5 MRI Scanners[①]

5.1 竞赛题

Introduction

Industrial and medical diagnostic machines known as Magnetic Resonance Imagers (MRI) scan a three-dimensional object such as a brain, and deliver their results in the form of a three-dimensional array of pixels. Each pixel consists of one number indicating a color or a shade of grap that encodes a measure of water concentration in a small region of the scanned object at the location of the pixel. For instance, 0 can picture high water concentration in black (ventricles, blood vessels), 128 can picture a medium water concentration in gray (brain nuclei and gray matter), and 255 can picture a low water density in white (lipid-rich white matter consisting of myelinated axons). Such MRI scanners also include facilities to picture on a screen any horizontal or vertical slice through the three-dimensional array (slices are parallel to any of the three Cartesian coordinate axes). Algorithms for picturing slices through oblique planes, however, are proprietary. Current algorithms are limited in terms of the angles and parameter options available; are implemented only on heavily used dedicated workstations; lack input capabilities for marking points in the picture before slicing; and tend to blur and "feather out" sharp boundaries between the original pixels.

① 1998年美国大学生数学建模竞赛A题。

A more faithful, flexible algorithm implemented on a personal computer would be useful

(1) for planning minimally invasive treatments,

(2) for calibrating the MRI machines,

(3) for investigating structures oriented obliquely in space, such as post-mortem tissue sections in animal research,

(4) for enabling cross-sections at any angle through a brain atlas consisting of black-and-white line drawings.

To design such an algorithm, one can access the values and locations of the pixels, but not the initial data gathered by the scanner.

Problem

Design and test an algorithm that produces sections of three-dimensional arrays by planes in any orientation in space, preserving the original gray-scale values as closely as possible.

Data Sets

The typical data set consists of a three-dimensional array A of numbers $A(i,j,k)$ which indicates the density $A(i,j,k)$ of the object at the location $(x,y,z)_{ijk}$. Typically, $A(i,j,k)$ can range from 0 through 255. In most applications, the data set is quite large.

Teams should design data sets to test and demonstrate their algorithms. The data sets should reflect conditions likely to be of diagnostic interest. Teams should also characterize data sets that limit the effectiveness of their algorithms.

Summary

The algorithm must produce a picture of the slice of the three-dimensional array by a plane in space. The plane can have any orienta-

tion and any location in space. (The plane can miss some or all data points.) The result of the algorithm should be a model of the density of the scanned object over the selected plane.

赛题简析

本题有着非常明确的医学背景,MRI 称为磁共振成像器,该仪器可对身体的各个部位进行扫描,以诊断体内的某些疾病及异常,这种手段已越来越多地被医学界所采用,例如人体肿瘤的探测,由于肿瘤出现的形态不同,在取截面时,边界的清晰度以及区域的光滑程度就显得尤为重要,本问题就是针对这类问题提出来的。

用磁共振成像器扫描三维物体,产生一个三维点阵。其中的每个点是代表颜色或灰度的数(0—255),它表示物体在该点的小邻域中水浓度的度量。例如 0 用黑色描绘高水浓度(脑室,血脉管),128 用灰色描绘中等水浓度(脑核与灰色物质),255 用白色描绘低水浓度(有髓鞘的轴突组成的白色物质)。核磁共振成象器扫描还包含在屏幕上画出穿过三维点阵的水平和垂直切片的设备。

要求设计一个算法,产生由任意方向的平面与三维点阵相交的截面,尽可能的保持原始的灰度值。要求参赛者设计的算法必须生成任意方向的平面与三维点阵相交的截面的图形,不能只是矩阵的形式。参赛者还应当:(1)设计数据集检验和证明其算法。(2)生成反映诊断值的数据集。(3)指出数据集中限制其算法有效性的特征。

一篇好的论文除了完成上述要求外,还应当给出算法的误差分析。文章写作应清晰流畅。

5.2 参赛论文一

Image Reconstruction in MRI

Hong Wei Sun Bo Guo Wei
Advisor He Xiaoliang

Summary

Our model takes the water concentration of the scanned object as a continuous-space three-dimensional signal. The given three-dimensional array of pixels from MRI is taken as three-dimensional sampling of the signal. The picture of the slice of the three-dimensional array by an oblique plane in space is also a two-dimensional array of pixels, which is also a two-dimensional sampling of the continuous-space signal on the plane. To get the picture we recovers the continuous-space signal from the given array with a certain low-pass-filter first. Then, we re-samples the recovered continuous-space signal on the oblique plane to get the required two-dimensional array.

