。图 2.10 是最后的裂纹图像。可以看到一幅较好的裂纹图像被提取。

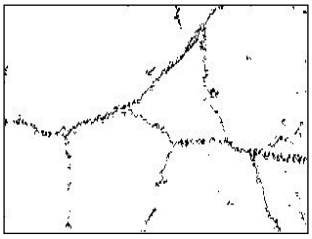


图 2.10 裂纹图像

三、裂纹的分类

第一章第一节中提到裂纹可以分成四类：横向裂缝、纵向裂缝、龟裂裂缝、块状裂缝四类。现实中，由于块状裂纹和龟裂裂缝有相似之处，所以被划归为龟裂裂纹。在图像上，横向裂缝、纵向裂缝、龟裂裂缝有如下特征：

横向裂纹每一行上的目标像素点个数差异大，而每一列上的像素点个数趋向致；纵向裂

纹恰好相反；而龟裂裂纹横纵方向的像素点个数有一定差异，但不大。据此，可以通过如下算法来分类：

（1）统计每一行每一列有多少个目标像素点。

（2）由于只统计一行可能出现极大或者极小值而影响最终结果，所以为了减少误差，可以统计连续的多行和多列。对应的操作就是图像的缩放。连续的行数过少则起不到减少误差的作用；过多则使各个数据之间的差异太小。在本文中，统计了连续的三行和三列。也就是将图像缩小成原来的三分之一。

（3）分别求出每列的最大最小值，记为 maxX，maxY 和每行的最大最小值 minX，minY。

（4）利用公式(2.21)判断裂纹类型：

$$\begin{aligned} retX &= \max X - \min X; \\ retY &= \max Y - \min Y; \\ dValue &= retX - retY; \end{aligned} \tag{2.21}$$

对于横向裂纹而言，retX 趋向于 0，retY 大于 0，所以 dValue 会小于 0；纵向裂纹相反；龟裂裂纹的 dValue 会趋向于 0。所以设定一个阈值就可以进行判定。本文取 70。对于本文的图像，rectX=114，rectY=136，dValue=-22。可见，是一个龟裂裂纹。

为了更直接地证明裂纹类型，需要介绍 Hough 检测法。对于一条直线，在笛卡尔坐标系中，常见的表现形式有点斜式，两点式两种。然而在 Hough 变换中，使用（r,theta）来表示一条直线。其中 r 为该直线到原点的距离，theta 为该直线的垂线与 x 轴的夹角。

使用 Hough 变换检测直线的思想是：假设每一个点都有 n 个方向的直线，通常 n=180，此时检测的直线的角度精度为 1°，分别计算这 n 条直线的（r,theta）坐标，得到 n 个坐标点。假设共有 N 个点，最终得到 N*n 个坐标。如果多个点在一条直线上，那么必有这多个点在（r,theta）附近。从数值上看是极大值，从图像上来看就是亮斑。如图 3.4。

从图 2.12 可见 Hough 图不止一个亮斑，说明该裂纹图像不止一条直线，显然不是横纵裂纹，是龟裂裂纹。图 2.11 的直线检测图也能证明。

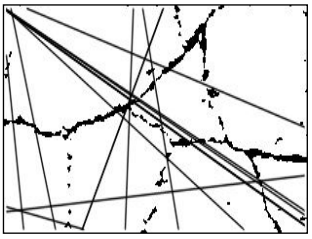


图 2.11 直线检测图

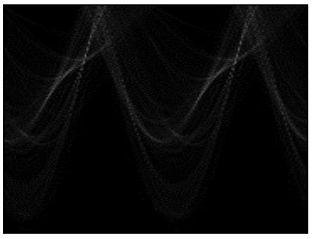


图 2.12 Hough 图

第四节　本章小结

受拍照过程的影响，路面图像会受到各种噪声的影响，从而使图像质量下降，无法有效

地进行裂纹的检测与识别。针对图像中的噪声问题，本章第一节使用了数字图像处理中常见的中值滤波、均值滤波、高斯滤波进行去噪处理。在实验前，先对三种方法进行理论上的推演和预测，也对实验效果进行了初步判断。经过实验发现均值，中值滤波确实有其缺陷，而高斯滤波取得较为理想的滤波效果。因此，本文采用高斯滤波作为理想的滤波函数。

第二节利用 Hessian 矩阵对已经滤波过的图像进行处理。首先介绍了 Hessian 矩阵的原理以及在几何图像中的运用，然后介绍了四个版本的 Hessian 矩阵的计算方法和三个用来选取最终裂纹点的特征函数。

第一个版本的 Hessian 矩阵计算方法是通过观察法总结出来的，通过观察法可知裂纹横截面呈 V 型，因此一阶二阶导数可以通过相邻两数相减得到，它的优点就是直观简单，缺点是数据粗糙，不精确。经典版是通过图像与一定尺度的高斯滤波器相应的二阶导数的卷积求得，通常采用多尺度方法，该算法的优点就是能较好的取得所需要的效果，缺点就是计算量大，因为其中有大量的乘法运算。库函数简化版是基于经典版改变过来的，也是本章最终采取的方法。它没有从根本上改变整个计算过程，因此可以保证得到和经典版同样的效果。它的简化主要在于将原本由计算机做的大量的计算替换成人工简化，通过人工化简的方式省去了调用库函数这一步骤，因此大大简化了计算量。模糊版是在库函数化简版的基础上做了进一步的简化，将原本的数据用近似值代替，破坏了数据的真实性，因此效果不好。

最后对比分析了上述实验结果。实验证明，Hessian 矩阵的计算确实可以简化，并且和经典版的计算方法取得同样的效果。

第三节主要是对两类杂质的处理——非裂纹区域和裂纹区域的杂质点。先介绍非裂纹区域的除杂方法，包括形态学方法和区域增长方法，然后介绍了裂纹区域的除杂方法。实验证明区域增长取得了好的效果。最后通过分类算法对裂纹进行了正确的分类。

两个非裂纹区域的除杂算法各有特点。开运算用来消除孤立的杂质点，虽然裂纹点分散，但是相对于裂纹本身而言不能算“细小”，因此在本课题中不适用。区域增长方法的运用主要是抓住了此时裂纹点比较分散的特点才能获得较好的效果。

最后，对于先去除哪一类杂质的问题，本文也做过测试，发现采取先非裂纹区域再裂纹区域的方法更好。因为区域增长虽会去除很多待判定点，但是由于裂纹的骨架里的点稠密，只要控制好 w 和 n 就不会对裂纹骨架产生什么影响。如果倒过来，由于归类会使裂纹骨架产生空洞，再使用区域增长方法的时候可能会误判骨架内的点。

第三章　实验结果与分析

第二章完整地介绍了本文提出的 Hessian 矩阵检测裂纹的流程。为了测试本文算法检测效果，另取一张图像做测试。图 3.1 是原图。高斯滤波后的图像效果如图 3.2。

