作业7

1. **解释两个像素的距离**

在图像处理中，两个像素之间的距离通常用来度量它们在空间或特征空间中的差异程度。有多种方式可以定义像素之间的距离，以下是两个常用的距离度量方法：

1-欧氏距离（Euclidean Distance）：欧氏距离是最常见的距离度量方法，也是我们通常在几何空间中使用的距离概念。对于二维图像，欧氏距离可以通过计算两个像素之间的直线距离来定义。假设有两个像素点P1(x1, y1)和P2(x2, y2)，则它们之间的欧氏距离可以表示为：

d = sqrt((x2 - x1)^2 + (y2 - y1)^2)

欧氏距离考虑了像素在空间上的位置差异，即两个像素点之间的直线距离。

2-曼哈顿距离（Manhattan Distance）：曼哈顿距离也被称为城市街区距离或L1距离。它表示两个像素点之间在网格上的距离，即通过水平和垂直方向的移动所需的步数。对于二维图像，曼哈顿距离可以表示为：

d = |x2 - x1| + |y2 - y1|

曼哈顿距离忽略了对角线方向上的移动，只考虑水平和垂直方向上的距离。

这些距离度量方法可根据具体的应用场景和需求选择。在图像处理中，我们可以根据像素的位置、颜色或其他特征来计算它们之间的距离，以便进行像素级别的分析、分类或聚类等操作。

1. **解释开和闭运算的用途.**

开运算（Opening）和闭运算（Closing）是数学形态学中常用的图像处理操作，主要用于图像的去噪、边缘保持、连通性处理等方面。

开运算：

开运算是先进行腐蚀操作，然后再进行膨胀操作。它主要用于去除图像中的小的噪点或细小的边缘部分，同时保持主要结构和边缘的形状。开运算能够平滑图像，消除细小的高频噪声，并分离比较密集的物体。在图像预处理中，开运算常用于去除噪点、孤立点或细小的连通区域，从而改善图像质量和减少后续处理的干扰。

闭运算：

闭运算是先进行膨胀操作，然后再进行腐蚀操作。它主要用于填充图像中的小孔或连接断裂的边缘，同时保持主要结构和边缘的形状。闭运算能够消除图像中的小的黑色空洞，填补图像中的断裂，并连接比较稀疏的物体。在图像处理中，闭运算常用于去除图像中的空洞、连接物体、填充缺失的区域，从而改善物体的连通性和完整性。

开运算和闭运算是一对互补的操作，可以相互配合使用。它们能够在保持主要结构和形状的同时，对图像进行去噪、边缘保持和形态处理，适用于多种图像分析和计算机视觉任务，如目标检测、图像分割、形状分析等。

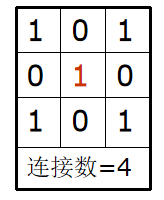
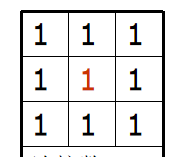
1. **计算下图像素的4-连接数**

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 1 |
| 1 | 1 | 0 |
| 1 | 1 | 0 |
| 连接数=3 | | |

