智能医学数字图像处理实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 | 19084129 | 班级 | 190841班 |
| 姓名 | 李奕澄 | 专业 | 生物医学工程 |
| 实验  名称 | 实验3：图像空域增强 | | |
| 实验  目的  和  内容 | 实验目的和要求：  1.掌握图像空域变换的⽅法和原理  1.掌握空域滤波中常⽤的平滑滤波⽅法和原理  2.掌握空域滤波中常⽤的锐化滤波⽅法和原理。  实验内容：  1.实现图像的算术运算和逻辑运算的代码  2.实现以下常⽤平滑滤波⽅法的代码：  （1）图像均值滤波  （2）图像中值滤波  （3）图像⾼斯滤波  3.实现以下常⽤锐化滤波⽅法的代码：  （1）roberts算⼦  （2）prewitt算⼦  （3）sobel算⼦  （4）拉普拉斯算子 | | |
| 实验  结果  与  分析 | 此次实验在进行算数运算与逻辑运算时需要保证两张图片的像素一致，否则将会报错。而且要需要注意的是cv.\*() 存在饱和机制，也就是说所有值超过255就保存为255，所有的值小于0就保存为0 | | |
| 实验  代码 | import cv2  import numpy as np  import matplotlib.pyplot as plt  #读取图片  img1 = cv2.imread('scenery.png')  img2 = cv2.imread('scenery1.png')  source1 = cv2.cvtColor(img1,cv2.COLOR\_BGR2RGB)  source2 = cv2.cvtColor(img2,cv2.COLOR\_BGR2RGB)  #算术运算  #算数加法  added = cv2.add(source1,source2)  #算数减法  subtract = cv2.subtract(img1,img2)  #算数乘法  multiply = cv2.multiply(img1,img2)  #算数除法  divide = cv2.divide(img1,img2)  #逻辑运算  #逻辑与  bitwise\_and = cv2.bitwise\_and(img1,img2)  #逻辑或  bitwise\_or = cv2.bitwise\_or(img1,img2)  #逻辑异或  bitwise\_xor = cv2.bitwise\_xor(img1,img2)  #逻辑非  bitwise\_not = cv2.bitwise\_not(img1)    #均值滤波  result1 = cv2.blur(source1, (10,10))  result2 = cv2.blur(source1, (20,20))  #高斯滤波  result3 = cv2.GaussianBlur(source1, (3,3), 0)  result4 = cv2.GaussianBlur(source1, (15,15), 0)  #中值滤波  result5 = cv2.medianBlur(source1, 3)  grayImage = cv2.cvtColor(img1, cv2.COLOR\_BGR2GRAY)  #高斯滤波  gaussianBlur = cv2.GaussianBlur(grayImage, (3,3), 0)  #阈值处理  ret, binary = cv2.threshold(gaussianBlur, 127, 255, cv2.THRESH\_BINARY)  #Roberts算子  kernelx = np.array([[-1,0],[0,1]], dtype=int)  kernely = np.array([[0,-1],[1,0]], dtype=int)  x = cv2.filter2D(binary, cv2.CV\_16S, kernelx)  y = cv2.filter2D(binary, cv2.CV\_16S, kernely)  absX = cv2.convertScaleAbs(x)  absY = cv2.convertScaleAbs(y)  Roberts = cv2.addWeighted(absX, 0.5, absY, 0.5, 0)  #Prewitt算子  kernelx = np.array([[1,1,1],[0,0,0],[-1,-1,-1]], dtype=int)  kernely = np.array([[-1,0,1],[-1,0,1],[-1,0,1]], dtype=int)  x = cv2.filter2D(binary, cv2.CV\_16S, kernelx)  y = cv2.filter2D(binary, cv2.CV\_16S, kernely)  absX = cv2.convertScaleAbs(x)  absY = cv2.convertScaleAbs(y)  Prewitt = cv2.addWeighted(absX,0.5,absY,0.5,0)  #Sobel算子  x = cv2.Sobel(binary, cv2.CV\_16S, 1, 0)  y = cv2.Sobel(binary, cv2.CV\_16S, 0, 1)  absX = cv2.convertScaleAbs(x)  absY = cv2.convertScaleAbs(y)  Sobel = cv2.addWeighted(absX, 0.5, absY, 0.5, 0)  #拉普拉斯算法  dst = cv2.Laplacian(binary, cv2.CV\_16S, ksize = 3)  Laplacian = cv2.convertScaleAbs(dst)  #显示图形  titles = ['Source1','Source2 ',  'Added','Subtract','Multiply', 'Divide',  'Bitwise\_and','Bitwise\_or','Bitwise\_xor','Bitwise\_not',  'Average filtering1', 'Average filtering2', 'Gaussian filter1',  'Gaussian filter1', 'Median filter',  'Binary', 'Roberts','Prewitt','Sobel', 'Laplacian']  images = [source1, source2,  added, subtract, multiply, divide,  bitwise\_and, bitwise\_or, bitwise\_xor, bitwise\_not,  result1, result2, result3, result4, result5,  binary, Roberts, Prewitt, Sobel, Laplacian]  for i in range(20):  plt.subplot(4,5,i+1), plt.imshow(images[i], 'gray')  plt.title(titles[i])  plt.axis('off')  plt.xticks([])  plt.yticks([])  plt.show() | | |
| 成绩  评定 | 教师签名：  年 月 日 | | |