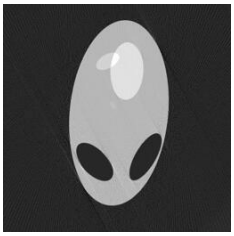
CT 系统参数标定及成像

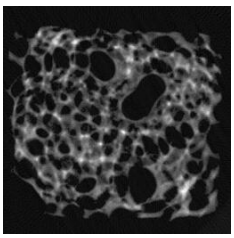
摘要

CT 以其无损、精确等优点广泛应用于医学诊断、工业检测、安检等各个领域。本文研究的是 CT 系统主要部件安装定位的精度和相关几何参数的测量精度，以及对图像的重建过程，并对标定模板进行合理的改进以提高精度和稳定性。

针对问题(1)，首先我们根据标定模板上的圆形模板在各角度平行入射的 X 射线下，投射在探测器上的长度相同，可得长度所包含的具体等距单元探测器个数 n=29。再由圆形模板的半径 r=4mm，求得探测器单元之间的距离为 0.2759mm；之后，选取附件 2 中第 1、137、180 列数据，通过两个图形的中心点在探测器上的投影距离与实际距离之间的几何关系，可得三个合适的 θ 值；然后由已知的 θ 值、与旋转中心以及探测器与正方形模板齐平的左端点坐标的几何关系，列出包含旋转中心坐标、左端点的三个线性方程。求得在以椭圆中心为原点，单位为 mm 的直角坐标系下，旋转中心坐标为(-9.2239,6.1489)（在以图像左下顶点为原点，单位为 mm 的直角坐标系下，旋转中心为（40.7761,56.1489））；最后，将附件 2 中 180 组数据反代回上述方程，在 matlab 中利用遍历的方法逼近各组数据的 θ 值并进行，其中初始 θ=29.8392°，终止 θ=208.7679° 具体见附录 1。

针对问题(2)(3)，首先运用以卷积定理和中心切片定理为基础的滤波反投影重建算法，选取 ramLak 滤波函数进行 CT 图像重建，确定几何形状。然后，通过坐标平移，使正方形托盘的中心与求得的重建图像中心重合；之后，还原图片真实比例，使得图像像素变为符合题意的 256×256。画出最终的还原图像，如下图，得出图形在正方形托盘中的位置。最后，我们利用重建图像中提取的灰度值，定义了吸收率。之后绘制灰度值与像素点个数的直方图，确定灰度的阈值，去除了灰度近似于 0 的像素点的影响；再对灰度分级，确定问题(2)(3)不同级别像素点的吸收率，结果见材料 2 和材料 4。





针对问题(4)，分析误差主要来源于圆半径较小、接收器单元离散等。不稳定性主要来源于标定模板的形状导致不能得到一个简单有效的方法求解 θ 值。本文增加了一个等距曲边三角形模板，对探测器单元之间的距离以及该 CT 系统使用的 X 射线的 180 个方向进行较为精确和稳定的求解。

模型在求解旋转中心、吸收率时有很大的优势；在对模板改进的时候避免了为求解角度使用的低效率算法，使得参数求解精简准确，对参数标定的改进有重大意义。但在角度求解时部分结果不理想，对(2)(3)问结果有一定的影响。吸收率的定义仍有改进的空间。

关键词：CT 图像重建，滤波反投影重建算法，灰度值，吸收率，
等距曲边三角形

一、问题重述

CT(Computed Tomography)可以在不破坏样品的情况下，利用样品对射线能量的吸收特性对生物组织和工程材料的样品进行断层成像，由此获取样品内部的结构信息。一种典型的二维 CT 系统如图 1 所示，平行入射的 X 射线垂直于探测器平面，每个探测器单元看成一个接收点，且等距排列。X 射线的发射器和探测器相对位置固定不变，整个发射-接收系统绕某固定的旋转中心逆时针旋转 180 次。对每一个 X 射线方向，在具有 512 个等距单元的探测器上测量经位置固定不动的二维待检测介质吸收衰减后的射线能量，并经过增益等处理后得到 180 组接收信息。

CT 系统安装时往往存在误差，从而影响成像质量，因此需要对安装好的 CT 系统进行参数标定，即借助于已知结构的样品（称为模板）标定 CT 系统的参数，并据此对未知结构的样品进行成像。

请建立相应的数学模型和算法，解决以下问题：

(1) 在正方形托盘上放置两个均匀固体介质组成的标定模板，模板的几何信息如图 2 所示，相应的数据文件见附件 1，其中每一点的数值反映了该点的吸收强度，这里称为“吸收率”。对应于该模板的接收信息见附件 2。请根据这一模板及其接收信息，确定 CT 系统旋转中心在正方形托盘中的位置、探测器单元之间的距离以及该 CT 系统使用的 X 射线的 180 个方向。

(2) 附件 3 是利用上述 CT 系统得到的某未知介质的接收信息。利用(1)中得到的标定参数，确定该未知介质在正方形托盘中的位置、几何形状和吸收率等信息。另外，请具体给出图 3 所给的 10 个位置处的吸收率，相应的数据文件见附件 4。

(3) 附件 5 是利用上述 CT 系统得到的另一个未知介质的接收信息。利用(1)中得到的标定参数，给出该未知介质的相关信息。另外，请具体给出图 3 所给的 10 个位置处的吸收率。

(4) 分析(1)中参数标定的精度和稳定性。在此基础上自行设计新模板、建立对应的标定模型，以改进标定精度和稳定性，并说明理由。

(1)-(4)中的所有数值结果均保留 4 位小数。同时提供(2)和(3)重建得到的

介质吸收率的数据文件（大小为 256×256，格式同附件 1，文件名分别为 problem2.xls 和 problem3.xls）

二、问题的分析

1、问题（1）分析

在应用 CT 进行各种检测时，不同几何参数误差在重建图像中的表现形式对系统几何参数的判定具有重要意义，而 CT 系统主要部件安装定位精度和相关几何参数的测量精度是影响重建图像质量的重要因素。[1]

首先，我们根据标定模板上的圆形模板在各角度平行入射的 X 射线下，投射在探测器上的长度相同，由附件 2 数据，经过数据筛选和分析可得长度所包含的具体等距单元探测器个数。再由圆形模板的半径，可知探测器单元之间的距离。

之后，选取三组合适的数据，通过两个图形的中心点在探测器上的投影距离与实际距离之间的几何关系，可得三个 θ 值。

然后由已知的 θ 值、与旋转中心以及探测器与正方形模板齐平的左端点的几何关系，列出包含旋转中心坐标(x_0,y_0)、左端点x_1的三个方程，由已知 θ 值以及附件 2 中数据可以进行求解旋转中心坐标。

最后，由求得的旋转中心(x_0,y_0)与相关数据，将附件 2 中 180 组数据反代回上述方程，在 matlab 中利用遍历的方法求解 θ 值。

2、问题（2）、（3）分析

①几何形状的确定：首先对 CT 断层成像的原理进行分析，然后运用以卷积定理和中心切片定理为基础的滤波反投影重建算法，选取 ramLak 滤波函数进行图像重建。

②在正方形托盘中位置的确定：首先，通过坐标平移，使正方形托盘的中心与求得的重建图像中心重合；之后，还原图片真实比例，使得图像像素变为符合题意的 256×256。画出最终的还原图像，得出图形在正方形托盘中的位置。

③吸收率的确定：首先我们根据题目中的相关信息自主定义吸收率，并从重建图像中提取出灰度值。之后绘制直方图，确定阈值，对灰度分级，确定吸收率。

3、问题（4）分析

问题（1）误差主要来源于圆半求解径较小、接收器单元离散以及球心确定等。不稳定性主要来源于标定模板的形状导致不能得到一个稳定的非线性方程进行求解 θ 值。针对其误差与不稳定性，我们增加了一个等距曲边三角形模板，对探测器单元之间的距离以及该 CT 系统使用的 X 射线的 180 个方向进行较为精确和稳定的求解。

三、模型的假设

（1）考虑到 X 射线的波动不明显，故忽略 X 射线的衍射现象，将 X 射线的传播做直线传播处理。

（2）忽略 X 射线光源器以及接收器的体积和形状，将其处理为质点。

（3）数据能客观反映实际情况。

四、符号说明

（1） θ：探测器左端点沿探测器所成的向量角

（2）r：圆的半径

（3）d：探测器单元之间的距离

（4）D：一个探测单元间距的毫米数

（5）l：探测器左端点（ θ 值为 0 时，探测器在模版的正上方时的左侧端点）到椭圆中心点的投影点的距离

五、模型的建立

1、问题（1）模型建立

由问题（1）分析中所言，我们根据模板的几何信息、吸收率以及接受信息之间的关系，对探测器单元之间的距离、旋转中心在正方形托盘中的位置、该 CT 系统使用的 X 射线的 180 个方向依次进行建立数学模型并求解，具体求解结果见第六部分——模型的求解。

①确定探测器单元之间距离的距离

标定模板上的圆形模板在各 θ 角度平行入射的 X 射线下，投射在探测器上的长度相同，且都等于该圆形半径长度的 2 倍。由圆形模板半径 r=4mm，以及附件 2 中可分辨且较完全接收到圆形模板信息的探测器个数 n，有公式：

$$d=\frac{2r}{n}$$

通过对附件 2 中的 180 组数据进行分析和筛选，计算 n 的值，代入上式，可得探测器单元之间的距离 d。

②旋转中心在正方形托盘中位置的确定

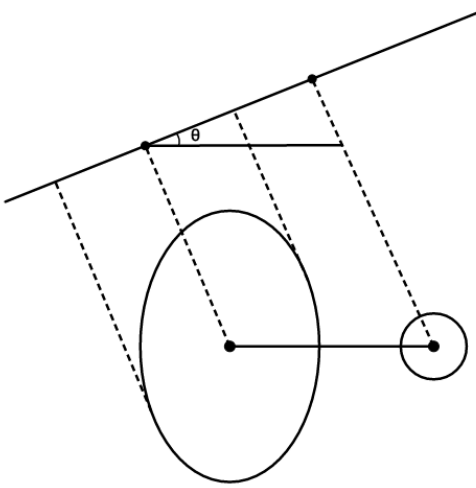


图 1 θ 值示意图

如图 1 情况中，θ 值如图所示。对附件 2 中数据进行删选和分析，可选取三组数据求得的三个旋转角度$\theta_1,\theta_2,\theta_3$

这里 θ 角是探测器左端点沿探测器方向所成向量与水平轴的夹角。

设旋转中心坐标(x_0,y_0)，当$\theta=0$时探测器左端点坐标为(x_1,y_1)，当θ发生变化时探测器左端点坐标为$(x_1^{'},y_1^{'})$ 椭圆中心坐标为(x_2,y_2)。θ 角意义同上。