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 连接数=1 | | |

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 连接数=0 | | |

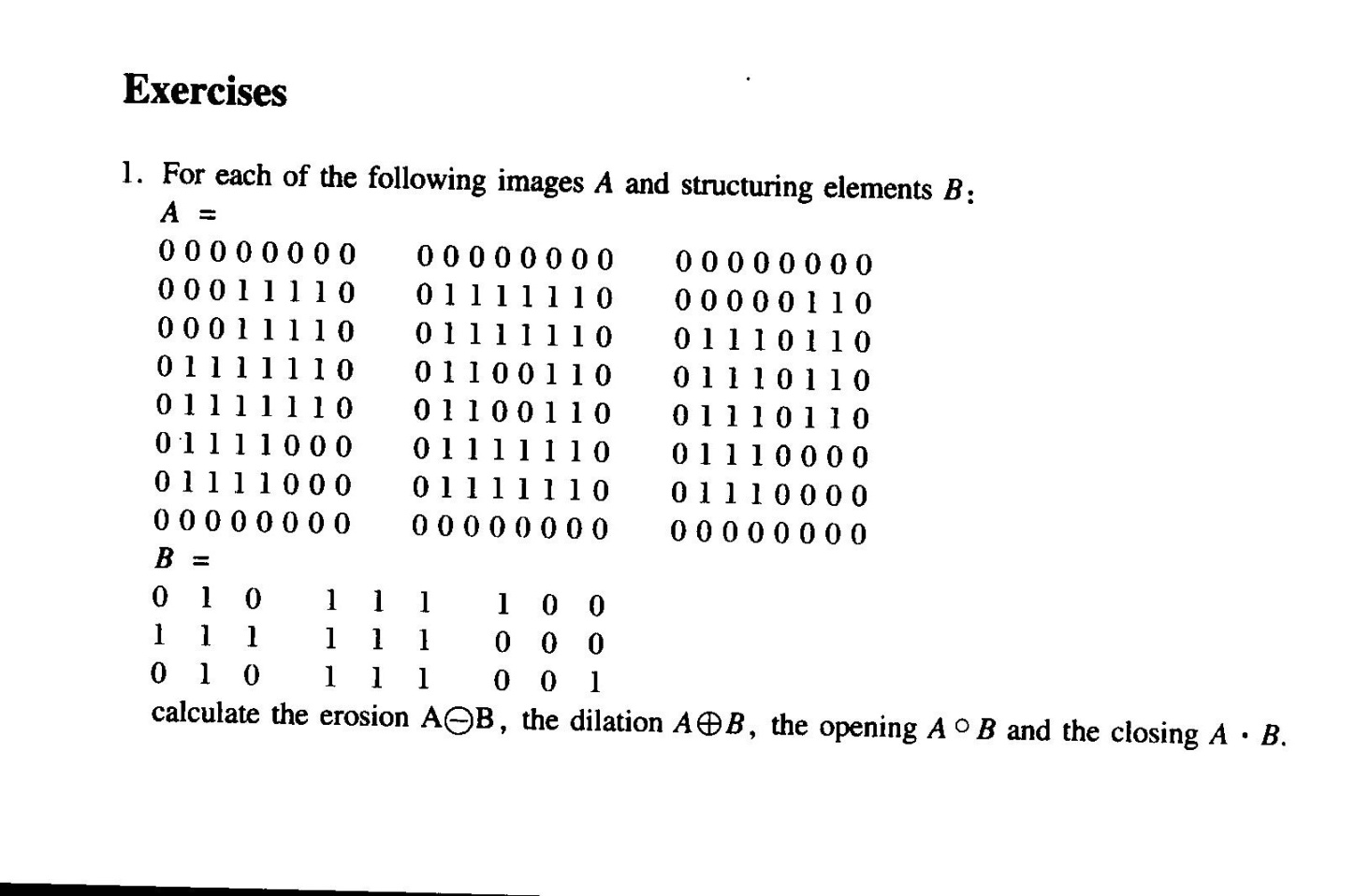
|  |  |  |
| --- | --- | --- |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 连接数=0 | | |



|  |  |  |
| --- | --- | --- |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 连接数=1 | | |

连接数=0 连接数=4

1. **进行以下的运算：**



**腐蚀运算的矩阵**A13\_E =[ 0 0 1 1 1 1 0 Inf 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 1 1 0 0 0 1 1 0 1 1 0 1 1 0 0 0 1 1 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 Inf 0 1 1 1 1 0 0]**膨胀运算的矩阵：**

A13 =[ 0 0 1 1 1 1 0 -Inf 0 0 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 0 0 -Inf 0 1 1 1 1 0 0]**开运算的矩阵：**A13\_O =[ 0 0 0 0 0 0 0 -Inf 0 0 0 1 1 1 1 0 0 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 1 1 1 0 0 0 -Inf 0 0 0 0 0 0 0]**闭运算的矩阵：**

A13\_C =[ 0 1 1 1 1 0 0 Inf 1 0 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1 1 1 1 0 0 1 Inf 0 0 1 1 1 1 0]

1. **查找资料了解“顶帽变换（top-hat）”，“黑帽变换（black-hat）”。简要说明顶帽变换，黑帽变换的基本思想及应用。**

顶帽变换（Top-Hat Transformation）和黑帽变换（Black-Hat Transformation）是数学形态学中常用的图像处理操作，用于图像的增强和特征提取。它们基于图像的开运算和闭运算操作，对图像进行亮度差异的突出或背景特征的提取。