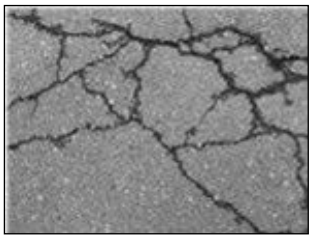


图 3.1 原图像

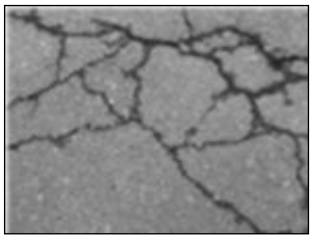


图 3.2 高斯滤波图像

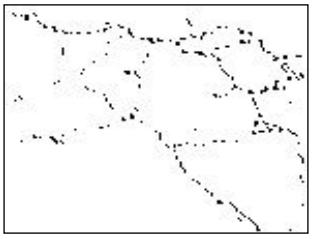
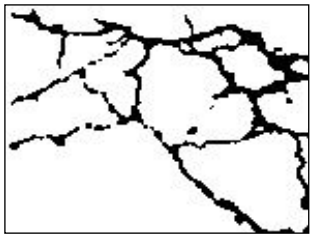


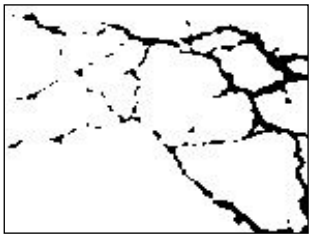
图 3.3 观察法

观察法 Hessian 矩阵如图 3.3。如第二章第四节所分析的那样，一方面计算简单，另一方面数据粗糙，效果差，无法满足检测要求。



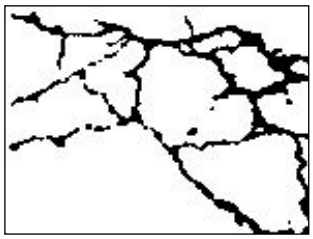
A c

A d

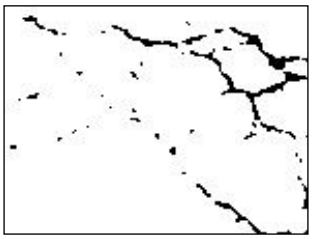


A e

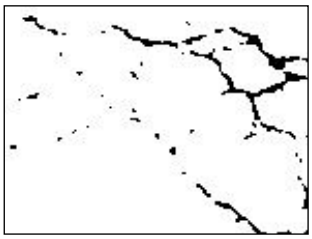
图 3.4 经典版效果图



B c



B d



B e

图 3.5 库函数化简板效果图

经典版记为 A，库函数简化版记为 B，三个特征函数记为 c，d，e。那么 Hessian 矩阵经典版的效果图就可以如图 3.4 标注，库函数简化版版的效果图如图 3.5 标注。

先比较图 3.4 的三幅图可知，Hessian 矩阵可以有效的检测裂纹。其中特征函数的选取也

尤为关键，选择第一个特征函数可以让效果达到最佳。

然后比较图 3.5，将 Bc 和 Bd，Be 比较可知，第一个特征函数也同样可以让简化版取得理想效果。当然，两次使用的阈值 T 是不同的。

比较 Ac，Bc，得出的结论就如第二章第四节的结论所述，库函数简化版之所以取得和经典版同样的效果，主要是因为通过人工的方式取代了库函数的频繁调用，因此大大简化了计算量，提高了效率。在这次测试中，从视觉上来说两幅图像几乎完全一样，最可能的原因就是原图像整体像素值偏低，在噪声比较少的情况下完全可以将噪声忽略不计。所以可以观察到 Ac，Bc 除了裂纹的整体骨架以外，在非裂纹区几乎看不到杂质点，这就是噪声比较少的证据。

模糊版的效果图如图 3.6。把原本复杂的卷积运算分成简单的加减运算，速度大大提升，有个整体的轮廓，但是效果不好，和 Bc 差距大。

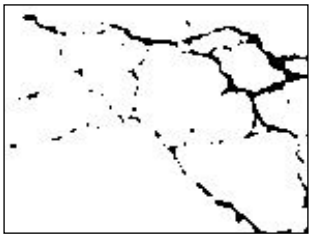


图 3.6 再简化版的效果图

Hessian 矩阵处理完之后就是非裂纹区和裂纹区的除杂。非裂纹区的除杂主要是针对那些离散的像素点，但观察 Bc 发现非裂纹区的杂质点几乎没有，所以，无论采取哪种方法，结果都是一样的，并且和 Bc 的差距也不会大。形态学处理和区域增长方法的效果图如图 3.7,3.8。可以看到，对于本张图像，区域增长的优势不大。

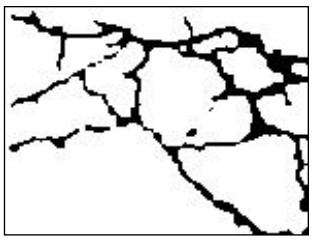


图 3.7 形态学处理

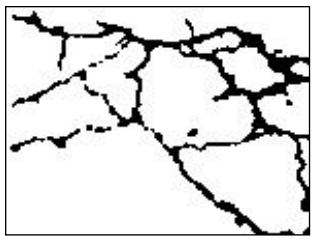


图 3.8 区域增长法

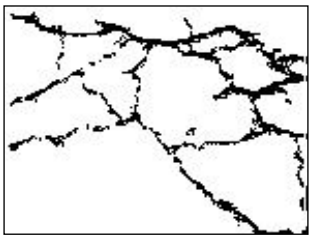


图 3.9 裂纹区的除杂

接着是裂纹区的除杂，主要是指方向上的归类。在这里 max=3，min=-2。这里的参数和之前的参数一模一样，可能的原因是计算 Hessian 矩阵时涉及到指数，数据很大，但是又被作为分母，使得整个的数据非常小，大约只有一万分之一到十万分之一之间。此时再乘上一

个 0 到 255 的像素值对最终结果也没什么影响。然后再经过特征向量的计算使得两次测试用的图像最终算出的结果趋向于一致。所以参数没变。本次除杂效果图如图 4.8。

