《数字图像处理》
冠脉中心线的提取算法
实验报告

自 61
张嘉玮
2018/12/01

目录

1. 需求分析

根据题目的提示和要求，冠脉中心线的提取算法有下面主要的环节。

1.1 对增强图像的二值化

由于题目所给的数据是 CT 图，其中的像素值反映的是该点为血管的概率，而为了提取中心线，首先需要将“概率图”转化为二值三维图像。即根据概率大小，将每一个位置的值令为 1 或者 0，其中 1 表示血管。

1.2 修复二值图中存在的空洞等问题

该步骤对二值图像进行初步的处理，主要解决其中的空洞等问题，即填补血管当中的空洞。

1.3 三维二值图像细化，提取其中的中心线

该环节是算法的核心环节，其实现的功能是将三维二值图像进行细化，得到中心线。为后面的分段和连接做准备。细化有两种途径，一种是将原始图像进行一层层的腐蚀，得到细化结果。另一种思路是提取三维二值图像的骨架。

1.4 分叉点、端点检测

受限于细化算法的精确程度，提取的中心线会出现断开、孤立分支等问题，因此需要处理这些状况。这时，需要提取细化图像得到的分叉点和端点，作为后续操作的依据。

1.5 断连分支重连、孤立分支删除

根据提取的端点和分叉点，判断孤立分支和毛刺，将其剔除。然

后对判断为正确的中心线信息进行连接，连接其中的中断的位置。

1.6 中心线分支模型构建

三维二值图像经过上面的步骤之后，得到了最终的中心线。该环节根据之前提取的中心线，判断得到其中的端点和分叉点，然后根据这些点，将中心线划分为多个子分支，同时需要判断每一个分支上做有点的顺序，使得最终显示的图像能够将不同的分支显示为不同的颜色。

2. 算法实现

本实验设计的算法基于对三维二进制图的充分分析，根据其特点设计对应的算法。其中，主要的环节是细化、端点分叉点检测以及分支的判断。对于细化算法，MATLBA 提供的算法**只有骨架的提取**，用骨架提取算法提取细小的血管是可以的，但是像根部这样的位置，由于血管比较粗，因此提取的骨架和中心线会有因一定的误差。我在分析了二维细化算法之后，充分分析了三维的对应的关系，**实现了三维细化算法**，以此和 MATLAB 内置的骨架提取算法相对比。而端点和分叉点的关系也是**基于对三维拓扑关系的分析**，在一个领域内的分析，得到对应的结果。而分支算法则是在分析了追踪算法后，结合本次实验设计的算法，得到一种**改进的追踪算法**，其效率更高更快，并且很好的保持了细化后的结果。

2.1 相关概念

首先说明下面的一些拓扑概念。

【邻接】

对于空间当中的两点，$p=(x1,y1,z1),q=(x2,y2,z2)$,根据两点的欧氏距离，有下面的分类：

$$若\left\|p-q\right\|\le 1,称p,q是6-邻接;$$

$$若\left\|p-q\right\|\le \sqrt{2},称p,q是18-邻接;$$

$$若\left\|p-q\right\|\le \sqrt{3},称p,q是26-邻接;$$

其中，本次试验常用的 26 领域即为一个 3*3*3 的立方体领域。

【领域】

基于三维空间邻接的定义和特点，若 p 点和 q 点 26-邻接，则称 p 点在 q 点的 26 领域当中。显然，q 点也在 p 点的 26 领域当中。

【目标点邻接】

在三维空间当中，像素值为 1 的点成为目标点；若两个目标点是 26-邻接的，则称这两个点是临街的。

2.2 填补空洞

空洞出现的情况等同于将血管“切开”,里面出现的非血管的情况。基于这种分析，在填补空洞是，也将其“切开”，分别分为三个方向的若干二维图，然后对其进行二维的填补，填补其中的空洞。

2.3 三维细化算法的实现

该细化算法由二维拓展而来，基本思路是从未处理的冠脉出发，由外向内的腐蚀，直到其变成一条单像素的“细线”为止。

腐蚀过程区别于书上的腐蚀,是用**不同的模板进行击中与否运算**,然后将击中与否得到的点删除。

在设计模板时，即要分析什么样的点是要删除的点，其应该具有下面的要求和特点。

A、其是**可以删除的点**，即删除后不影响细化后的图和原图的拓扑关系相似的要求。

B、根据 A 的要求，其是边缘点。这样的点**至少与一个白点 6 领域邻接**。

根据这两个要求，在每一个目标点的 3*3*3 的领域内进行判断，若其满足一定的模板特点，便可判断其为满足上面 要求的点，即可以删除的点。

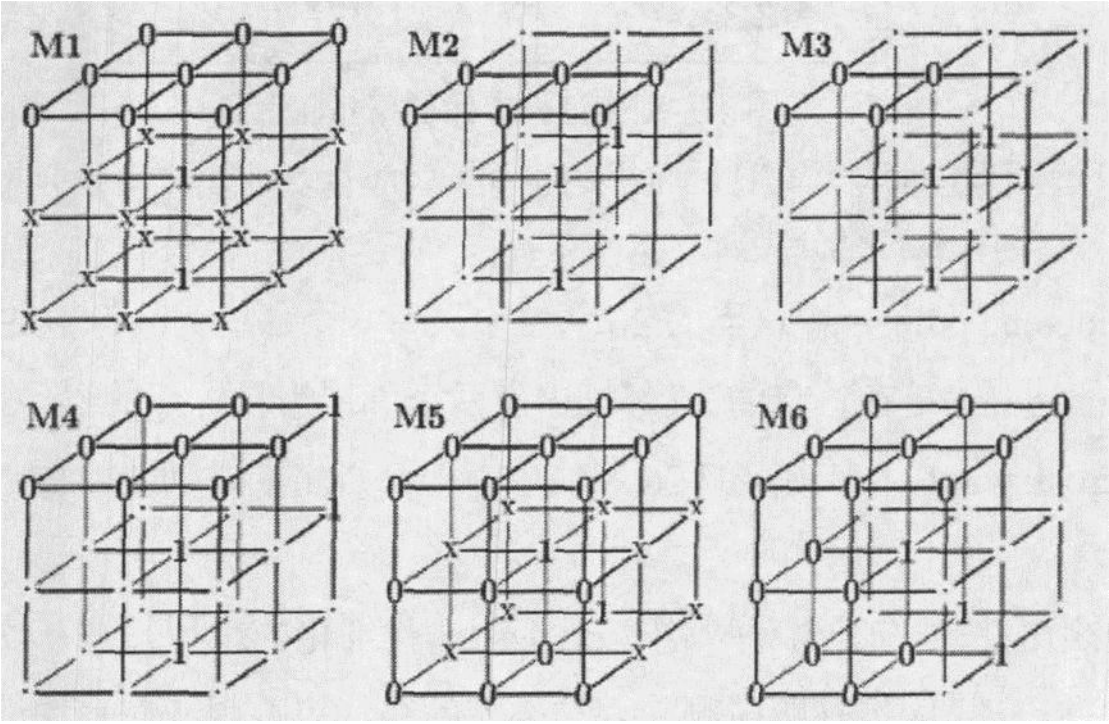
对于一个 3*3*3 的二值立方体，不包括中心的点（中心的点肯定是目标点，即为 1），其**形状有** $2^{26}=67108864$ **种可能性**。显然，穷尽列举执行击中与否在实现上是不可能的，特别是对于 CPU，也是很难在有限的时间里遍历完的。那么就需要其对其进行分类，分为不同的类进行击中与否运算。

在击中与否细化过程当中，为了使得细化的线条为冠脉的中心线，需要从不同的方向依次“苞去”最外面的一层。本算法从六个方向（上下左右前后六个方向）依次细化，分别设为 U、D、W、E、S、N。下面以上方向（U）为例，说明其模板。

若一个目标点的 3*3*3 的领域符合下面的模板之一，判断其为“上面的”可以删除的点。

下面的 6 个模板当中，“1”表示像素值为 1 第点，“0”表示像素值为 0 的点，“.”表示不关心其像素值，0 或者 1 均可，“X”表示所

有的这些 X 位置，至少有一个像素值为 1 的点。上表面的点应满足下面的 6 个模板及其各自旋转 90、180、270 所得的 24 种模板。



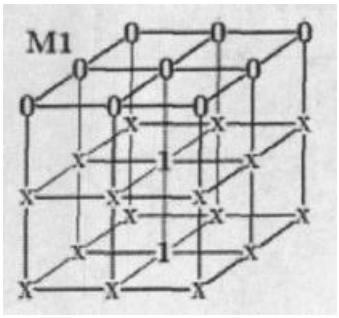
（图片截图自参考文献 1）

其余 5 个方向的模板只需将上面的模板旋转对应的方向即可。

根据上面的几种模板，设计对应的击中与否模板。在 6 个方向依次做击中与否运算，循环多次操作，直到所有的方向的击中与否运算所得结果为 0 矩阵，说明细化过程结束，结束细化。