顶帽变换（Top-Hat Transformation）：

顶帽变换的基本思想是通过对图像进行开运算，然后从原始图像中减去开运算的结果，以突出亮度较小的细节或局部特征。顶帽变换能够提取出比周围背景亮一些的图像细节，例如小的斑点、边缘、血管等。它在图像增强、细胞分析、纹理分析等领域具有广泛的应用。

黑帽变换（Black-Hat Transformation）：

黑帽变换的基本思想是通过对图像进行闭运算，然后从闭运算的结果中减去原始图像，以突出亮度较大的背景特征或目标形状。黑帽变换能够提取出比周围前景暗一些的图像细节，例如大的斑块、背景纹理等。它在图像增强、背景减除、形状分析等领域具有广泛的应用。

顶帽变换和黑帽变换可以通过以下形式表示：

顶帽变换：T = I - (I ⊖ B)

黑帽变换：T = (I ⊖ B) - I

其中，I是原始图像，B是结构元素（用于定义开运算和闭运算的形状和大小），⊖表示形态学腐蚀操作。

顶帽变换和黑帽变换对图像中的亮度变化和背景特征提取具有很好的效果，能够帮助分析和处理图像中的细节和形态。它们常用于计算机视觉、医学图像分析、工业检测等领域，以提高图像质量、增强目标特征、检测异常区域等。

1. **用C++编程实现：在8位灰度图像中求直方图，选择直方图的谷底作为阈值进行图像的二值化。**

#include <iostream>

#include <opencv2/opencv.hpp>

using namespace cv;

int main()

{

// 读取灰度图像

Mat image = imread("input\_image.jpg", IMREAD\_GRAYSCALE);

if (image.empty())

{

std::cout << "Failed to read image" << std::endl;

return -1;

}

// 计算直方图

int histSize = 256; // 直方图大小

float range[] = {0, 256};

const float\* histRange = {range};

bool uniform = true;

bool accumulate = false;

Mat hist;

calcHist(&image, 1, 0, Mat(), hist, 1, &histSize, &histRange, uniform, accumulate);

// 查找直方图谷底的阈值

int threshold = 0;

float minVal = FLT\_MAX;

for (int i = 1; i < histSize - 1; i++)

{

float prevVal = hist.at<float>(i - 1);

float currVal = hist.at<float>(i);

float nextVal = hist.at<float>(i + 1);

if (currVal < prevVal && currVal < nextVal && currVal < minVal)

{

minVal = currVal;

threshold = i;

}

}

// 图像二值化

Mat binaryImage;

threshold(image, binaryImage, threshold, 255, THRESH\_BINARY);

// 显示原始图像和二值化图像

namedWindow("Original Image", WINDOW\_AUTOSIZE);

namedWindow("Binary Image", WINDOW\_AUTOSIZE);

imshow("Original Image", image);

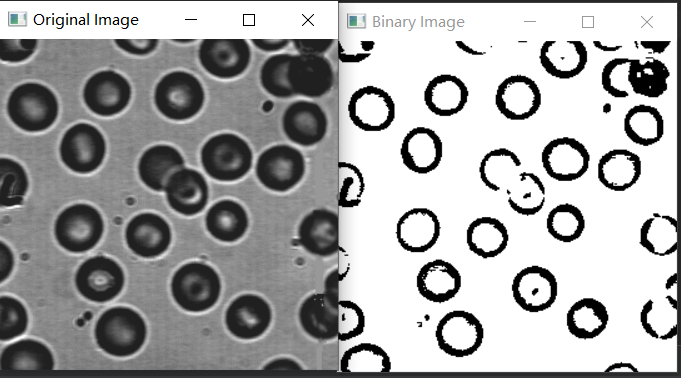
imshow("Binary Image", binaryImage);

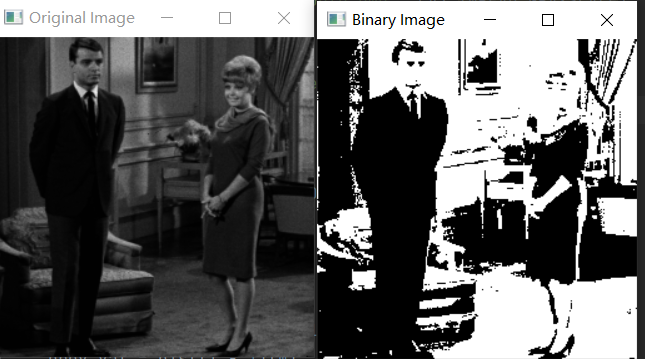
waitKey(0);

return 0;