The most important part of our algorithm is how to select the low-pass-filter to recover the continuous-space signal as precisely and quickly as possible. We use linear interpolation, rectangle window function and Hamming window function to recover the continuous-space signal and compare their performances. Besides the analysis of those interpolation functions in theory, we also give two criterion to judge the quality of the output array.

The actual three-dimensional array from MRI contains high frequency noises. We should restrain the noise when recover the signal. Taking

all these factors as a whole, we find that Hamming window function is a better choice as an interpolation function both in theory and from the result of out simulations on a personal computer.

Assumptions

- The distribution of the water concentration of the scanned object is a three-dimensional function which satisfy the Dirichlet condition.
- The sampling intervals when MRI scans the object is so small that the details of the object is preserved in the given array.
- It is possible that high frequency noise is added to the samples when MRI scans the object.
- The resolution of the slice is equal to the resolution of the given array.
- The sampling intervals when MRI scans the object is the same in the three Cartesian axes directions.

Symbols, expressions

$A_u(x,y,z)$	the water concentration in a small region of the scanned object at the location (x,y,z)
$A(i,j,k)$	the three-dimensional array of pixels got from MRI
P	the specified plane to picture on it
n	the unit normal vector of plane $P(n=(n_x,n_y,n_z))$
$A(m,n)$	the final output two-dimensional array of pixel
$w(x)$	the interpolation function used to recover the continuous-space signal
$N(x,y,z)$	the high frequency noise distributed in three-dimensional space

Analysis of the Problem

- Provided that the water concentration of the scanned object is a

continuous-space signal $A_a(x, y, z)$ in the three-dimensional space, the given three-dimensional array of pixels $A(i, j, k)$ is taken as samples of $A_a(x, y, z)$ in the three-dimensional space; i.e.,

$$A(i, j, k) = A_a(i\delta x, j\delta y, k\delta z) \tag{1}$$

δx, δy, δz represent the sampling intervals in the three Cartesian axes. To simplify the discussion of the problem, we suppose that $\delta x = \delta y = \delta z = 1$. Thus,

$$A(i, j, k) = A_a(i, j, k) \tag{2}$$

• Suppose the equation of the oblique plane P is

$$n_x x + n_y y + n_z z + c = 0 \tag{3}$$

$n = (n_x, n_y, n_z)$ represent the unit normal vector of the plane(No matter how the plane is specified, Eq. (3) can be derived from the input). Construct a two-dimensional Cartesian coordination in the plane(the method of the construction is described later). If (u, v) represent a pixel on the plane in that coordination and $A_a(u, v)$ represent the water concentration in the small region at the location of the pixel, we get

$$A_a(u, v) = A_a(x, y, z)\,|_{(x,y,z)\in P} \tag{4}$$

The final result of the algorithm should be samples of $A_a(u, v)$; i.e., the output two-dimensional array

$$A(m, n) = A_a(m\delta u, n\delta v) \tag{5}$$

To have the same resolution with the resulting array from MRI, we take $\delta u = \delta v = 1$. Then,

$$A(m, n) = A_a(m, n) \tag{6}$$

• As mentioned above, our algorithm takes two steps to get the two-dimensional array $A(m, n)$ from the original three dimensional array $A(i, j, k)$,

i) recover the continuous-space signal $A_a(x, y, z)$;

ii) take the samples of $A_a(u, v)$ got from $A_a(x, y, z)$ as the final two-dimensional array.

