

基于新型 FBP 算法的快速 CT 系统标定与成像

摘要

电子计算机断层扫描 (Computed Tomography, CT) 系统利用样品对射线能量的吸收特性, 从多个角度进行断层成像, 经过重建获取样品内部的结构信息。对于一个参数未知的 CT 系统, 本文首先借助于已知结构的样品 (称为模板), 利用投影原理对其进行标定, 接着使用滤波反投影、傅里叶变换方法建立了完整的数学模型, 并据此对未知结构的样本进行成像。

对于问题一: 首先, 利用多次差分运算和多个修正线性单元 (Rectified Linear Unit, ReLU) 提取断层成像的边缘信息, 采用局部预测重排算法将分割标定模板的投影信息, 根据投影原理算得探测器阵列的间距为 0.2844mm, 并进一步求出 180 次投影的角度。接着, 以探测器阵列中心为原点建立新坐标系, 将发射-接收系统的旋转转化为标定模板的旋转, 使用三角函数拟合得到 2 个固体中心和旋转中心间的距离, 通过多方法综合分析, 最终求得旋转中心在正方形托盘中的位置为 (39.1272, 56.3769)。

对于问题二: 首先, 平行入射 X 射线检测未知介质的过程与 Radon 变换很相似, 但未知介质的旋转中心不一定是该介质的几何中心。未知介质的吸收信息实际上是未知介质各点吸收率在一个平面内沿不同的直线对做线积分。这种积分变换的逆变换可得到未知介质上任意一点的吸收率。相关重建算法有先滤波再反投影、先反投影后滤波、希尔伯特变换再反投影等, 本模型初步采用傅里叶变换进行斜坡滤波, 对滤波后的数据施行反投影就可得到未知介质在正方形托盘中的相关信息。

对于问题三: 由于提供的未知介质的吸收信息更加复杂, 根据傅里叶变换理论, 在 ω 域中做乘法等价于在 s 域中做卷积。滤波器的设计会影响重建质量。为了消除直接反投影造成的图像模糊, 要用 Ramp 滤波函数作滤波处理。本文比较了几种常用滤波器, 最终参考 Ramp-Lak 滤波器设计了一种新的滤波器, 实现较为简单, 实验效果较好。采用滤波反投影 (Filter Back Projection, FBP) 算法得到未知介质的相关信息。

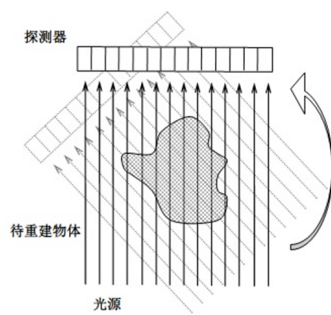
对于问题四: 首先, 根据求解问题一的流程, 筛选出探测器单元间距、投影角度、三角函数拟合结果这三个受误差影响较大的关键变量, 并据此有针对性地提出了五点基本的设计原则。根据这些原则, 设计出由 1 个大圆、4 个大小不一致的小圆组成的新标定模板。使用正态分布函数模拟产生批量数据, 对新、旧模型均进行多次标定测试, 通过比较分析计算结果与真实数据间的误差, 证明新模板提高了标定的精度和稳定性。

关键字: 计算机断层成像; 滤波反投影; Radon 逆变换; 傅里叶变换; Ramp-Lak 滤波器

一、问题重述

1.1 问题背景

电子计算机断层扫描 (Computed Tomography, CT) 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进行断层成像, 由此获取样品内部的结构信息, CT 具有扫描时间快, 图像清晰等特点, 可用于金属探伤和多种疾病的检查。如图 1(a) (b) 所示, 典型的 CT 系统包括产生平行 X 射线的发射器, 以及与射线方向垂直的探测器阵列。在成像过程中, 整个系统旋转至各个角度检测介质吸收衰减后的射线能量, 经过增益等处理后得到吸收信息, 由此可以对未知结构的样品进行重建。



(a) 原理示意图



(b) 实物示意图

图 1 CT 系统示意图

1.2 问题的提出

CT 系统在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。

对于一个参数未知的二维 CT 系统, 需要建立相应的数学模型和算法, 解决以下问题:

- (1) 对于在正方形托盘上放置的由两个均匀固体介质组成的标定模板, 根据它的几何性质及其接收信息, 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) 对于由吸收率分布较为均匀、几何形状规则的样本经上述 CT 系统成像得到的特征明显的接收信息, 利用问题 (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。
- (3) 对于由吸收率分布散乱、几何形状不规则的固体介质样本经上述 CT 系统成像得到的没有明显特征信息的数据, 设计算法进行图形重建, 得出吸收率的分布信息。
- (4) 对问题 (1) 中参数标定的精度和稳定性进行数值分析。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。

二、问题分析

本题以 CT 系统的标定与成像为背景，实质上是利用一维投影信息的二维重建问题。

2.1 问题一分析

问题一要求根据标定模板的几何性质、探测器在各个角度的接收信息，确定探测器单元之间的距离、X 射线的 180 个入射方向以及 CT 系统旋转中心在正方形托盘中的位置。

首先，在两个固体介质分别是标准椭圆、圆的假设下，将标定模板中 2 个均匀固体介质的投影信息分割开来，由圆形固体介质的直径和投影宽度（以覆盖的探测器个数计），计算探测器单元的间距；接着，利用椭圆与圆中心对称的特性，确定两固体介质中心坐标，根据其连线的投影长度与实际长度之比，求出 180 次投影的角度；然后，以探测器阵列中心为原点建立新坐标系，将发射-接收系统的逆时针旋转转化为标定模板的顺时针旋转，利用先前计算出的投影方向，分别对固体介质中心的极坐标参数进行三角函数拟合，得出它们的旋转半径；最后，通过多方法综合分析，确定旋转中心在正方形托盘中的位置。

2.2 问题二分析

问题二要求利用问题一中得到的标定参数和附件 3 中的提供的未知介质的吸收信息，确定该未知介质在正方形托盘中的坐标位置、几何形状和吸收率等信息。此外，还需要给出 10 个指定位置的吸收率。

将附件 3 提供的未知介质吸收信息可视化，结合问题一的结论，观察所得可视化图像，初步分析可知，未知介质中可能包含一个较大的椭圆，除此以外，还包含多个不规则的小几何形状。

未知介质的吸收信息实际上是未知介质各点吸收率的一种积分变换，抽象地，该变换将二维平面函数 f 变换成一个定义在二维空间上的一个线性函数 Rf ，在具体的使用中表现为未知介质在一个平面内沿不同的直线（直线与原点的距离为 d ，方向角为 α ）对 f 做线积分。自然地，我们考虑这种积分变换的逆变换，即找到未知介质上某一点 p 在不同角度 α 时在探测器上的投影点位置 $p(\alpha)$ ，将 $p(\alpha)$ 点的吸收信息累加后平均，即可得到未知介质上某一点 p 的吸收率，同理可算得未知介质上所有点的吸收率。

2.3 问题三分析

问题三要求利用附件 5 提供的 CT 系统的另一个未知介质的吸收信息和问题一得到的标定参数，给出该未知介质的位置、几何形状和吸收率等信息，并给出 10 个指定位置的吸收率。

将附件 5 提供的未知介质吸收信息可视化，发现附件 5 的未知介质吸收信息非常复杂，仅凭肉眼很难观察得到初步结论。考虑现有的 CT 图像重建算法，CT 图象重建有四种基本的算法：矩阵法、迭代法、傅立叶算法、反投影算法 [2]。反投影算法包括平行光束直接反投影法和滤波反投影算法，也包含等角扇形光投影的重建算法和平行光束投影

的滤波反投影算法，本文所使用的模型主要是基于平行光束投影的 FBP 算法。滤波反投影由于是在反投影的基础上先进行卷积运算，因而具有更好的效果。

采用滤波反投影重建图像的具体过程是，先把由检测器上获得的原始数据与一个滤波函数进行卷积运算，得到各方向卷积的投影函数；然后再把它们从各方向进行反投影，即按其原路径平均分配到每一矩阵元上，进行叠加后得到每一矩阵元的 CT 值；再经过适当处理后就可以得到被扫描物体的断层图像，滤波反投影可消除单纯的反投影产生的边缘失锐效应，补偿投影中的高频成分和降低投影中心密度，并保证重建图像边缘清晰和内部分布均匀。我们采用滤波反投影法解决复杂未知介质的重建问题，为问题三提供解决方案。

2.4 问题四分析

问题四要求分析问题一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。

注意到问题一的求解过程是环环相扣的，每一步求解的准确性都依赖于上一步的计算结果，因此误差会在逐级的传递中被不断放大。首先，对问题一的求解流程进行分析，筛选出探测器单元间距、投影角度、三角函数拟合结果这三个较容易引进误差的关键变量。接着，针对这些变量优化模板的设计方案，提出圆形介质半径尽可能大、几何形状中心对称、投影特征易识别、介质之间相距尽可能远、物体数目尽可能少这五项设计原则，由此设计出新的标定模板。

设计出新的标定模板后，模拟产生大量不同的初始条件，然后分别按照新、旧模板生成多组吸收信息，用于标定分析。分析两份模板在探测器单元间距、投影角度、旋转中心计算结果上误差的大小，由此可以证明新的标定模板提高了标定的精度和稳定性。

三、模型假设

1. 入射 X 射线为沿直线传播的平行光，且干涉、衍射、反射、折射现象均不显著，可以忽略。
2. 探测器阵列与样本位于同一平面，与样本距离充分远，不会发生接触或碰撞。
3. 探测器单元具有一定宽度，在宽度范围内发生衰减的 X 射线都会被检测出。
4. 每个探测器采集到的吸收信息仅与穿透它的 X 射线所经过介质的吸收率有关。
5. 空气对 X 射线的吸收极少，吸收率可视为 0。

四、主要符号说明

表 1 列举了本文中主要使用的符号以及含义。

表 1 符号含义说明表

符号	意义	单位
θ	X 射线入射方向与 x 轴负方向夹角	$^{\circ}$
A	样本对 X 射线吸收程度	
A'	探测器实际接收信息	
d_i	第 i 个探测器的实际接收信息, $i = 1, 2, \dots, 512$	
I	X 射线光照强度	lux
I_0	入射 X 射线初始光照强度	lux
$\mu(\cdot)$	物质对 X 射线吸收系数	m^{-1}
w	物体在探测器阵列上投影宽度	mm
$ReLU(x)$	修正线性单元函数	
$edge_{i,j}$	第 i 个角度, 第 j 个边界坐标 ($1 \leq i \leq 180, 1 \leq j \leq 4$)	
$target_{i,j}$	$edge_{i,j}$ 的预测值	
n	投影覆盖的探测器个数	
r	探测器半径	mm
D	圆形固体介质的真实半径	mm
l	探测器单元间距	mm
(x, y)	平面直角坐标系中的坐标	
y'	探测器阵列坐标系中的坐标	
r_0	旋转中心与旋转点之间的间距	mm
b	旋转中心投影坐标	
$f(\cdot)$	平行光束吸收函数	
I_0	初始光照强度	lux
I_1	照射后光照强度	lux
R	Radon 变换算子	
$\mathbf{p}(\mathbf{s}, \theta)$	投影数据	
$\mathbf{P}(\omega, \theta)$	一维傅里叶变换结果	
$\mathbf{q}(\mathbf{s}, \theta)$	滤波后的数据	
$h(k)$	滤波器函数	
A_i	指定十个点的吸收率, ($i = 1, 2, \dots, 10$)	
π	对 $edge_{i,1:4}$ 的一种排列方法	
π^*	对 $edge_{i,1:4}$ 的最佳排列方法	
z	X 射线射入物体的深度	m
a_0	传感器坐标系下, 旋转点相对于旋转中心的初始方位角	$^{\circ}$

五、模型建立和求解

5.1 模型一：未知 CT 系统参数标定模型

5.1.1 模型简介

本模型独立地看待每个角度下探测器阵列采集到的吸收信息，利用多次差分运算和多个修正线性单元 (Rectified Linear Unit, ReLU) 提取断层成像的边缘信息。考虑到发射-接收系统始终沿同一方向转动，且采样角度间隔不大，使用局部预测算法对两固体介质投影的始、末位置进行归纳，分割出两固体介质的投影信息。在根据投影定理算得探测器间距与采样角度后，通过坐标系变换考虑样本相对发射-接收装置的旋转，对两固体介质中心的投影位置进行三角函数拟合，得到它们的旋转半径，以此求解旋转中心位置。

5.1.2 参照系与入射角定义

本文采用如图 2 所示的平面直角坐标系与如图 3 所示的探测器阵列坐标系。对于探测器阵列坐标系，利用 X 射线探测器等间距排列的特点，将探测器的编号作为其对应的坐标，建立一维坐标系。此外，将入射 X 射线的角度定义为入射方向与坐标系 x 轴负方向的夹角。

