# 2024数字图像处理实验报告

**学号：09022107**

**姓名：梁耀欣**

1. 实验题目（实验题目具体内容）
2. 基于图像形态学处理算法提取下面血管图像的中心线，并将提取的中心线以红色重 叠显示在原图中。注意:若读取到计算机中的血管图像不是二值图像，通过灰度变换算 法(阈值处理)将其转换成二值图像。



1. 实验环境（实验所使用的平台和相关软件）

Pycharm 2024.2.0.1

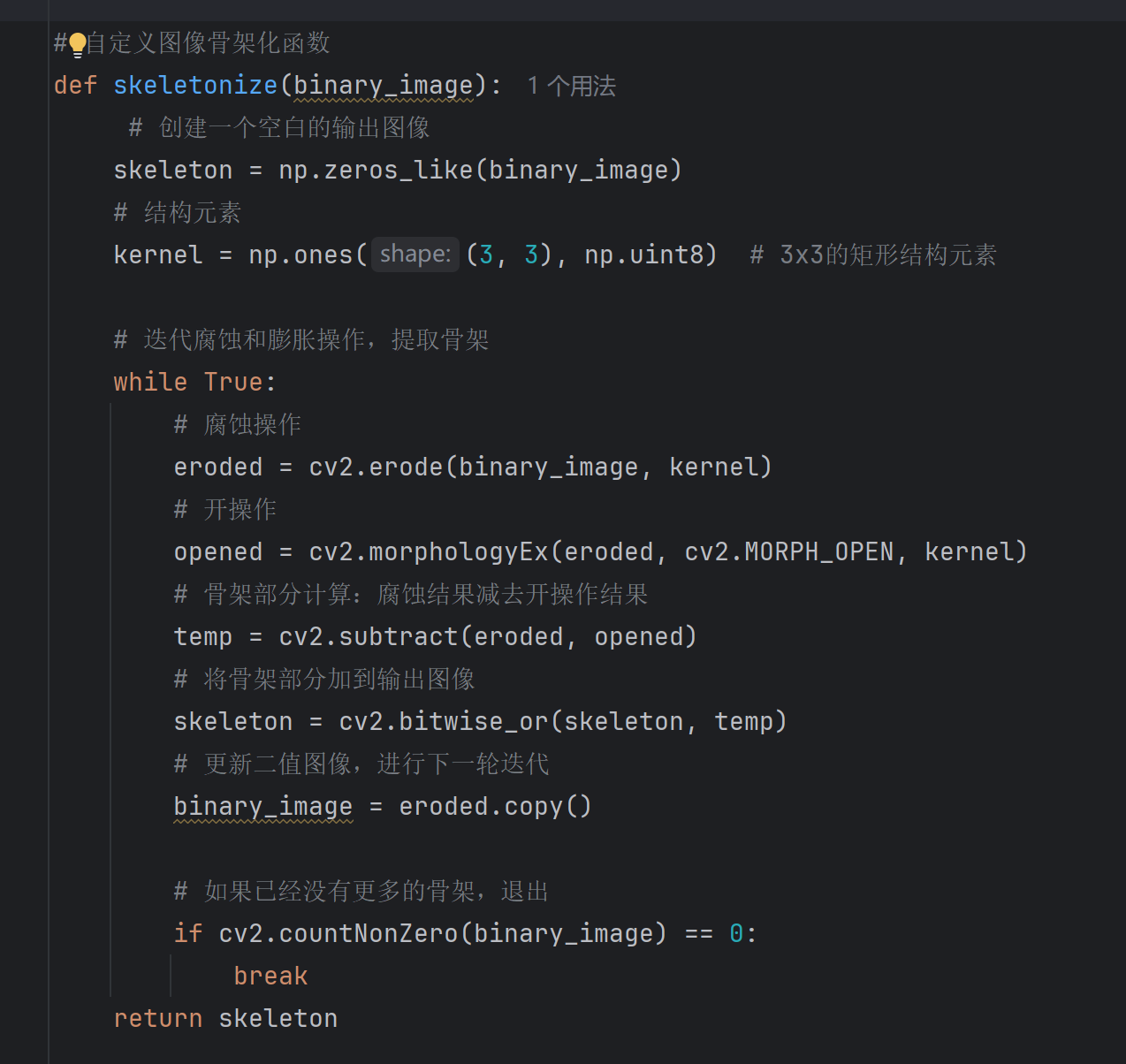
Python 3.12