Recovering Signal

According to Nyquist principle, to recover $A_a(x, y, z)$ from $A(i, j, k)$ precisely, $A_a(x, y, z)$ must be a band-limited signal and the sampling frequency must at least twice the highest frequency of $A_a(x, y, z)$. Since the sampling intervals $\delta x = \delta y = \delta z = 1$, the sampling frequencies in the three Cartesian axes are all "1". Suppose the spectrum of of $A_a(x, y, z)$ is $F_a(w_x, w_y, w_z)$. Then

$$F_a(w_x, w_y, w_z) = \begin{cases} A_a(w_x, w_y, w_z), \ if \begin{cases} |w_x| < \frac{1}{2}, |w_y| < \frac{1}{2}, \\ |w_z| < \frac{1}{2} \end{cases} \\ 0, \ otherwise \end{cases} \tag{7}$$

We can recover $A_a(x, y, z)$ from $A(i, j, k)$ precisely by a low-pass-filter (with 0.5 as its cut-off frequency) and we get

$$A_a(x, y, z) = \sum_{i=-\infty}^{+\infty} \sum_{j=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(i, j, k) w(x - i) w(y - j) w(z - k) \tag{8}$$

where $w(x)$ is the interpolation function whose spectrum is the low-pass-filter.

Selection of a Practical Low-Pass-Filter

Since addition is computed in an infinite domain, it's impossible to recover $A_a(x, y, z)$ from $A(i, j, k)$ through Eq. (8) on a personal computer. To make the algorithm realizable, we must make some approximations and optimizations. Giving

$$A_c(x, y, z) = \sum_{i=-E}^{+E} \sum_{j=-E}^{+E} \sum_{k=-E}^{+E} A(i, j, k) w(x - i) w(y - j) w(z - k) \tag{9}$$

we have

$$A_a(x,y,z) = \lim_{E\to\infty} A_c(x,y,z) \tag{10}$$

Since $w(x)\to 0$ when $x\to\infty$ if E is large enough, $A_c(x,y,z)$ is very approximate to $A_a(x,y,z)$. From Eq. (9), we find that the computation for one pixel in space is proportional to $(2E+1)^3$. To get a faster algorithm, we should decrease E as much as possible while maintaining the precision in a certain degree.

Rectangle window

• The ideal low-pass-filter in theory is a rectangle window. Then

$$w(x) = \mathrm{sinc}(x) \tag{11}$$

where

$$\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \tag{12}$$

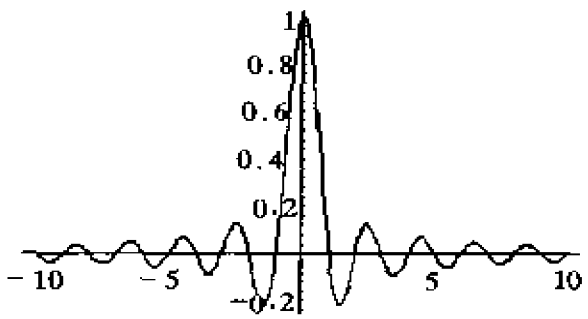

Fig. 1 $\mathrm{sinc}(x)$

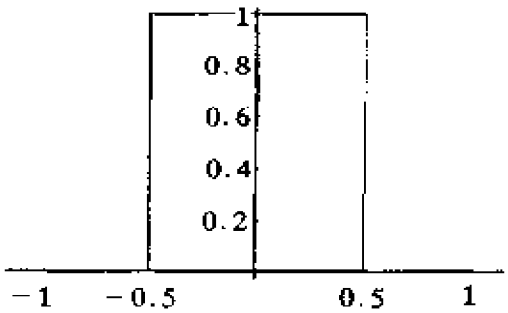

Fig. 2 Spectrum of rectangle window

But as shown by Fig. 1, the side lobes of $\mathrm{sinc}(x)$ are not insignificant and the attenuation of $\mathrm{sinc}(x)$ is not very fast, E must be relatively large not to lose precision. So the computation work is great and the speed is slow. According to our simulation, if E is greater than 5, the increase of E has little affection to the result. So we use $(2\times 5+1)^3$ pixels to recover one pixel on the oblique plane.