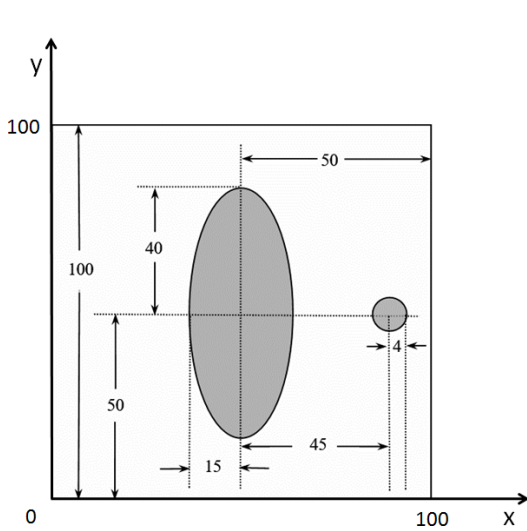


图 2 平面直角坐标系

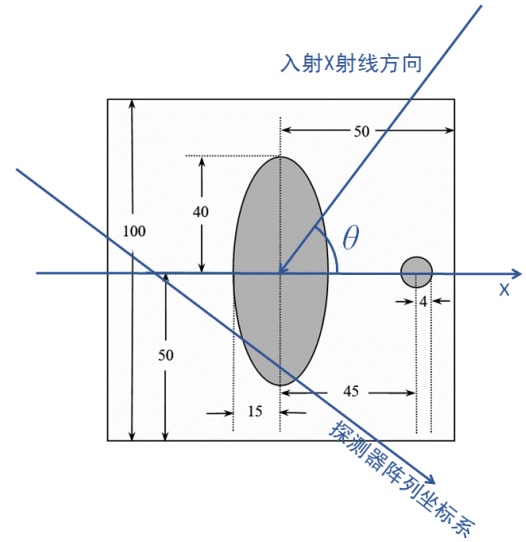


图 3 入射角定义

5.1.3 投影模型

首先，引入 X 射线在物质中的衰减定理。

当 X 射线射入物质时，由于发生光电效应、康普顿-吴有训效应、电子对效应，其光照强度随入射深度的增加而减弱，这种现象称为 X 射线衰减，即 X 射线被物质所吸收。X 射线被物质吸收遵循负指数衰减定律 [1]：

$$I = I_0 e^{-\mu z} \quad (1)$$

其中： I_0 为入射 X 射线光照强度， μ 为物质对 X 射线的吸收系数， z 为 X 射线射入物体的深度， I 为入射 z 深度处的 X 射线光照强度。

据此，本文定义样本对任意 X 射线的吸收程度 (Absorption Degree) A 为：

$$A = \int_{\vec{l}} \mu(x, y) ds, \quad (2)$$

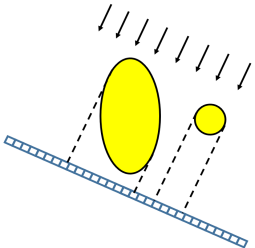
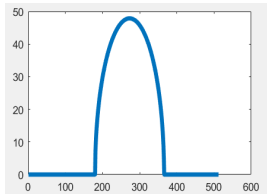
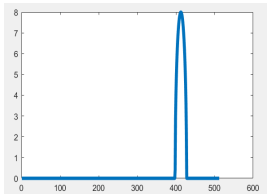
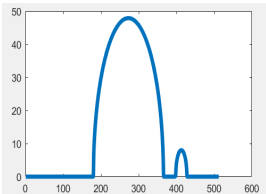
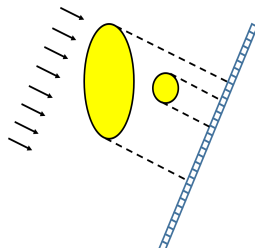
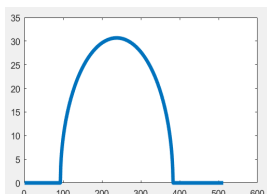
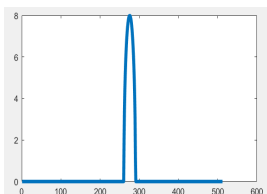
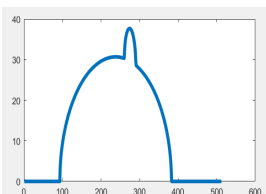
并将 (1) 式改写为：

$$I = I_0 e^{-A}, A = -\ln \frac{I}{I_0}, \quad (3)$$

其中， \vec{l} 为 X 射线穿透样本的长度， $\mu(x, y)$ 是位于坐标 (x, y) 处样本的吸收强度。

按照此模型，对于不同的投影角度，椭圆固体介质和圆形固体介质都会在接收平面上形成轴对称的单峰函数。如表 2 所示，投影曲线的横坐标是 X 射线探测器的编号，而纵坐标就是该探测器接收到的吸收强度。对于给定的投影方向，两固体介质的投影信息在探测器表面线性相加，得到实际的投影信息。由于两个固体介质对应的单峰函数在峰值、宽度方面存在显著的差异，根据合成图像可以直接分割出两个单峰函数的始、末位置。

表 2 不同方向的投影结果

投影方向	椭圆介质投影	圆形介质投影	实际投影信息
			
			

此外，经过观察表 2，还注意到投影模型具有以下特性：

- 圆形固体介质的投影宽度与投影方向无关，可以作为标定探测器间距的依据。
- 由于图形中心对称的性质，两固体介质几何中心的投影位置恰好是其对应单峰函数的对称轴（同时也是峰值位置）。

为了验证这一模型，使用题目所给数据 2 与之进行比对，从附件 2 中选取第 52、第 128 个投影角度对应的吸收数据，作出折线图，如图 4 (a) (b) (c) 所示。不难发现，探测

器得到的接收信息同样可以视为两个单峰函数相加的结果。

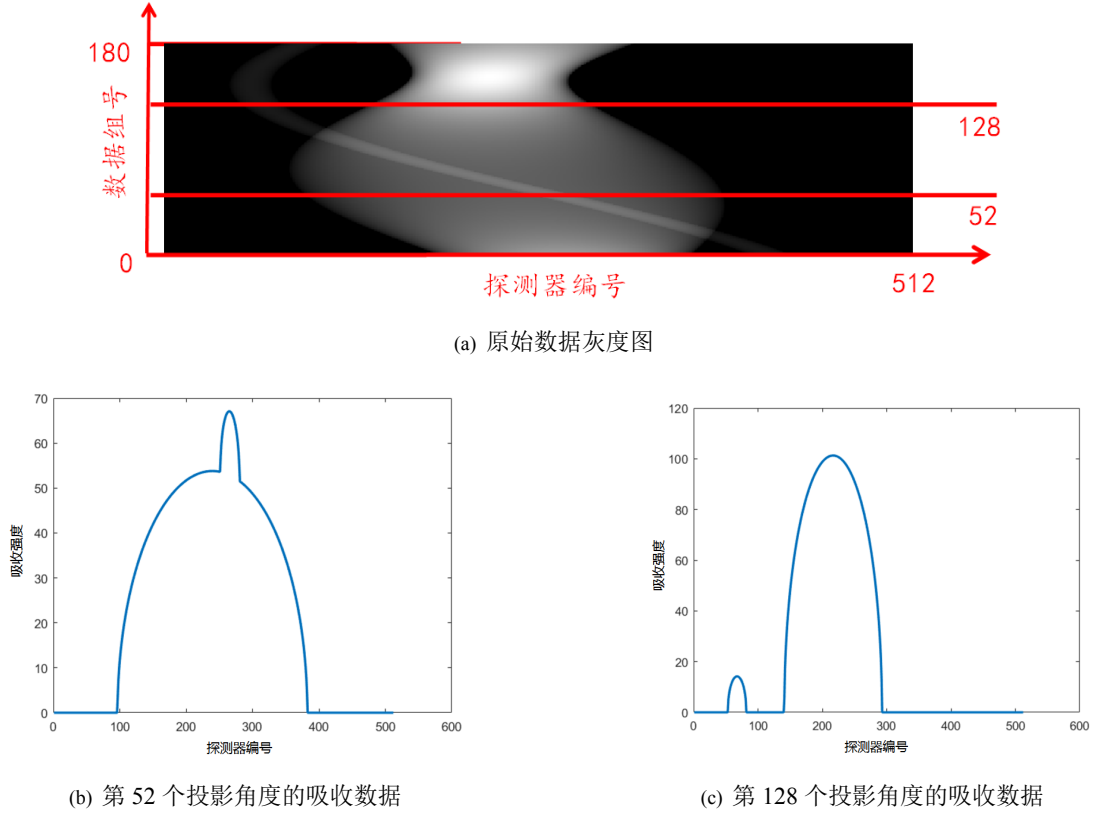


图 4 原始吸收数据可视化示意图

比对结果表明，由探测器所采集的吸收数据基本与本投影模型相符。进一步地，由图像形状、凹凸性可以判断：实际接收信息 A' 应当与先前定义的吸收程度 A 存在正相关关系，函数 $A' = f(A)$ 满足：

$$f(0) = 0 \quad (4)$$

$$\forall a_1, a_2 \in [0, +\infty) : a_1 < a_2 \rightarrow f(a_1) < f(a_2) \quad (5)$$

在不对函数 f 进行求解的情况下，由式 (4)、(5) 可以粗略地推导出以下结论：

$$A' = f(A) \begin{cases} = 0, & A = 0 \\ > 0, & A > 0 \end{cases} \quad (6)$$

5.1.4 吸收信息差分分割模型

首先，对于某个确定的投影方向，定义

$$d_i = \text{第 } i \text{ 个探测器得到的接收信息}, i = 1, 2, \dots, 512 \quad (7)$$

将物体在探测器上投影的起始位置定义为首个探测来自该物体接收信息的探测器之编号，类似地，定义它的结束位置为从起始位置起，首个探测不到来自该物体接收

信息的探测器之编号。于是有：

$$\text{投影宽度 } w = \text{结束位置} - \text{起始位置} \quad (8)$$

由上节投影模型可知， d_1, d_2, d_{512} 可视作两个轴对称单峰函数的叠加，由于它们峰值的数量级相差不大，每个峰的单调性变化非常明显。由此将接收信息序列与时间序列进行类比，两个单峰函数的叠加可视作时间序列模型中的非平稳序列。已经证明，差分方法是一种非常简便、有效的确定性信息提取方法 [3]。

离散序列的 d 阶差分就相当于连续变量的 d 阶求导，而差分运算的实质是使用自回归的方式提取确定性信息。Cramer 分解定理 [4] 在理论上保证了适当阶数的差分一定可以充分提取确定性信息，然而由于每次差分都会有信息的损失，过高阶数的差分会导致过差分的现象。

按照定义，一阶差分可展开为

$$\nabla d_t = d_t - d_{t-1} \quad (9)$$

对于题目所给数据，一阶差分运算能够有效提取椭圆、圆形固体介质接收信息的边界。图 5 (a) (b) 是第 14 组接收信息经过一阶差分运算的结果。经过一阶差分运算以后，固体投影的起始位置表现为序列的极大值，结束位置表现为序列的极小值。从图像上看，两种位置均表现为尖锐的峰形。

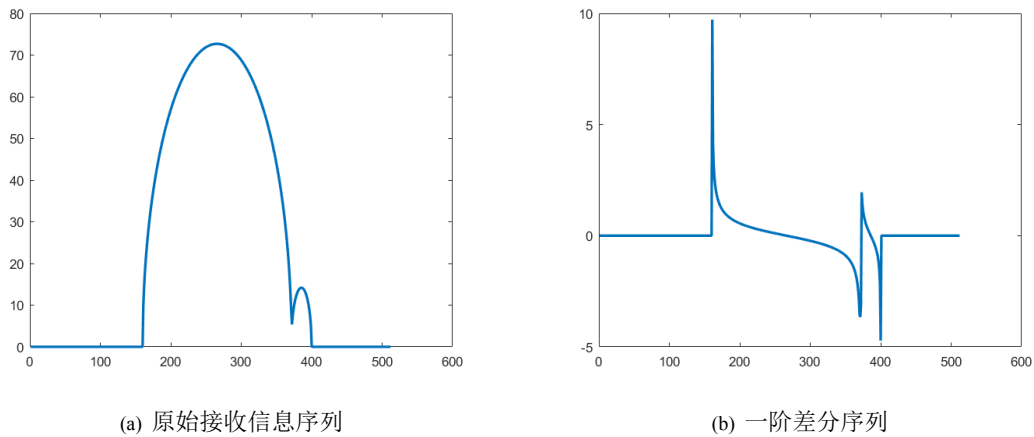
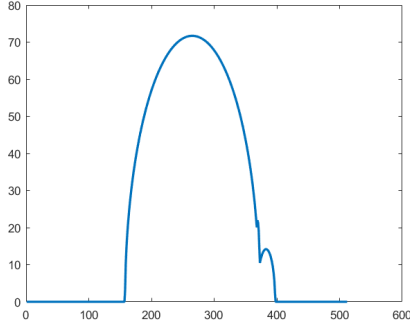


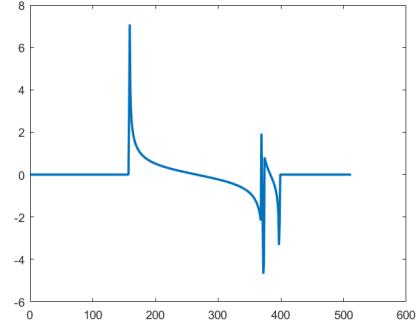
图 5 一阶差分运算效果示意图

一种启发式的方法是：在序列中寻找局部的极大值与极小值，作为两个固体介质投影的始、末位置。在实际应用中，这种算法尽管能够正确求出大部分数据（经过参数调节，正确率可达 90% 以上）的边界，但由于一些异常现象的影响，少部分数据仍然需要人工干预。主要的异常现象包括：

- (1) 部分数据的极大值或极小值多于 2 个（如第 15 组数据，见图 6），这是由一个固体介质的投影末尾部分恰好与另一个固体介质投影的起始部分交叠导致的。对于这种情况，从中选取正确的极大/极小值需要额外的经验。