最后就是裂纹的分类。阈值不变，还是取 70。本次测试 retX=101，retY=133，dValue=-32。证明是龟裂裂纹。

图 3.11 显示不止一个亮斑，说明不是横纵裂纹，是龟裂裂纹。图 3.10 的直线检测图也可以看到确实是一个龟裂裂纹。

图 3.10 直线检测图

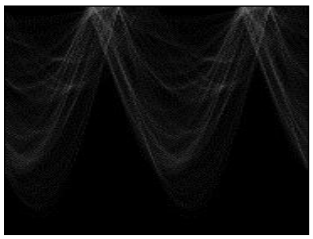
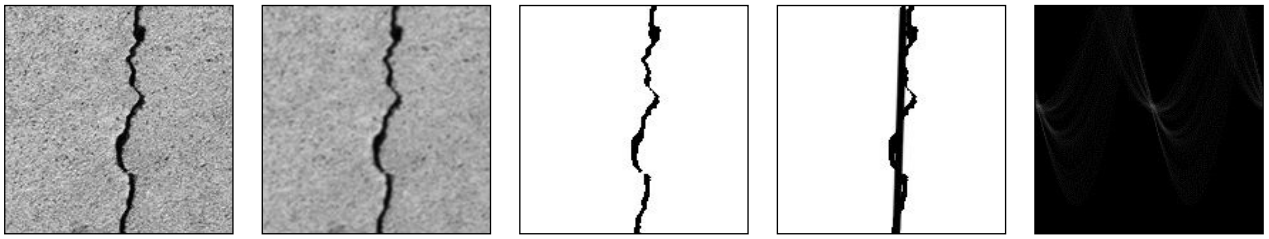


图 3.11 Hough 图

接下来再用一张简单点的图像来检测。图 3.12 从左到右分别是(a)原图，(b)高斯滤波图，(c)Hessian 矩阵检测图，(d)直线检测图和(e)Hough 图。对于此次检测，rectX=153，rectY=21，dValue=132，是一张标准的纵向裂缝。



(a)原图　(b)高斯滤波　(c)Hessian 矩阵检测图　(d)直线检测图　(e)Hough 图

图 3.12

从 Hough 图可以看出有一个很明显的亮斑，说明裂纹图像中存在一条可以检测出来的直线。参考直线检测图可以证明之前的判断——这是一张标准的纵向裂缝。为了让直线检测图便于观察，已经通过修改阈值的方式将其他直线略去并且加粗了检测出来的直线。

第四章　结论

随着公路使用时间的增加，路面破损程度也日渐严重，因此路面裂纹的检测也就成了解决此类问题的关键。论文利用 Hessian 矩阵提取裂纹，由于 Hessian 矩阵的计算过于复杂，所以裂纹检测的难点就在于如何简化它的计算，同时取得相同的效果。文章中通过实验比较了若干种简化版的 Hessian 矩阵计算方法，并结合区域增长的处理方法较好的提取出了路面裂纹，为后续工作奠定了良好的基础。

一、文章算法总结

文章中尝试了若干种 Hessian 矩阵的计算方法进行裂纹检测。在这些过程当中，通过测试选择了合适的特征函数和除杂操作，最终很好的检测出了裂纹。

路面图像会由于光照，镜头移动等因素影响而产生高斯噪声，通过多种滤波方法的测试和对比，最终采用高斯滤波。

本文的最终目的是要提取图像中的裂纹，所以 Hessian 矩阵的计算尤为重要。但是传统的计算方法效果好但是复杂度大，因此需要化简。通过查阅资料和观察法，发现了两种简化方法——库函数化简版和模糊版。模糊版由于用加减法替换了乘法导致了数据的很不精确，因此实验效果差。而库函数简化版版是将大量由计算机处理的重复计算用手工化简然后代入，对于每一个像素点而言就是减少了一次库函数的调用，对于整幅图像而言就是节约了时间，提高了效率。但是就整个过程而言并没有数据上的出入，因此同样可以获得好的效果。

最后是除杂和分类操作。对于经过 Hessian 矩阵处理的图像中存在的伪裂纹点，经过图像的开闭运算以及归类提取方法的处理，基本上消除了伪裂纹点的影响，较好地提取出裂纹，达到了预期的效果。分类的结果也和预期相符合。

二、算法分析

本文算法很好的实现了裂纹提取的目的，但是也有弊端。第一：观察图 3.2 发现左上角的那部分颜色较浅的裂纹未被成功检测出来，而右边颜色较浅的裂纹却检测出来了。问题还是出在了除杂操作上。区域增长操作一方面去除了杂质点，但另一方面也把细小的裂纹当杂质去除。究其原因还是区域增长算法本身很依赖于种子点的选取以及增长方法的选择。本文中增长方法虽然基于一定的属性和特征，但是比较简单，无法适应复杂多变的裂纹。第二：算

法中多次用到阈值，需要阈值说明无法自动化，只能人工操作，自适应性差。

三、后续工作的建议

论文的主要目的是通过 Hessian 矩阵来进行裂纹的检测，但是仅靠 Hessian 矩阵难以做出很好的判断，比如算法分析中提到的，有些检测出来而有些没有。其实在区域增长方法使用前，可以对此时的裂纹杂质点的特征，例如几何特征，进行分析，然后可以有针对性的进行处理。因此，在后续工作中，可以考虑其他方法，比如前言中提到的形态学方法，或者寻找一种更好的增长方法。将这些方法结合考虑或许能进一步完善裂纹的提取效果。同时，如何减少阈值的使用来提高自适应性也是未来需要解决的问题。

致　谢

历时将近一学期的时间终于将这篇论文写完，在论文的写作过程中遇到了无数的困难和障碍，都在同学和老师的帮助下度过了。尤其要感谢我的论文指导老师—秦红星老师，他对我进行了无私的指导和帮助，指导本人进行论文的修改和改进。另外，还要感谢研究生学长学姐，感谢诸位在我论文完成的过程中的无私帮助。在此向帮助和指导过我的各位老师和同学表示最衷心的感谢！

感谢这篇论文所涉及到的各位学者。本文引用了数位学者的研究文献，如果没有各位学者的研究成果的帮助和启发，我将很难完成本篇论文的写作。

由于我的学术水平有限，所写论文难免有不足之处，恳请各位老师和学友批评和指正！

参考文献

[1] 宋蓓蓓，韦娜.FCM 分割和形态学的沥青路面图像裂缝提取[J].计算机工程与应用.2013（04）

[2] 徐威，唐振民，吕建勇.基于图像显著性的路面裂缝检测[J].中国图象图形学报.2013(01)

[3] 蔡光明.高速铁路桥梁底面裂缝的视频检测系统研究[D].北京交通大学:机械制造及其自动化,2011

[4] Henrique Oliveira. Automatic Road Crack Detection and Characterization [J]. IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.2013(14)

[5] 周琳.基于 Hessian 矩阵的视网膜血管中心线提取[D].南京航空航天大学;机电学院. 2011

[6] 张钦.冠脉造影图像血管树提取方法研究[J].中国制造业信息化.2011(01)

[7] 程骏.基于机器视觉的耐火砖裂纹检测[J].微计算机信息.2010(35)

[8] 李文涛, 范慧杰. 基于 Hessian 矩阵的高压线检测算法[J]. 仪器仪表学报, 2012, 33(8)

[9] 吴青, 李金宗. 高分辨率航空图像的道路中心线提取算法[D]. 哈尔滨:计算机工程与运用, 2006

[10] 刘德超.基于图像处理的路面裂纹识别研究[D].长沙理工大学:交通信息工程及控制. 2009

[11] 孟乔. 沥青路面裂纹检测算法研究[D]. 长安大学:计算机应用技术, 2010

[12] Derek Bradley. Adaptive Thresholding Using the IntegralImage[D]. Carleton University: Carleton University. 2007