• The size of an actual object is finite so the signal isn't band-limited. The frequency $F_a(w_x, w_y, w_z)$ is aliased; i.e., the high frequency part of $F_a(w_x, w_y, w_z)$ will be strengthened after the sampling. Thus it's impossible to recovering $A_a(x, y, z)$ from $A(i, j, k)$ precisely by an ideal low-pass-filter.

• Taking the high frequency noises into account, the given array $A(i, j, k)$ is the samples of $A_a(x, y, z)$ added by a three-dimensional distributed noise $N(x, y, z)$ at the location (i, j, k); i.e.

$$A(i, j, k) = A_a(i, j, k) + N(i, j, k) \tag{13}$$

So after the sampling, the high frequency part of the spectrum of $A_a(x, y, z)$ is added by the spectrum of the high frequency noise. ! We should restrain that high frequency noise when recovering $A_a(x, y, z)$. Since the magnitude of the rectangle window is unchanged inside the pass-band, there is no high frequency restraint to the noise.

• If there are sharp boundaries in the original signal, there will be Gibbs phenomenon near the boundaries when recovering the signal using Eq. (9) with a finite E. In Fig. 3, the black dot represent samples of a signal with two sharp boundaries. Recover the signal from those samples with $\mathrm{sinc}(x)$ as the interpolation function, we get the continuous curve shown in the figure. Oscillation near the boundaries is very obvious. It makes the boundaries even sharper. But it also has bad effect at the smooth part of the signal. If the boundaries is very important than the smooth part, $\mathrm{sinc}(x)$ should be used as the interpolation function.

Linear interpolation

To decrease the computation, we can select other interpolation function w(x) whose side lobe is more insignificant than $\mathrm{sinc}(x)$.

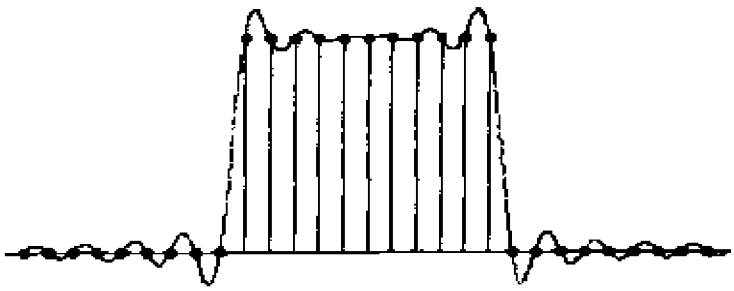

Fig. 3 Recovering a signal with two sharp boundaries from samples with $\mathrm{sinc}(x)$

One choice is to use the linear interpolation function; i. e.,

$$w(x) = \begin{cases} 1 - |x|, & |x| \leqslant 1 \\ 0, & |x| > 1 \end{cases} \tag{14}$$

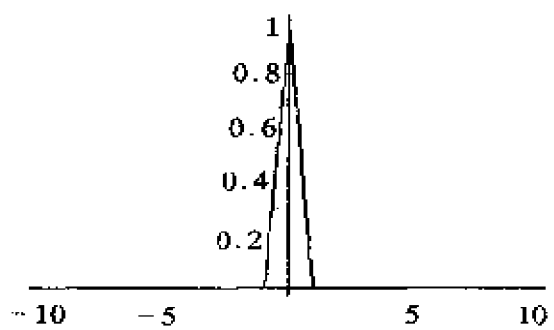

Fig. 4 Linear interpolation function

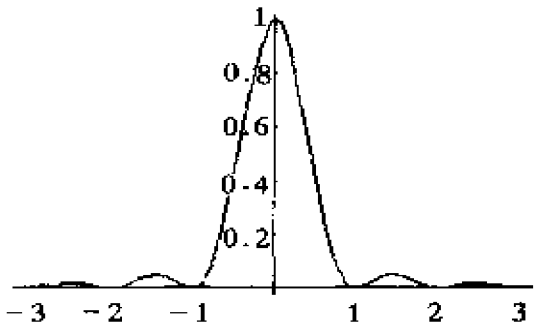

Fig. 5 Spectrum of the linear interpolation function

• Its side lobe is 0, so we can specify $E = 1$ and get the least computation. Since $E = 1$, we need only compute $(2 + 1)^3$ pixel to recover one pixel on the oblique plane.