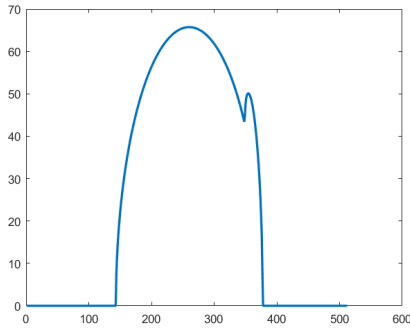
(a) 第 15 组接收信息序列



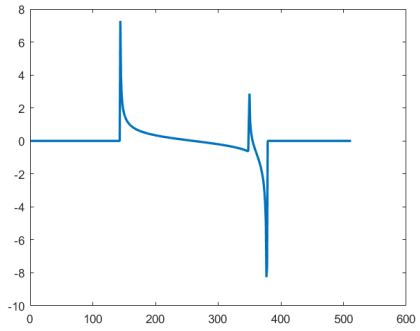
(b) 第 15 组一阶差分序列

图 6 第一类异常现象示意图

(2) 部分数据的极大值或极小值少于 2 个（如第 22 组数据，见图 7），这是由两个固体介质投影的起始或结束位置恰好重合导致的。对于这种情况，添加正确的极大/极小值也需要额外的经验。



(a) 第 22 组接收信息序列



(b) 第 22 组一阶差分序列

图 7 第二类异常现象示意图

为了进一步提取有效的边界信息，同时缩小差分后序列里峰值的宽度，需要进一步对序列进行差分处理。然而，重复进行差分运算不仅容易导致过差分现象，也会在丢失信息的同时产生新的极小/极大值，这使得边界信息的提取更加困难。为了在保留关键边界信息的同时滤去由差分运算产生的新噪声，引入神经网络模型中广泛运用的修正线性单元 (Rectified Linear Unit, ReLU) [5] 作为激活函数，添加在差分运算后，其代数表示如式 (10)：

$$ReLU(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (10)$$

实践表明，ReLU 函数的引入在保留关键边界信息的同时抑制了噪声的产生，能够快速收敛、有效提取出所有边界信息，并且其正确性不会受到第一类异常现象的影响。本文采用图 8 所示的序贯模型 (Sequential Model) 进行边界信息提取，原始的投影信息序列，在进行一阶差分运算后，会交替进行差分 and ReLU 运算。其中，差分运算是用于

信息的提取，而 ReLU 函数则用来保证噪声的滤除。在实际应用中， n 通常取值在 20 以上，以确保结果收敛。本模型能够准确提取吸收数据的所有边界信息，并且对于第二类异常现象，它会提取出所有的三个边界坐标。这样一来，提取的边界坐标个数可以作为后续算法识别和处理异常的依据。

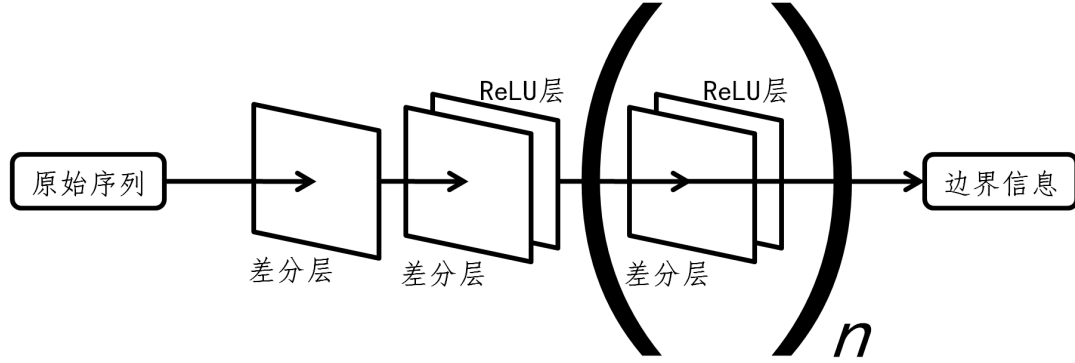


图 8 序贯模型示意图

对于属于第一类异常现象的第 15 组数据，分别用含 ReLU 层与不含 ReLU 层的差分函数对原始数据迭代计算 4 次和 20 次。在图 9 中，横坐标是探测器的编号，而纵坐标是由差分及 ReLU 运算得到的新序列。由图 9 (a) (b) 可知，不使用 ReLU 层的差分函数在迭代 4 次时就已经过度差分，具有明显的震荡现象，迭代 20 次后边界信息已被差分运算产生的噪声淹没；而图 9 (c) (d) 说明在引入 ReLU 层后，噪声信息显著减少，迭代 20 次后只有真正的边界信息会保留下来。

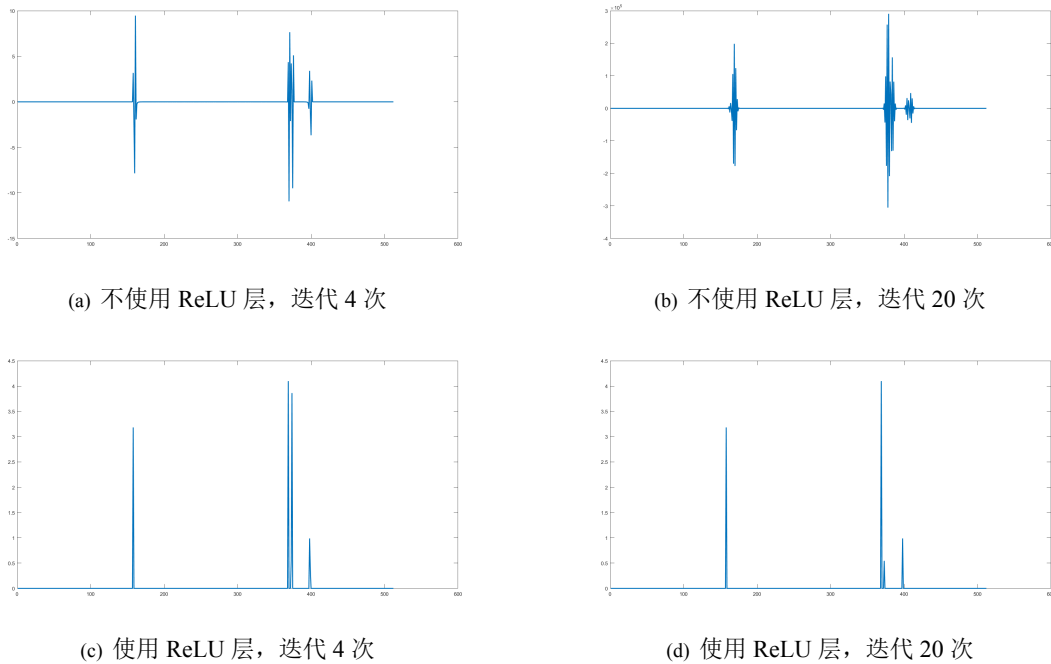


图 9 ReLU 激活函数效果示意图

值得注意的是，在多次迭代后，保留的边界信息仍是不使用 ReLU 层、对原数据二阶差分后得到的数值，将式 (9) 迭代两次，得到：

$$d'_t = d_t + d_{t-2} - 2d_{t-1} \quad (11)$$

因此，求得的投影起始位置坐标是正确无误的，而结束位置的坐标比真实值大了 1，需要在之后减去。

5.1.5 局部预测重排算法

利用上述模型求得边界信息后，还需要将每一组数据的 3~4 个边界坐标分配给两个物体，作为它们的起始和结束位置。本文利用基于线性插值的局部预测算法，以初始的两组数据（正确性一目了然）作为起始条件，能够在不需要人工干预的情况下，自动对后续数据进行补充、分类和排列，正确率达到 100%。

首先，定义 $edge_{i,j}$ 为第 i 组数据的第 j 个边界坐标，其中 $1 \leq i \leq 180, 1 \leq j \leq 4$ 。数据经过预处理，对于确定的 i ， $edge_{i,j}$ 关于 j 单调增。另外，定义 $target_{i,j}$ 为 $edge_{i,j}$ 的预测值。

表 3 前两组数据边界坐标

$edge_{i,j} \backslash j$	1	2	3	4
i				
1	189	360	402	430
2	186	361	399	428

由表 3 可得，前两组数据都是按照椭圆投影起始、结束位置，圆形投影起始、结束位置来排列的，将这一排列方式作为标准格式，对后续数据进行重新排列。

由于所有数据是 X 射线发射-接收系统沿同一方向旋转产生的，而且分析数据后，可以认为相邻的两个投影角度相差不大。根据以上结论，本文提出一种局部预测算法：对于待预测的数据 $target_{i,j}$ ，将完成排列的 $edge_{i-2,j}$ 和 $edge_{i-1,j}$ 作为 ground truth 进行预测，预测时使用如下函数：

$$target_{i,j} = 2edge_{i-1,j} - edge_{i-2,j}, \quad (12)$$

即：对点 $(i-2, edge_{i-2,j})$ 与 $(i-1, edge_{i-1,j})$ 进行线性插值，用得到的线性函数预测横坐标为 i 时的纵坐标。根据 $target$ 的第 i 行可以对 $edge$ 的第 i 行重新排列（对于异常现象一，还需要补充 1 个坐标），将优化目标定义为：

$$\pi^* = \arg \min_{\pi \in \Omega} \sum_{j=1}^4 abs(edge_{i,j}^{\pi} - target_{i,j}), \quad (13)$$

并不加证明地使用下述定理：

定理 1 当且仅当 $\forall j_1, j_2 \in \{1, 2, 3, 4\} : target_{i,j_1} \leq target_{i,j_2} \iff edge_{i,j_1} \leq edge_{i,j_2}$ 时 $\sum_{j=1}^4 abs(edge_{i,j}^{\pi} - target_{i,j})$ 取得最小值。

对于具有异常现象的数据，本算法首先利用距离最近原则将 3 个真实坐标与 3 个预测坐标配对，对于剩余的 1 个预测坐标，再从真实坐标中选取最接近的一个作为其配对。这一方法的正确性来源于：4 个真实的边界坐标中必然有 2 个发生了重叠，才会只形成 3 个坐标。具体算法如下：

算法 1: 局部预测重排算法

输入: 抽取得到的边界坐标 $edges[1, \dots, 180][1, \dots, 4]$ 。

输出: 按照标准格式重新排列的边界坐标 $edges[1, \dots, 180][1, \dots, 4]$ 。

```

1 for  $i = 3$  to  $180$  do
2    $target_{i,1:4} \leftarrow 2 \times edge_{i-1,1:4} - edge_{i-2,1:4}$ ;
3   根据  $target_{i,1:4}$  补充和重新排列  $edge_{i,1:4}$ ;
4 return  $edges$ ;

```

经局部预测算法对坐标重排后，生成 4 个坐标序列，在原图上以不同的彩色标识出轨迹，得到图 10 所示的结果，可见结果非常准确，该算法具有较强的鲁棒性 (Robostness)。

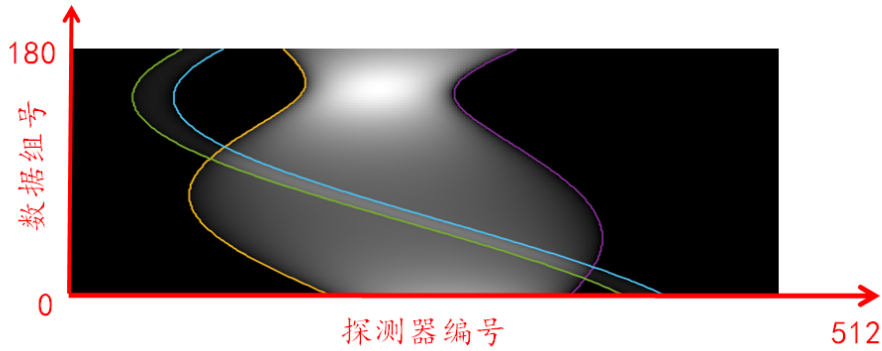


图 10 局部预测算法重排结果

5.1.6 基于数值计算的标定模型

根据已有信息与待求信息画出算法框图，如图 11 所示。首先，根据圆形固体介质的直径和投影宽度（以覆盖的探测器个数计），计算出探测器单元的间距；接着，利用椭圆与圆中心对称的特性，确定两固体介质中心坐标，根据其连线的投影长度与实际长度之比，求出 180 次投影的角度；然后，以探测器阵列中心为原点建立新坐标系，将发射-接收系统的逆时针旋转转化为标定模板的顺时针旋转，利用先前计算出的投影方向，分别对固体介质中心的极坐标参数进行三角函数拟合，得出它们的旋转半径；最后，通过平面解析几何，确定旋转中心在正方形托盘中的位置。