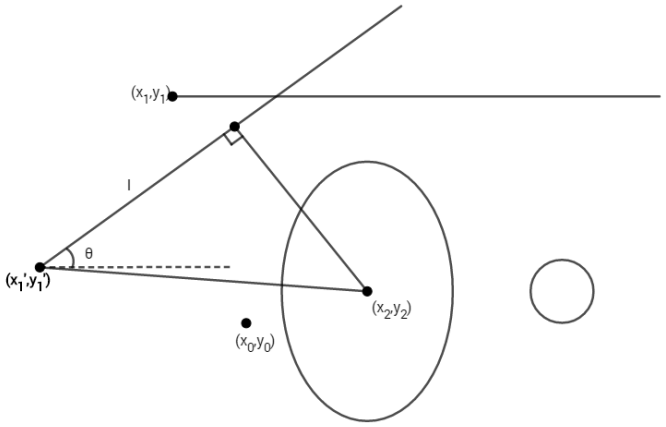


图 2 旋转中心求解示意图

以椭圆中心(x_2,y_2)为原点建立直角坐标系。

由几何关系及向量运算可知：

$$l=(\cos\theta,\sin\theta)\begin{pmatrix}x_2-x_1^{'}\\ y_2-y_1^{'}\end{pmatrix}$$

由旋转关系可知：

$$\begin{pmatrix}x_1^{'}\\ y_1^{'}\end{pmatrix}=\begin{pmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{pmatrix}\begin{pmatrix}x_1-x_0\\ y_1-y_0\end{pmatrix}+\begin{pmatrix}x_0\\ y_0\end{pmatrix}$$

整理得：

$$\begin{cases}x_1^{'}=\cos\theta_1(x_1-x_0)-\sin\theta_1(y_1-y_0)+x_0\\ y_1^{'}=\sin\theta_1(x_1-x_0)+\cos\theta_1(y_1-y_0)+y_0\\ l_1=\cos\theta_1(x_2-x_1^{'})+\sin\theta_1(y_2-y_1^{'})\end{cases}$$

对上式进行化简：

$$l_1=\cos\theta_1[x_2-\cos\theta_1(x_1-x_0)+\sin\theta_1(y_1-y_0)-x_0]+\sin\theta_1[y_2-\sin\theta_1(x_1-x_0)-\cos\theta_1(y_1-y_0)-y_0]$$

⇓

$$l_1=x_2\cos\theta_1-(x_1-x_0)\cos^2\theta_1+(y_1-y_0)\sin\theta_1\cos\theta_1-x_0\cos\theta_1+y_2\sin\theta_1-(x_1-x_0)\sin^2\theta_1-(y_1-y_0)\sin\theta_1\cos\theta_1-y_0\sin\theta_1$$

⇓

$$l_1=(x_2-x_0)\cos\theta_1+(y_2-y_0)\sin\theta_1-(x_1-x_0)$$

即有函数$f(x_0,y_0,x_1)$，由公式表示如下：

$$l_i=(x_2-x_0)\cos\theta_i+(y_2-y_0)\sin\theta_i-(x_1-x_0)$$
$$(i=1,2,3)$$

(I)

对附件 2 中数据进行删选和分析选取的三组数据求得的三个旋转角度$\theta_1,\theta_2,\theta_3$以及三个$l_1,l_2,l_3$，代入上述公式，可得旋转中心坐标$(x_0,y_0)$。

③该 CT 系统使用的 X 射线的 180 个方向的确定

由求得的旋转中心(x_0,y_0)与(x_1,y_1)、(x_2,y_2)，使函数$f(x_0,y_0,x_1)$可转化为函数$f(\theta_i,l_i)$。将附件 2 中 180 组数据反代回公式(Ⅰ)，可转化为公式(Ⅱ)，如下：

$$l_i = (x_2 - x_0)\cos\theta_i + (y_2 - y_0)\sin\theta_i - (x_1 - x_0)$$
$$(i = 1,2,3...180)$$

(II)

最后，在 matlab 中利用遍历的方法求解 180 个 θ 值。

2、问题（2）、（3）模型建立

通过对问题整体的分析，我们对问题（2）、问题（3）采用同样的方法进行模型的建立与求解。

①几何形状的确定

a.CT 断层图像原理 [2]

CT 成像过程可归纳为图 3。如下：

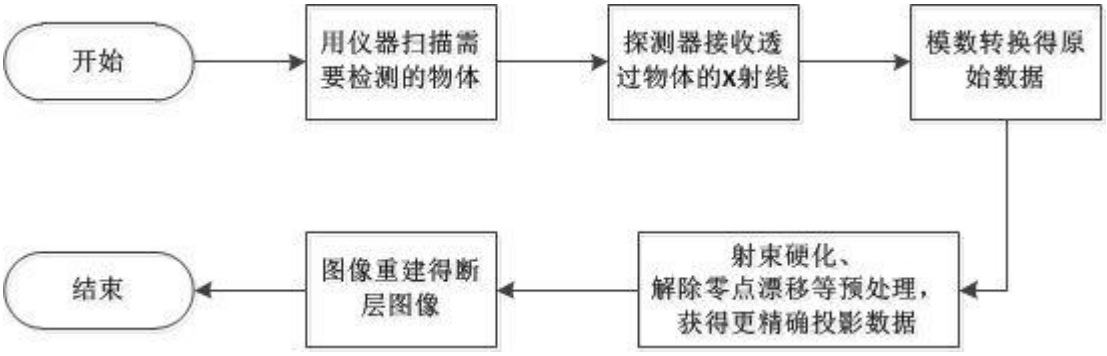


图 3 CT 成像过程流程图

断层成像技术就是从不同射线角度θ，不同检测器的位置的许多投影值重建原始图像的过程。

b.滤波反投影重建算法

反投影重建的基本思想是断层内该点的密度值就是经过该平面上的一点的射线投影和。我们将“取投影”→“反投影重建”→“重建后图像”看作一个系统。图像重建的过程可以看成是由一系列坐标变换得到的。为保证图像的质量，使不易失真，我们采用滤波反投影算法。该算法使在一定视角下投影，然后进行滤波投影，再做反投影，把这些反投影值累加就可以得到重建图像。

在进滤波反投影时，运用卷积定理，并以中心切片定理为基础，进行图像重建。

由二维傅里叶变换的一个重要定理——卷积定理（如下）

$$F\left(f(x,y)*h(x,y)\right)=F(u,v)\cdot H(u,v)$$

则需要重建图像（具体符号代表含义如图 4）为：

$$\hat{f}(r,\theta)=f(x,y)=F_2^{-1}\left[F(\omega_1,\omega_2)\right]=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}F(\omega_1,\omega_2)e^{i2\pi(\omega_1 x+\omega_2 y)}d\omega_1 d\omega_2$$

$$\omega_1=2\pi\rho\cos\phi\ \ \omega_2=2\pi\rho\sin\phi\ \ x_r=r\cos(\theta-\phi)$$

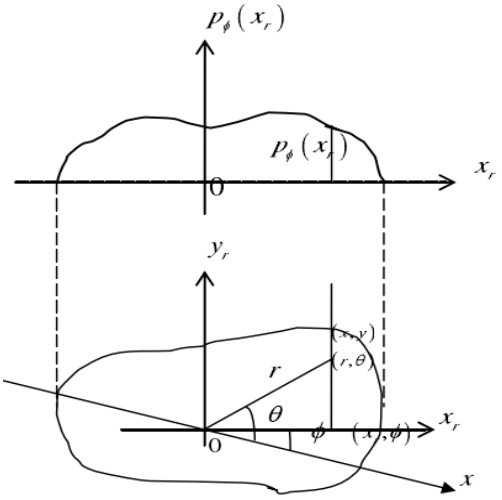


图 4 滤波反投影重建算法坐标系

将公式进行整理，有：

$$\hat{f}(r,\theta)=f(x,y)=F_2^{-1}\left[F(\omega_1,\omega_2)\right]=\int_0^{\pi}g\left[r\cos(\theta=\phi),\phi\right]d\phi$$

且有 $h(x_r)=F_1^{-1}(|\rho|)$，$p(x_r,\phi)=F_1^{-1}(P(\rho,\phi))$，$g\left[r\cos(\theta-\phi),\phi\right]=h(x_r)*p(x_r,\phi)$

为使重建图像更加清楚，我们选择 ramLak 滤波函数进行图像重建。

ramLak 滤波函数系统函数表示为：

$$H_{R-L}(\rho)=|\rho|W(\rho)=|\rho|rect(\tfrac{\rho}{2B})$$

其中

$$rect\left(\tfrac{\rho}{2B}\right)=\begin{Bmatrix}1,|\rho|<B=\dfrac{1}{2d}\\0,\text{其他}\end{Bmatrix}$$

d 代表采样间隔，$B=1/(2d)$ 为最高不失真频率。

系统冲激函数表为：

$$h_{R-L}(x_r)=\int_{-B}^{B}|\rho|e^{i2\pi\rho x_r}d\rho=\int_{-B}^{B}(-\rho)e^{i2\pi\rho x_r}+\int_{0}^{B}\rho e^{i2\pi\rho x_r}d\rho=2B^2\sin c(2x_r,B)-B^2\sin c^2(x_r,B)$$