• The magnitude of its spectrum attenuates from 1 at 0 to 0.41 at 0.5(See Fig. 5). So it restrains high frequency noise in a certain degree.

The attenuation of its spectrum near 0.5 can also compensate for the

aliasing.

Because the spectrum of the linear interpolation function is not 0 in the exterior of $(-0.5,0.5)$(See Fig. 5), the periodic repetitions of the spectrum of $A_a(x,y,z)$ is not filtered completely. Then extra high-frequency part will be introduced in the spectrum of the recovered signal.

• Fig. 6 demonstrates the recovering of a signal with two sharp boundaries from samples with linear interpolation. It can be seen that the sharp boundaries are well maintained while preserving the smooth part of the signal.

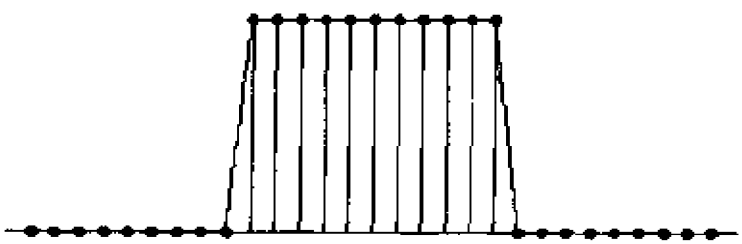

Fig. 6 Recovering a signal with two sharp boundaries by samples with linear interpolation

Hamming window

To have a greater restraint of the high frequency noise and filter the periodic repetitions of the spectrum of the original signal, consider *Hamming window function*

$$w(x)=\begin{cases}0.54\sin c(x)+0.23\sin c(x-1)+0.23\sin c(x+1), \\ \qquad\qquad\qquad\qquad\qquad\qquad |x|\leqslant 0.5 \\ 0, \text{ if } |x|>0.5\end{cases} \tag{15}$$

• It can be seen from Fig. 7 that Hamming function has a more insignificant side lobe than $\sin c(x)$, so E can be smaller and the com-

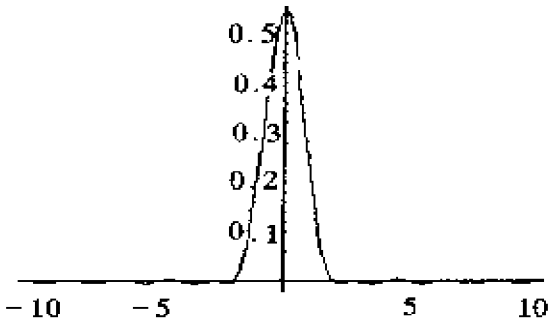

Fig. 7 Interpolation function using Hamming window

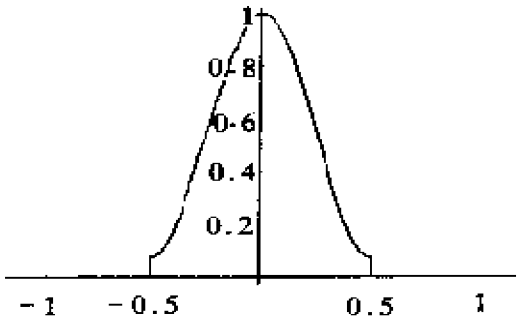

Fig. 8 Spectrum of Hamming window

putation will decrease. According to our simulation, if E is greater than 2, the increase of E has little affection to the result. So we use $(2 \times 2 + 1)^3$ pixel to recover one pixel on the oblique plane with Hamming window as the interpolation function.