[13] 孙晓明. 基于结构光的公路路面裂缝检测关键技术研究[D]. 哈尔滨工业大学;仪器科学与技术.2012

[14] 陈丽平. 基于 Hessian 矩阵的血管图像增强与水平集分割算法研究[D]. 湖南大学:生物医学工程, 2012

[15] 康长青, 袁磊. 一种新的血管造影图像 Hessian 矩阵增强算法[J]. 计算机工程与科学, 2010, 34(10)

[16] 韩冰, 王永明, 孙继银. 加速的 Fast Hessian 多尺度斑点特征检测[J]. 光学精密工程, 2011, 19(7)

[17] 曹容菲, 王醒策等. 基于 Hessian 矩阵特征值聚类的脑血管分割方法[J]. 系统仿真学报, 2014

[18] Yang, SF, Cheng, CH. Fast computation of Hessian-based enhancement filters for medical images[J]. Computer Methods and Programs in Biomedicine, 2014, 116(3)

[19] Y. Sun, E. Salari, and E. Chou, “Automated pavement distress detection using advanced image

processing techniques,” in Proc. IEEE Int. Conf. Electro/Inf. Technol., Windsor, ON, Canada, 2009

[20] 刘晏丽.基于 Hessian 矩阵和区域生长的肝血管树的分割算法研究[J].计算机与现代化.2011(01)

[21] 胡正平.区域进化自适应高精度区域增长图像分割算法[J].系统工程与电子技术.2007(06)

[22] 刘佳.冠状动脉造影图像中血管中心线跟踪算法的研究[D];东北大学;2008

[23] P. Orlowski, M. Orkisz, Efficient computation of Hessian-based enhancement filters for tubular structures in 3D images, IRBM 30 (2009)

附 录

一．英文原文：

I. INTRODUCTION

Cracks are the most common road pavement surface defects, and capturing imagery of pavement surface during road surveys, for crack detection and characterization, is considered an adequate procedure for the collection of data about the condition of the pavement surface [1], [2]. Automatic crack detection and characterization systems are being developed for fast and reliable pavement surface defect analysis, instead of relying solely in the slower and subjective traditional human inspection procedures, also contributing to the development of a safer survey methodology, particularly when monitoring high-speed roads like highways [3]. These procedures can be combined with the usage of intelligent transportation systems,which can detect and identify vehicle types (e.g., heavy vehicles like tractor trailers) [4], together with the monitoring of traffic intensity [5], for instance, to help identify roads where more frequent pavement status surveys are required.

Automatic procedures relying on neural networks, Markov random fields, edge detectors, or morphological operators have been used to characterize road pavement surface distresses like cracking [6], as discussed in Section II. The fully integrated system for crack detection and characterization proposed here, i.e., crack detection tool of Institutode Telecomunicaçoes (IT) (CrackIT), requires no manually provided labeled examples for the training phase, automatically selecting training images, and learning the statistics about crack and noncrack image blocks. Only basic prior radiometric knowledge about crack characteristics is necessary (image regions darker than their surroundings). In fact, image samples for training the system are automatically selected, and the crack blocks they contain are preliminary labeled using a simple system that may miss some portions of the existing cracks but still is able to collect enough statistical information for the detection stage.

An unsupervised two-step pattern recognition system is presented. First, the detection of image blocks containing crack pixels is addressed. Then, cracks are characterized into types as specified by the Portuguese Distress Catalog [7], and CrackIT is able to classify the different crack types present in a given image, as well as to estimate the average width of each detected crack for a crack severity level assignment. The 2-D feature space used for crack detection considers two simple but effective local statistics: mean and standard deviation of gray level values inside nonoverlapping image blocks; then, six clustering techniques are confronted to infer about their crack detection

suitability. Crack type characterization exploits the spatial distribution of image blocks composing the detected cracks’ connected components using another 2-D feature space. Finally, a novel crack severity assignment procedure is applied, automatically calculating the average width of each crack.

The proposed crack detection strategy follows a learning from samples paradigm, with unlabeled data being used for training the CrackIT system. The crack type and severity characterization steps require no training. Performance evaluation is conducted using pavement surface images captured during a visual survey along a Portuguese road, a subset of which has been manually labeled by human experts, which will be taken as ground truth, for comparison purposes.

This paper is structured as follows. After this brief introduction to the crack detection and characterization problem, Section II presents an overview of the relevant literature.Section III describes the proposed crack detection approach, notably in terms of parameter estimation and the classifiers used. Section IV presents the proposed crack type characterization and severity level assignment strategies. Experimental results are discussed in Section V, and Section VI draws conclusions and provides hints for future work.

II. LITERATURE REVIEW

The number of recently published papers dealing with crack detection and characterization of pavement surface distresses shows an increasing interest in this area. A recent publication [8] proposes a methodology to detect cracks using a multiscale approach based on Markov random fields to segment fine structures (cracks) in road pavement surface images. Cracks are enhanced using a 1-D Gaussian smoothing filter and then processed by a 2-D matched filter to detect them. A total of 64 road pavement surface images representing several crack types are considered for experimentation, producing a qualitative evaluation. Details on image characteristics or the type of sensor used to capture them are not provided.

In [9], crack detection starts with a nonlinear filter to enhance the contrast between distresses and the image background. Then, segmentation by thresholding and finally a pixel-based connectivity analysis is performed to identify cracks. The actual crack detection performance is difficult to evaluate as no objective evaluation metrics are used, and a global analysis of the image database is not included.

A fractional differential and wavelet transform is presented in [10], being qualitatively compared with other edge detection operators like Sobel, Prewitt, and Logarithm of Gaussian(LoG), to show its superior performance. A Sobel edge detector after image smoothing with a bidimensional empirical mode decomposition is proposed in [11] to detect cracks. The reported results use 15 images, but no details about image spatial resolution are provided, and the results make evident some of the challenges faced by these types of systems, with false crack region edges

appearing due to the presence of white lane markings, and irregularities (more difficult to handle) always present in this type of pavement surface due to the appearance of coarse surface aggregates caused by the loss of fine aggregates (raveling). Another author [12] evidences the difficulty of detecting cracks of less than 3 mm width when using edge detectors. A nonsubsampled contourlet transform is adopted in [13] to detect cracks, wherein a limited set of experimental results is presented.

