图像视差匹配实验报告

实验目的

通过立体匹配（Stereo Matching）算法，计算两张图像之间的视差图，进而分析深度信息。

实验环境

硬件环境：个人计算机

软件环境：Python 3.8, OpenCV 4.5

开发工具：Pycharm

实验数据：实验使用一对立体图片（左图和右图），假设已经保存在当前目录下，文件名分别为 imgs.jpg 和 imgss.jpg

实验步骤

1、导入必要的库

|  |
| --- |
| import cv2  import numpy as np  import matplotlib.pyplot as plt |

2、读取图像，并转化为灰度图

|  |
| --- |
| left\_img = cv2.imread('imgs.jpg', cv2.IMREAD\_GRAYSCALE)  right\_img = cv2.imread('imgss.jpg', cv2.IMREAD\_GRAYSCALE) |

3、创建StereoBM对象，并设置相关参数

|  |
| --- |
| stereo = cv2.StereoBM\_create(numDisparities=16, blockSize=15) |

4、使用StereoBM对象计算视差图。

|  |
| --- |
| disparity = stereo.compute(left\_img, right\_img) |

5、使用Matplotlib可视化视差图。

|  |
| --- |
| plt.imshow(disparity, 'gray')  plt.colorbar()  plt.title('Disparity Map')  plt.show() |

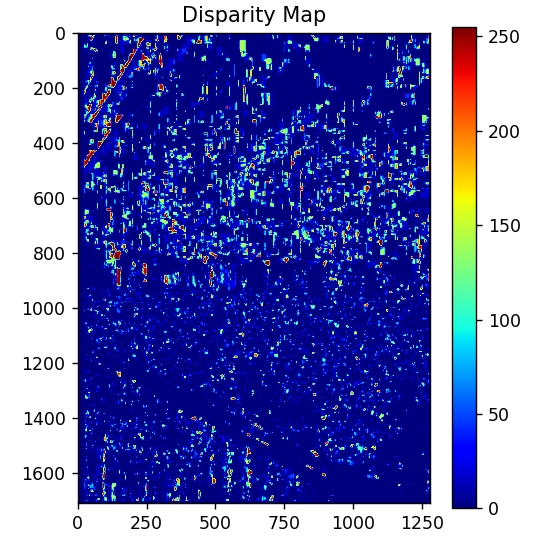
6、完整的实验代码如下：

|  |
| --- |
| import cv2  import numpy as np  import matplotlib.pyplot as plt  # 读取图像  left\_img = cv2.imread('left\_image.jpg', cv2.IMREAD\_GRAYSCALE)  right\_img = cv2.imread('right\_image.jpg', cv2.IMREAD\_GRAYSCALE)  # 创建StereoBM对象并计算视差图  stereo = cv2.StereoBM\_create(numDisparities=16, blockSize=15)  disparity = stereo.compute(left\_img, right\_img)  # 可视化视差图  plt.imshow(disparity, 'gray')  plt.colorbar()  plt.title('Disparity Map')  plt.show() |

实验结果

通过上述步骤，成功生成并可视化了视差图。视差图展示了场景中的深度信息，颜色越亮表示物体距离相机越近，颜色越暗表示物体距离相机越远。

视差图的计算结果如下：



结果分析

1、视差图概述：

视差图是通过比较左、右图像中对应像素的视差来生成的。视差值越大，表示物体离相机越近；视差值越小，表示物体离相机越远。

2、视差图特点：

在视差图中，前景物体（离相机较近）呈现为较亮的区域，背景物体（离相机较远）呈现为较暗的区域。

在理想情况下，视差图中的噪声应当较少，物体边界应当清晰。然而，由于立体匹配算法的局限性，视差图中可能会出现噪声和错误匹配。

3、影响因素：

视差图的质量受多种因素影响，包括图像分辨率、相机校正精度、立体匹配算法参数等。

在本实验中，使用了OpenCV的StereoBM算法，算法参数如numDisparities和blockSize对结果有显著影响。适当调整这些参数可以优化视差图质量。

4、优化与改进：

使用更高级的立体匹配算法，如StereoSGBM，可以获得更好的视差图。预处理图像（如直方图均衡化）和后处理视差图（如视差图过滤）也可以显著提升结果质量。

结论

通过本次实验，成功使用立体匹配算法计算了两张图像的视差图，并详细分析了视差图的结果。实验展示了视差图在深度信息提取中的应用，并指出了影响视差图质量的关键因素和优化方向。