• In frequency domain, it is a low-pass-filter band-limited in $(-0.5, 0.5)$(See Fig. 8), thus it will not introduce extra high frequency part.

• The magnitude of the spectrum of Hamming window attenuates from 1 at 0 to 0.08 at 0.5(See Fig. 8). So it restrains the high frequency noise more thoroughly than linear interpolation.

The attenuation of its spectrum near 0.5 can also compensate for the aliasing.

• Fig. 9 shows the recovering of a signal with two sharp boundaries from samples with Hamming window as the interpolation function. The sharp boundaries are not recovered clearly.

Compensation at the edge of the array

Since $A(i, j, k)$ is obtained through the output of MRI, the value of $A(i, j, k)$ is only specified in that array; but in the computation

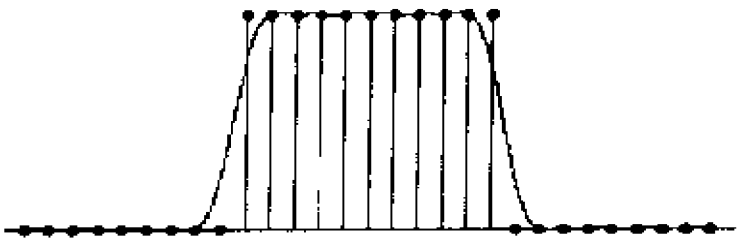

Fig. 9 Recovering a signal with two sharp boundaries by samples with Hamming window

of $A_c(x, y, z)$ by Eq. 2, it's possible to access the value of $A(i, j, k)$ *beyond the array. It's not reasonable to take the value* 0 in such situations. We take the value of $A(i, j, k)$ at the edge of the array as the value beyond the array.

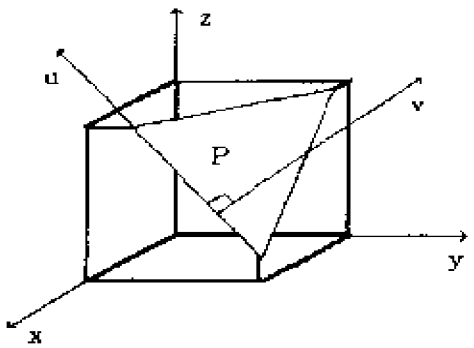

Fig. 10 The plane P in the three-dimensional space

Obtaining the Location of Sampling Pixels

The final output of our algorithm is $A(m, n)$, the samplings of $A_a(u, v)$. So we need to construct a two-dimensional Cartesian coordinate on Plane P. We take the direction perpendicular to the x axis in the three-dimensional coordinate as the direction of the u axes in the two-dimensional Cartesian coordinate on P. The unit vec-

tor $u=(0, u_y, u_z)$ of the u axis on P can be expressed in the three-dimensional coordinate through equation

$$u \cdot n = 0 \tag{16}$$

When u is specified, the unit vector $v=(v_x, v_y, v_z)$ of the v axis on P can be expressed in the three-dimensional coordinate through equations

$$u \cdot v = 0 \tag{17}$$

$$v \cdot n = 0 \tag{18}$$

Finally, we get

$$u_y = \frac{1}{\sqrt{1+(\frac{n_y}{n_z})^2}} \tag{19}$$

$$v_x = \sqrt{\frac{1+(\frac{n_z}{n_y})^2}{1+(\frac{n_z}{n_y})^2+(\frac{n_x}{n_y})^2}} \tag{20}$$

$$v_y = -\frac{n_x}{n_y(1+(\frac{n_z}{n_y})^2)} v_x \tag{21}$$

When u and v specified, we can choose an arbitrary pixel $O(o_x, o_y, o_z)$ on P as the origin of the two-dimensional coordinate. Then pixel $Q(q_x, q_x, q_z)$ will be a sampling pixel if

$$\begin{cases} q_x = o_x + mv_x \\ q_y = o_y + mv_y + nu_y \\ n_x q_x + n_y q_y + n_z q_z + c = 0 \end{cases} \tag{22}$$

and we get

$$A(m, n) = A_o(q_x, q_y, q_z) \tag{23}$$

How to Judge the Result

In order to compare those interpolation functions in this particular

problem, we introduce two criterion to represent the deviation between the recovered $A(u,v)$ and the original $A_a(u,v)$.