Other approaches, based on segmentation by thresholding to automatically detect cracks, are proposed in [14] and [15], where results using the Otsu threshold selection method [16] and the entropic method of Kapur [17] are compared with the neighboring difference histogram method proposed in [14]. The authors of [15] claim that their segment extending approach achieves better results because they are capable of dealing with noisy images and detecting unclear and disconnected cracks, but no comparative results are included, and information about the ground truth used is omitted. Another approach extracts linear features (cracks) [18] comparing two methodologies: one based on holistic thresholding and the second employing the Otsu algorithm. The second methodology proves to be more efficient in terms of both the threshold value and the required computational effort for crack detection. Again, no details about the image database are provided, and the analysis of results is qualitative and based on a single image. An improved dynamic programming (DP)-based algorithm is proposed in [19] to automatically segment pavement distresses, where the authors claim that their algorithm performs three times faster than the original DP-based method while providing the same accuracy. No severity levels are assigned for the segmented cracks.

A complete methodology to automatically detect and characterize pavement defects is proposed in [20], using grayscale images captured by line scan cameras illuminated by lasers during road surveys performed using a high-speed image acquisition system. Crack detection uses a conditional texture anisotropy measurement to each image individually, and defect characterization uses a multilayer perceptron neural network with two hidden layers. The results presented are promising, but the experimental evaluation does not support the distinction of multiple cracks in the same image.

Another neural-network-based crack imaging system to classify cracks is proposed in [21], where a proximity-based neural network, trained with a computer-generated artificial data set, following the Federal Highway Administration guidelines, comprising 60 images for each crack type (longitudinal, transversal, blocks, alligator, and no crack), performed well in comparison with image- and histogram-based neural networks. Although the reported results are good using 124 real images (for which no details or acquisition information is provided), there is no indication if the system is capable of dealing with more than one distinct crack type per image, and no severity

levels are assigned to the detected cracks.

In summary, most of the automatic crack detection strategies reported in the literature, as indicated in [22], where a classification of existing methods is also proposed, perform a pixel-based analysis of road pavement surface images, producing global image analysis results, to indicate the type of distress detected in that image. Usually, just a qualitative analysis of the results is done.

This paper proposes to overcome some of the identified limitations, performing a (fast) block-based crack detection capable of identifying the presence of multiple cracks in a given image and providing a detailed quantitative evaluation of the results obtained, highlighting which blocks in each image contain cracks, identifying the type of each detected crack, and assigning it a severity level. The obtained results are quantitatively evaluated using a set of well-known metrics and taking into account a ground truth provided by a skilled operator who manually labeled a subset of the database images, identifying all blocks in each of those images as containing crack pixels or not.

III. UNSUPERVISED CRACK DETECTION

The proposed automatic system (CrackIT) to detect and characterize cracks found in flexible road pavement surfaces follows a block-based image analysis approach, and its general architecture is illustrated in Fig. 1.

The image database considered is composed by gray-level images of fixed size (1536 × 2048 pixels) with pixel intensities ranging from 0 (black) to 255 (white). They were captured by a digital camera at sunlight with its optical axis perpendicular to the road surface and its lateral edges parallel to the road axis. Each pixel corresponds to a road area of about 1 mm2. For processing purposes, nonoverlapping 75 × 75 pixel blocks were considered, as this represents a good compromise between computational performance (faster processing time and lower memory storage requirements) and accuracy of the detection and crack characterization results [23]—notice that blocks that are too small tend to increase the number of false positive crack detections, while for blocks that are too large, the impact of small cracks tends to vanish in the computation of statistical features. Two 20 × 27 feature matrices are computed for each image: 1) the mean matrix (Mm), with each block’s average intensity, and 2) the standard deviation matrix (STDm), with the corresponding standard deviation (std) values. The blockbased analysis ignores the incomplete blocks at the right (corresponding to part of the curb) and at the bottom (which is repeated at the top of the following image) sides of the images.

In the crack detection task, a training stage is implemented, where automatically selected image samples are used for training the system, composing the training set (TrS), whereas the remaining database images are used for testing, composing the test set (TstS), to provide a quantitative measure of the crack detection quality. This strategy envisages the estimation of more reliable

parameters of the probability density functions used to model the stochastic pattern recognition system used for crack detection, allowing the system to adjust itself to the different textural characteristics found in the surfaces of different roads.

The rest of this section details the main steps of the crack detection stage according to the system architecture in Fig. 1.Crack type characterization and assignment of severity levels are discussed in Section IV.

A. Preliminary Image Labeling

The proposed algorithm starts by automatically splitting the image database into two subsets: TrS and TstS. The selection of TrS images, i.e., those images to be used for training the system, is based on the analysis of the image’ s mean matrices (Mm). The methodology used, presented in [23], considers that training images should contain relevant road pavement cracks, i.e., images allowing an easy detection of a sufficient number of “crack” blocks during the preliminary labeling stage, to ensure an adequate learning stage. Mm matrices are vertically and horizontally scanned to find blocks with relevant crack pixels, producing binary matrices (one per image), where blocks containing evident crack pixels are labeled “1,” and the remaining are labeled “0.” A sample of the results obtained after this preliminary image labeling procedure is included in Fig. 2.Equations (1) and (2) are used for the preliminary labeling of blocks (plb) along the horizontal and vertical scannings of the rows and columns of the Mm matrix, respectively, i.e.,

$$plb_H^{(i,j)} = [std(Ah^{(i,j)}) > (k_1 * std(Bh^i) + k_2 * mean(Bh^i))] \wedge [(Ah^{(i,j)}[1] - Ah^{(i,j)}[2]) > 0] \quad (1)$$

$$plb_V^{(i,j)} = [std(Av^{(i,j)}) > (k_1 * std(Bv^j) + k_2 * mean(Bv^j))] \wedge [(Av^{(i,j)}[1] - Av^{(i,j)}[2]) > 0] \quad (2)$$

where i and j are the row and column indexes, and Ah, Bh, Av, and Bv are given by

$$Ah^{(i,j)} = [\frac{M_{m(i,j-1)} + M_{m(i,j+1)}}{2}; M_{m(i,j)}]$$

$$Bh^i = [0; std(Ah^{(i,2)}); .. std(Ah^{(i,26)}); 0] \quad (3)$$

$$Av^{(i,j)} = [\frac{M_{m(i-1,j)} + M_{m(i+1,j)}}{2}; M_{m(i,j)}]$$

$$Bv^j = [0; std(Av^{(2,j)}); .. std(Av^{(19,j)}); 0] \quad (4)$$

with the values of parameters k1 and k2 for optimal operation being related to each other. This relationship can be determined from the TrS set by analysis of the F-measure (Fm) metric, which is defined as

$$Fm = \frac{2 * precision * recall}{precision + recall} \quad (5)$$

where precision is the rate between the number of crack blocks correctly detected and the total amount of blocks detected as cracks, and recall stands for the rate between the number of crack

blocks correctly detected and the number of crack blocks in the ground truth [24].

Intuitively, the “A” matrices register each block’s mean intensity as well as the average of the block’s immediate neighbor mean intensities. The local mean intensity value variation can thus be computed by std(A). Those local variations (of blocks) can then be compared against the average variation and the standard deviation observed along a line/column. All local variation values along a line/column are stored in the “B” matrices. This is the basis for identifying the prominent crack blocks in the preliminary labeling stage.