我们在对图像滤波时，使用离散化的函数，将 $x_r=nd$ 带入上述式子，这样离散化的冲激函数表示为：

$$h_{R-L}(nd)=\begin{cases}\frac{1}{4d^2},n=0\\0,n=\text{偶数}\\-\frac{1}{n^2\pi^2d^2},\ n=\text{奇数}\end{cases}$$

②在正方形托盘中位置的确定

首先，通过坐标平移，使正方形托盘的中心与求得的重建图像中心重合。

还原图片真实比例，使得图像像素变为符合题意的 256×256。画出最终的还原图像，得出图形在正方形托盘中的位置。

③吸收率的确定

a.定义吸收率

$$\rho=\frac{\sum Q^{'}/n_i}{\sum Q/n}$$

其中：

Q 为大于阈值的像素的灰度值之和

n 为灰度值大于阈值的像素个数

$Q^{'}$ 为某级总灰度值之和

n_i 为某级下灰度值大于阈值的像素个数

上式为我们给出的吸收率定义[3]，根据附件 1 中的吸收率，我们对问题（2）和问题（3）的吸收率进行大胆且有根据得定义。

b.绘制直方图

对所得重建图像进行灰度值提取[4]，绘制直方图，以进行吸收率的阈值的确定和分级。

c.确定阈值

通过直方图的分布情况，可直观地观测出阈值，将阈值以下的出数据归零处

理。

d.吸收率分级

通过直方图中灰度的聚集情况，可对灰度大小进行分级，我们使同一等级下的像素点具有相同的吸收率。

e.确定吸收率

通过不同等级的灰度值以及我们给出的吸收率的定义，确定吸收率。

3、问题（4）模型建立

①精度分析

问题（1）中参数标定的误差主要来源于以下几个方面：

a.计算探测器单元之间的距离 d 误差主要来源于：

圆的半径较小导致圆在探测器上投影所覆盖的单元个数较少

探测器单元离散导致对圆在 X 射线下的投影信息接受不完全（圆在探测器上的投影长度小于圆的半径）。

b.椭圆中心点位置确定不准确，主要误差来源于：

当圆与椭圆在探测器上的投影半覆盖（如图 6）时，椭圆中心点无法准确判断位置。

当圆与椭圆在探测器上的投影不覆盖（如图 7）时，椭圆中心点是利用椭圆在探测器上投影覆盖的探测器单元位置的中点确定，存在误差，导致球心距的计算不准确。

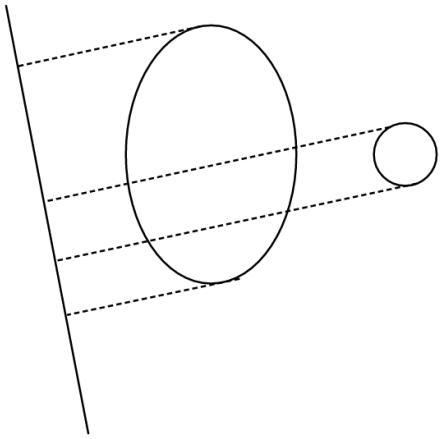


图 5 圆与椭圆在探测器上的投影全覆盖情况示意图

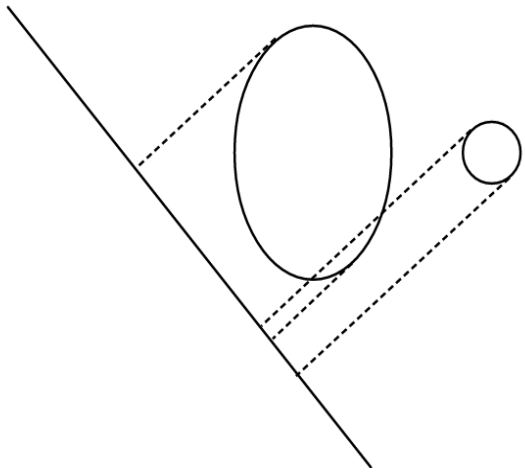


图 6 圆与椭圆在探测器上的投影半覆盖情况示意图

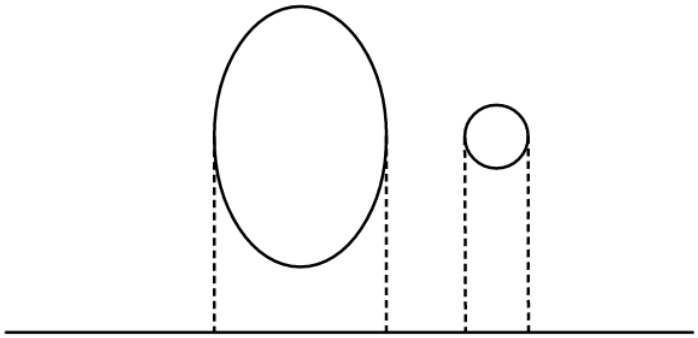


图 7 圆与椭圆在探测器上的投影不覆盖情况示意图

②稳定性分析

在问题（1）中对 180 个 θ 值求解时，非线性方程组无法求解，且在遍历的方法中，结果存在异常值，说明其稳定性较差。

③设计新模板并标定

模板 a 沿用问题（1）中所给模板，确定旋转中心在正方形在正方形托盘中的位置、180 个 θ 值的范围。

模板 b 为每条半径为 60（㎜）的等距曲边三角形（其几何信息如图 8），确定探测器单元之间的距离 d，并利用模板 a 求得的信息，可求 180 个 θ 具体值。

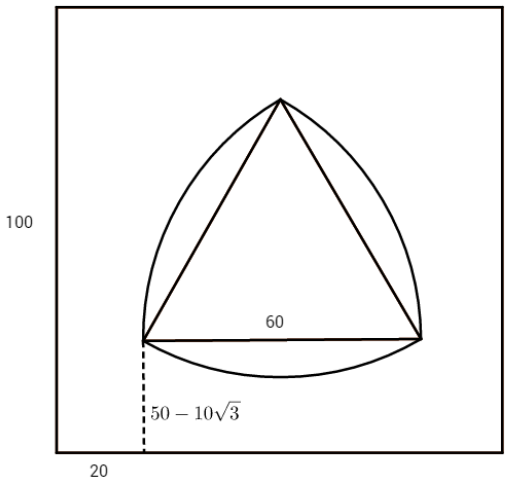


图 8 模板 b 的几何信息图

等距曲边三角形定义及性质：等距曲边三角形的每个顶角到对边的距离相等，并且该距离是等距曲边三角形的最大弦长。考虑到题目中的 CT 机为同一台且正方形托盘的大小不变，为保证模板能完全被 X 射线照射，基于稳健性，将该最大弦长定为 60（mm）。

模板 b 上的等距曲边三角形模板在各 θ 角度平行入射的 X 射线下，投射在探测器上的长度相同，且都等于等距曲边三角形的最大弦长，由半径 R=60mm，以及模板 b 上的等距曲边三角形在探测器上的投影覆盖的探测器单元个数 n，有公式：

$$d=\frac{R}{n}$$

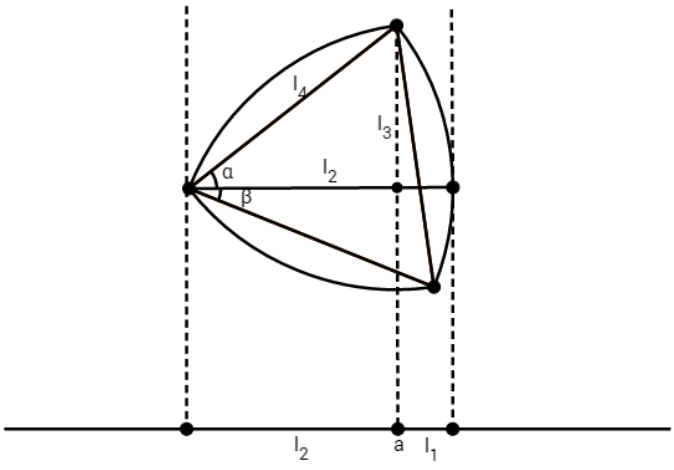


图 9 X 射线与等距曲边三角形相切情况示意图

其中：

$$a:\text{最大吸收信息位置点}$$
$$l_1 \le l_2 (l_1 + l_2 \text{已知})$$
$$\alpha = \arccos l_2 / l_4$$
$$\beta = 60^o - \alpha$$

如图 9，某方向上的 X 射线与等距曲边三角形相切时，相切的两条射线必经过等距三角形的一个或两个顶点，即相切的射线必经过一个顶点，且与该点所对的圆弧相切。

连接顶点与切点，该线段与切线 X 光线垂直，则有 $\cos\alpha = l_2 / l_4$，α 的角度就可以由 l_2 确定。

$l_1 + l_2$ 为顶点与切点所连线段在探测器上的投影长度，l_1 与 l_2 的分界点为最大接受信息在屏上的位置 a，且 $l_1 \le l_2$。则在某方向 X 射线下，l_2 的长度可以唯一确定。