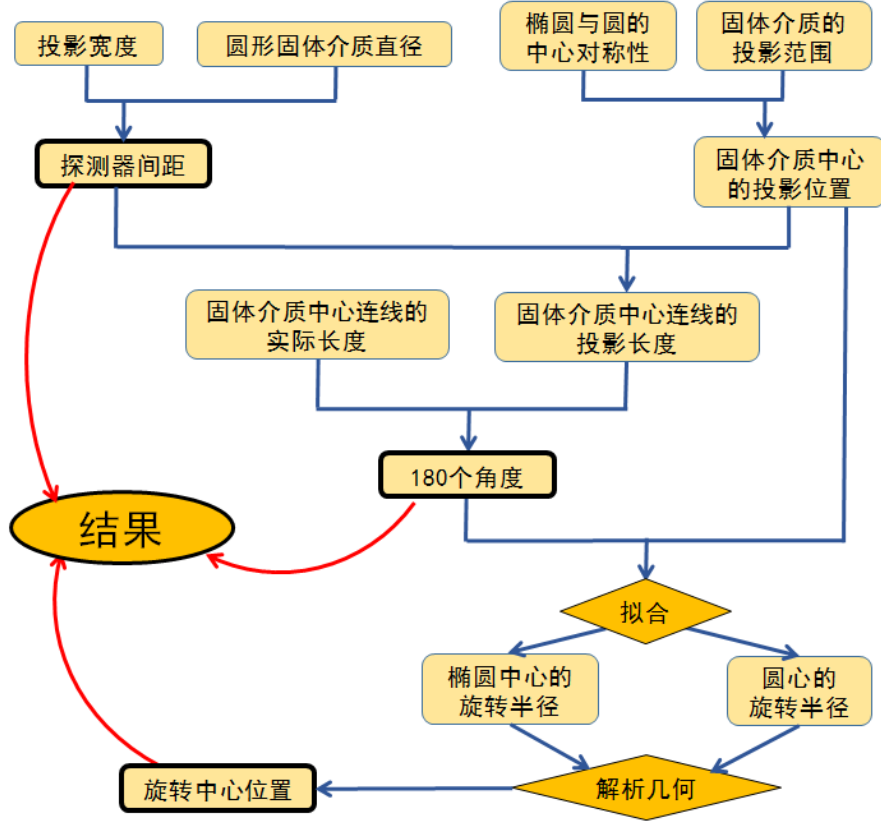


图 11 算法框图

首先，根据圆形固体介质在探测器阵列上投影的 128 组起始和结束位置，计算出平均每次投影覆盖的探测器个数为 n 。为了根据 n 、探测器半径 r 与圆形固体介质真实直径 D 求解探测器单元的实际间距 l ，给出下述定理和证明。

定理 2 $\frac{D+2r}{l}$ 是 n 的无偏估计量。

证明 1 将圆形固体介质投影覆盖探测器的个数 n 视作长为 $\frac{D}{l}$ 的闭区间与 $\{[x - \frac{r}{l}, x + \frac{r}{l}] | x \in \mathbb{Z}\}$ 重叠的个数。设闭区间为 $[s, s + \frac{D}{l}]$ ，令 $k = [\frac{D}{l}]$, $p = \frac{D}{l} - k$ ，不妨设闭区间的起始位置 s 在 $(0, 1]$ 内，先验地假设它为均匀分布，于是有

$$n = \begin{cases} k, & \frac{r}{l} < s < 1 - p - \frac{r}{l} \\ k + 1, & 0 < s \leq \frac{r}{l} \text{ 或 } 1 - p - \frac{r}{l} \leq s \leq 1 \end{cases} \quad (14)$$

因此， n 的数学期望

$$E(n) = k \times (1 - p - 2 \times \frac{r}{l}) + (k + 1) \times (p + 2 \times \frac{r}{l}) = k + p + 2 \frac{r}{l} = \frac{D + 2r}{l} \quad (15)$$

所以， $\frac{D+2r}{l}$ 是 n 的无偏估计量。 ■

根据上述定理，可以得到 X 射线探测器间距 l 的计算公式：

$$l = \frac{D + 2r}{n} \quad (16)$$

根据提取的边界信息，计算得 $n = 28.6056$ ；又查阅资料得，当前普通 X 射线平板探测器的直径（即 $2r$ ）约为 $120\sim 150\mu m$ [6]，不妨取中间值 $135\mu m$ 作为 X 射线探测器单元的直径，代入式 (16) 后算得：

计算结果 1 探测器单元的间距 l 为 $0.2844mm$ 。

如图 12 所示，对于给定的投影方向 θ ，记圆与椭圆固体介质在原坐标系中中心坐标之差为 \vec{X} ，投影中心在探测器轴上坐标之差为 \vec{X}' ，显然有

$$\vec{X}'l = \vec{X}\cos(\theta), \quad (17)$$

其中， l 是探测器单元的间距。

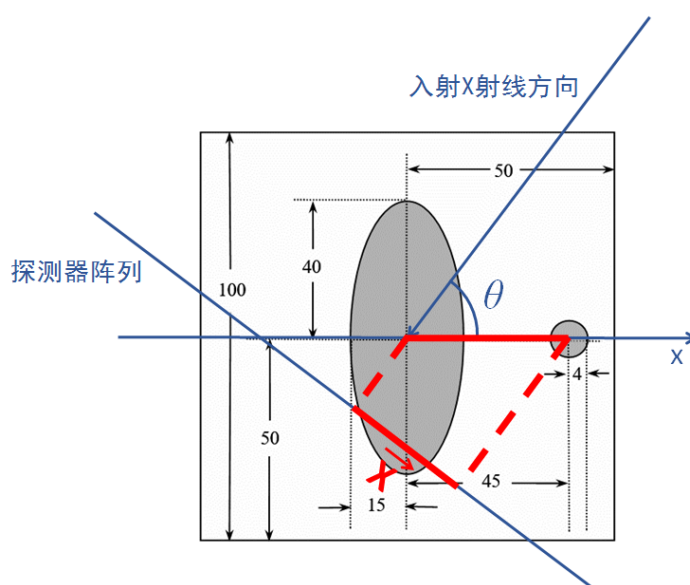


图 12 计算投影角度原理示意图

利用式 (17) 和投影方向始终沿逆时针旋转的特性，可以求得 180 个投影角度，其中部分数据见表 4，完整的结果见职支撑材料中的 *degree.xls*。

计算结果 2 180 个投影角度，见表 4。

表 4 部分投影角度计算结果

编号	1	2	3	4	5	6	7	8	9	10
角度/ $^{\circ}$	119.7616	120.8072	121.8217	123.1310	123.7687	124.7059	125.9220	127.1036	127.6824	128.5368
编号	171	172	173	174	175	176	177	178	179	180
角度/ $^{\circ}$	290.1739	291.1696	292.1226	293.0380	294.3501	294.7729	296.0027	296.7934	297.9404	299.0457

最后，为了计算出旋转中心在平面直角坐标系中的位置，对坐标系进行变换，考虑圆形固体介质中心在探测器阵列坐标系上投影的位置规律。由于 X 射线发射-接收系统相对样本进行逆时针旋转，因此如果以旋转的探测器为参照系，样本是绕着旋转中心在

顺时针旋转的。如图 13 所示，样本中的旋转点以半径 r 、初始相位 a_0 ，绕旋转中心顺时针旋转，它在探测器阵列上的投影坐标 y' 满足：

$$y' = r \cos(a_0 - \theta) + b \quad (18)$$

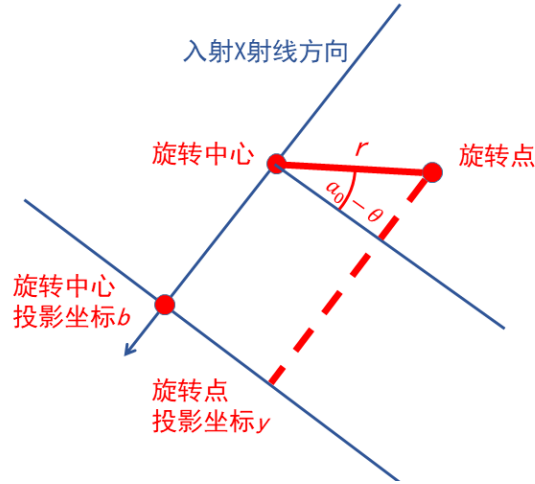


图 13 坐标系变换示意图

在原平面直角坐标系中，旋转点相对于旋转中心的方位角 α 可表示为：

$$\alpha = a_0 + \theta_0 - \frac{\pi}{2}, \quad (19)$$

其中， θ_0 是第一组吸收数据对应的投影角度， a_0 是在探测器阵列坐标系下，旋转点相对于旋转中心的初始方位角。由表 4 可知， $\theta_0 = 119.7616^\circ$ 。使用式 (18) 分别对两固体介质中心的投影坐标进行余弦函数拟合，得到表 5 所示的结果：

表 5 拟合结果

	r	a_0	b
圆	197.7442	-0.6331	256.6706
椭圆	40.2861	-1.1097	256.6706

由于两次拟合得到的 b 惊人地一致，可以认为旋转中心在探测器阵列坐标系上投影的位置就是 256.6706。

图 14 的两组序列，振幅较小的是椭圆中心的投影坐标序列，另一组则是圆心的投影坐标序列。用式 (18) 分别拟合这两组数据，拟合结果与原数据非常吻合。接下来，使用 4 种方法分别对旋转中心坐标进行求解：

1. 利用圆心投影坐标拟合得到的 r 、 a_0 ，定位旋转中心坐标为 (39.1272, 56.3769)。
2. 利用椭圆中心投影坐标拟合得到的 r 、 a_0 ，定位旋转中心坐标为 (40.4820, 56.3769)。
3. 利用两次拟合得到的不同旋转半径 (r)，定位旋转中心坐标为 (38.8202, 52.5041)。
4. 利用两次拟合得到的不同初始相位 (a_0)，定位旋转中心坐标为 (40.7602, 56.1905)。

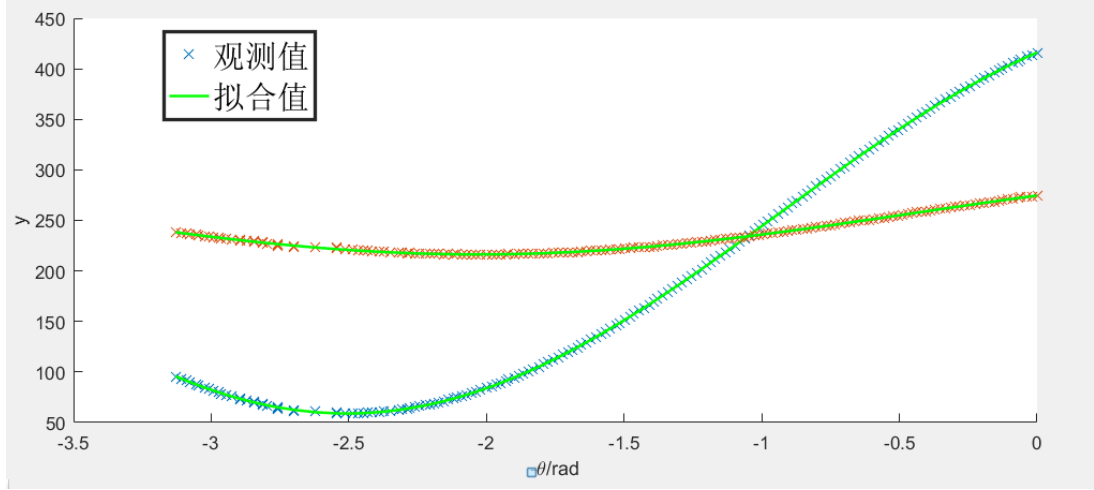


图 14 拟合结果示意图

对比上述计算结果，不难发现：

- 椭圆中心的旋转半径较小，因而定位旋转中心的误差相对较大；
- 相比于半径信息，利用相位信息的定位更加准确。

根据上述结论，认为由第 1 种方法计算出的旋转中心坐标最为可信。

计算结果 3 旋转中心的坐标 l 为 $(39.1272, 56.3769)$ 。

5.2 模型二：基于傅里叶变换的反投影模型

5.2.1 平行光束吸收率信息——Radon 变换

CT 扫描技术的基本原理是奥地利数学家 Radon 分别在 1917 年和 1919 年提出的 Radon 变换 [9] 和 Radon 逆变换 [8]。设一束平行光束的吸收函数为

$$f(x, y) \begin{cases} \neq 0, (x, y) \in \Omega \\ = 0, (x, y) \notin \Omega \end{cases}, \quad (20)$$

其中 Ω 为介质。

为了便于计算，可以建立两个坐标系： $X - Y$ 和 $s - u$ ，它们之间的关系是

$$\begin{cases} s = x \cos \theta + y \sin \theta \\ u = -x \sin \theta + y \cos \theta \end{cases}, \text{ 或 } \begin{cases} x = s \cos \theta - u \sin \theta \\ y = s \sin \theta + u \cos \theta \end{cases}, \quad (21)$$

令 R 为 Radon 变换的运算符，我们采用 $R - \text{Radon}$ 算子进行 Radon 变换，其形式为式 (22) 或式 (23)

$$g(s, \theta) = Rf = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy, \quad (22)$$

$$-\infty < s < +\infty, 0 \leq \theta < \pi,$$

$$g(s, \theta) = Rf = \int_{-\theta}^{+\theta} f(s \cos \theta - u \sin \theta, s \sin \theta + u \cos \theta) du, \quad (23)$$

$$-\infty < s < +\infty, 0 \leq \theta < \pi;$$

其中, Rf 表示函数 $f(x, y)$ 在 $s - u$ 坐标系下经过 (s, θ) 确定的点对坐标 u 的一维投影。

5.2.2 图像重建算法描述

反投影从不同角度测得的投影数据做成了一个图像。这个图像与真实图像相差不多, 只是模糊了一些。这模糊的效果是因为在二维傅里叶空间 (即 $\omega_x - \omega_y$ 平面) 里有个非均匀的 $\frac{1}{|\omega|}$ 密度分布, 这里 $|\omega| = \sqrt{\omega_x^2 + \omega_y^2}$ 。矫正 $1/|\omega|$ 密度分布函数可以通过对投影数据 $p(s, \theta)$ 或其一维傅里叶变换 $P(\omega, \theta)$ 进行斜坡滤波 (即在频率域乘以 $|\omega|$) 完成。对滤波后的数据施行反投影就可得到精确的图像。反投影 (FBP) 算法的傅里叶变换实现如算法2所示。