Fig. 3 shows the Fm results obtained after 2009 simulations of the preliminary labeling using all database images for which a ground truth has been manually provided by a skilled technician, who labeled all image blocks containing crack pixels. The best Fm values for the preliminary labeling step (higher than 74%) are obtained for k1 and k2 values in the ranges [0;3] and [0;2.5], respectively [see the isoline labeled 74 in Fig. 3(b)], leading to a linear fit [see line in Fig. 3(c)] represented by

$$k_1 = -0.7671 * k_2 + 2.3636 \quad (6)$$

For instance, the results presented in Fig. 2(b) were obtained using 0.4 and 2.1 for k1 and k2, respectively. Notice that the k1 and k2 values suggested by the CrackIT application (and which may be manually adjusted by the system operator) always propose to use k1 smaller than k2 to favor recall instead of precision, as for this type of application the more important is not to miss any existing road crack (see Section V). Moreover, for different types of pavement surfaces, different values for the k1 and k2 parameters may be selected.

Both plbH and plbV (binary matrices storing the partial preliminary labeling results) are merged after removing isolated blocks in each of them separately, i.e.,

$$plb^{(i,j)} = plb_H^{(i,j)} \vee plb_V^{(i,j)} \quad (7)$$

A preliminary labeling binary matrix (plb) is thus obtained for each image, and the length of each image’s longest connect component (lcc), after applying a connected components algorithm, is computed. The preliminary labeling step ends after sorting all the images in descending order of their lcc value.

Now, the training set TrS can be automatically selected, being composed by a number of images (controlled by the system operator) taken from the top of this sorted list. Once the composition of TrS and TstS sets is known, both are further preprocessed to keep the adequate level of detail for crack detection and characterization, as described next.

B. Preprocessing

The proposed preprocessing step follows the architecture illustrated in Fig. 4. This is an extended version of the strategy adopted in [23], envisaging better crack detection and

characterization results.

Intensity Normalization and Pixel Saturation: Preprocessing starts with pixel intensity normalization, independently applied to each image, to deal with the nonuniform background illumination, which is mainly due to the type of sensor used for image capture. The goal is to obtain the same average pixel intensity for blocks preliminary labeled with “0” (considered as background) in each image, whereas the remaining (crack) blocks keep a lower average intensity due to the presence of relevant (darker) crack pixels. The preliminary labeling step needs to be done prior to this normalization as an initial location of crack blocks is crucial to prevent a substantial increase of pixel intensity values in blocks preliminary classified as containing cracks and thus to enhance the subsequent detection task performance [23].

Road pavement surface images often include bright pixels due to specular reflections on some surface materials. That increases the standard deviation of pixel intensities in blocks without cracks, eventually leading to similar std values in both crack and noncrack blocks, which is undesirable since this paper intends to explore std as a feature for crack detection.

A simple way to handle this problem consists of applying a saturation function, such that pixels with intensities higher than a certain threshold (automatically selected as the average of all pixel intensities in each image) are replaced by the threshold value. This results in slightly darker images, without losing information regarding the presence of cracks, as notably, the “std” feature becomes more discriminative [23].

Feature Extraction and Normalization: The proposed crack detection step relies on two simple but effective features: the mean and the standard deviation values of pixel-normalized intensities within an image block. The resulting 2-D feature space is illustrated in Fig. 5(a) for the input image of Fig. 2(a), where each point identifies one image block. To reduce the scattering of the computed features among images, due to different individual average intensities, which would influence negatively the classification results, also a feature space normalization procedure is proposed. Each database image undergoes three actions: 1) a clustering analysis to distinguish between target (no crack) and outlier (crack) points, 2) computation of a global centroid for the target cluster, and 3) linear fitting applied to the target points to compute a reference angle for each 2-D feature space (being the lowest angle between the line and the horizontal axis).

二. 英文翻译：

第一部分.介绍

裂纹是最常见的路面缺陷。在道路调查期间捕获路面图像用于裂纹检测和特性描述,对于路面数据采集这一过程被认为是一个恰当的步骤。为了进行快速有效的路面裂纹分析，而不

是完全依靠缓慢的主观的传统的肉眼检测规程，自动路面裂纹检测和特性描述系统被开发出来。同时它也促进了一种更安全的检测方法的诞生，尤其是当检测高速公路时。这些系统由智能交通系统（它可以检测车辆类型）和交通流量密度监视器组成，来帮助确认哪条道路需要更频繁的路面情况检测。

依赖神经网络，马尔可夫随机域，边沿检测设备，或者形态学算子的自动检测过程已经被用来显示路面瑕疵，比如第二部分讨论的裂纹。这里提出的完全集成的裂纹检测系统，也就是 CrackIT 系统，不需要人工为训练集提供标示过的图例，它可以自动选择训练集图像，然后获取裂纹和非裂纹区域的相关数据。只需要基本的裂纹特征的先验知识（图像区域比四周暗）。事实上，该系统的图例是自动选择的，通过使用一个简单的虽然可能丢失一部分存在的裂纹但依然可以为检测阶段搜集到足够信息的系统将它们所包含的裂纹区事先标示。

现在提供一个无人监督的两步式的模式识别系统。首先，图像区域，包括裂纹像素点被编址处理。然后，裂纹就依据葡萄牙分类目录被分类成特定类型。CrackIT 可以将一幅给定的图像区分成不不同的裂纹类型，也可以估计每个检测出来的裂纹的平均宽度。用来检测裂纹的 2D 特征空间认同两个简单但是有效的数据：在不重叠的图像区域内的灰度级的平均值和标准差。然后，六个聚类技术被面临着去推断裂纹检测的稳定性。通过使用另外的 2D 特征空间，图像块的空间分布被裂纹类型充分识别并分类。最后,裂纹严重程度的分类程序被应用,自动计算每个裂缝的平均宽度。

提出来的裂纹检测策略遵循样本范式,即未被标记的数据是用来测试 CrackIt 这个系统的。裂纹的类型和严重程度分类这些步骤不包括在这个系统里面。性能评估被执行是为了比较不同的结果。使用路面调查得来的路面图像，然后被专家人工标记好，就被认为是地面真实值。

本文结构如下。在这个裂纹检测和特征表述问题的简要介绍之后，第二部分展示了文献综述的主要内容。第三部分描述了提出来的裂纹检测方法，尤其是在参数评估和分类标准的使用方面。第四部分展示了裂缝类型特征和严重性级别分类策略。第五部分讨论实验结果。第六部分下结论，并且为未来工作提出建议。

第二部分.文献综述

近年来发表的有关路面裂纹检测的论文显示这一领域受到人们的广泛关注。最近的发表期刊文献 8 提出了一种基于马尔可夫随机域的多尺度的方法来分割裂纹图像中的完整裂纹。裂纹先被一维高斯滤波器加强，然后被一个合适的二维滤波器处理来检测他们。总共 64 幅代表了多种不同的裂纹类型的裂纹图像被用来做测试，由此产生一个定性评估。图像的细节特征或使用的用来捕获它们的传感器类型没有提供。