（MATLAB 内置有骨骼提取算法，实验部分会将两者的结果加以对比）

下面距离说明其中的一个集中与否的模板，以 U 方向的 M1 为例。



第一种情况将（1,1,2）位置的点令为 1，此时，其余 X 位置为 0 或者为 1 都可以。设击中函数的模板为 B1、B2，则：

$$B1(:,:,1)=\begin{Bmatrix}0&0&0\\0&0&0\\0&0&0\end{Bmatrix}$$

$$B1(:,:,2)=\begin{Bmatrix}1&0&0\\0&1&0\\0&0&0\end{Bmatrix}$$

$$B1(:,:,3)=\begin{Bmatrix}0&0&0\\0&1&0\\0&0&0\end{Bmatrix}$$

而对应的 B2 为：

$$B2(:,:,1)=\begin{Bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{Bmatrix}$$

$$B2(:,:,2)=\begin{Bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{Bmatrix}$$

$$B2(:,:,3)=\begin{Bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{Bmatrix}$$

即不关心的位置在两个模板当中始终为 0。以这 B1，B2 为模板，进行击中预算，即可获得所有这种情况下的上边面的点，然后将原图减去这些点，即可得到这一步的细化结果。循环遍历 6 个方向，每一个方向遍历以上所有的模板，直到 6 个方向的所有模板击中与否运算结果都是 0 矩阵，细化过程结束。

2.4 分叉点、端点检测

根据细化后的结果，检测其中的端点和交叉点。

端点是在该点的 26-领域当中只有一个像素值为 1 的点。

交叉点是该点的 26-领域当中，有等于或者多于 3 个像素位置为 1 的点。

根据这样的分析，除了端点和交叉点，剩下的所有像素值为 1 的点便是线点，其 26-领域当中有 2 个像素值为 1 的点。

（这个函数在 MATLAB 里面有内置的，其对端点和分叉点的检测

结果和我自己实现的检测算法结果一样，但 MATLAB 没有检测“线点”的内置函数）

2.5 断连分支重连、孤立分支删除

步骤一：此环节首先删除一较短的孤立短线，这些点大都是原始数据当中的误差造成的，通过此过程，将其删除。

步骤二：除了删除上面的孤立的线，还要将一些毛刺删除，毛刺主要出现的一些长线上，且毛刺基于下面的分析：

A、如果其为孤立的点，其 26-领域当中没有其他任何目标点，应删除。

B、如有其为端点，其 26-领域当中有 1 个目标点，当这个位置的 5*5*5- 领域当中有多于 2 个的目标点时，则判断其是长线上的毛刺，应删除。

迭代这一环节，直到删除所有的毛刺。

步骤三：连接。连接即是将一些空缺的短线进行连接，找距离端点最近的、不属于其线段上的端点或线点进行连接。而连接方式是直线连接（满足最短路径的要求），以确保其在之后的检测当中，是线点。

这里判断断点需要连接，是作为参数进行调节的。（实验当中端点和线点线距离设为欧式距离 20 以下时连接；端点和端点的距离设为曼哈顿距离小于 45 时进行连接）

2.6 中心线分支模型构建

本次算法实现了改进的追踪算法。原本的追踪算法是根据曲线走

势判断点的顺序，这**在一些角度比较大的区域会出现误差**，同时，这对细化算法的要求很苛刻，对上面环节的误差很敏感。

而改进之后的算法任然充分利用空间拓扑结构。考虑上一环节得到的结果。如果将其中的分叉点全部抹点，那么剩下的便是线点和端点，根据这样的分析设计算法。

从任意端点出发，找其 26 领域当中的下一个点，对于线点，找下一个 26-领域内的点(线点的 26-领域会有 2 个目标点,此处需要判断)，直到找到端点，那么这是一个线段。以此循环迭代，直到找完所有的点。将每一条线段存储在一个 cell 当中，迭代循环，便得到的所有的线段，因此也就将所有点根据端点和分叉点进行了分类，同时做了排序。