1. 实验内容与方法（实验算法实现代码等）

首先为了避免图像不是标准的黑白图像，对图像进行二值化，用固定阈值127把输入的灰度图像转化为二值图像，确保目标和背景清晰区分。



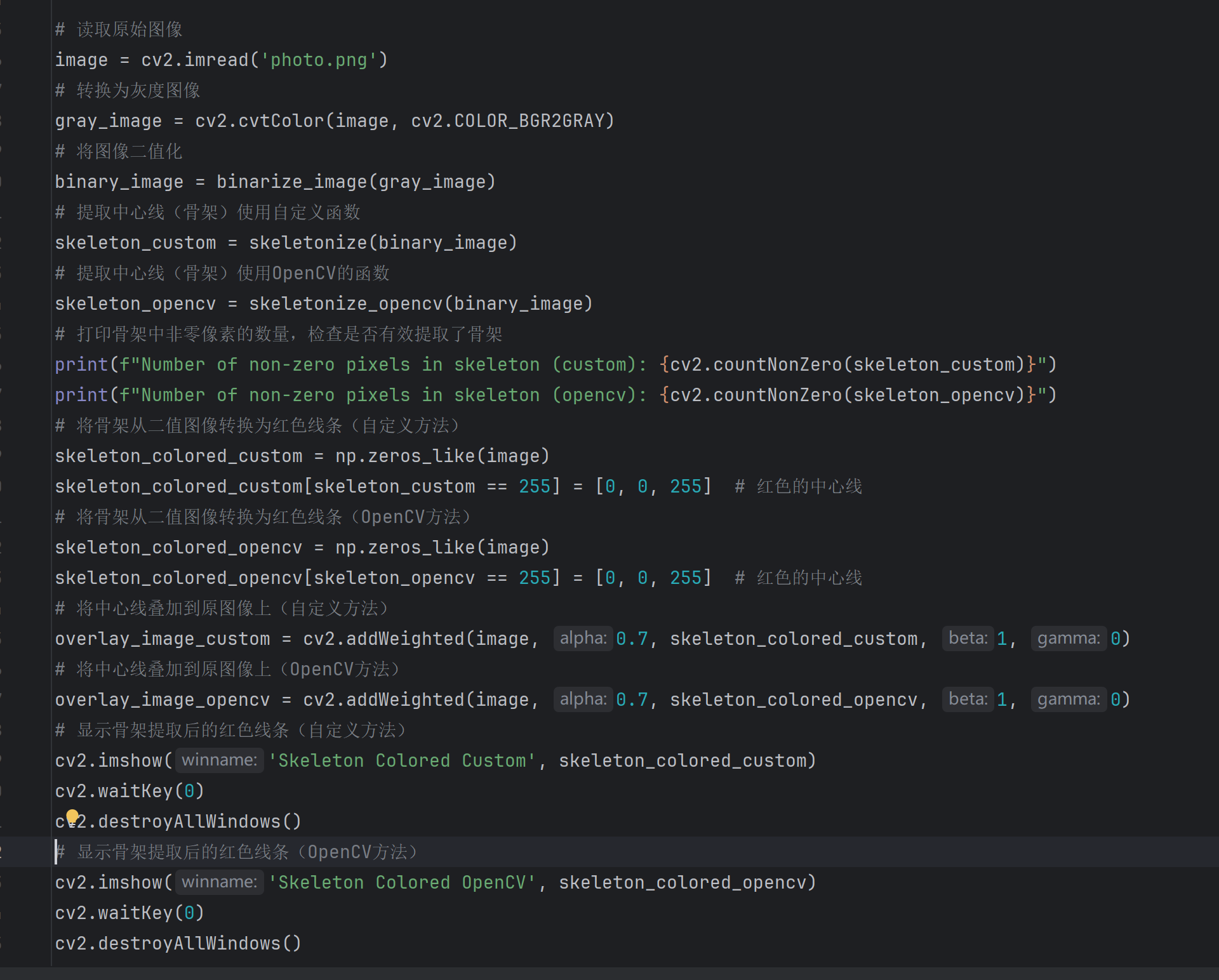
然后自定义一个骨架化函数，创建一个空图像来储存骨架再定义一个3x3的结构元素，进行迭代过程：对二值图像先腐蚀操作，然后对腐蚀之后的图像执行开操作，通过腐蚀结果减去开操作计算的结果来计算骨架子集，把骨架的子集累加到刚定义的空骨架图像，更新这个图像再进入下一轮迭代。当二值图像所有像素为0时停止迭代。