在文献 9 中，裂纹检测始于一个用来增强裂纹和背景的对比的一维线性滤波器。然后分割阈值，最后使用一个基于像素的连通性分析器分析裂纹。实际的裂纹检测性能很难评价因为使用了不客观评价指标，以及全球图像数据库没有包括进来。

分数微分和小波变换在文献 10 中被提出了,在定性地与其他边缘检测算法如 Soble，普瑞

维特，对数高斯相比之后，显示其优越性能。用二维经验模态分解的方法对图像进行滤波后，Sobel 边缘探测器在文献 11 被提出用来检测裂缝。报告结果使用 15 幅图像，但是没有任何有关于图像空间分辨率的细节信息提供，而且结果使得这个系统所面临的挑战更加明显，因为错误的裂纹边界出现了，原因是出现了白色的车道标记以及违法行为在这条道路上经常出现。另外一篇文献 12，证据在使用边缘探测器时最大的困难是检测不到 3 毫米宽度的裂缝。一个图像增强方法在文献 13 中被采用来检测裂纹，其中一组有限的实验结果集被展示出来。

其他的方法，基于分割阈值的自动检测裂缝，在文献 14,15 中被提出，使用了大津阈值选择方法和 Kapur 熵的方法的结果用来和邻近差分直方图的方法作比较。文献 15 的作者声称它们的分割方法获得了更好的效果因为他们的方法可以处理噪声图像，检测不清晰和不连续的裂纹，但没有比较结果，而且使用地面实况信息省略了。另一种方法通过比较两种方法提取线性特性：一种是基于整体阈值和大津算法，第二种就阈值和裂纹检测所需的计算工作而言被证明是更有效的。同样的，图像库的细节没有提供，而且结果定性分析以及基于单个图像。一种改进的动态规划(DP)的算法在文献 19 中被提出，用来自动分割路面祸患，而且作者声称它们的算法在同样的精度下比原本的动态规划(DP)的算法快三倍。裂纹的分割没有指定严重程度。

一个完整的自动检测和描述路面缺陷特征的方法在文献 20 中被提出，在利用高速图像采集系统进行调查期间使用线扫描相机拍摄到的灰度图像。裂纹检测采用一个条件结构各向异性的测量标准衡量每张图像，缺陷描述使用多层感知器神经网络和两个隐藏层。展示的结果很有前途，但是实验评价在同一幅图像中不支持多裂缝的差异。

另一个基于神经元网络裂缝成像系统对裂缝进行分类的方法在文献 21 中被提出。一个基于临近值的，以电脑生成的人工数据集为训练集的，遵守联邦高速公路管理局的指导方针的，由 60 张不同裂纹类型(纵向、横向、块、皱裂和无裂纹)的图像组成的神经网络，与基于直方图的神经网络图像相比,表现良好。尽管用了 124 张图像结果很好，但是没有迹象表明系统是否能够处理多个不同的裂纹类型。

总的来说，大多数的在文献中提到的自动路面检测策略，如何分类的方法也提出了。这些策略完成了基于像素分析的路面图像的检测，产生全局的分析结果，来显示图像中的裂纹是哪一种。

本文提出要克服一些裂纹检测的识别局限性，展示一种基于分块的，能够在一个给定的图像中识别多裂缝的存在的，提供一个详细的定量评价结果的，高亮显示在每个图像块包含裂纹的，可以识别每个检测裂缝的类型,并给它分配一个严重级别的检测策略。通过使用一组著名的指标以及把由一个熟练的操作员通过手动标记数据库的一个子集的图像来提供的地面实况考虑在内，,识别包含或者不包含裂纹的所有图像块，所得出来的结果会被定量评价。

第三部分.无人监督的裂纹检测

提出来的用来检测和表述裂纹特征的 CrackIT 系统遵循一个基于块的图像分析方法。它的

主体架构如图 1 所示。

图像数据库被认为是由固定大小的(1536×2048 像素)，像素强度从 0（黑色）到 255（白）的灰度图像。他们在阳光下被数码相机的光轴以某种角度被抓取。每个像素对应于道路面积约 1 平方毫米。为了便于处理，不重叠的 75*75 个像素块被考虑进去，因为这是一个在计算性能和检测和裂缝表征结果的准确性中的折中方案。注意块太小会增加伪裂纹点的检测,而对于块太大,小裂缝的影响往往消失在计算过程中。对于每幅图像计算两个 20*27 的特征矩阵：1 均值矩阵 Mm，代表每一块的平均强度。2 标准差矩阵 STDm。基于块的分析忽略了不完整的块在右边的和底部的图片。

在裂纹检测任务中,培训阶段（自动选择图例，组成训练集，同时余下的用来测试，组成测试集 TstS）被实现用来提供定量测量裂纹检测的质量。随机模式识别系统用于裂纹检测,允许系统本身需要调整不同的在不同道路的表面结构特征。本节的其余部分细节裂纹检测的主要步骤阶段根据图 1 的系统架构。裂缝类型特征和评价严重性级别在第四部分讨论。

A.初步图象标识

提出的算法始于将图像数据库分成两部分：TrS 和 TstS。TrS 图像集的选择是基于图像的均值矩阵 Mm 的分析的。该方法认为训练集图像必须包含相关的路面裂纹的信息，图像允许在初步标示阶段使用一种简单的，可以检测出足够的裂纹的检测策略，以此来保证充分的学习阶段。Mm 矩阵是通过垂直和水平扫描来发现与裂缝有关的像素块,以此产生二进制矩阵(每个图像一个),块包含明显的裂缝像素标记为“1”,其余标为“0”。示例图像标记过程包括在图 2 中。

方程(1)和(2)用于初步标记块(plb)沿着水平和垂直扫描 Mm 矩阵的行和列,公式如下：

$$plb_H^{(i,j)} = [std(Ah^{(i,j)}) > (k_1 * std(Bh^i) + k_2 * mean(Bh^i))] \wedge [(Ah^{(i,j)}[1] - Ah^{(i,j)}[2]) > 0] \quad (1)$$

$$plb_V^{(i,j)} = [std(Av^{(i,j)}) > (k_1 * std(Bv^j) + k_2 * mean(Bv^j))] \wedge [(Av^{(i,j)}[1] - Av^{(i,j)}[2]) > 0] \quad (2)$$