}

实验结果：





* 讨论阈值选择是否合理

优点：

直方图谷底通常对应于图像中的背景区域或低频信息，选择谷底作为阈值可以有效地将背景与前景分离，实现二值化效果。（见第一幅图片）

直方图谷底对应于图像中的暗区域，因此可以适用于一些场景中需要提取较暗目标或背景的应用。

缺点：

直方图的谷底并不总是代表合适的阈值，特别是在图像具有复杂背景或多个目标的情况下，谷底可能并不能很好地区分前景和背景。（见第二幅图的实验结果）

选择谷底作为阈值可能会导致过度分割或欠分割的情况，使得部分目标或背景信息丢失。

谷底的选择对噪声敏感，如果图像中存在噪声或不均匀的光照条件，可能会导致阈值选择不准确。

* 扩展（自学）：查找资料研究利用直方图进行阈值计算有哪些方法。

Otsu's Method：

Otsu's方法是一种自适应阈值选择算法，旨在最大化图像类间方差。它通过对图像的直方图进行分析，自动找到最佳阈值，将图像分为前景和背景两个类别。该方法适用于图像具有双峰直方图的情况，即存在明显的前景和背景分离。

Triangle Method：

三角形法是一种基于直方图形状的阈值选择方法。它将直方图的斜率视为直方图中的峰值，并根据峰值的位置计算阈值。该方法适用于直方图具有单峰或近似单峰的情况。

Ridler-Calvard Method：

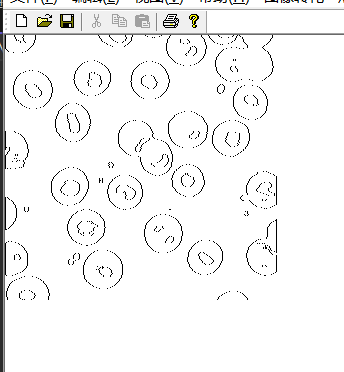
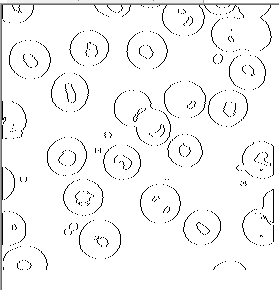
Ridler-Calvard方法是一种迭代阈值选择算法，通过迭代计算直方图的均值来确定阈值。它将图像的灰度值分为两个类别，并在每次迭代中更新阈值，直到阈值收敛。

Mean Method：

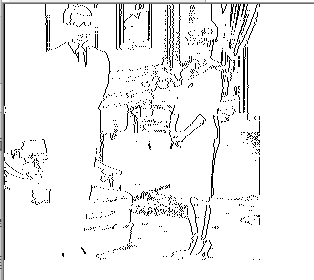
均值法是一种简单的阈值选择方法，它根据图像的平均灰度值计算全局阈值。该方法适用于图像的前景和背景具有明显的灰度差异的情况。