3. 实验及分析

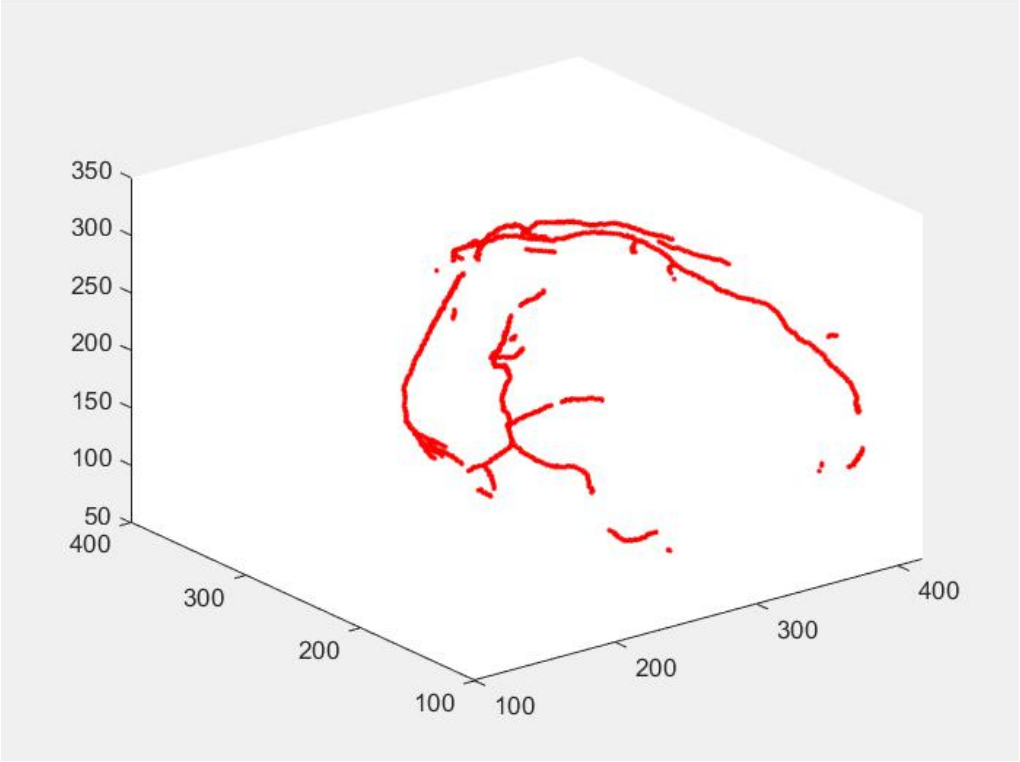
【以 ours_054_C1.mha 为例】

3.1 使用 MATLAB 内置的骨架提取函数

【二值图像】

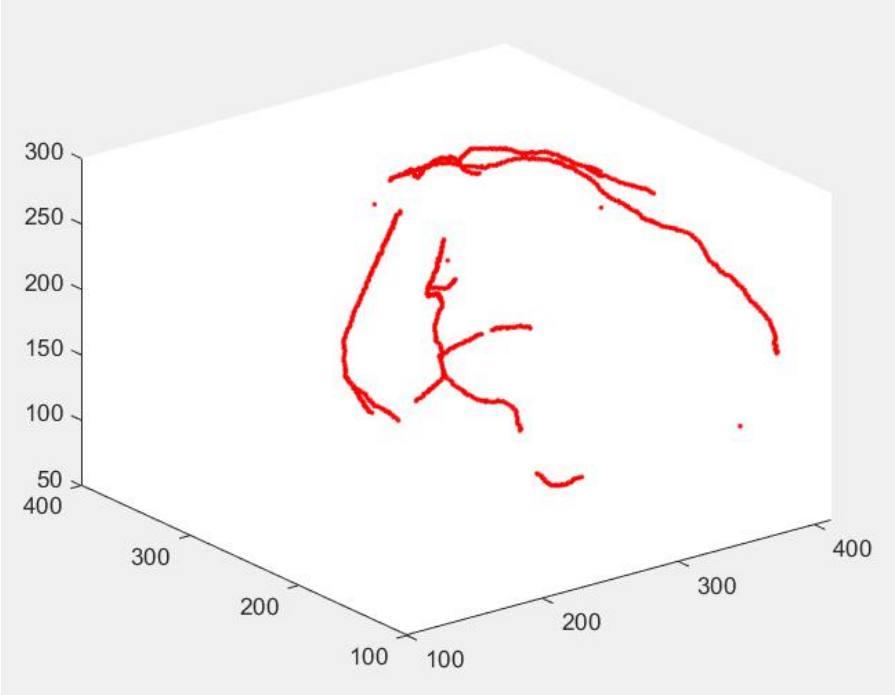
可见，其中有部分血管应该连起来的地方，出现了断连的情况。

【细化结果】



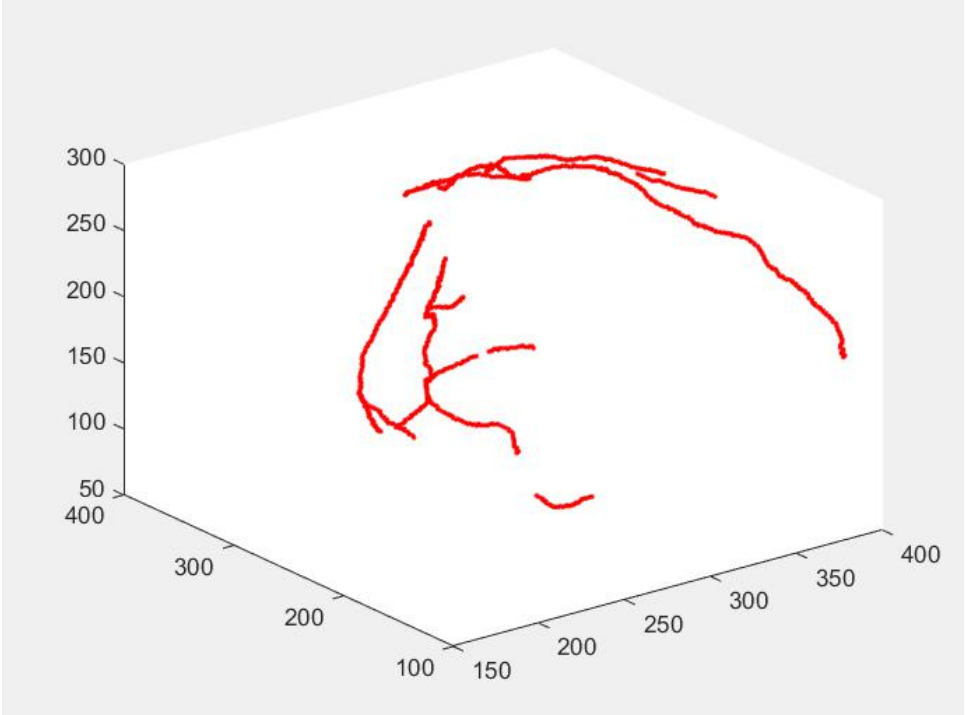
自带的骨架函数细化结果如上，可见其有曲线较为光滑，且符合血管走势。

【删除短线】



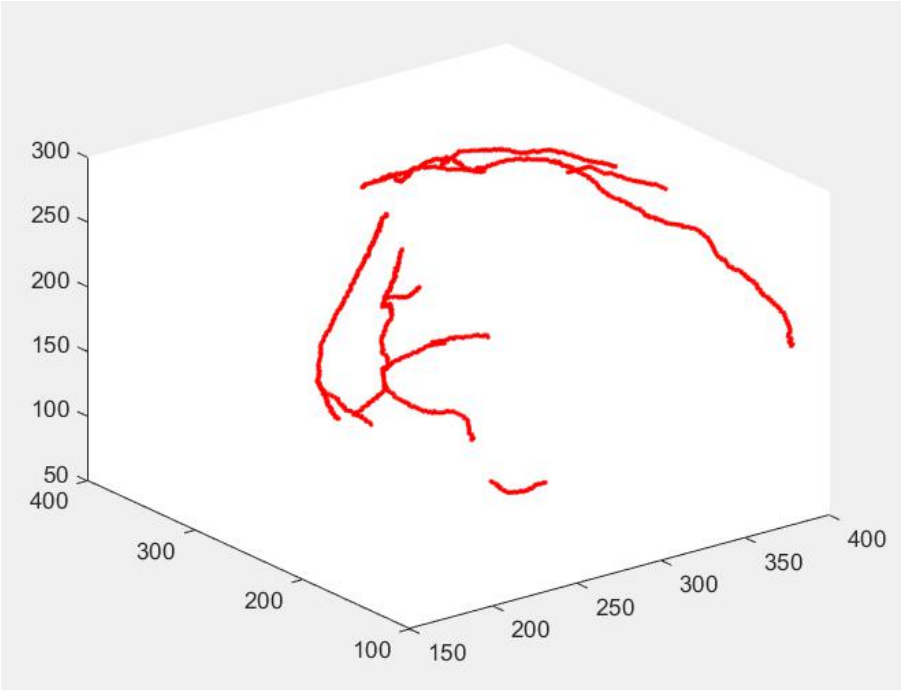
删除了部分短线和一些短的毛刺。

【删除毛刺】



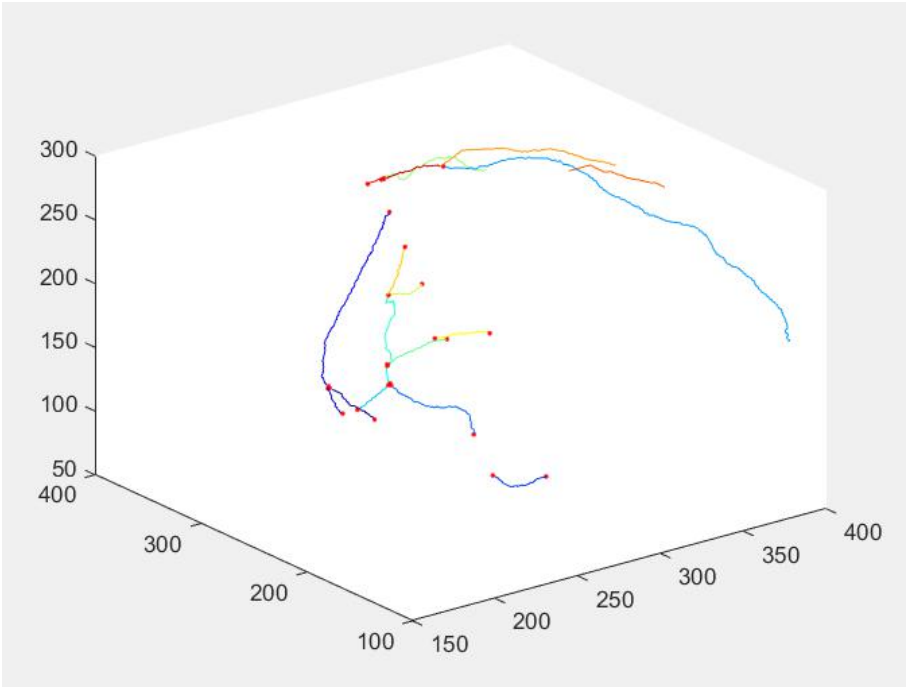
删除了孤立点和端点毛刺。

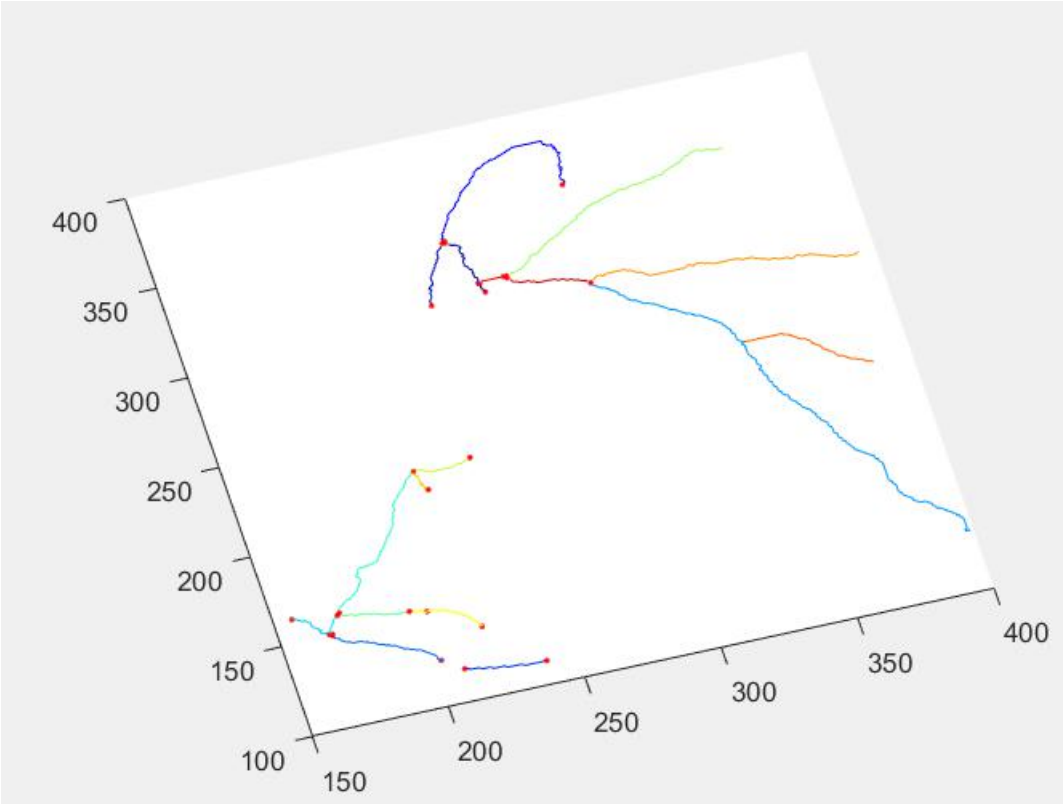
【连接】