其中 i 和 j 的行和列索引,Ah,Bh,Av,和 Bv 由如下公式给出：

$$Ah^{(i,j)} = [\frac{M_{m(i,j-1)} + M_{m(i,j+1)}}{2}; M_{m(i,j)}]$$

$$Bh^i = [0; std(Ah^{(i,2)}); ... std(Ah^{(i,26)}); 0] \quad (3)$$

$$Av^{(i,j)} = [\frac{M_{m(i-1,j)} + M_{m(i+1,j)}}{2}; M_{m(i,j)}]$$

$$Bv^i = [0; std(Av^{(2,j)}); ... std(Av^{(19,j)}); 0] \quad (4)$$

参数 k_1 和 k2 是彼此相关的。这种关系通过分析设定 F-measure(Fm)指标可以确定 TrS 集。该指标定义为如下：

$$Fm = \frac{2 * precision * recall}{precision + recall} \quad (5)$$

其中 precision 是被正确检测的裂纹块和检测出的块的总量的比值,recall 代表被正确检测的裂

纹块和块的总数的真实值的比值。

直观地说,A 矩阵记录每一个块的平均强度以及块的临近点的平均强度。平均强度值的变化可以由 std(A)计算。那些变化的块可以化和观察到的标准差进行比较。所有变化的按沿行/列存储在“B”矩阵。这是在标记阶段确定初步的裂纹块的基础。

2009 年后模拟初步标签阶段使用所有的已经被一个娴熟的技术人员提供的手动标识并包含所有图像块裂缝像素数据库图像，并得到了 Fm 的结果。最好的 Fm 值(高于 74%)被获得了，并且 k1 和 k2 值的范围(0;3)和[0,2.5],从而导致如下线性拟合方程式：

$$k_1 = -0.7671 * k_2 + 2.3636 \quad (6)$$

例如,从图 2(b)可知，k_1 和 k_2 使用 0.4 和 0.4。注意,k1 和 k2 值 CrackIT 建议使用 k1 小于 k2，是为了支持 recall 而不是 precision，因为这样做不会错过任何现有道路裂纹(见第五节)。此外,对于不同类型的路面表面,k_1 和 k2 可能选择不同的值。

plb_H 和 plb_V(二进制矩阵存储部分初步标记结果)合并后可以去除孤立块，公式如下：

$$plb^{(i,j)} = plb_H^{(i,j)} \vee plb_V^{(i,j)} (7)$$

初步标签二进制矩阵从每个图像中获得每个图像的最大连通区域(lcc)。初步标记步骤结束后整理所有的图片，把他们的 lcc 值降序排列。

现在，由这个排了序的序列的前几幅图像组成的训练集可以被自动选择。一旦 TrS 集合 TstS 集被确定，就为进一步的裂缝检测和表征的预处理保证足够的细节。

B.预处理

提出的预处理步骤遵循的架构见图 4。这是一个扩展版的策略，应该会有更好的裂纹检测和表征结果。

强度归一化和像素饱和度：预处理从像素强度归一化开始处理非均匀背景照明，独立应用于每个图像，这主要是受用于图像捕获的传感器类型影响。目标是获得相同的块的平均像素强度，对于初步标有“0”(视为背景)的块，而剩下的(裂纹)块保持较低的平均强度。初步标记步骤之前需要做这个正常化因为作为初始裂纹的位置块是至关重要的，它可以防止大幅增加的像素强度值,从而提高后续检测任务性能。

路面图像通常包括一些由于镜面反射材料而产生的明亮的像素。增加非裂纹块的像素强度标准差块,最终导致类似 std 值在裂纹和非裂纹块相似,这是不可取的。因为本文打算探讨 std 作为裂纹检测的性能。

一个简单的方法来处理这个问题可以应用饱和函数,这样,像素强度高于某个阈值(自动选择每幅图像的像素强度的平均值)则被取而代之。这导致图像略暗,没有丢失信息关于裂缝的存在,值得注意的是,std 作为性能标准的功能变得不准确。

特征提取和标准化:提出的裂纹检测步骤依赖于两个简单但有效的特点:强度的平均值和标准差。产生的二维特征空间如图 5 所示,其中每个点标识一个图像块。为了减少的计算中的

误差，由于不同个体的平均强度,还提出了一种特征空间标准化过程。每个数据库图像经过三个操作:1)聚类分析来区分目标(无裂纹)和离群值(裂纹)点,2)全局目标质心的计算集群,3)线性拟合来计算每个二维特征空间的参考角度(最低线之间的角度和水平轴)。

三．源程序

Hessian 矩阵经典版代码：

```
void Extra(IplImage *src, CvMat *nx, CvMat *ny)
{
    CvMat *mat = cvCreateMat(src->height, src->width, CV_64FC1);
    cvConvertScale( (src), (mat), 1, 0 );

    CvMat *Hessian = cvCreateMat(2, 2, CV_64FC1);
    CvMat *W = cvCreateMat(2, 2, CV_64FC1);
    CvMat *U = cvCreateMat(2, 2, CV_64FC1);

    for(int y = 0; y < src->height; y++){
        for(int x = 0; x < src->width; x++){
            double V = 0;
            for(int sigma = 300; sigma <= 500; sigma += 100 )
            {
                double P = cvmGet(mat, y, x);
                double Ixx = (1.0*x*x-sigma*sigma)/(6.28*pow(1.0*sigma,
4))*P*exp(-1.0*(x*x+y*y)/2/sigma/sigma);
                double Ixy = (1.0*x*y)/(6.28*pow(1.0*sigma,
4))*P*exp(-1.0*(x*x+y*y)/2/sigma/sigma);
                double Iyy = (1.0*y*y-sigma*sigma)/(6.28*pow(1.0*sigma,
4))*P*exp(-1.0*(x*x+y*y)/2/sigma/sigma);

                cvmSet(Hessian, 0, 0, Ixx );
                cvmSet(Hessian, 0, 1, Ixy );
                cvmSet(Hessian, 1, 0, Ixy );
                cvmSet(Hessian, 1, 1, Iyy );
                cvSVD(Hessian, W, U, NULL, CV_SVD_V_T);

                double renmda = cvmGet(W, 0, 0);//max    1,1min
                //cout << renmda << "      " << cvmGet(W, 1, 1) << endl;
                if(renmda > 0.0000005)//6.28*sigma*sigma
                {
```

```
                    V = log( fabs(renmda)+1 );//后面 0.75
                    //V = fabs(renmda)-fabs( cvmGet(W, 1, 1) );//后面 0.55
                    //V = cvmGet(W, 1, 1)*cvmGet(W, 1, 1)/renmda;//后面 0.48
                    cvmSet(nx, y, x, cvmGet(U, 0, 1) );
                    cvmSet(ny, y, x, cvmGet(U, 1, 1) );
                }
            }
            //cout << V << endl;
            cvmSet(mat, y, x, V );
            //cout << x << "  " << y << endl;
        }
    }
    cvNormalize(mat, mat, 255, 0);
    for(int y = 0; y < src->height; y++){
        for(int x = 0; x < src->width; x++){
            //cout << cvmGet(mat, y, x) << endl;
            if( cvmGet(mat, y, x) > 1.5 )
                cvmSet(mat, y, x, 255);
        }
    }
    cvConvert(mat, src);
}
```