算法 2: 反投影算法的傅里叶变换实现

输入: 吸收信息数据 $\mathbf{p}(\mathbf{s}, \theta)$ 。

输出: 反投影结果 $\mathbf{q}(\mathbf{s}, \theta)$ 。

- 1 求投影数据 $\mathbf{p}(\mathbf{s}, \theta)$ 的以 \mathbf{s} 为变量的一维傅里叶变换, 得到 $\mathbf{P}(\omega, \theta)$;
 - 2 对 $\mathbf{P}(\omega, \theta)$ 乘以斜坡滤波器的传递函数 $|\omega|$, 得到 $\mathbf{Q}(\omega, \theta)$;
 - 3 求 $\mathbf{Q}(\omega, \theta)$ 的以 ω 为变量的一维傅里叶反变换, 得到 $\mathbf{q}(\mathbf{s}, \theta)$;
 - 4 **return** $\mathbf{q}(\mathbf{s}, \theta)$;
-

问题二中采用与问题一类似的坐标系, 都是以平行光束传播方向为 y 轴正方向, 即 y 轴正方向与平行光束传播方向相一致, 坐标系与光线同方向旋转。不同的是坐标原点平移到问题一计算得到的旋转中心 O , 即 $(50, 50)$, 见图 15。

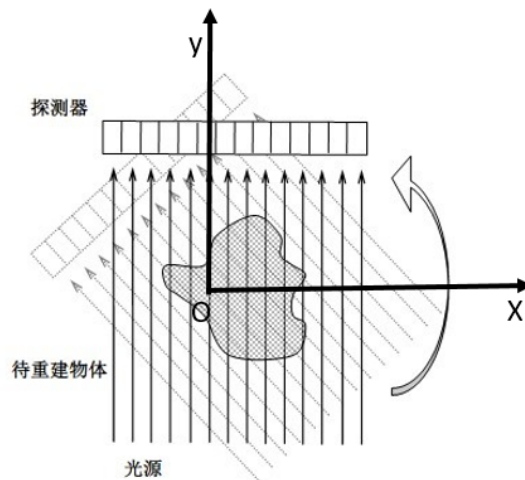


图 15 问题二平面直角坐标系

坐标系建立后, 计算未知介质上一点旋转后在探测器上的投影坐标。设平面上一点

(x_1, y_1) ，绕另一点 (x_0, y_0) 逆时针旋转 α 角度后的点为 (x_2, y_2) ，则：

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} \begin{pmatrix} x_1 - x_0 \\ y_1 - y_0 \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \quad (24)$$

即

$$\begin{aligned} x_2 &= (x_1 - x_0) \cos \alpha - (y_1 - y_0) \sin \alpha + x_0 \\ y_2 &= (x_1 - x_0) \sin \alpha + (y_1 - y_0) \cos \alpha + y_0 \end{aligned} \quad (25)$$

对于数字图像，一般是从左上角为起点，故顺时针旋转 α 度的公式也是式(25)。本模型采用问题一的计算结果得到的旋转中心，计算得到未知介质上的点绕该旋转中心旋转一定角度后在探测器上的投影坐标。

5.2.3 图像重建结果

为了提高重建图像的吸收率信息准确度，首先尝试从已知的附件 2 进行重建，并与附件 1 中的真实值进行对比和校准。图 16 显示了附件 1 中吸收率分布，图 17 的内容则是本模型初步得到的吸收率分布。在这两幅图中，横坐标为每个像素点的编号，而纵坐标则是该像素点对应的吸收率。两幅图像具有非常相似的形状，都有大部分的点分布在某个非零的常数附近，因此考虑它们之间的线性关系。

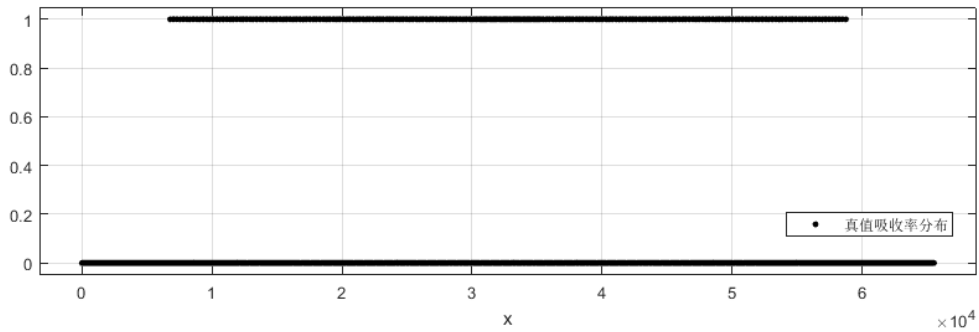


图 16 附件 1 提供的吸收率分布

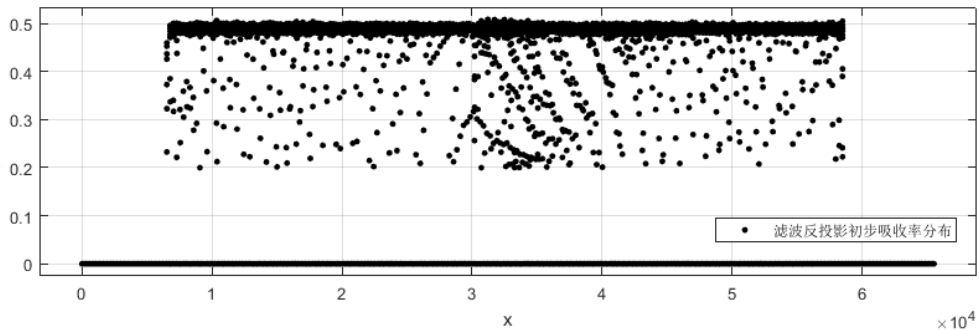


图 17 本模型初步得到的吸收率分布

本模型以附件 1 中提供的吸收率为真值，对模型初步得到的吸收率做线性变换进行校准，从附件 3 重建得到的校准后的最终吸收率分布如图 18：

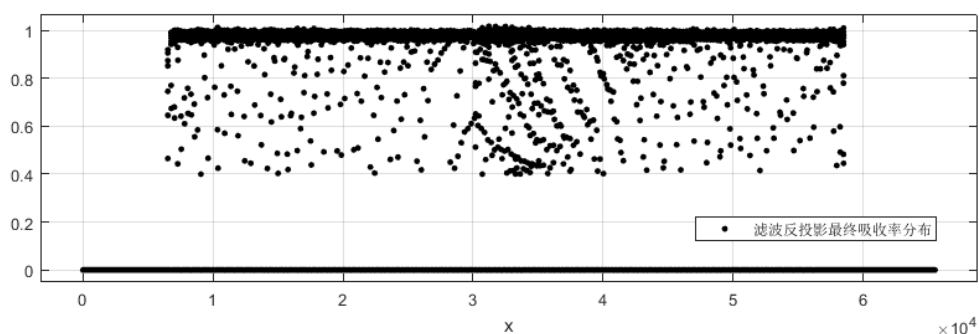


图 18 本模型最终得到的吸收率分布

采用上述模型对附件 2 进行图像重建，可视化结果如图 19(b)所示：

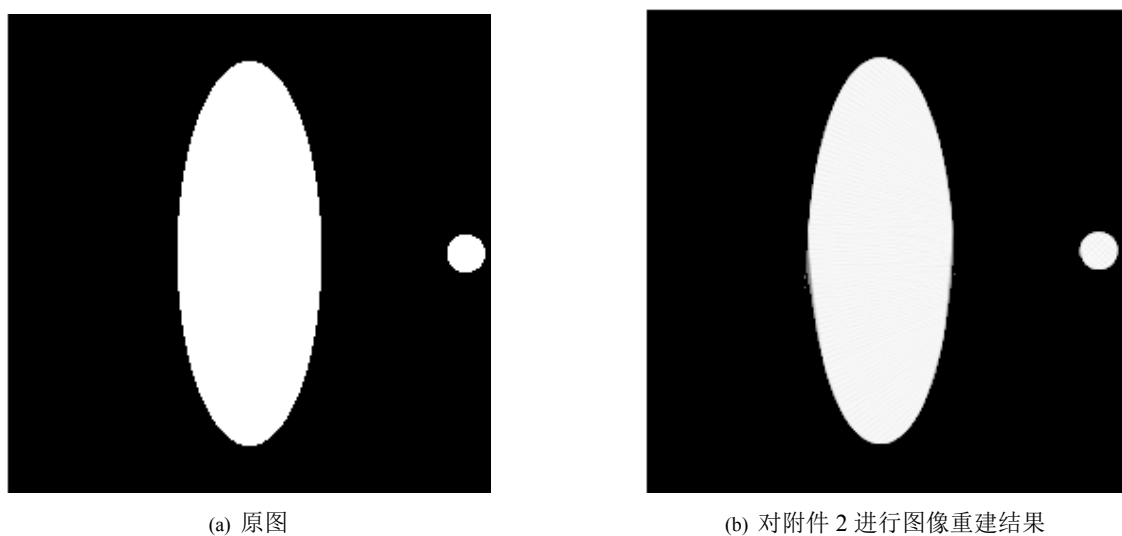


图 19 重建图像与原图对比效果

与原图进行对比可知，本模型重建效果比较理想。

采用上述模型对附件 3 进行重建得到的未知介质在托盘中的位置、几何形状和吸收率等相关信息可视化结果如图 20 所示：

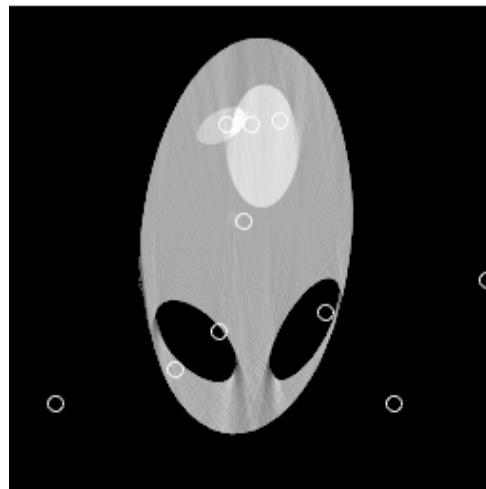
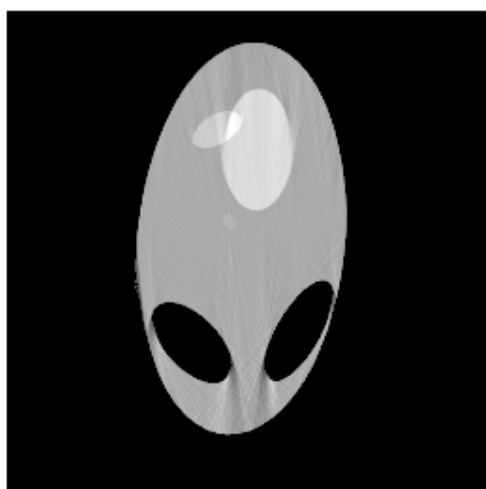


图 20 本模型对附件 3 未知介质重建结果 图 21 问题二指定 10 个点在重建后图像的分布

观察可知，

1. 未知介质在托盘中的位置接近中央。
2. 未知介质的几何形状是由一个近似的大椭圆，其中分布着几个小椭圆组合而成。
3. 未知介质的吸收率从图20初步判断是不均匀的，大椭圆吸收率较小，小椭圆上吸收率相对较大，具体介质的吸收率文件请见附件 problem2.xls。

指定 10 个点在重建后图像的分布如图21所示：

指定 10 个点的吸收率请见表 6。

表 6 指定 10 个点的吸收率

编号	坐标 x	坐标 y	吸收率
1	10.0000	18.0000	0.0000
2	34.5000	25.0000	1.0284
3	43.5000	33.0000	0.0000
4	45.0000	75.5000	1.2416
5	48.5000	55.5000	0.9844
6	50.0000	75.5000	1.2770
7	56.0000	76.5000	1.3045
8	65.5000	37.0000	0.0000
9	79.5000	18.0000	0.0000
10	98.5000	43.5000	0.0000

5.3 模型三：滤波反投影模型

5.3.1 算法描述

问题三提供的未知介质的吸收信息比问题二复杂得多，但主要原理还是相似的。

根据傅里叶变换理论，在 ω 域中做乘法等价于在 s 域中做卷积 [12]。滤波后的数据 (s, θ) 可以通过卷积而得到：

$$q(s, \theta) = p(s, \theta) * h(s), \quad (26)$$

“*” 是卷积运算的记号，这个卷积运算是以 s 为变量的积分运算。其中 $h(s)$ 是卷积积分中的卷积核，它是 $H(\omega) = |\omega|$ 的一维傅里叶反变换。

通过问题一得到的旋转中心，可以算得旋转中心在探测器阵列中的投影位置。采用滤波反投影变换即可得到需要的结果。

采用滤波反投影重建图像的具体过程是，先把由检测器上获得的原始数据与一个滤波函数进行卷积运算，得到各方向卷积的投影函数；然后再把它们从各方向进行反投影，即按其原路径平均分配到每一矩阵元上，进行叠加后得到每一矩阵元的 CT 值；再经过适当处理后可以得到被扫描物体的断层图像，我们采用滤波反投影法解决复杂未知介质的重建问题，为问题三提供解决方案。算法描述如算法3所示。