再定义一个opencv库中的骨架化方法进行对比



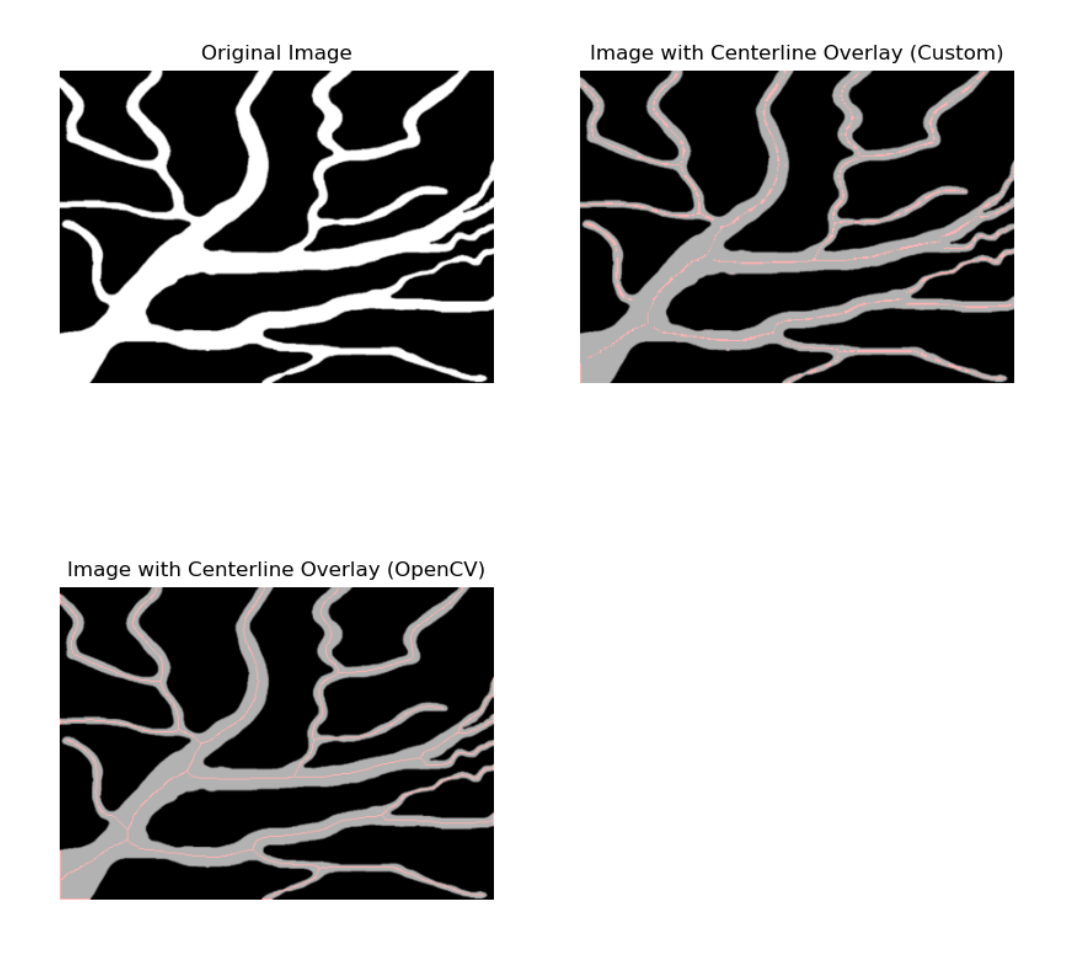
主函数：



产生的骨架图像如图：



把骨架图像的红色线条叠加到原图像上（这里一开始一直不显示红色，是因为白色再叠加依然是白色，然后设计对图像像素值进行一个加权，最终可以显示红色值了：



1. 实验结果与分析

实验结果显示自定义的方法和调库做的方法都能效果比较好的提取到图像的骨架，但是调库的速度相对更快，自定义的骨架相对更清晰.

1. 总结与思考

本实验的目的是比较自定义骨架化方法和OpenCV内置的骨架化函数在提取图像骨架时的效果。通过图像二值化后，两种方法分别提取了图像中的中心线。自定义方法采用逐步的腐蚀和膨胀操作，能够更好地去除边界噪声和细节，适合对骨架精度要求较高的场景；而OpenCV的 cv2.ximgproc.thinning 函数在效率和计算速度上表现更优，但可能在边界区域出现断点，导致骨架结构不够完整。通过比较原图与骨架线条叠加后的图像，可以直观地观察到两种方法的差异。