(1) Normalized mean square distance criterion d.

$$d = \sqrt{\frac{\sum_{(m,n)\in P}[A_a(m,n)-A(m,n)]^2}{\sum_{(m,n)\in P}[A_a(m,n)-\bar{A}]^2}} \tag{24}$$

where $\bar{A}$ represent the average of the water concentration of the scanned object. If $d=0$, it means that the recovering is extremely precise and there is no deviation. The larger d is, the larger the diviation is.

(2) Normalized mean absolute distance criterion r.

$$r = \frac{\sum_{(m,n)\in P}|A_a(m,n)-A(m,n)|}{\sum_{(m,n)\in P}|A_a(m,n)|} \tag{25}$$

If $r=0$, it means that the original signal is precisely recovered. The larger r is, the larger the diviation is.

Though both criteria can be used to represent the deviation between the original signal and the recovered signal, they give different judging of the quality of the recovered signal on the oblique plane. d is more sensitive to the situation where there are little pixels with relatively large deviations while r is more sensitive to the situation where there are lots of pixels with relatively small deviations.

Test of our model

We simulate a three-dimensional object, consisting of a water concentration distribution produced by function

$$A_a(x,y,z) = 130 + 100\sin(x/2)\sin(y/2) \tag{26}$$

in a sphere and

$$A_a(x,y,z) = 50 \tag{27}$$

outside the sphere to test our model.

Provided the given array is a $36\times36\times36$ array and the oblique plane P in space is specified with $n=(1/2,1/2,1/\sqrt{2})$ as its unit normal vector and pixel $(35,35,0)$ on P. The ideal slice (Fig. 11)is calculated from the distribution function $A_a(x,y,z)$ on the slice.

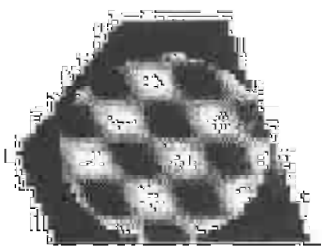

Fig. 11 the original water concentration distribution of the object on P

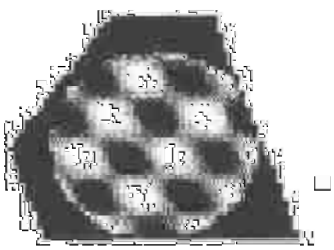

Fig. 12.1 The slice recovered with rectangle window

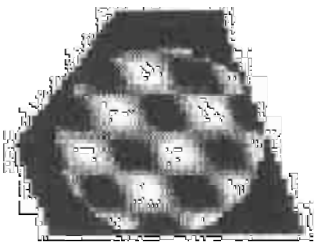

Fig. 12.2 The slice recovered with linear interpolation

• Provided no high frequency noise is introduced in sampling. we get the following result from the three interpolation functions:

Table 1

	rectangle window function	linear interpolation	Hamming window function
d	0.195	0.176	0.231
r	0.046	0.032	0.062

From Table 1, we find that the linear interpolation produces the best slice. It recovers the signal most closely and the sharp boundary is preserved. We can see the oscillation near the boundaries of the object clearly when recovering with rectangle window.