算法 3: 基于 Radon 逆变换的图像重建算法

输入: 模板介质的接收信息矩阵 \mathbf{R} 。

输出: 模板图像重建结果 \mathbf{P} 。

```

1  $\mathbf{h} \leftarrow$  空间脉冲响应滤波器;
2  $\mathbf{x} \leftarrow \mathbf{h}$  和  $\mathbf{R}$  中每一列数据进行卷积所得结果;
3  $\mathbf{P} \leftarrow$  初始化为大小为  $256 \times 256$  的零矩阵;
4 for  $\mathbf{i} \in \mathbf{P}$  中逐行从左到右每一个像素点 do
5   for  $\mathbf{k} \in \mathbf{R}$  中的每一列 do
6      $\mathbf{t} \leftarrow$  像素  $\mathbf{i}$  旋转后在探测器的投影位置;
7      $\mathbf{i} \leftarrow \mathbf{t}$  相邻两个探测器吸收率加权之和;
8 return  $\mathbf{P}$ ;
```

重建处理是从投影影像重建得到原图像，是 Radon 变换的一种逆问题。对于二维 Radon 变换，最常被使用的解析公式是滤波反投影方法的 Radon 反变换公式 f ，反变换公式为

$$f(x) = \int_0^\pi (Rf(\cdot, \theta) * h)(\langle x, n_\theta \rangle) d\theta \quad (27)$$

函数 h 满足 $\hat{h}(k) = |k|$ ，卷积核 h 也被称作 Ramp 滤波器 [7]。具体表达式为

$$f(x, y) = R^{-1}g(s, \theta) = \left(\frac{1}{2\pi^2}\right) \int_0^\pi \int_{-\infty}^{+\infty} \frac{\frac{\partial g}{\partial s}(s, \theta)}{x \cos \theta + y \sin \theta - s} ds d\theta, \quad (28)$$

极坐标下可表示为

$$f_\rho(x, \phi) = f(r \cos \phi, r \sin \phi) = \left(\frac{1}{2\pi^2}\right) \int_0^\pi \int_{-\infty}^{+\infty} \frac{\frac{\partial g}{\partial s}(s, \theta)}{r \cos(\theta - \phi) - s} ds d\theta, \quad (29)$$

5.3.2 滤波器比较与设计

采用滤波反投影方法，滤波器的设计关系到重建质量 [11]。为了消除反投影造成的图像模糊，要用 Ramp 滤波函数作滤波处理。目前常用的是普通型 Ramp 滤波函数 (简称

RF). 另一种称为定量分析型的 Ramp 滤波函数 (简称 RFQ) 也有应用。

目前已有的几种滤波器及其特点如下：

- **巴特沃思高通滤波器** 标准巴特沃思高通滤波器低频部分过于陡峭，部分低频信息的损失会导致重建图像噪声较多和过渡不够平滑，因而需要改进。改进的巴特沃思滤波器在高频处的幅值有所降低，且上升部分更平滑。有利于获得细节更丰富的重建图像，不过也会造成边缘模糊，实际使用中仍有待改进。
- **平滑滤波器** 平滑滤波器重建图像平滑、噪声少。滤波函数曲线大致成拱形。图像清晰而平滑，重建效果理想。
- **Ramp-Hamming 结合滤波** 相比斜坡滤波器，在高频部分对噪声具有很好的抑制作用。
- **Ramp-Sinc 结合滤波** 重建质量更高，环状伪影也逐渐消失。
- **Butterworth-Sinc 结合滤波** 重建图像边缘清晰。该滤波器可以通过设置不同的参数值来获得边缘清晰或整体比较平滑的图像。

本文参考以上滤波器设计方案，同时考虑简便性等因素，根据 CT 图像处理中的滤波器设计 [10] 等文献资料，设计滤波器如下：

$$h(k) = \begin{cases} \frac{1}{4}, & k = 512 \\ 0, & k \text{ 为其他偶数} \\ -\frac{1}{[(512-k)\pi]^2}, & k \text{ 为奇数} \end{cases} \quad (30)$$

采用上述滤波器后，可消除单纯的反投影产生的边缘失锐效应，补偿投影中的高频成分和降低投影中心密度，并保证重建图像边缘清晰和内部分布均匀。

5.3.3 重建结果

采用上述模型得到的未知介质相关信息可视化结果如图22所示：

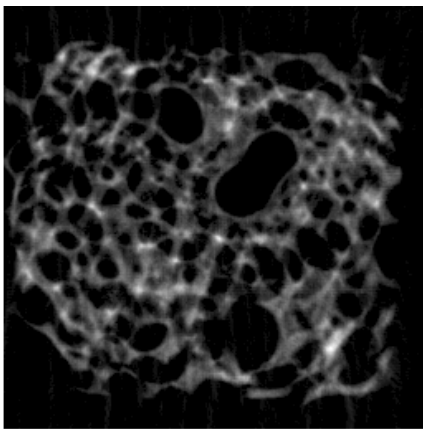


图 22 问题三重建结果

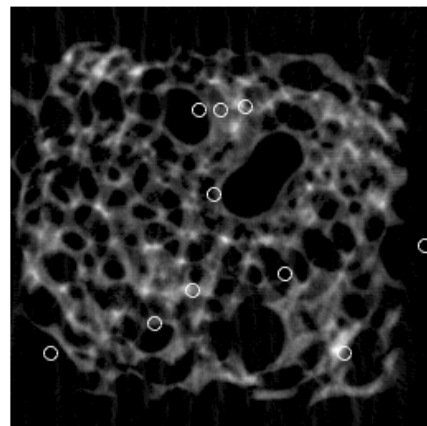


图 23 问题三指定 10 个点在重建后图像中的分布

观察可知，

1. 位置：未知介质在托盘中的位置分布覆盖托盘大部分区域。
2. 几何形状：未知介质的外部边缘呈正方形，中间成孔穴状或海绵状，疏松多孔的结构。
3. 未知介质的吸收率从图20初步判断是不均匀的，具体介质的吸收率文件请见附件 problem3.xls。

由于模板大小为 $100mm \times 100mm$ ，指定 10 个点的坐标也是以毫米为单位，本模型得到的吸收率规格为 256×256 （单位：像素），因此取指定坐标对应的点及周围共 9 个点的吸收率平均值作为所求点的吸收率，可以减小误差。指定 10 个点在重建后图像的分布如图23所示：

计算得指定 10 个点的吸收率请见表7。

表 7 指定 10 个点的吸收率

编号	坐标 x	坐标 y	吸收率
1	10.0000	18.0000	0.0000
2	34.5000	25.0000	1.0699
3	43.5000	33.0000	2.1302
4	45.0000	75.5000	0.0000
5	48.5000	55.5000	0.5958
6	50.0000	75.5000	1.2285
7	56.0000	76.5000	1.7914
8	65.5000	37.0000	0.0000
9	79.5000	18.0000	1.5872
10	98.5000	43.5000	0.0000

具体介质的吸收率文件请见补充材料中的 problem3.xls。

5.4 模型四：改进的样本标定模型

问题一的求解结果实际上是比较粗略的，原始模型为标定计算提供的信息不够丰富，致使整个计算过程必须分步进行，并且后一步的计算对前一步计算结果的依赖性非常强，如图 24 所示，整个计算模型使用了探测器单元间距、投影角度、样本相对探测器旋转半径和初始相位这三个关键变量。误差经过这些中间变量不断累积，最后会对结果造成不容忽视的影响。



图 24 引入误差的关键中间变量

误差的根源本质上来源于连续二维样本在一维离散探测器上投影这一事实。尽管投影宽度可以作为样品实际直径的无偏估计量，但误差仍然是不可避免的。接下来分别从系统因素与偶然因素两方面对误差进行分析。

对于系统误差，回顾第一题的解题思路，不难发现：作为一切后续计算结果根基的探测器单元间距，是仅仅使用圆形固体介质的投影宽度而计算出来的，而圆形固体介质直径相当小，很难不引入测量误差；至于后续的计算，几乎也完全是利用投影宽度计算得到的。对于探测器阵列，这一绝对误差与探测器本身灵敏度、探测器单元的间距相关，相对来说比较固定、难以消除。但通过增大投影宽度，可以有效减小相对误差，提高计算准确程度。

对于偶然因素引起的误差，可以通过多次测量的方法消除。即使在同一次测量中，增加信息的丰富程度也可以有效减少偶然因素对计算结果的影响。考虑问题一的求解过程，投影角度的计算完全依赖于两固体介质中心连线的投影、或是椭圆自身的投影宽度，用于计算的信息非常缺乏。如果样品中有更多的物体，能够进行交叉检验，那么偶然误差将会大大减小。

由此，出于尽可能消除误差的考虑，对标定模板的设计提出以下五项原则：

- 应当有一块半径尽可能大的原形介质，用于精确标定探测器单元间距。
- 介质的几何形状应当中心对称，这样其中心的投影位置必然位于整个物体投影的中心，方便计算。
- 介质与介质之间应当相隔尽可能远，以便利用其中心连线的投影信息计算投影方向。
- 介质的投影特征应当容易识别。
- 物体数目不应超过 5 个，否则识别和计算都过于复杂。

基于上述原则，对原始模板进行改进，设计出如图 25 所示的新模板：它有 1 个位于正方形托盘中央的大圆，用于对探测器单元的间距进行标定。在托盘的四个角，各有一个半径稍小的小圆。小圆按照半径分为 10mm 和 15mm 两种，既保证了足够的直径，又能够互相区分。

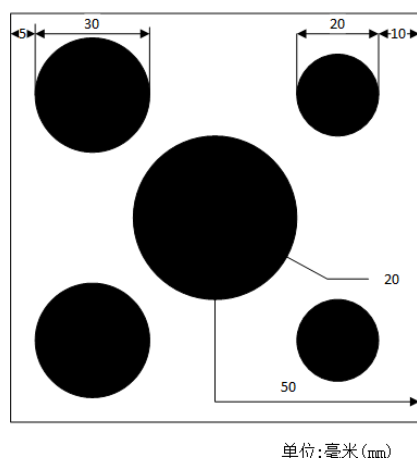


图 25 新的标定模板几何特征示意图

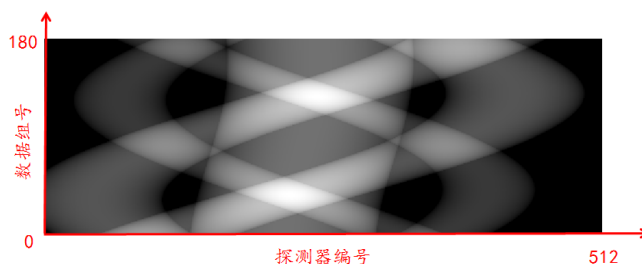


图 26 新的标定模板吸收信息示意图

按照问题一中求得的 CT 系统参数，绘制出新标定模板在 180 个角度的吸收信息，

如图 26 所示。首先，根据中央大圆的投影信息可以计算探测器单元的实际间距；然后由四个小圆投影信息的宽度、相位确定它们分别对应了哪一块固体介质；接着考虑位于正方形托盘对角的小圆圆心连线的投影宽度，由两对小圆得到**两组角度信息**，取均值以减小偶然误差；最后，和问题一类似地，拟合每个小圆圆心绕旋转中心的顺时针旋转参数，对求得的 4 个坐标依然取均值以减小误差。

为了验证新的标定模型对能够有效提高标定的精度和稳定性，使用随机的初始参数比较两种标定模型计算误差的大小：

- 旋转中心的横、纵坐标 x_0, y_0 服从分布 $N(50, 100)$ 。
- 旋转中心在探测器坐标系是投影坐标 b 服从分布 $N(256.5, 100)$ 。
- 探测器单元间距 l 服从分布 $N(0.3, 0.0025)$ 。
- 投影角度 θ 间隔为 1° ，初始角度 θ_0 服从分布 $N(100, 25)$ 。

对两种标定模板均进行 1000 次测试，平均误差如表 27 所示。由实验数据可证明新的模板提高了标定的精度和稳定性。

图 27 新、旧标定模板误差对比表

	探测器间距 l 平均绝对误差/mm	探测器间距 l 平均相对误差/%	投影角度 θ 绝对误差/ $^\circ$	投影角度 θ 相对误差/%	旋转中心 平均误差/mm
原始模板	0.0404	14.0121	4.9915	2.41	18.1317
新模板	0.0017	0.5449	0.348	0.17	2.3768

六、模型检验与定量分析

6.1 问题一数据的重建检验

首先，用求解问题一得到的旋转中心坐标、探测器单元间距、旋转中心在探测器坐标轴上的投影位置、180 个投影角度，计算每个角度下探测器阵列接收到的 X 射线吸收程度 (Absorption Degree)，如图 28 所示。