断链的部分基本连接，只有一段距离较长的一段没有连接起来。这是本算法连接部分的一个参数（上面以有阐述），为了符合原数据，保留了一些较大的断点，以供医生判断病情。

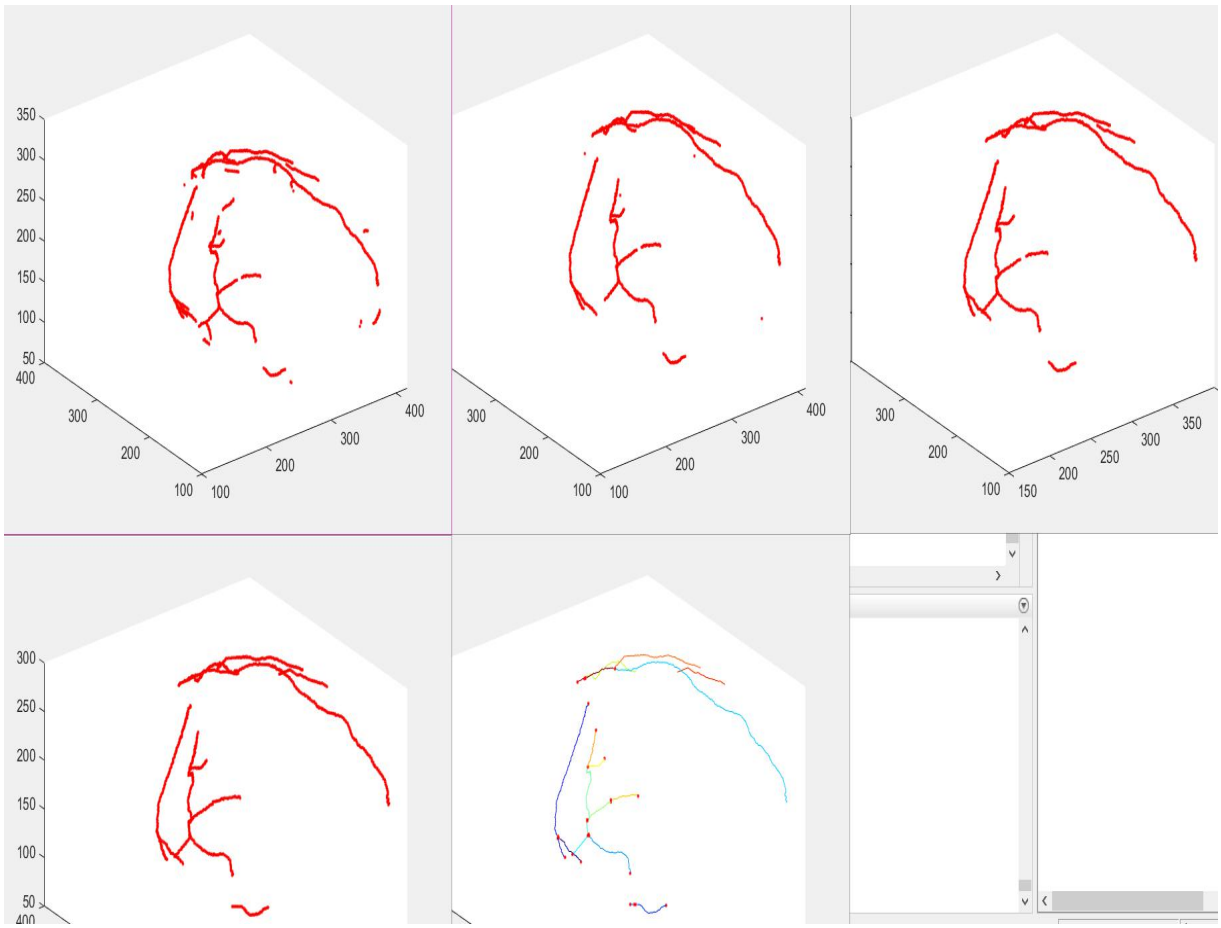
【树状图显示】





在提供的代码之上做了修改，将断点和分叉点显示为红色。

【对比显示】



以上六张图为各环节的结果，对比可知效果较好，符合预期效果。

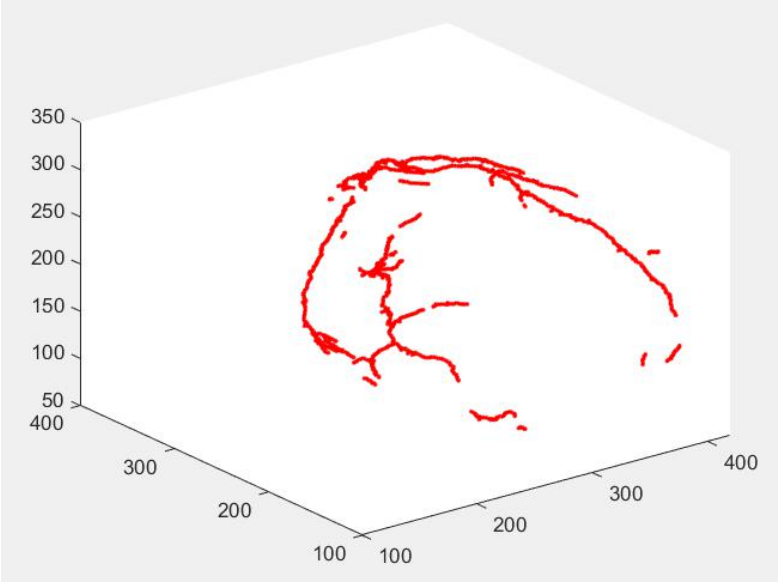
3.2 使用自己实现的细化算法

由于需要一层一层细化，因此效率是一个问题，如果能在 6 个方向（上下左右前后）同时并行细化，效率将会大大加快，但受限于笔记本 CPU 的性能，不能使用“parfor”并行计算，只好串行细化。

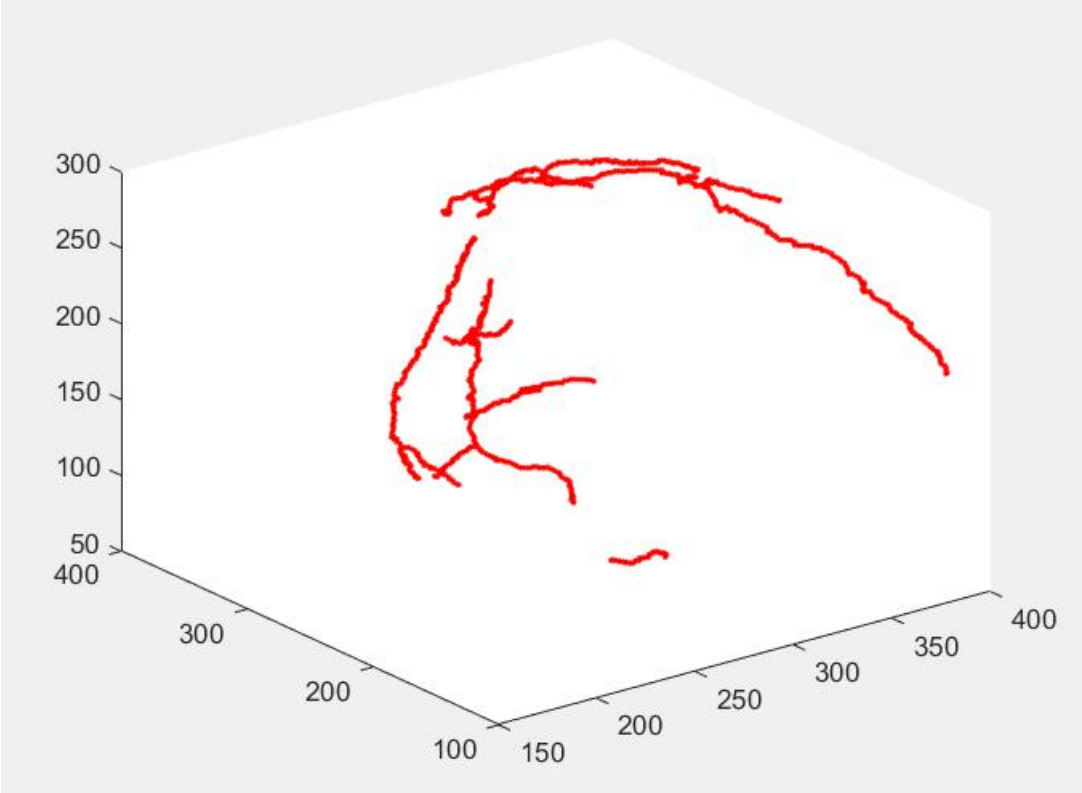
大概需要在 40 分钟左右，可以将原始的 37153 个目标点细化到 1450 个目标点左右。(MATLAB 内置的骨架提取函数到骨架结果在 1200 个目标左右左右)