$\beta = 60^o - \alpha$，且当 β 角确定以后，θ 角有如图 10 所示 6 种可能取值。

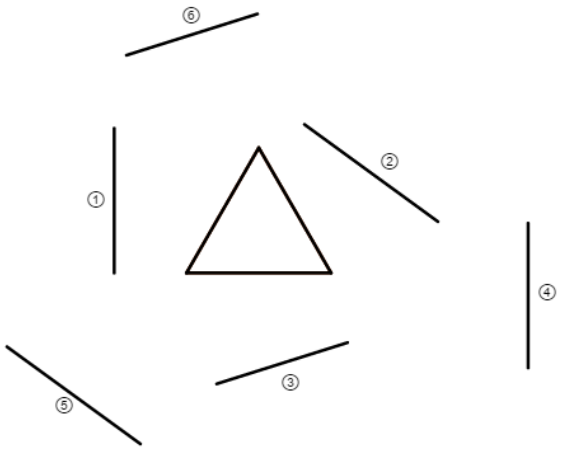


图 10 θ 角 6 种可能取值情况

但由模板 a，以问题（1）的求解中方法确定 θ 角范围后，θ 角可由 β 角唯一确定。

记算法时间复杂度为 $o(n)$。经此模板的改进，可知在确定探测器单元之间的

距离 d 时，圆的半径变大，使得其所对应的探测器单元的个数变大，使相对误差减小。并且改进了模板形状，不采用非线性方程进行求解，而是采用几何方法求解，算法复杂度$o(n)$减小，使得模型的稳定性和普适性提高。

六、模型的求解

1、问题（1）模型求解

①确定探测器单元之间距离的距离

标定模板上的圆形模板在各 θ 角度平行入射的 X 射线下，投射在探测器上的长度相同，且都等于该圆形半径长度（r=4mm）的 2 倍。，以及附件 2 中可分辨且较完全接收到圆形模板信息的探测器个数 n，以及公式：

$$d=\frac{2r}{n}$$

通过对附件 2 中的 180 组数据进行分析和筛选，我们统计了当圆与椭圆在探测器上的投影不覆盖的情况下，所有圆的投影所对应的探测器单元个数，n 取最大值 29，得出$d=0.2759mm$

②旋转中心在正方形托盘中位置的确定

$$l_i=(x_2-x_0)\cos\theta_i+(y_2-y_0)\sin\theta_i-(x_1-x_0)$$
$$(i=1,2,3)$$

我们对附件 2 中数据进行删选和分析，选取了三组数据，为附件 2 中的第 1、137、180 列。

求得的三个旋转角度$\theta_1,\theta_2,\theta_3$分别为 0.5208（29.8383 度）、2.8782（164.9098 度）、3.6436（208.7617 度），及三个l_1,l_2,l_3分别 274.5000 、218.5000 、238.0000 个探测器单元间距。

代入上述公式，可得坐标系单位为探测单元间距的旋转中心坐标为$(x_0,y_0)=(-33.4365,2.2289)$

（$x_1=-290.0228$）

③该 CT 系统使用的 X 射线的 180 个方向的确定

由求得的旋转中心(x_0,y_0)与(x_1,y_1)、(x_2,y_2)，使函数$f(x_0,y_0,x_1)$可转化为函数$f(\theta_i,l_i)$。将附件 2 中 180 组数据反代回公式，并在 matlab 中利用遍历的方法求解 180 个 θ 值。部分结果如表 1，全部结果见附录 1。

表 1 θ值部分结果

旋转序数	弧度制（rad）	角度制（°）	旋转序数	弧度制（rad）	角度制（°）
1	0.5208	29.8392	11	0.6948	39.8120
2	0.5484	31.4214	12	0.7207	41.2936
3	0.5621	32.2049	13	0.7335	42.0305
4	0.5757	32.9836	14	0.7335	42.0305
5	0.6026	34.5276	15	0.7335	42.0305
6	0.6160	35.2933	16	0.7335	42.0305
7	0.6293	36.0551	17	0.7335	42.0305
8	0.6425	36.8132	18	0.7335	42.0305
9	0.6557	37.5678	19	0.7335	42.0305
10	0.6818	39.0670	20	0.7591	43.4972

我们发现 180 个 θ 值中有部分数值粘连，进行处理。部分处理过程如下，全部处理后的结果见附录 2。

表 2 θ值数据处理过程（1）

1.4637	83.8640	θ_1
1.4777	84.6711	θ_2
1.4777	84.6711	θ_3
1.5062	86.3041	θ_4

例如上表，θ_2 和 θ_3 结果在保留四位小数的情况下相等。所以参考与 θ_2 和 θ_3 相邻的 θ_3 和 θ_4，做均匀处理，让 $\theta_2=\theta_1+(\theta_4-\theta_1)/3$，$\theta_3=\theta_4-(\theta_4-\theta_1)/3$，得到 θ_2 和 θ_3 的修正值，结果如下：

表 3 θ值数据处理过程（2）

1.4637	83.8640	θ_1
1.4211	81.4239	θ_2'
1.4921	85.4908	θ_3'
1.5062	86.3041	θ_4

2、问题（2）模型求解

①几何形状的确定

首先根据 CT 断层成像原理的分析，然后运用以卷积定理和中心切片定理为基础的滤波反投影重建算法，选取 ramLak 滤波函数进行图像重建，利用 matlab 程序进行求解。具体 matlab 程序[5]。

求得的问题（2）几何形状（512×512）如图 11。

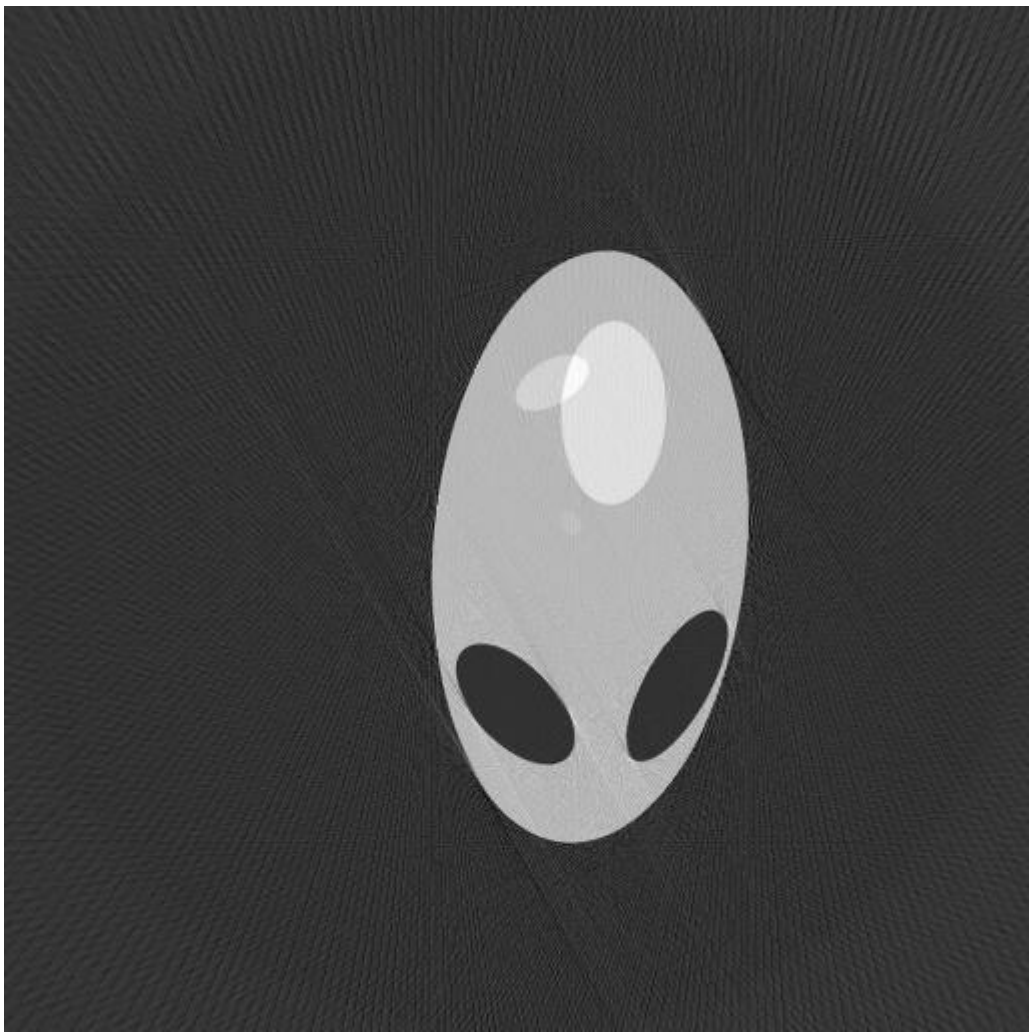


图 11 问题（2）求得的几何形状图

②在正方形托盘中位置的确定

首先，通过坐标平移，使正方形托盘的中心与求得的重建图像中心重合；之后，还原图片真实比例，使得图像像素变为符合题意的 256×256。[6]