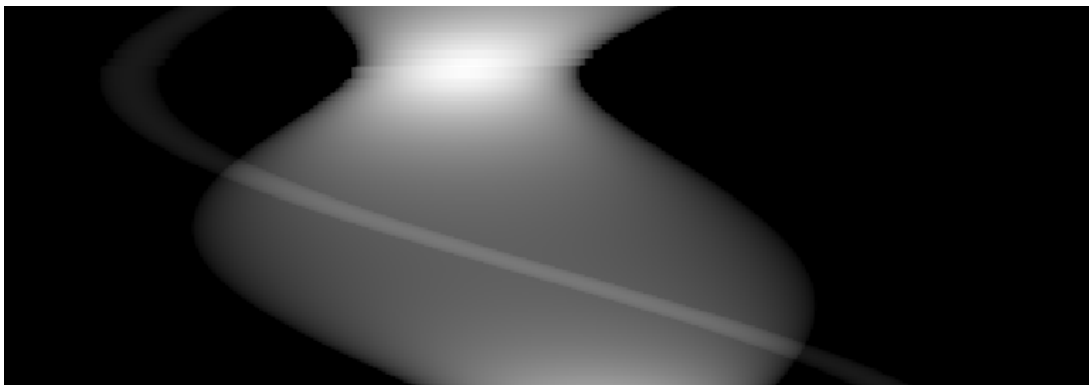


图 28 重建数据灰度图

定性地看，重建的数据灰度图与原数据在形状上非常相似；定量地看，本文定义的吸收程度与题目所给的吸收信息之间的相关系数高达 0.9926，说明两组数据大致上线性相关。根据式 (4)，它们之间应成正比例关系。

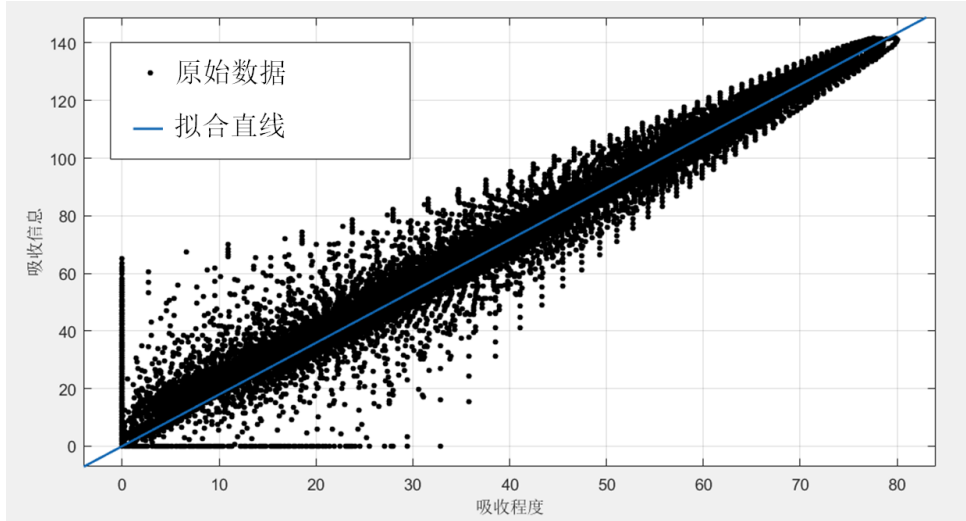


图 29 吸收程度与吸收信息拟合示意图

经过最小二乘法拟合，得到如图 29 所示的拟合直线，由于 R-square 达到 0.9851，可认为拟合程度较好，吸收程度 A 、吸收信息 A' 之间近似满足：

$$A' = 1.793A \quad (31)$$

这一拟合结果证明了计算结果的精确性，说明该模型能够满足实际应用需求。

七、模型的评价

7.1 模型的优点

1. 能够自动提取数据的边界信息，并自动排列坐标，该部分完全不需要人工干预。
2. 设计的新模板对未知 CT 系统参数标定的准确性、稳定性非常高。
3. 与传统的迭代重建算法相比，本模型轻便而简约，在得到较好视觉效果的同时提高了计算速度。
4. 投影数据是对物体的线积分（或称 Radon 变换）。投影数据是用探测器采集的。探测器上所成的像是物体叠加后的图像。反投影是一个求和的过程，它把所有角度上的数据都加在一起。它把投影域中的数据沿着原来的投影路径“涂抹”回去，但不改变数据的值。本模型在图像重建时充分考虑旋转中心不在几何中心的情况，采用问题一中计算得到的旋转中心和 180 个方向，使得重建误差更小。
5. 在计算未知几何体吸收率时，本模型用已知的附件 1 作为吸收率真值，将重建后的结果与之对比，进行校准，使得吸收率的计算更加准确。
6. 本模型采用滤波反投影算法，设计了一种便于实现、效果良好的滤波器，消除图像数字化时所混入的噪声，滤波反投影可消除直接反投影产生的边缘失锐效应，补偿投影中的高频成分和降低投影中心密度，并保证重建图像边缘清晰和内部分布均匀。

7.2 模型的缺点

1. 滤波器的使用虽然改善了可视化效果，同时也可能造成最后得到的未知几何体吸收率不准确的问题，产生误差。
2. 拟合三角函数的过程中可能产生误差。

八、参考文献

- [1] 百度百科. X 射线吸收 [EB/OL]. [2016-11-22] [https://baike.baidu.com/item/X 射线吸收](https://baike.baidu.com/item/X%20射线吸收).
- [2] 张朋, 张兆田. 几种 CT 图像重建算法的研究和比较 [J]. CT 理论与应用研究, 2001, (04): 4-9. [2017-09-18].
- [3] Box G. Box and Jenkins: Time Series Analysis, Forecasting and Control[M]. London: Palgrave Macmillan, 2013. 125.
- [4] 王燕. 时间序列分析——基于 R[M]. 北京: 中国人民大学出版社, 2015. 136 - 141.
- [5] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[A]. NIPS. 2012: 1097-1105.
- [6] 张军毅, 王同乐. X 射线平板探测器性能比较研究 [EB/OL]. [2013-01-19] <https://wenku.baidu.com/view/c8d311fcf61fb7360b4c6518.html>.
- [7] G. L. Zeng. Revisit of the Ramp Filter[J]. IEEE Transactions on Nuclear Science, 2015, 62: 131 - 136.
- [8] Holschneider. Inverse Radon transforms through inverse wavelet transforms[J]. Inverse problems, 1991, 7: 853.
- [9] Deans, Stanley R. The Radon transform and some of its applications[M] North Chelmsford: Courier Corporation, 2007. 338 - 339.
- [10] 文斌, 吴胜利, 潘瑞谊. CT 图像重建预处理滤波器设计 [J]. CT 理论与应用研究, 2008, 17(03): 57-64.
- [11] 马兴荣, 张虹. 两种 Ramp 滤波函数重建 SPECT 图像的比较 [J]. 医学争鸣, 1998(s1): 10-12.
- [12] 曾更生. 医学图像重建 [M]. 高等教育出版社, 2010.

程序 1 : 问题一代码

```
1 % 边界信息抽取
2 datasheet1 = xlsread('.../A/A题附件.xls','附件1');
3 datasheet2 = xlsread('.../A/A题附件.xls','附件2');
4 datasheet3 = xlsread('.../A/A题附件.xls','附件3');
5 datasheet5 = xlsread('.../A/A题附件.xls','附件5');
6
7 % 边界提取
8 diff_res = datasheet2;
9 diff_res_2 = Diff(diff_res, false);
10 diff_res_3 = Diff(diff_res_2, true);
11 diff_res_4 = diff_res_3;
12 for i = 1:200
13     diff_res_4 = Diff(diff_res_4, true);
14 end
15
16 % 寻找边界
17 line_num = 4;
18 loc = max_n(diff_res_4, line_num);
19 [x, y] = find(loc == 1); % [14, 22, 109]
20 loc = preprocess(loc, line_num);
21
22 image = zeros(512, 180);
23 for i = 1:180
24     image(loc(:, i), i) = 1;
25 end
26 figure('Name', 'loc');
27 imshow(image, [])
28 loc([2,4], 1:180) = loc([2,4], 1:180) - 1;
29 save loc loc
30 clc; clear;
31 % 探测器单元之间的距离
32 load loc loc
33 coord = loc;
34 % 小圆的投影探测器个数
35 circle_projection=coord(4,:) - coord(3,:);
36 % 椭圆的投影探测器个数
37 oval_projection=coord(2,:) - coord(1,:);
38 mid_value_circle = coord(3,:) + circle_projection ./ 2;
39 mid_value_oval = coord(1,:) + oval_projection ./ 2;
40
41 x = 1:180;
42 % 小圆投影平均覆盖探测器个数
43 circle_mean_num = mean(circle_projection)
44 % 因为是平行投影, 单位: mm
45 detector_real_distance = (4 * 2 + 0.135) / circle_mean_num
46
47 % CT 系统使用的 X 射线的 180 个方向
```

```

48 % 椭圆和小圆投影中心之间的实际距离
49
50 rescale_factor = max(abs(mid_value_circle - mid_value_oval))
    * detector_real_distance / 45;
51 mid_distance = (mid_value_circle - mid_value_oval) .* (
    detector_real_distance / rescale_factor) ;
52 % 以椭圆中心和小圆中心连线为 0 度, 逆时针方向为度数增大方向
53 degree1 = rad2deg(asin(mid_distance(1:152) ./ 45));
54 degree2 = rad2deg(asin(mid_distance(153:180) ./ 45));
55 degree2 = degree1(152) + degree1(152) - degree2;
56 degree = [degree1 degree2];
57 degree = 180 - degree;
58 save degree degree
59
60 % 确定 CT 系统旋转中心在正方形托盘中的位置
61 %  $y = a * \sin(b * x + c) + d$ 
62 % sinparam = [2 10 5 4];
63 xdata = deg2rad(degree(1)) - deg2rad(degree);
64 % 拟合小圆中心正弦函数
65 % 改写一个参数, 用于下面函数 lsqcurvefit 的设置
66 options = optimset('MaxFunEvals', 10000000);
67 cosparam_est1 = lsqcurvefit(@(cosparam, xdata) cosfunc(
    cosparam, xdata), [206 0 239], xdata, mid_value_circle);
68 figure('Name', '拟合椭圆和小圆中心正弦函数');
69 scatter(xdata, mid_value_circle, 'x');
70 hold on
71 plot(xdata, cosfunc(cosparam_est1, xdata), 'g-', 'LineWidth',
    1.5) ;
72 legend({'观测值', '拟合值'});
73 % 拟合椭圆中心正弦函数
74 cosparam_est2 = lsqcurvefit(@(cosparam, xdata) cosfunc(
    cosparam, xdata), [46 0 247], xdata, mid_value_oval);
75 scatter(xdata, mid_value_oval, 'x') ;
76 hold on
77 plot(xdata, cosfunc(cosparam_est2, xdata), 'g-', 'LineWidth',
    , 1.5);
78 legend({'观测值', '拟合值'});
79
80
81 % 计算在原坐标系中, 2 个固体介质的中心相对旋转中心的坐标
82 rotate_center_projection = (cosparam_est1(3) + cosparam_est2
    (3)) / 2
83 radius_circle = detector_real_distance * cosparam_est1(1);
84 theta_circle = cosparam_est1(2) + deg2rad(degree(1)) - pi/2;
85 pos_from_circle = [95 - radius_circle * cos(theta_circle)
    ; 50 - radius_circle * sin(theta_circle)]
86 radius_oval = detector_real_distance * cosparam_est2(1);
87 theta_oval = cosparam_est2(2) + deg2rad(degree(1)) - pi/2;
88 pos_from_oval = [50 - radius_oval * cos(theta_oval) ; 50 -

```

```

    radius_oval * sin(theta_oval)]
89
90 % 利用半径信息，利用固体介质中心坐标，计算旋转中心坐标
91 ang = acos((radius_oval*radius_oval + 45*45 - radius_circle*
    radius_circle)/2/45/radius_oval);
92 final_pos = [50 + radius_oval * cos(ang); 50 + radius_oval *
    sin(ang)]
93
94 ttt = theta_circle - theta_oval
95 radius_circle = 45 / abs(sin(ttt)) * abs(sin(theta_oval));
96 final_pos_2 = [95 - radius_circle * cos(theta_circle) ;50 -
    radius_circle * sin(theta_circle)]
97
98 % 重新绘制图像
99 circle_image = zeros(512, 128);
100 for i = 1:180
101     for j = 1:512
102         circle_image(j, i) = cross_circle(j, deg2rad(degree(
            i)));
103     end
104 end
105 figure('Name','circle_reconstruction');
106 imshow(circle_image,[])
107 oval_image = zeros(512, 128);
108 for i = 1:180
109     for j = 1:512
110         oval_image(j, i) = cross_oval(j, deg2rad(degree(i)))
            ;
111     end
112 end
113 figure('Name','oval_reconstruction');
114 imshow(oval_image,[])
115 sum_image = circle_image + oval_image;
116 figure('Name','final_reconstruction');
117 imshow(sum_image,[])
118 datasheet2 = xlsread('../A/A题附件.xls','附件2');
119 datasheet2 = reshape(datasheet2, 1, 512*180);
120 rc = reshape(sum_image, 1, 512*180);
121 figure('Name','cmp');
122 plot(datasheet2, rc);

```

程序 2：差分 ReLU 运算代码

```

1 function [ output ] = Diff( input, relu )
2 %DIFF 对矩阵作一次差分处理
3 % 对矩阵的每个元素，都减去它的前一个元素。
4     slide = [zeros(1, 180); input(1 : 511, :)];
5     output = input - slide;
6     if relu