1. **在8位二值图像中用形态学的方法提取物体的内边界和外边界。结构元素的选择自己定义。**具体代码见实验5

实验结果：

内边界和外边界



* 用多幅图像实验，讨论边界提取是否合理。

内边界提取合理性：

内边界提取可以突出物体的轮廓并捕捉其内部细节。这对于物体分割、形状分析和目标识别等任务非常有用。

内边界提取也可能捕捉到物体内部的孔洞或空洞，这可能是不需要的，因为这些孔洞与物体的轮廓无关。

在某些情况下，内边界提取可能会导致边界的腐蚀，使得物体边缘变得不连续或模糊。这可能会影响后续处理或分析任务的结果。

外边界提取合理性：

外边界提取可以突出物体与其周围环境之间的边界，有助于物体分割和图像分析。

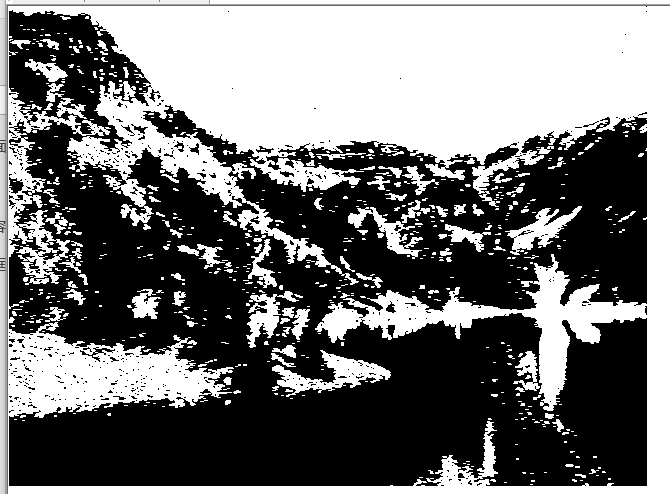
外边界提取可能包括周围环境中的噪声或其他不相关的元素，这可能会导致提取到的边界不准确或不理想。

在某些情况下，外边界提取可能会导致边界的膨胀，使得物体边缘变得粗糙或增加其厚度。这可能会影响后续处理或分析任务的结果。

1. **在8位二值图像中用形态学的方法去除噪声。结构元素的选择自己定义。**实验5中已经写过相关代码，这里只展示实验结果：



运用腐蚀运算处理得到的实验结果



运用腐蚀运算处理得到的实验结果

适用于小尺寸噪声：形态学方法通常适用于去除小尺寸的噪声，如孤立的噪声点或小区域的噪声。这包括盐噪声、椒噪声和孤立噪声点等。

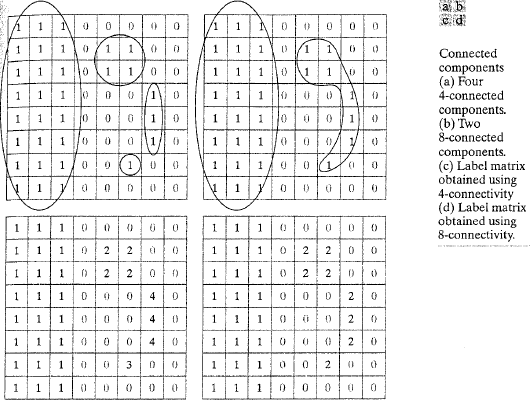
结构元素的选择：选择适当的结构元素非常重要，它应该与噪声的尺寸和形状相匹配。常见的结构元素形状包括矩形、圆形和十字形。结构元素的大小和形状选择不当可能导致噪声无法有效去除或过度去除。

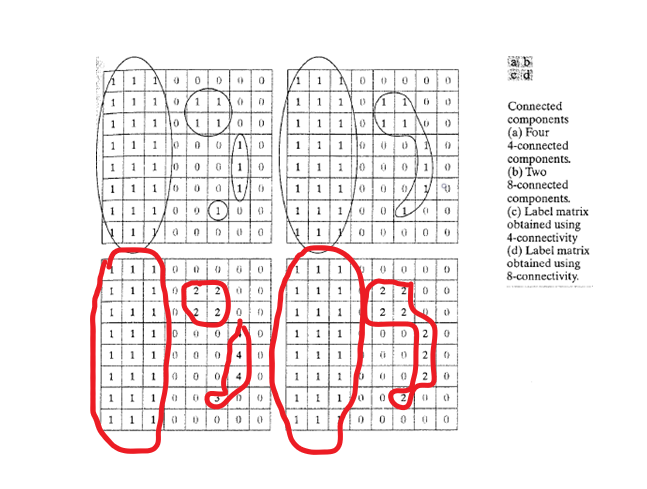
参数调整：形态学操作的参数需要根据噪声的强度和图像的特征进行调整。通常可以调整腐蚀和膨胀操作的迭代次数或使用其他操作方法，如闭操作、开操作或顶帽变换等。

不适用于连续噪声：形态学方法在处理连续噪声（如线状噪声或区域噪声）时可能效果不佳。对于这种类型的噪声，其他方法如滤波器（如中值滤波或高斯滤波）可能更合适。

去除噪声与保留细节之间的权衡：在进行噪声去除时，需要平衡去除噪声和保留图像细节之间的权衡。过度去除噪声可能导致图像细节的损失，而过度保留噪声可能导致噪声仍然可见。

1. （选做）对一个二值图像进行标注并计算图像中有几个物体。比如下图。





第一幅图片有3个物体

第二幅图片有2个物体

第三幅图片有3个物体

第四幅图片有2个物体