最终的还原图像如图 12，得出图形在正方形托盘中的位置。

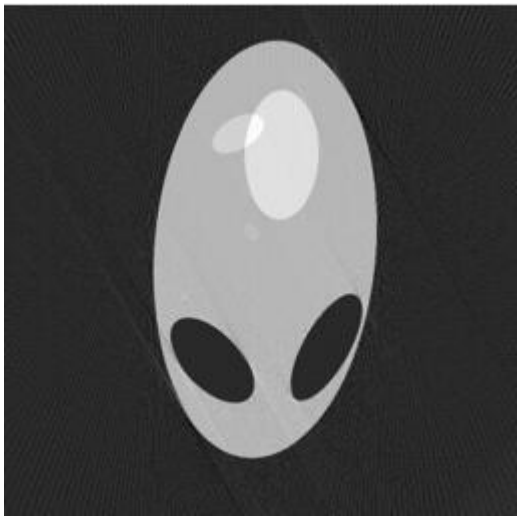


图 12 问题（2）还原图像

③吸收率的确定

a.定义吸收率

$$\rho=\frac{\sum Q^{'}/n_i}{\sum Q/n}$$

其中：

Q 为大于阈值的像素的灰度值之和

n 为灰度值大于阈值的像素个数

$Q^{'}$ 为某级总灰度值之和

n_i 为某级下灰度值大于阈值的像素个数

上式为我们给出的吸收率定义。

b.绘制直方图

对所得重建图像进行灰度值提取，具体结果见材料 1。绘制的直方图如图 13。

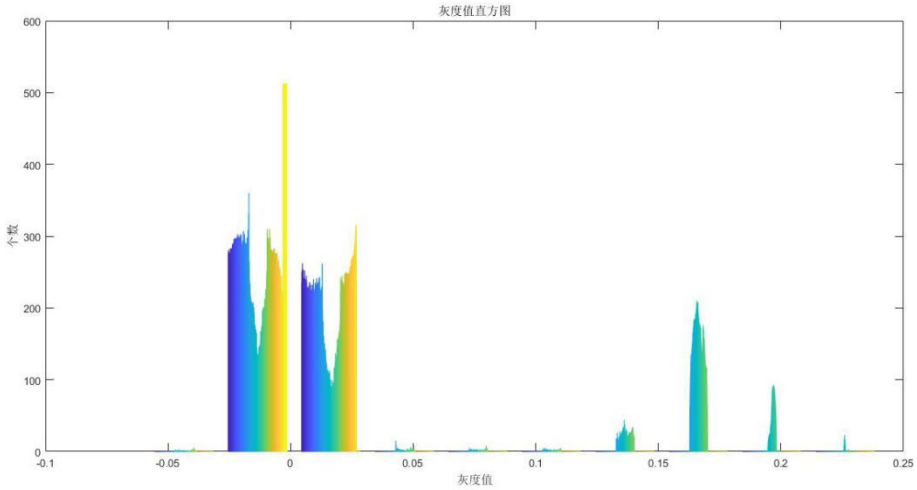


图 13 问题（2）灰度直方图

c.确定阈值

通过直方图的分布情况，可直观地观测出阈值。

d.吸收率分级

通过直方图中灰度的聚集情况，可对灰度大小进行分级，我们使同一等级下的像素点具有相同的吸收率。

e.确定吸收率

计算吸收率，每个级别的吸收率如表 4。10 个点的吸收率如表 5。具体吸收率结果见材料 2。

表 4 问题（2）各级别吸收率

分类	阈值区间	吸收率
1	(0.125, 0.15)	0.8749
2	[0.15, 0.18)	0.9652
3	[0.18, 0.225)	1.2314
4	[0.225, +∞)	1.4285

表 5 问题（2）10 个点吸收率

x	y	吸收率
10.0000	18.0000	0.0000
34.5000	25.0000	0.9652
43.5000	33.0000	0.0000
45.0000	75.5000	1.2314
48.5000	55.5000	0.9652
50.0000	75.5000	1.4285
56.0000	76.5000	1.2314
65.5000	37.0000	0.0000
79.5000	18.0000	0.0000
98.5000	43.5000	0.0000

3、问题（3）模型求解

①几何形状的确定

首先根据 CT 断层成像原理的分析，然后运用以卷积定理和中心切片定理为基础的滤波反投影重建算法，选取 ramLak 滤波函数进行图像重建，利用 matlab 程序进行求解。具体 matlab 程序见附录。

求得的问题（3）几何形状如图 14。

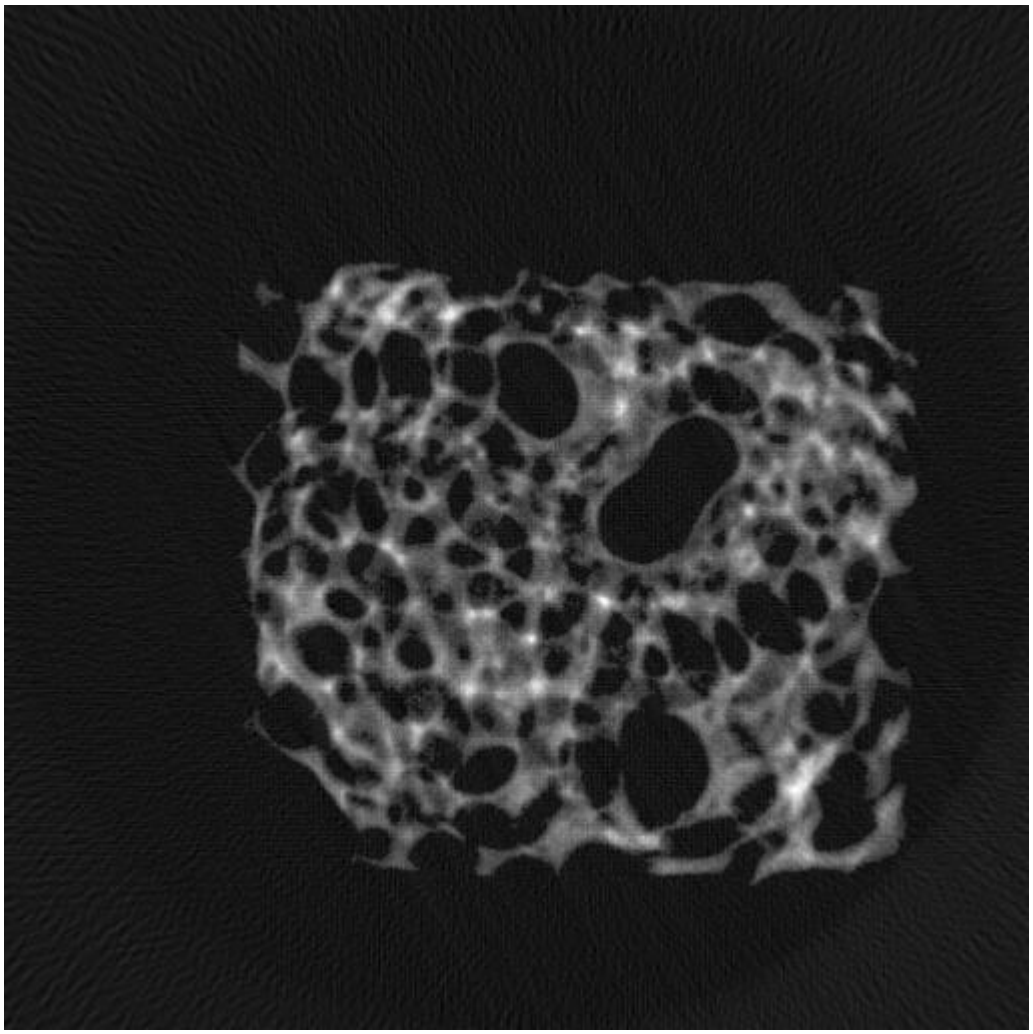


图 14 问题（3）求得的几何形状图

②在正方形托盘中位置的确定

首先，通过坐标平移，使正方形托盘的中心与求得的重建图像中心重合；之后，还原图片真实比例，使得图像像素变为符合题意的 256×256。

最终的还原图像如图 15，得出图形在正方形托盘中的位置。

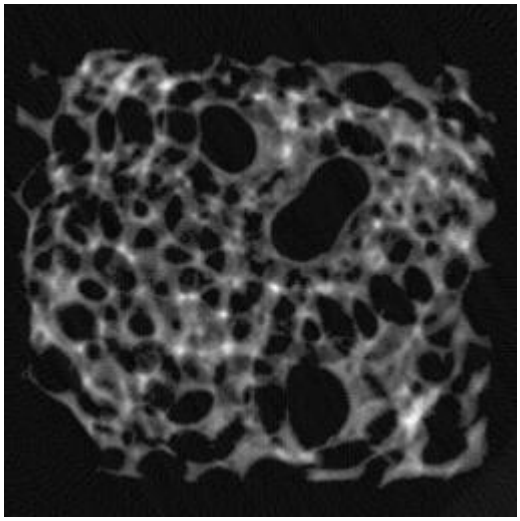


图 15 问题（3）还原图像

③吸收率的确定

a.定义吸收率

b.绘制直方图

对所得重建图像进行灰度值提取，具体结果见材料 3。绘制的直方图如下图 16。

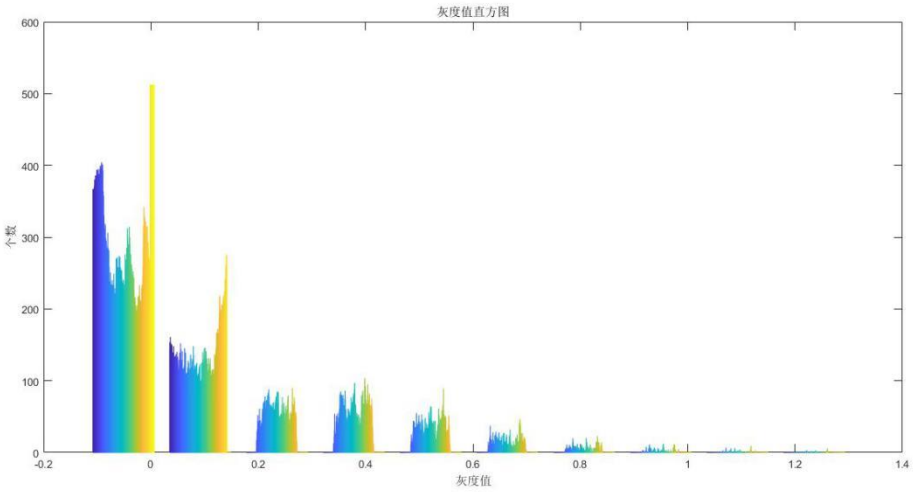


图 16 问题（3）灰度直方图

c.确定阈值

通过直方图的分布情况，可直观地观测出阈值。

d.吸收率分级

通过直方图中灰度的聚集情况，可对灰度大小进行分级，我们使同一等级下的像素点具有相同的吸收率。

e.确定吸收率

计算吸收率，每个级别的吸收率如表 6。10 个点的吸收率如表 7。具体吸收率结果见材料 4。

表 6 问题（3）各级别吸收率

分类	阈值区间	吸收率
1	(0.3,0.5)	0.8557
2	[0.5,0.6)	1.1890
3	[0.6,0.7)	1.4026
4	[0.7,0.9)	2.0598
5	[0.9,1)	2.2723
6	[1,1.17)	2.4618
7	[1.17,+∞)	2.6630