【二值图像】

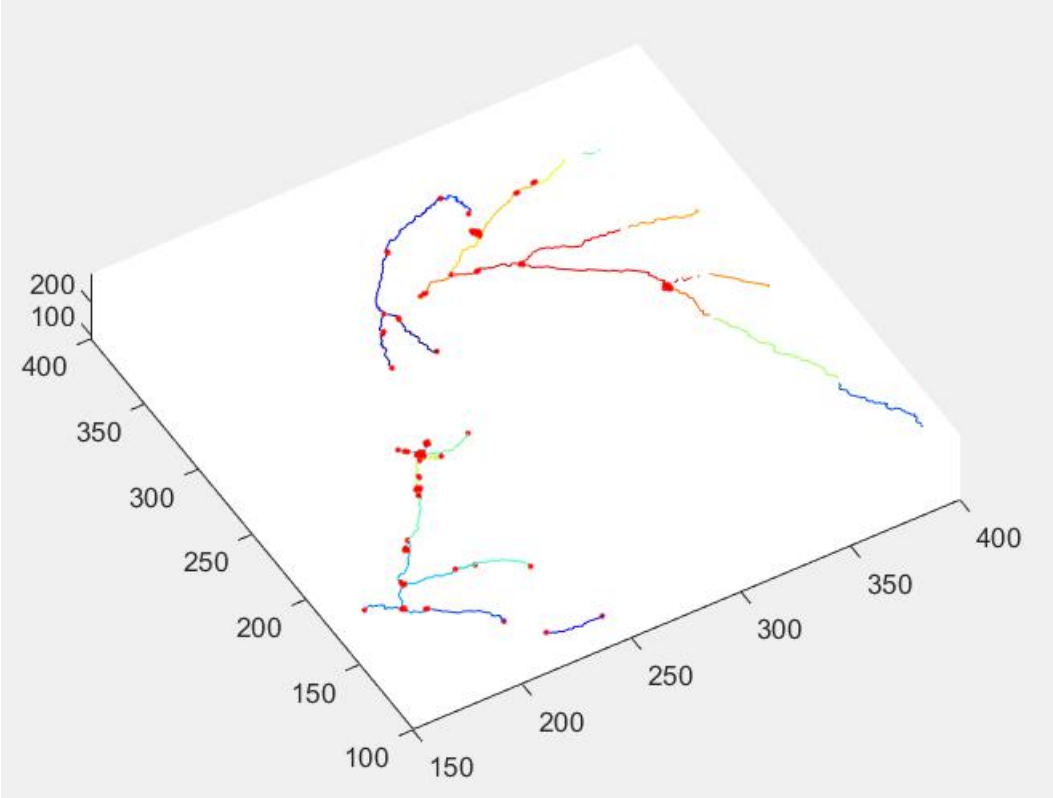
【细化结果】



【去除毛刺、连接断点】

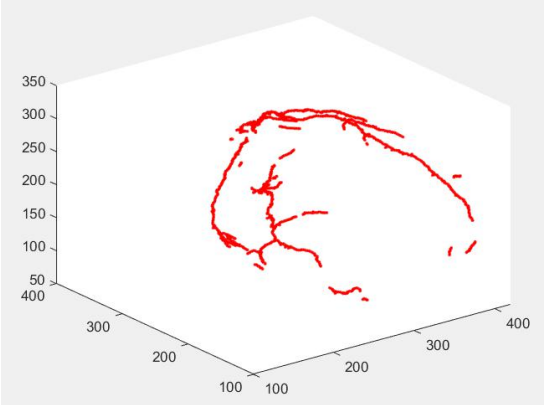

300
250
200
150
100
50
400
300
200
100
150
200
250
300
350
400

【树状图显示】

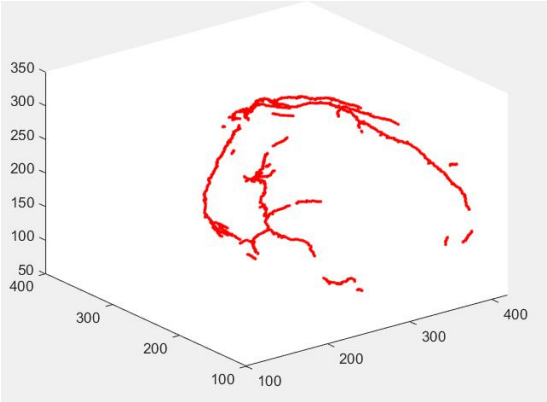

200
100
400
350
300
250
200
150
100
150
200
250
300
350
400

3.3 实验结果比较

【细化结果】



（内置骨架提取算法）

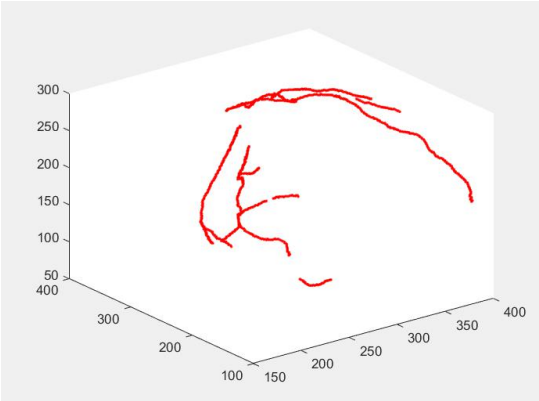


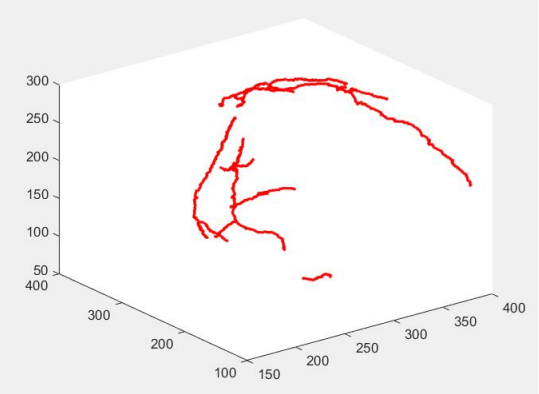
（自己实现的细化算法）

观察两种方法的细化结果，可以看出，整体细化一致。放大可以看出内置的骨架提取函数较为光滑，毛刺较少。相比而言，实现的细化算法则完全保留了原本的弯曲走势信息，因为是从六个方向以此细化，因此最后的细化结果也是中心位置的线，对一些血管的细节信息得到了很好的保留，这在医生分析当中很有用的。但是有其固有的缺点，即对一些噪声信息也得到了保留，这使得在长线上有许多毛刺，这一点直接对比两张图当中的信息也可以得到。

左边比右边的目标点少了大概 200 多点，这些点大多是毛刺，在接下来的去除毛刺环节会删除。

【去除毛刺环节】

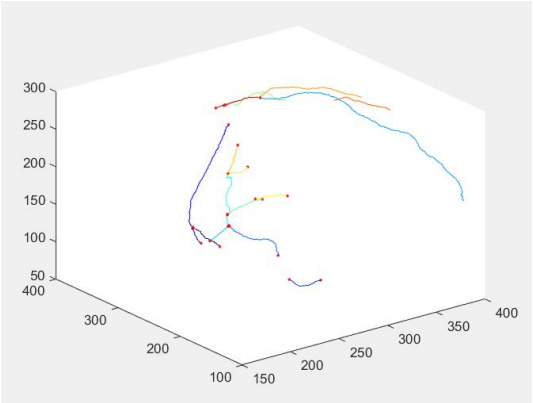


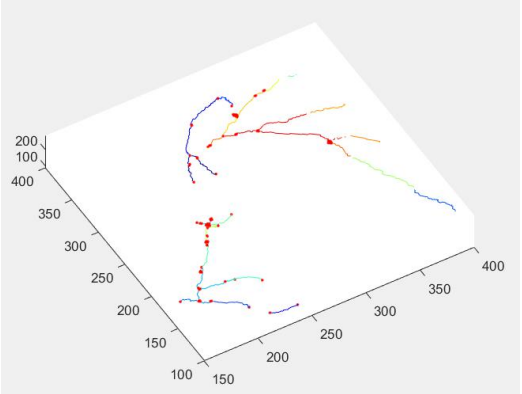


（内置骨架提取算法）　　（自己实现的细化算法）

任然是细化函数会保留原本的细节信息，因此会使得中心线看起来不够光滑，整体结构相似，说明细化函数有效。

【最终结果】





（内置骨架提取算法）　　（自己实现的细化算法）

对别两种细化方法最终的拓扑架构，两者基本相似，实现的细化算法对关节点和端点的处理没有内置的骨架函数好，这主要是没有穷举完所有的 3*3*3 领域的可能性造成的，但是在线段上的点则较好的保留了血管的粗细变化信息。这对于医生的分析是至关重要的信息。