Provided that the noise distribution is

$$A_a(x,y,z)=20\sin(3z) \tag{28}$$

We get the following result from the three interpolation functions:

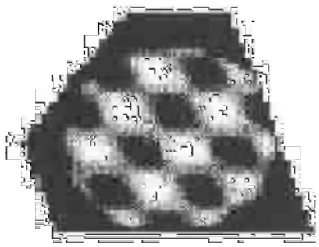

Fig. 12.3 The slice recovered with Hamming window

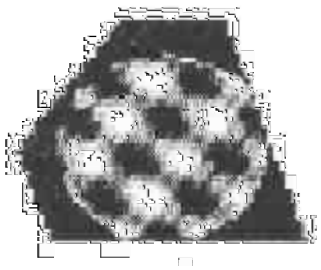

Fig. 13.1 The slice recovered with rectangle window

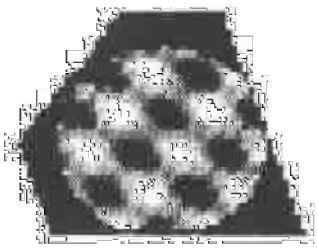

Fig. 13.2 The slice recovered with linear interpolation

Table 2

	rectangle window function	linear interpolation	Hamming window function
d	0.286	0.235	0.233
r	0.115	0.083	0.066

From Table 2 and Fig. 13, the Hamming window restrains the high frequency noise and recovers the original signal most exactly. The effect of the noises is obvious in the signal recovered with rectangle window or linear interpolation.

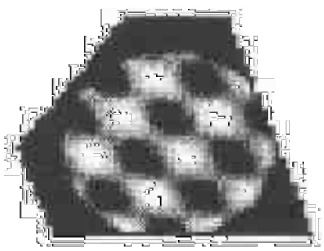

Fig. 13.3 The slice recovered Hamming window

Final Conclusions

From the analysis in theory and the result from our simulation, we find

• If the high frequency noise is insignificant, linear interpolation is a good choice in practice.

• If the high frequency noise must be taken into account, using Hamming window as the interpolation function is proper to filter the noise.

In practice, the high frequency noise isn't insignificant, so we think Hamming window is favorite as an interpolation function.

Strengths and Weaknesses of Our Model

Strengths

• We takes the effect of high frequency noise into account and restrains the noise if necessary.

• When the size of the given array increases, the computation needed to recover one pixel does not increase.

• Our method of recovering and re-sampling is not related to the shape of the surface. So if we need the distribution of the water concentration of an object on an arbitrary curved surface, our algorithm is still useful.

Weaknesses

• The preserving of sharp boundaries and restraining of high frequency noise are conflicting requirements. Using Hamming window as the interpolation function, the sharp boundaries are not recovered very clearly.

References

Oppenheim A. V., Schafer R. W.: Digital Signal Processing. New York: Prentice Hall, 1975.

Zhuang Tiange. Principles and Algorithms of CT. Shanghai: The Press of Shanghai Jiaotong University.

Oppenheim A. V., Willsky A. S., Young. I. T., Signals and Systems. New York: Prentice Hall, 1983.

论文点评

论文建立的模型是将被扫描物体作为三维信号的连续空间，将 MRI 扫描出的点阵作为三维信号的采样。倾斜平面与三维点阵相交的截面是二维连续空间信号的采样。

文中采用了低 — 通 — 滤波器从给定的点阵来恢复连续信号，再从二维截面上恢复的连续信号进行采样以得到需要的二维点阵。

文中算法的关键点是选择低 — 通 — 滤波器使得尽量精确和迅速地恢复连续信号。通过使用线性插值、矩形窗口函数、Hamming 窗口函数来恢复连续信号并比较它们的性能。通过比较发现，Hamming 窗口函数在理论和模拟的结果上都是最好的。文中还给出了判别输出点阵质量的两个准则，并且讨论了高频噪声的限制问题。

论文论证正确，写作简明流畅。

论文特点：

1. 建模中使用的方法是有效的。

2. 论文不仅很好地完成了题目的要求，即设计的算法能生成任意方向的平面与三维点阵相交的截面的图形，而且该算法还能生成三维点阵中曲面的图形。

3. 高频噪声的限制使结果更精确。

使用 Hamming 窗口作为插值函数，对带尖点的边界不能很好地恢复，这是一个不足之处。

本篇论文获得 1998 年美国数学建模竞赛的一等奖。