表 7 问题（3）10 个点吸收率

x	y	吸收率
10.0000	18.0000	0.0000
34.5000	25.0000	0.8557
43.5000	33.0000	2.2723
45.0000	75.5000	0.0000
48.5000	55.5000	0.0000
50.0000	75.5000	0.8557
56.0000	76.5000	2.0598
65.5000	37.0000	0.0000
79.5000	18.0000	2.2723
98.5000	43.5000	0.0000

七、模型的评价

1、模型的优点

（1）由于 randon 反变换方法对灰度大小的分层效果不佳，导致求解方法不好，所以我们避免使用该方法，改而采用滤波反投影重建算法。

（2）求解旋转中心时充分利用了数据中的信息，用线性方程组，简单地求解出了旋转中心，结果准确。

（3）通过对吸收率的定义，简化了问题的复杂度，在对重建图像的灰度分布进行分析后，求得的 10 个目标点的吸收率能充分反映这些点的材质的性质。

（4）问题（4）另辟蹊径，构造一个全新的模板后，避免了对角度值的非线性方程的求解或对角度值的遍历。利用等距曲边三角形，用简单的初等方法，可以确定唯一的角度值，具有普适性。

2、模型的缺点

（1）用遍历的方法求解角度时，对遍历的步长要求很高；为去掉异常值，对角度进行了人为控制，对结果的处理有一定的主观性。

（2）对于定义的吸收率，只能反映同一次 CT 扫描下材料的性质，不能反映不同 CT 扫描下的材料的性质，具有一定的局限性。

八、参考文献

[1] 池明辉.锥束工业 CT 系统几何参数误差矫正算法研究[D].重庆:重庆大学, 2015.

[2] 刘明帅.CT 断层图像重建算法研究 [EB/OL].2014[2017-9-17].https://wenku.baidu.com/view/f6d4ec1acf84b9d529ea7a18.html.

[3] 姜启源,谢金星.数学模型[M].第四版.北京:高等教育出版社, 2011 :187-191.

[4] 司守奎,孙兆亮.数学建模算法与应用[M].第 2 版.北京:国防工业出版社,

2015 :342-345.

[5] Mark Bangert.CT reconstruction package [EB/OL].2017[2017-9-16].https://cn.mathworks.com/matlabcentral/fileexchange/34608-ct-reconstruction-package?s_tid=srchtitle.

[6] 佚名.图像频域变换 MATLAB 版 [EB/OL].2015[2017-9-16].https://wenku.baidu.com/view/4414227beff9aef8941e06a5.html.

附录

附录 1

旋转序数	弧度制（rad）	角度制（°）	旋转序数	弧度制（rad）	角度制（°）
1	0.5208	29.8392	91	2.1272	121.8830
2	0.5484	31.4214	92	2.1272	121.8831
3	0.5621	32.2049	93	2.1584	123.6681
4	0.5757	32.9836	94	2.1584	123.6682
5	0.6026	34.5276	95	2.1584	123.6682
6	0.6160	35.2933	96	2.2290	127.7172
7	0.6293	36.0551	97	2.2290	127.7173
8	0.6425	36.8132	98	2.2706	130.1006
9	0.6557	37.5678	99	2.2706	130.1006
10	0.6818	39.0670	100	2.2706	130.1006
11	0.6948	39.8120	101	2.3808	136.4146
12	0.7207	41.2936	102	2.5536	146.3160
13	0.7335	42.0305	103	2.5536	146.3160
14	0.7335	42.0305	104	2.5536	146.3161
15	0.7335	42.0305	105	2.5536	146.3161
16	0.7335	42.0305	106	2.5536	146.3162
17	0.7335	42.0305	107	2.5536	146.3163
18	0.7335	42.0305	108	2.5536	146.3163
19	0.7335	42.0305	109	2.6237	150.3288
20	0.7591	43.4972	110	2.6237	150.3288
21	0.8225	47.1291	111	2.6237	150.3289
22	0.8853	50.7242	112	2.6237	150.3290

23	0.9103	52.1552	113	2.5536	146.3160
24	0.9227	52.8697	114	2.5536	146.3160
25	0.9476	54.2972	115	2.5536	146.3161
26	0.9601	55.0105	116	2.5536	146.3161
27	0.9850	56.4364	117	2.5536	146.3160
28	0.9974	57.1494	118	2.5536	146.3160
29	1.0223	58.5758	119	2.5536	146.3160
30	1.0348	59.2895	120	2.5536	146.3160
31	1.0472	60.0036	121	2.5536	146.3160
32	1.0722	61.4337	122	2.6237	150.3288
33	1.0847	62.1499	123	2.6237	150.3288
34	1.1097	63.5852	124	2.6237	150.3289
35	1.1223	64.3045	125	2.6237	150.3290
36	1.1349	65.0250	126	2.6237	150.3288
37	1.1475	65.7470	127	2.7264	156.2174
38	1.1727	67.1956	128	2.7264	156.2174
39	1.1854	67.9225	129	2.7264	156.2175
40	1.2109	69.3821	130	2.7264	156.2175
41	1.2237	70.1150	131	2.7264	156.2176
42	1.2494	71.5881	132	2.7880	159.7441
43	1.2623	72.3285	133	2.7880	159.7441
44	1.2883	73.8176	134	2.8366	162.5313
45	1.3014	74.5667	135	2.8366	162.5314
46	1.3145	75.3190	136	2.8366	162.5314
47	1.3277	76.0748	137	2.8782	164.9147
48	1.3543	77.5973	138	2.9152	167.0340
49	1.3677	78.3644	139	2.9152	167.0341
50	1.3947	79.9115	140	2.9152	167.0342
51	1.4083	80.6918	141	2.9489	168.9638
52	1.4220	81.4769	142	2.9800	170.7489
53	1.4358	82.2672	143	2.9800	170.7490
54	1.4637	83.8640	144	3.0092	172.4192
55	1.4777	84.6711	145	3.0092	172.4193
56	1.4777	84.6711	146	3.0367	173.9955
57	1.5062	86.3041	147	3.0628	175.4932
58	1.5207	87.1308	148	3.0628	175.4932
59	1.5499	88.8059	149	3.0878	176.9237
60	1.5499	88.8059	150	3.1118	178.2963

61	1.5797	90.5128	151	3.1348	179.6181
62	1.6101	92.2549	152	3.1571	180.8950
63	1.6101	92.2549	153	3.1787	182.1320
64	1.6412	94.0364	154	3.1787	182.1320
65	1.6412	94.0364	155	3.1997	183.3329
66	1.6570	94.9433	156	3.2201	184.5014
67	1.6893	96.7927	157	3.2399	185.6403
68	1.7058	97.7366	158	3.2593	186.7522
69	1.7058	97.7366	159	3.2783	187.8394
70	1.7395	99.6672	160	3.2783	187.8394
71	1.7567	100.6557	161	3.2969	188.9037
72	1.7743	101.6612	162	3.3330	190.9707
73	1.7921	102.6849	163	3.3330	190.9707
74	1.8103	103.7281	164	3.3678	192.9648
75	1.8289	104.7925	165	3.3847	193.9374
76	1.8479	105.8796	166	3.3847	193.9375
77	1.8673	106.9916	167	3.4179	195.8392
78	1.8872	108.1305	168	3.4342	196.7701
79	1.8872	108.1305	169	3.4502	197.6886
80	1.9076	109.2990	170	3.4660	198.5955
81	1.9285	110.4999	171	3.4971	200.3769
82	1.9501	111.7368	172	3.5124	201.2527
83	1.9724	113.0138	173	3.5275	202.1191
84	1.9955	114.3356	174	3.5425	202.9767
85	1.9955	114.3356	175	3.5573	203.8260
86	2.0194	115.7082	176	3.5720	204.6674
87	2.0444	117.1388	177	3.5866	205.5012
88	2.0705	118.6364	178	3.6010	206.3277
89	2.0705	118.6365	179	3.6153	207.1475
90	2.0705	118.6365	180	3.6436	208.7679

附录 2

旋转序数	弧度制（rad）	角度制（°）	旋转序数	弧度制（rad）	角度制（°）
1	0.5208	29.8392	91	2.1272	121.8830
2	0.5484	31.4214	92	2.1272	121.8831
3	0.5621	32.2049	93	2.1584	123.6681
4	0.5757	32.9836	94	2.1584	123.6682
5	0.6026	34.5276	95	2.1584	123.6682
6	0.6160	35.2933	96	2.2290	127.7172
7	0.6293	36.0551	97	2.2290	127.7173
8	0.6425	36.8132	98	2.2670	129.8916
9	0.6557	37.5678	99	2.3049	132.0659
10	0.6818	39.0670	100	2.3429	134.2402
11	0.6948	39.8120	101	2.3808	136.4146
12	0.7207	41.2936	102	2.5536	146.3160
13	0.7335	42.0305	103	2.5536	146.3160
14	0.7335	42.0305	104	2.5536	146.3161
15	0.7335	42.0305	105	2.5536	146.3161
16	0.7335	42.0305	106	2.5536	146.3162
17	0.7335	42.0305	107	2.5536	146.3163
18	0.7335	42.0305	108	2.5536	146.3163
19	0.7335	42.0305	109	2.6237	150.3288
20	0.7591	43.4972	110	2.6237	150.3288
21	0.8225	47.1291	111	2.6237	150.3289
22	0.8853	50.7242	112	2.6237	150.3290
23	0.9103	52.1552	113	2.5536	146.3160
24	0.9227	52.8697	114	2.5536	146.3160
25	0.9476	54.2972	115	2.5536	146.3161
26	0.9601	55.0105	116	2.5536	146.3161
27	0.9850	56.4364	117	2.5536	146.3160
28	0.9974	57.1494	118	2.5536	146.3160
29	1.0223	58.5758	119	2.5536	146.3160
30	1.0348	59.2895	120	2.5536	146.3160
31	1.0472	60.0036	121	2.5536	146.3160
32	1.0722	61.4337	122	2.6237	150.3288
33	1.0847	62.1499	123	2.6237	150.3288
34	1.1097	63.5852	124	2.6237	150.3289
35	1.1223	64.3045	125	2.6237	150.3290