对比可知自己实现的细化算法还有一定的缺陷，主要是对分叉

的处理上。但可以说明实现的细化算法行之有效，实现了“细化”的功能，保留了原本的拓扑信息，并且较好的保留了原本的血管走势信息。至于速度，如果用性能比较好的处理器或者图形计算器的话，使用并行细化，将大大加快细化环节。

4. 关键问题分析和解决

4.1 细化算法

三维图像的细化算法没有内置的函数，而三维的骨架提取函数在一些较粗的领域会出现误差，因此在细化环节就遇到了困难。

在查找资料后了解到，细化算法是一层一层腐蚀，因此速度较慢，我想这也是 MATLAB 没有内置的原因。但是理论和方法都是成熟的，因此我先分析二维细化过程，然后将其往三维推广。

这里主要的困难在于三维的细化模板太多，穷举又不显示，只好查血文献。在文献[1]当中有所体现，虽然这篇文献当中的分类有错误（有重复的情况），但给了我很大的提示。然后按照不同的情况在设计对应的模板。模板的设计花费了较多的时间，主要原因在于三维情况的击中模板情况比二维要复杂的多。

最终成功实现可细化算法，也是本次算法当中我收获最大的部分。虽然没有 MATLAB 内置的骨架函数那样速度快，但是通过自己手动实现细化算法，对二维三维的形态学处理。特备是腐蚀、击中与否等操作有了更加深刻的认识和理解。同时也对后面的改进版本的追踪算法提供了思路。

4.2 旋转函数

在细化算法当中由于模板太多，需要许多矩阵的旋转操作。原本使用了内置的 Rot90（）函数，但是跑出来的细化结果总是不尽人意，不够“光滑”。在一步步 DeBug 之后发现，内置的旋转函数得到的并不是自己想要的结果，并且内置的三维旋转函数只有沿 Z 轴旋转的。之一环节只好自己实现沿不同轴的旋转算法。改正之后，细化效果好了许多。

4.3 改进追踪算法

在分析了追中算法之后，发现其并不是很适合本次任务，只好回归到原本的拓扑分析当中，沿着端点往下找，这样完全保留了细化和连接之后的结果，并且该环节的所有代码均有自己实现，没有使用内置的连通分量的搜索方法。虽然在速度上可能没有内置的函数速度快，当中通过自己编写提取代码，对整个过程的每一个细节都有比较清楚的理解和认识。

5. 总结和收获

这次大作业正如老师所讲，收获颇丰。从刚开始的无从下手到一步步分析、一步步实现，虽然过程当中有点“痛苦”，但收获确实很多。

本次作业比较满意的地方是自己实现了细化算法，虽然效果没有内置骨架函数好、速度也有点慢，但是在设计细化算法的过程当中，让我对形态学这一章有了充分的复习和理解，同时对三维的拓扑关系

也有了更加深刻的认识。

同时使用“按图索骥”的方法实现了改进版本的“追踪”算法，对端点、交叉点、线点的结构和关系有了具体到像素点的认识。

另外本次大作业也然后感受到了交流的重要性，在刚刚开始大作业时，完全没有思路。在和同学讨论和分析过后，思路也渐渐清晰。同时，和同学之间的交流使得我们的算法一步步优化，不仅提高了效率，同时也互相学习，共同进步。

6. 参考文献

[1] 赵宏伟. 基于 CT 数据的冠脉提取和细化方法的研究和实现.2008.06

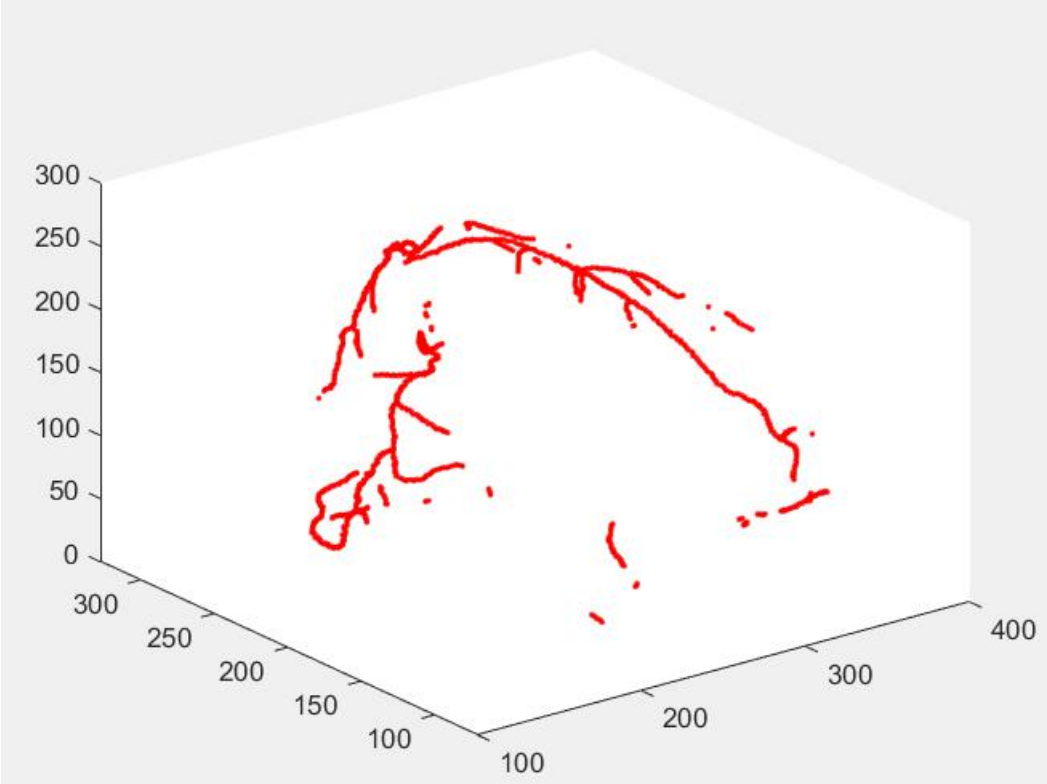
[2] 付玲. 基于 CT 影像的管状组织分割与中心线提取研究. 2012.09