```



```

7         output = max(output, 0);
8     end
9 end

```

程序 3 : 滤波反投影图像重建代码

```

1  clc; clear
2  %load datasheet1
3  load datasheet2
4  load datasheet3
5  load datasheet5
6  load degree
7  R = datasheet5; % 512 x 180
8
9  l = pow2(nextpow2(size(R,1))-1); %重构图像的大小 256 x 256
10
11 %滤波反投影法
12 N=180;
13 %滤波
14 H=size(R,1); % 512
15 h=zeros((H*2-1),1);
16 for i=0:H-1
17     if i==0
18         h(H-i)=1/4;
19     elseif rem(i,2)==0
20         h(H-i)=0;
21         h(H+i)=0;
22     else
23         h(H-i)=-1/(i*pi)^2;
24         h(H+i)=-1/(i*pi)^2;
25     end
26 end
27 x=zeros(H,N); % 512 x 180
28 for i=1:N
29     s=R(:,i);
30     xx=conv(s',h'); % 1 x 1534
31     x(:,i)=xx(H:2*H-1);
32 end
33
34 %反投影
35 Projection=zeros(l,l);
36 tmp = [];
37 overUpperBound=0;
38 overLowerBound=0;
39 for i=1:l
40     for j=1:l
41         for k=1:180
42             theta=(degree(k)-90)/180*pi;
43             % 中心坐标 (39.1272, 56.3769) 转化后的坐标

```



```

44         (100.1656, 111.6751)
45         % 39.1272/100*256 = 100.1656
46         % (100-56.3769)/100*256 = 111.6751
47         % 512 - 111.6751 * 2 = 288.6498
48         % 因为 256 * sqrt(2) * sqrt(2) = 512
49         t=((j - 100.1656)*cos(theta)+(111.6751-i)*sin(
50             theta))*sqrt(2)+256.5;
51         if t > 511
52             t = 511;
53             overUpperBound=overUpperBound+1;
54         elseif t < 1
55             t = 1;
56             overLowerBound=overLowerBound+1;
57         end
58         t1=floor(t);
59         t2=floor(t+1);
60         Projection(i,j)=Projection(i,j)+(t2-t)*x(t1,k)+(
61             t-t1)*x(t2,k);
62         %tmp=[tmp [i;j;k]];
63     end
64 end
65 Projection=pi/N*Projection;
66 save('Projection_Mat','Projection');
67 tmp_datasheet1 = zeros([1, 256*256]);
68 tmp_datasheet2 = zeros([1, 256*256]);
69
70 for i = 1:256
71     for j = 1:256
72         if (Projection(i,j)<0.2)
73             Projection(i,j)=0;
74         end
75     end
76 end
77 %Projection = Projection .* 2;
78
79 x = 1:256*256;
80 for i = 1:256
81     for j = 1:256
82         tmp_datasheet1((i-1)*256+j)=datasheet1(i,j);
83         tmp_datasheet2((i-1)*256+j)=Projection(i,j);
84     end
85 end
86 figure;imshow(Projection,[]);
87 xlswrite('result/problem3.xls', Projection , 'sheet1');

```

程序 4：局部预测重排程序代码

```

1 function [ output ] = preprocess( input, n )
2 %PREPROCESS 对抽取出的边界坐标进行预处理
3 % 输出矩阵每一列包括：[椭圆投影起始坐标，椭圆投影结束坐标，圆投影起始坐
   标，圆投影结束坐标]
4 % 每次根据前 2 列结果对下一列坐标进行预测，并根据预测结果重新排列（需要
   时补正）下一列坐标
5 % 本算法将初始 2 列数据的排列作为 ground truth，自动排列所有后续数据
6 output = input;
7 target = zeros(n, 1); % 存储预测位置
8 for i = 3:180
9     % 使用（局部）线性插值预测下一个坐标
10    for j = 1:n
11        target(j, 1) = 2 * output(j, i-1) - output(j, i-2);
12    end
13    vacancy = length(find(input(:, i) == 1));
14    res = zeros(n, 1); % 存储对实际位置的排列结果
15    occupy = zeros(n, 1); % 用于对候选坐标进行分配
16    if vacancy > 0 % 存在空坐标，说明有坐标重合了
17        for j = vacancy + 1 : n
18            current = 0;
19            dist = inf;
20            for k = 1:n
21                if occupy(k, 1) == 0 && abs(input(j, i) -
                    target(k, 1)) < dist
22                    dist = abs(input(j, i) - target(k, 1));
23                    current = k;
24                end
25            end
26            occupy(current, 1) = 1;
27            res(current, 1) = input(j, i);
28        end
29        [x, y] = find(occupy == 0);
30        current = vacancy+1;
31        for j = vacancy+2:n
32            if abs(input(j, i) - target(x, 1)) < abs(input(
                current, i) - target(x, 1))
33                current = j;
34            end
35        end
36        res(x, 1) = input(current, i);
37    else % 直接根据预测坐标大小顺序，排列实际坐标
38        [value, index] = sort(target, 1, 'ascend');
39        res(index, 1) = input(:, i);
40    end
41    output(:, i) = res;
42
43 end

```

程序 5：拟合函数代码

```

1 function F = cosfunc(cosparam, x)
2 F = cosparam(1) * cos(x + cosparam(2)) + cosparam(3) ;

```

程序 6 : 计算吸收率代码

```

1 clc;clear;
2 %x=[10.0000 34.5000 43.5000 45.0000 48.5000 50.0000 56.0000
   65.5000 79.5000 98.5000];
3 y=[18.0000 25.0000 33.0000 75.5000 55.5000 75.5000 76.5000
   37.0000 18.0000 43.5000];
4
5 Projection = xlsread('result/problem3.xls');
6
7 y_new = floor(x .* 2.56);
8 x_new = 256 - floor(y .* 2.56);
9
10 result = zeros([1 10]);
11 for i=1:10
12     value=Projection(x_new(i), y_new(i)) + Projection(x_new(
        i)+1,y_new(i)) + Projection(x_new(i)-1, y_new(i)) ...
13         + Projection(x_new(i), y_new(i)+1) + Projection(
        x_new(i)+1,y_new(i)+1) + Projection(x_new(i)-1,
        y_new(i)+1) ...
14         + Projection(x_new(i), y_new(i)-1) + Projection(
        x_new(i)+1,y_new(i)-1) + Projection(x_new(i)-1,
        y_new(i)-1);
15     result(i)=value/9;
16 end
17
18 figure('Name','Fig1');
19 imshow(Projection,[]);
20 hold on
21 scatter(y_new, x_new, 'w');
22 result

```

程序 7 : 问题四入口代码

```

1 for x = 1:100
2     new_template
3     res(1,:) = res(1,:) + [res1, res2, res3, res4, res5];
4     old_template
5     res(2,:) = res(2,:) + [res1, res2, res3, res4, res5];
6 end
7 storage = res

```

程序 8 : 新标定模板测试代码

```

1 params = [[50 50 20];[20 80 15];[80 80 10];[20 20 15];[80 20
    10]];
2 interval = normrnd(0.3, 0.05);
3 begin_deg = normrnd(100, 5);
4 original_degree = begin_deg:begin_deg+179;
5 zero = normrnd(256.5, 10);
6 posX = normrnd(50, 10);;
7 posY = normrnd(50, 10);
8
9 circle_image = zeros(5, 512, 128);
10 diff_image = zeros(5, 512, 128);
11 loc = zeros(5, 2, 180);
12 for k = 1:5
13     for i = 1:180
14         for j = 1:512
15             circle_image(k, j, i) = cross_circle_2(j,
                deg2rad(original_degree(i)), posX, posY,
                params(k, 1), params(k, 2), zero, interval,
                params(k, 3));
16         end
17     end
18     diff_image = Diff(reshape(circle_image(k, :, :), 512,
        180), false);
19     [value, index] = sort(diff_image, 1, 'descend');
20     loc(k, :, :) = index([1 512], :);
21 end
22 sum_image = reshape(sum(circle_image), 512, 180);
23 imshow(sum_image, []);
24
25 Interval = 2 * params(1,3) / mean(loc(1, 2, :) - loc(1, 1,
    :), 3);
26
27 cent = reshape((loc(2:5, 2, :) + loc(2:5, 1, :)) / 2, 4,
    180);
28 proj_1 = (cent(2, :, :) - cent(3, :, :)) / 60 / sqrt(2) *
    Interval;
29 proj_1 = proj_1 / max(abs(proj_1));
30 degree1 = rad2deg(asin(proj_1) + pi/4);
31 b = find(degree1 > 130, 1, 'first');
32 degree1 = [degree1(1:b), degree1(b)*2-degree1(b+1:180)];
33 proj_2 = (cent(4, :, :) - cent(1, :, :)) / 60 / sqrt(2) *
    Interval;
34 proj_2 = proj_2 / max(abs(proj_2));
35 degree2 = 180 - rad2deg(asin(proj_2) + pi/4);
36 b = find(degree2 > 220, 1, 'first');
37 degree2 = [degree2(1:b), degree2(b)*2-degree2(b+1:180)];
38
39 degree = (degree1 + degree2) / 2;
40 xdata = deg2rad(degree(1))-deg2rad(degree);

```

```

41| options = optimset('MaxFunEvals',10000000); %改写一个参数，用于
    下面函数 lsqcurvefit 的设置
42|
43| radius = zeros(1, 4);
44| theta = zeros(1, 4);
45| pos = zeros(4, 2);
46| for k=1:4
47|     cosparam_est1 = lsqcurvefit(@(cosparam, xdata) cosfunc(
        cosparam, xdata), [141 0 256], xdata, cent(k,:));
48|     radius(1, k) = Interval * cosparam_est1(1);
49|     theta(1, k) = cosparam_est1(2) + deg2rad(degree(1)) - pi
        /2;
50|     pos(k, :) = [params(k+1, 1) - radius(1, k) * cos(theta
        (1, k)) ;params(k+1, 2) - radius(1, k) * sin(theta(1,
        k))];
51| end
52| pos = mean(pos, 1);
53|
54| res1 = abs(interval-Interval);
55| res2 = res1/interval*100;
56| res3 = mean(abs(degree - original_degree));
57| res4 = mean(abs((degree - original_degree) ./
        original_degree));
58| res5 = sqrt((pos(1)-posX)*(pos(1)-posX)+(pos(2)-posY)*(pos
        (2)-posY));

```

程序 9：原始标定模板测试代码

```

1| circle_image = zeros(512, 128);
2| oval_image = zeros(512, 128);
3| loc = zeros(4, 180);
4| for i = 1:180
5|     for j = 1:512
6|         circle_image(j, i) = cross_circle(j, deg2rad(
            original_degree(i)));
7|         oval_image(j, i) = cross_oval(j, deg2rad(
            original_degree(i)));
8|     end
9| end
10| diff_image = Diff(oval_image, false);
11| [value, index] = sort(diff_image, 1, 'descend');
12| loc([1 2], :) = index([1 512], :);
13| diff_image = Diff(circle_image, false);
14| [value, index] = sort(diff_image, 1, 'descend');
15| loc([3 4], :) = index([1 512], :);
16| coord = loc;
17| % 小圆的投影探测器个数
18| circle_projection=coord(4,:) - coord(3,:);
19| % 椭圆的投影探测器个数

```

```

20 oval_projection=coord(2,:) - coord(1,:);
21
22 mid_value_circle = coord(3,:) + circle_projection ./ 2;
23 mid_value_oval = coord(1,:) + oval_projection ./ 2;
24 circle_mean_num = mean(circle_projection);
25 detector_real_distance = (4 * 2) / circle_mean_num;
26 mid_distance = (mid_value_circle - mid_value_oval) .* (
    detector_real_distance);
27 mid_distance = mid_distance / max(abs(mid_distance));
28 degree = rad2deg(asin(mid_distance(1:180)));
29 degree = 180 - degree;
30 b = find(degree > 268, 1, 'first');
31 degree = [degree(1:b), degree(b)*2-degree(b+1:180)];
32 xdata = deg2rad(degree(1))-deg2rad(degree);
33 options = optimset('MaxFunEvals',10000000); %改写一个参数,用于
    下面函数 lsqcurvefit 的设置
34 cosparam_est1 = lsqcurvefit(@(cosparam, xdata) cosfunc(
    cosparam, xdata), [206 0 239], xdata, mid_value_circle);
35 cosparam_est2 = lsqcurvefit(@(cosparam, xdata) cosfunc(
    cosparam, xdata), [46 0 247], xdata, mid_value_oval);
36
37 radius_circle = detector_real_distance * cosparam_est1(1);
38 theta_circle = cosparam_est1(2) + deg2rad(degree(1)) - pi/2;
39 pos_from_circle = [95 - radius_circle * cos(theta_circle)
    ;50 - radius_circle * sin(theta_circle)]
40 radius_oval = detector_real_distance * cosparam_est2(1);
41 theta_oval = cosparam_est2(2) + deg2rad(degree(1)) - pi/2;
42 pos_from_oval = [50 - radius_oval * cos(theta_oval) ;50 -
    radius_oval * sin(theta_oval)];
43 ang = acos((radius_oval*radius_oval + 45*45 - radius_circle*
    radius_circle)/2/45/radius_oval);
44 final_pos = [50 + radius_oval * cos(ang); 50 + radius_oval *
    sin(ang)];
45
46 ttt = theta_circle - theta_oval;
47 radius_circle = 45 / abs(sin(ttt)) * abs(sin(theta_oval));
48 final_pos_2 = [95 - radius_circle * cos(theta_circle) ;50 -
    radius_circle * sin(theta_circle)];
49
50 pos = (pos_from_circle+pos_from_oval+final_pos+final_pos_2)
    /4
51
52 res1 = abs(interval-detector_real_distance);
53 res2 = res1/interval*100;
54 res3 = mean(abs(degree - original_degree));
55 res4 = mean(abs((degree - original_degree) ./
    original_degree));
56 res5 = sqrt((pos(1)-posX)*(pos(1)-posX)+(pos(2)-posY)*(pos
    (2)-posY));

```