36	1.1349	65.0250	126	2.6237	150.3288
37	1.1475	65.7470	127	2.7264	156.2174
38	1.1727	67.1956	128	2.7264	156.2174
39	1.1854	67.9225	129	2.7264	156.2175
40	1.2109	69.3821	130	2.7264	156.2175
41	1.2237	70.1150	131	2.7264	156.2176
42	1.2494	71.5881	132	2.7880	159.7441
43	1.2623	72.3285	133	2.7880	159.7441
44	1.2883	73.8176	134	2.8366	162.5313
45	1.3014	74.5667	135	2.8366	162.5314
46	1.3145	75.3190	136	2.8366	162.5314
47	1.3277	76.0748	137	2.8782	164.9147
48	1.3543	77.5973	138	2.8959	165.9269
49	1.3677	78.3644	139	2.9136	166.9392
50	1.3947	79.9115	140	2.9312	167.9515
51	1.4083	80.6918	141	2.9489	168.9638
52	1.4220	81.4769	142	2.9800	170.7489
53	1.4358	82.2672	143	2.9800	170.7490
54	1.4637	83.8640	144	3.0092	172.4192
55	1.4211	81.4239	145	3.0092	172.4193
56	1.4921	85.4908	146	3.0367	173.9955
57	1.5062	86.3041	147	3.0537	174.9716
58	1.5207	87.1308	148	3.0708	175.9476
59	1.5499	88.8059	149	3.0878	176.9237
60	1.5499	88.8059	150	3.1118	178.2963
61	1.5797	90.5128	151	3.1348	179.6181
62	1.5952	91.3989	152	3.1571	180.8950
63	1.6106	92.2850	153	3.1713	181.7077
64	1.6261	93.1711	154	3.1855	182.5203
65	1.6416	94.0572	155	3.1997	183.3329
66	1.6570	94.9433	156	3.2201	184.5014
67	1.6893	96.7927	157	3.2399	185.6403
68	1.7058	97.7366	158	3.2593	186.7522
69	1.7058	97.7366	159	3.2719	187.4694
70	1.7395	99.6672	160	3.2844	188.1866
71	1.7567	100.6557	161	3.2969	188.9037
72	1.7743	101.6612	162	3.3205	190.2574
73	1.7921	102.6849	163	3.3441	191.6111

74	1.8103	103.7281	164	3.3678	192.9648
75	1.8289	104.7925	165	3.3845	193.9229
76	1.8479	105.8796	166	3.4012	194.8811
77	1.8673	106.9916	167	3.4179	195.8392
78	1.8807	107.7607	168	3.4342	196.7701
79	1.8941	108.5298	169	3.4502	197.6886
80	1.9076	109.2990	170	3.4660	198.5955
81	1.9285	110.4999	171	3.4971	200.3769
82	1.9501	111.7368	172	3.5124	201.2527
83	1.9724	113.0138	173	3.5275	202.1191
84	1.9955	114.3356	174	3.5425	202.9767
85	1.9955	114.3356	175	3.5573	203.8260
86	2.0194	115.7082	176	3.5720	204.6674
87	2.0444	117.1388	177	3.5866	205.5012
88	2.0705	118.6364	178	3.6010	206.3277
89	2.0705	118.6365	179	3.6153	207.1475
90	2.0705	118.6365	180	3.6436	208.7679

附录 3——问题（1）的 matlab 程序

```
%问题 1 求解未覆盖圆情况下的旋转角度 theta
clear
clc

A=xlsread('fujian_2.xls');

%找出所有数据的两个或四个端点值，并存入数组 M 中
M=zeros(4,180);  %建立空矩阵存放符合要求的数据
for j=1:180
    b=0;
```

```
    for i=2:511
        if
(A(i,j)>0&A(i-1,j)==0)|(A(i,j)>0&A(i+1,j)==0)
           b=b+1;
           M(b,j)=i;
        end
    end
end

%将符合要求的数据（即有四个端点值的数据）所对应的列存入
数组 N 中
N=zeros(180,1);

b=0;
for j=1:180
    if M(1,j)>0&M(2,j)>0&M(3,j)>0&M(4,j)>0
       b=b+1;
       N(b)=j;
    end
end

%统计符合要求数据的个数
s=0;
```

```
for j=1:180;
    if N(j)>0
        s=s+1;
    end
end

%找出圆对应的接收单位个数最多的一组，将其求得的间距作为探测单元之间的距离
max=M(4,N(1))-M(3,N(1));
for i=1:s
    if M(2,N(i))-M(1,N(i))<M(4,N(i))-M(3,N(i))
        if max<M(2,N(i))-M(1,N(i))
            max=M(2,N(i))-M(1,N(i));
        end
    else
        if max<M(4,N(i))-M(3,N(i))
            max=M(4,N(i))-M(3,N(i));
        end
    end
end
D=8/(max+1)          %求出探测单元之间的距离(单位：mm)
```

```
D_zhongxin=45;          %椭圆中心和圆心的距离
jiaodu=zeros(s,1);
for i=1:s

d(i)=abs((M(1,N(i))+M(2,N(i)))/2-(M(3,N(i))+M(
4,N(i)))/2);
    jiaodu(i)=(d(i).*D./D_zhongxin);          %求
出 cos 值
end

%从符合要求数据中选取三组进行求解
ss=[1  42  85];

real(jiaodu);
theta1=acos(jiaodu(ss(1)))
theta2=pi-acos(jiaodu(ss(2)))
theta3=pi+acos(jiaodu(ss(3)))

%确定椭圆中心对应的探测器上的位置
if
M(2,N(ss(1)))-M(1,N(ss(1)))<M(4,N(ss(1)))-M(3,
N(ss(1)))
   l1=(M(3,N(ss(1)))+M(4,N(ss(1))))/2
```

```
else
   l1=(M(1,N(ss(1)))+M(2,N(ss(1))))/2
end

if
M(2,N(ss(2)))-M(1,N(ss(2)))<M(4,N(ss(2)))-M(3,
N(ss(2)))
   l2=(M(3,N(ss(2)))+M(4,N(ss(2))))/2
else
   l2=(M(1,N(ss(2)))+M(2,N(ss(2))))/2
end

if
M(2,N(ss(3)))-M(1,N(ss(3)))<M(4,N(ss(3)))-M(3,
N(ss(3)))
   l3=(M(3,N(ss(3)))+M(4,N(ss(3))))/2
else
   l3=(M(1,N(ss(3)))+M(2,N(ss(3))))/2
end

%先运行文件 juli.m
%问题 1CT 系统旋转中心的求解
%设旋转中心为(x0,y0),接收器的左端点为(x1,y1) ， y1 在
```

列方程中消去

```
%设椭圆的中心为原点，即（x2,y2）=(0,0)

x2=0;

y2=0;

A=[l1,l2,l3]';

B=[1-cos(theta1) -sin(theta1) -1;

   1-cos(theta2) -sin(theta2) -1;

   1-cos(theta3) -sin(theta3) -1];

X=B\A


%先运行文件 juli.m 和 zhongxin.m

%遍历 belta，使得

abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-

X(1)) 尽可能的小

%求解 180 个方向，求解时间约为 5mins

l=zeros(180,1);

for i=1:180

    if M(2,i)-M(1,i)<M(4,i)-M(3,i)

        l(i)=(M(4,i)+M(3,i))/2;
```

```
    else
        l(i)=(M(2,i)+M(1,i))/2;
    end
end
l=l;

theta_180=zeros(180,1);

min=zeros(180,1);
jiao=zeros(180,1);
%对 belta 进行分段求解，并进行控制，去除异常值。因为逆时针旋转，前面的 belta 比后面的 belta 值小

alpha=0;
for i=1:49
    min(i,1)=10000000;
     for belta=0:0.0000001:1.2*pi;  %1.2*pi 是由最后一个角度的 belta 值确定，减少遍历次数
         if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-X(1))<min(i)&belta>alpha  %控制使得后一个 belta>前一个 belta
```

```
min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
            jiao(i)=belta;
            alpha=belta;
        end
    end
 end

alpha=0;

for i=49:51
   min(i,1)=10000000;
   for belta=0:0.000001:1.2*pi;
       if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha&belta<3 %根据 i=1:49 时
的 belta 值，控制 belta 的大小，去除异常值

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
           jiao(i)=belta;
           alpha=belta;
```

```
        end
    end
end

alpha=0;

for i=49:51
    min(i,1)=10000000;
    for belta=0:0.000001:1.2*pi;
        if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha&belta<3

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
            jiao(i)=belta;
            alpha=belta;
        end
    end
end

alpha=0;
for i=52:53
```

```
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
      if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta<3

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
          jiao(i)=belta;
         alpha=belta;
      end
  end
end

alpha=0;
for i=54:60
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
      if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta<3
```

```
min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
          jiao(i)=belta;
         alpha=belta;
      end
  end
end

%54--60
alpha=0;
for i=61:87
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
      if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta<3

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
          jiao(i)=belta;
          alpha=belta;
```

```
      end
  end
end
%i=61:87

alpha=0;

for i=88:97
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
      if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta<2.2707&belta>alpha %对 belta
值进行控制，保证后一个 belta>前一个 belta

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
          jiao(i)=belta;
         alpha=belta;
      end
  end
end
```

```
alpha=0;

for i=98
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
      if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta<3

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
          jiao(i)=belta;
         alpha=belta;
      end
  end
end

alpha=0;

for i=99
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
```

```
      if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
          jiao(i)=belta;
         alpha=belta;
      end
 end
end
%i=99

alpha=0;
for i=100:112
min(i,1)=10000000;
for belta=0:0.000001:1.2*pi;
    if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
```

```
)+X(3)-X(1));
        jiao(i)=belta;
        alpha=belta;
    end
end
end
%100:112

alpha=0;
for i=113:116
min(i,1)=10000000;
for belta=0:0.000001:1.2*pi;
     if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha&belta<2.56%对 belta 值
进行控制，保证后一个 belta>前一个 belta

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
        jiao(i)=belta;
        alpha=belta;
     end
```

```
 end
end
%113:116

i=117;
jiao(i)=1/2*(jiao(i-1)+jiao(i+1));  %i=117 时，
值异常。用 118 和 119 的值的平均来代替 i=117 时的 belta

alpha=0;
for i=118:119
min(i,1)=10000000;
 for belta=0:0.000001:1.2*pi;
     if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha&belta<2.60&belta>2.55

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
        jiao(i)=belta;
         alpha=belta;
     end
 end
end
```

```
%118:119


alpha=0;
for i=120:125
 min(i,1)=10000000;
 for belta=0:0.000001:1.2*pi;
     if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
        jiao(i)=belta;
         alpha=belta;
     end
 end
end

%120:125
alpha=0;
```

```
for i=126:180
    min(i,1)=10000000;
    for belta=0:0.000001:1.2*pi;
        if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
            jiao(i)=belta;
            alpha=belta;
        end
   end
end

%147-180

alpha=0;

for i=126:146
  min(i,1)=10000000;
  for belta=0:0.000001:1.2*pi;
      if
```

```
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
X(1))<min(i)&belta>alpha

min(i)=abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta
)+X(3)-X(1));
         jiao(i)=belta;
          alpha=belta;
      end
  end
end
min;

%126:146,覆盖之前所求i=126:147时,所求得的异常belta
值



%E=zeros(90,3);
%for i=91:180
 %    fa=0.0001;
 %   for belta=0:0.000001:pi;
 %       if
abs(l(i)+X(1)*cos(belta)+X(2)*sin(belta)+X(3)-
```

```
X(1))<fa&belta>2.1
 %       if E(i-89,1)>0
 %           if E(i-89,2)>0
 %               E(i-89,3)=belta;
 %           else
 %               E(i-89,2)=belta;
 %           end
 %       else
 %           E(i-89,1)=belta;
 %       end
 %       end
 %   end
%end
```