【附录】

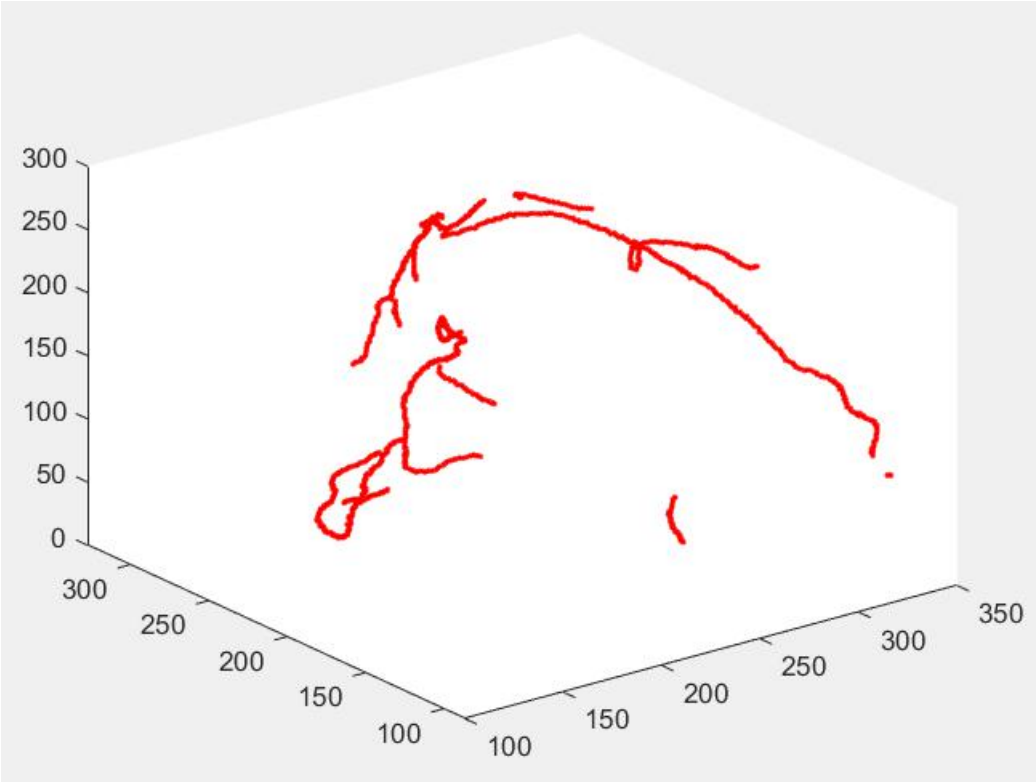
1. ours_066_C1.mha 实验结果

1.1 使用 MATLAB 内置的骨架提取函数

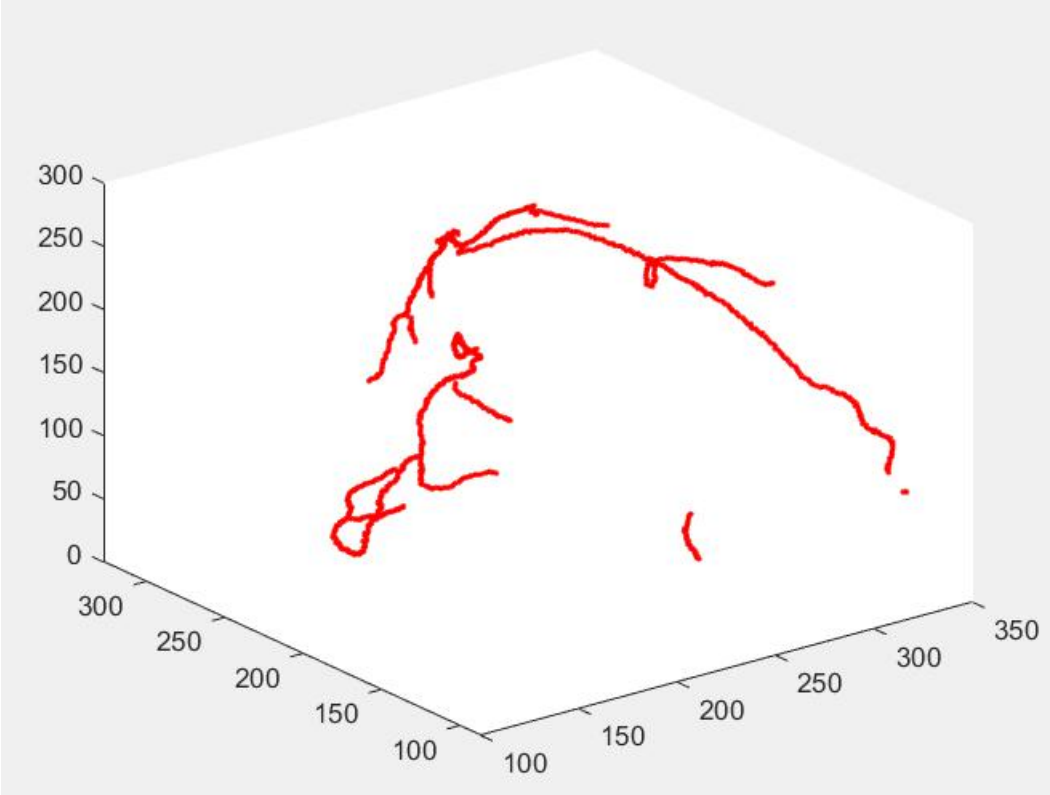
【细化结果】


300
250
200
150
100
50
0
300
250
200
150
100
100
200
300
400

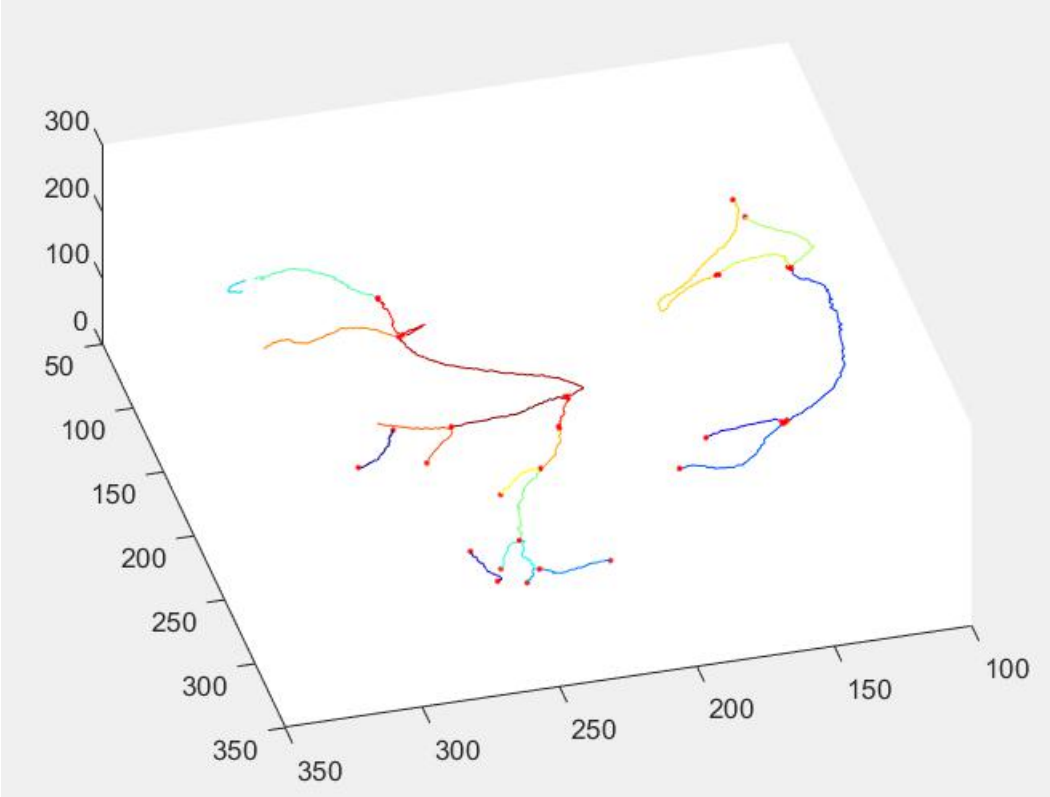
【去除毛刺】


300
250
200
150
100
50
0
300
250
200
150
100
100
150
200
250
300
350

【连接断点】

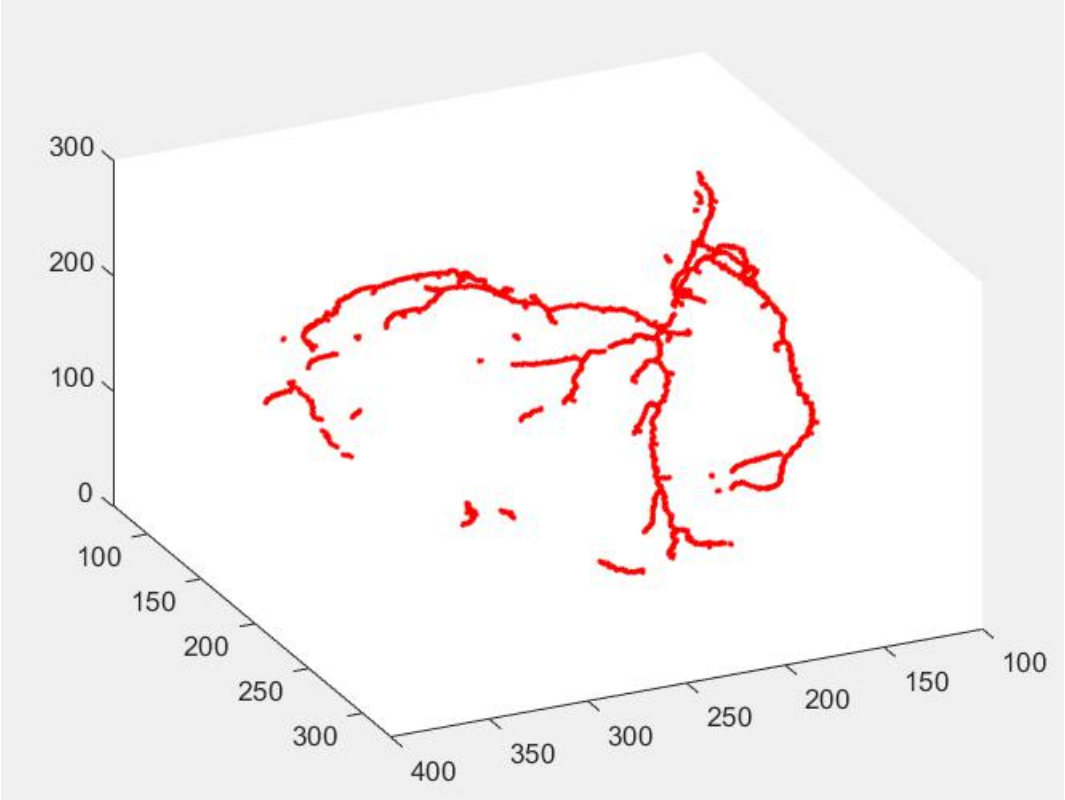

300
250
200
150
100
50
0
300