附录 4——问题（2）（3）的 matlab 程序

```
%CT 图像重建主程序
clear
clc

sinogram=xlsread('fujian_3.xls');  %将附件的接收信息作为 radon 变化的 sinogram 值/'fujian_5.xls'
theta=xlsread('Angles.xls','B1:B180');  %第一问
```

```
求得的角度
%theta=29.8391:208.8391;
%theta=theta';

%滤波降噪下的 CT 重建图像
FilteredBackprojection(sinogram,theta)

%问题 2、3，进行图像重建/反投影 （滤波降噪）
%统计不同灰度值的个数
%函数所需的参数为 sinogram（用的是接收信息，即假设接收信息与物体厚度成正比）
%另个一个参数为 theta(theta 为角度制，为第一问所求)

function [finalct] =
FilteredBackprojection(sinogram,theta);

theta = (pi/180)*theta;          %将角度制转化为弧度制

%设定 CT 重建图像的大小，注：像素 512×512 是因为与探测器的个数以及接收信息行数相同
%同时，要注意长度单位的转化。这里的 512 应该指探测单元间的间距,而不是像素
CT = zeros(512,512);
```

```
zhongxindian = 257;      %将图像的中心点作为旋转中心，图像重建后再进行坐标变换，得到具体位置
[x,y] = meshgrid(-256:255);      %生成坐标网格

%选择合适的滤波器对 CT 图像进行滤波降噪（用爬坡滤波器'ramLak'或者'sheppLogan'）
%这里选择爬坡滤波器 ramLak
halfFilterSize = 513;
filter = zeros(1,halfFilterSize);
filter(1:2:halfFilterSize) =
-1./([1:2:halfFilterSize].^2 * pi^2);
filter = [fliplr(filter) 1/4 filter];

%开始迭代重建，包含滤波降噪的过程
for i=1:180
    %确定旋转过程中的坐标网格大小，并且限制其不超出范围
    xuanzhuan = round(zhongxindian +
x*sin(theta(i))+y*cos(theta(i)));
    xianzhi =
find((xuanzhuan>0)&(xuanzhuan<=512));
    quanxin = xuanzhuan(xianzhi);

    %滤波降噪（卷积）
```

```
    juanji = conv(sinogram(:,i),filter,'same');

    CT(xianzhi) = CT(xianzhi) +
juanji(quanxin)./180;
end

%将图片逆时针旋转 90 度得重建图像（MATLAB 中图像处理中的
坐标系与习惯坐标系有区别）
%注意该图像只能反映介质的形状大小，不能反映具体的位置
ct = zeros(512,512);
for i = 1:512
    for j = 1:512
        m = -j+513;
        n = i;
        ct(m,n) = CT(i,j);
    end
end

%将图像进行平移，得到最终的图像位置
x0=ceil(-33.4365);
y0=ceil(22.2896);
finalct=zeros(512,512);

for j=1:512
```

```
    if j<512+x0
        finalct(:,j)=ct(:,j-x0);
    else
        finalct(:,j)=0;
    end
end

for i=1:512
    if i<512-y0
        finalct(i,:)=finalct(i+y0,:);
    else
        finalct(i,:)=0;
    end
end

%显示 CT 重建的灰度图像
subplot(1,3,1)
imshow(ct,[])
xlabel('X')
ylabel('Y')
title('滤波降噪后的 CT 重建图像')
```

```
%显示平移后的最终图像（形状、大小，不能反映在托盘的位置）
subplot(1,3,2)
imshow(finalct,[])
xlabel('X')
ylabel('Y')
title('平移后的最终图像')

%画出不同灰度值的直方图，为下面的吸收率分级做铺垫
subplot(1,3,3)
hist(finalct)
xlabel('灰度值')
ylabel('个数')
title('灰度值直方图')

 %求每个图的相对吸收率
% ‘finalct.xlsx’和'finalct3.xlsx'文件是
chongjian.m 所得的灰度图像 ct 的矩阵
%阈值的划分从 chonjian.m 的灰度值直方图得出
clear
clc

%还原出重建图像的大小、形状、以及在托盘中的相对位置
D=0.2759;
```

```
finalct=xlsread('finalct_2.xlsx');      %问题 3 用
'finalct_3.xlsx'

x0=ceil((512-100/D)/2);
y0=ceil(((512-100/D)/2));
zuizhong=zeros(512,512);

for j=1:512
    if j<512-x0
        zuizhong(:,j)=finalct(:,j+x0);
    else
        zuizhong(:,j)=0;
    end
end

for i=1:512
    if i>y0
        zuizhong(513-i+y0,:)=zuizhong(513-i,:);
    else
        zuizhong(513-i,:)=0;
    end
end
```

```
ct=zeros(ceil(100/D),ceil(100/D));
for i=1:ceil(100/D)
    for j=1:ceil(100/D)

ct(ceil(100/D)+1-i,j)=zuizhong(513-i,j);
    end
end
ct=imresize(ct,[256,256]);
imshow(ct,[])

%吸收率分级
s=zeros(4,1);                    %灰度值求和
num=zeros(4,1);                  %统计个数
ave=zeros(4,1);
fenji=zeros(256,256);
for i=1:256
    for j=1:256
        if ct(i,j)>0.125&ct(i,j)<0.15
            s(1)=s(1)+ct(i,j);
            num(1)=num(1)+1;
            fenji(i,j)=1;
```

```
        elseif ct(i,j)>=0.15&ct(i,j)<0.18
            s(2)=s(2)+ct(i,j);
            num(2)=num(2)+1;
            fenji(i,j)=2;
        elseif ct(i,j)>=0.18&ct(i,j)<0.225
            s(3)=s(3)+ct(i,j);
            num(3)=num(3)+1;
            fenji(i,j)=3;
        elseif ct(i,j)>=0.225
            s(4)=s(4)+ct(i,j);
            num(4)=num(4)+1;
            fenji(i,j)=4;
        end
    end
end

for i=1:4
    ave(i)=s(i)/num(i);
end
zong=(s(1)+s(2)+s(3)+s(4))/(num(1)+num(2)+num(
3)+num(4));

xishoulv=zeros(4,1);
```

```
for i=1:4
    xishoulv(i)=ave(i)/zong;
end
xishoulv

for i=1:256
    for j=1:256
        if fenji(i,j)==1
            fenji(i,j)=xishoulv(1);
        elseif fenji(i,j)==2
            fenji(i,j)=xishoulv(2);
        elseif fenji(i,j)==3
            fenji(i,j)=xishoulv(3);
        elseif fenji(i,j)==4
            fenji(i,j)=xishoulv(4);
        end
    end
end

 %先运行 wenti2.m 或者 wenti3.m
%求出十个坐标对应的行和列
D=0.2759;
```

```
tenCoords=xlsread('fujian_4.xls');
tenRowColumn=zeros(10,2);
for i=1:10

tenRowColumn(i,1)=ceil(257-tenCoords(i,2)/(100
/256));

tenRowColumn(i,2)=ceil(tenCoords(i,1)/(100/256
)+1);
end
tenRowColumn
tenxishoulv=zeros(10,1);
for i=1:10

tenxishoulv(i)=fenji(tenRowColumn(i,1),tenRowC
olumn(i,2));
end
tenxishoulv
```