250
200
150
100
100
150
200
250
300
350

【最终结果】

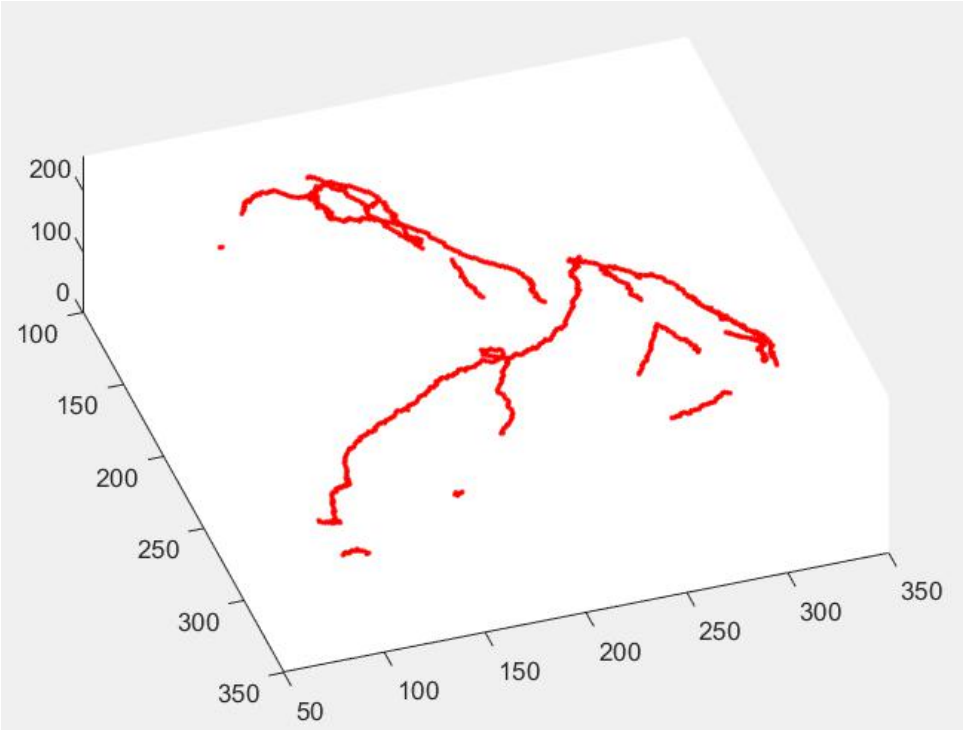

300
200
100
0
50
100
150
200
250
300
350
350
300
250
200
150
100

1.2 自己实现的细化算法

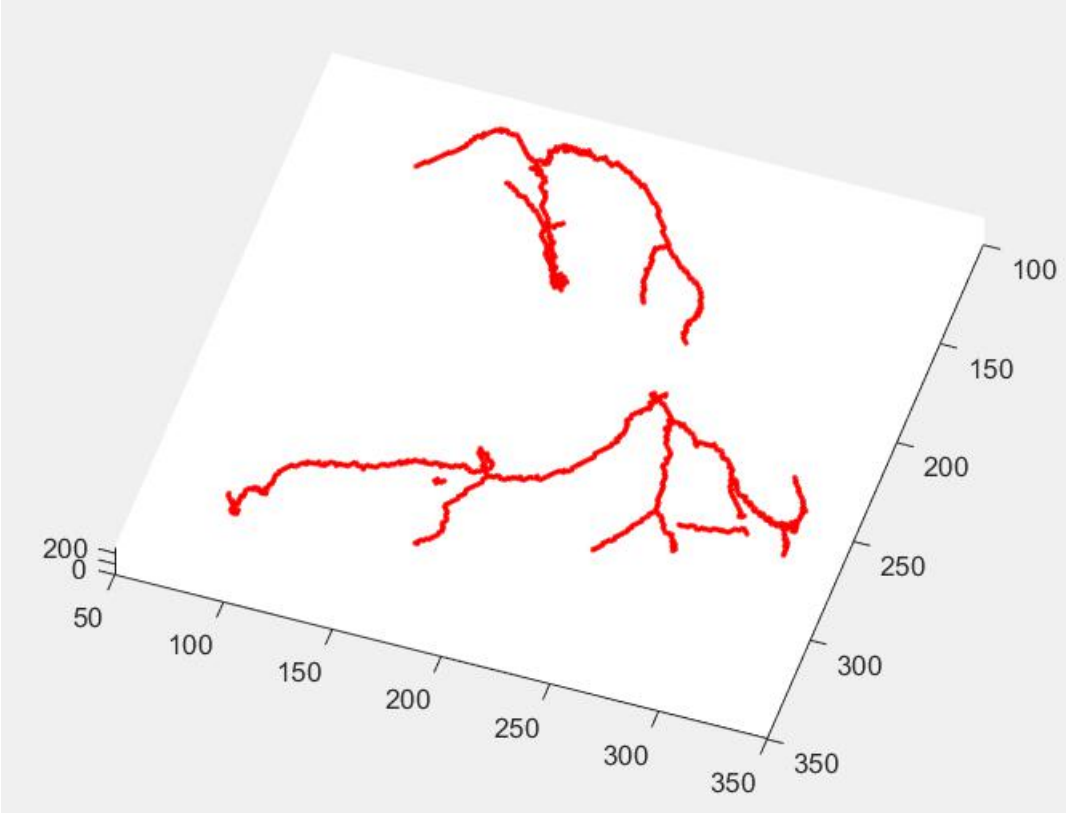
【细化结果】



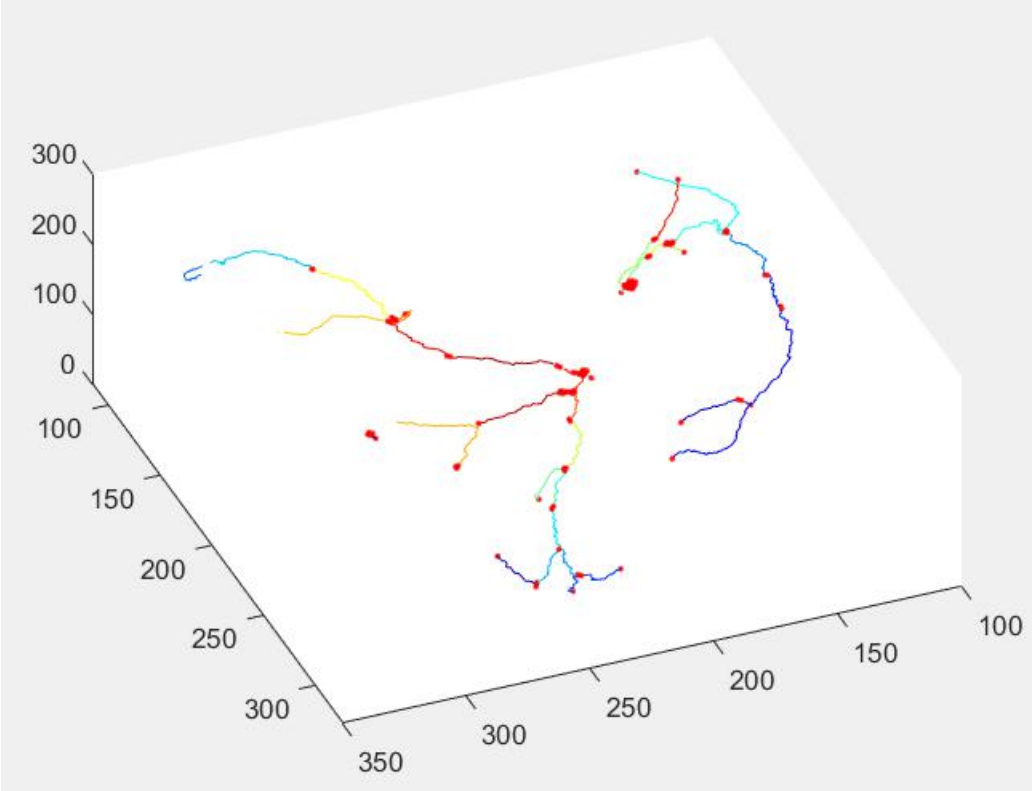
【去除毛刺】



【连接断点】


200
0
50
100
150
200
250
300
350
100
150
200
250
300
350

【最终结果】


300
200
100
0
100
150
200
250
300
350
300
250
200
150
100