学号：200920311007

《高级操作系统》课程报告



论文题目：书法轮廓点对匹配的三维可视化设计与实现

学 院： 信息工程学院

专 业： 计算机科学与技术

班 级： 计算机111专

姓 名： 徐培

指导老师： 章夏芬

完成日期： 2013年5月

**摘 要**

中国的书法字作品具有很悠久的历史，对于书法字作品来说，每一个字都纷繁复杂，有着不同的形态，这对鉴别来说，是一个相当复杂的过程。传统的文字识别算法可以用于识别单一文字，但对于同一文字的不同风格很难识别出来，因此，需要研究算法来实现书法字的匹配。

对于传统的匹配算法研究，研究过程相对漫长，要花费相当多的时间于匹配结果验证，并且，对于中间结果以及算法的准确性验证比较困难。而当前，可视化的研究对于算法的匹配情况，能够更为直观的呈现出来，对于算法研究和匹配展现来说，都有着相当重要的意义。

而相较于以前的数字化的匹配方式，本毕业设计实现了将匹配结果和中间过程三维可视化显示，以便于校验算法的准确性。本系统主要完成了以下工作：

1. 书法字图像轮廓点特征提取：将书法字图像的处理过程（原图、二值化、归一化、轮廓提取）显示在程序中，并将书法字轮廓线和采样点显示在三维空间内。
2. 匹配点对可视化：进行书法字匹配，将匹配结果用直线把采样点连接起来进行三维可视化处理。提供匹配点对的信息，并将匹配的点对单独动态的做二维可视化处理。
3. 点对分段显示：分段显示匹配结，根据系统配置，以及用户的人机交互操作，改变图形图像的显示视角、范围以及显示效果。

匹配点对的三维可视化显示，对于匹配算法的研究有着相当大的帮助，并且也为研究者提供了一个更好的匹配结果展示平台。

**关键词：轮廓匹配，分段，可视化显示**

**Abstract**

Chinese calligraphy character has a long period of history. In terms of calligraphy character images, every character has a unique shape. So it is difficult to recognize character images. Traditional OCR (Optical Character Recognition) technologies can be used to identify an individual character image, but for Chinese characters in different styles, it is difficult to identify. Therefore, this paper researches the algorithm to achieve matching of calligraphy.

The conventional matching algorithm cost a lot of time relatively in research process. It takes considerable time to verify the matching and the intermediate results. Also, it is difficult to verify the accuracy of the algorithm. The current visualization research for the matching algorithm can be more intuitively display the matching process. The Display of matching algorithms has a very important significance.

Compared to the previously digitized matching, the graduation design will match the results achieved during the middle of three-dimensional visualization and display, in order to verify the accuracy of the algorithm. The main contributions of this system are as follows:

(1) Contour points extraction: Calligraphy character image processing (image, binarization, normalization, contour extraction) shown in the program. The calligraphy contour and the sampling points are displayed in three-dimensional space.

(2) 3D Peer-points Visualization: To calligraphy character match, the sampling points of match result with a straight line connecting the three-dimensional visualization. Provide matching points of information, and a separate matching pairs of points to make two-dimensional dynamic visualization.

(3) Detail display of corresponding points: Segment display matching results. Depending on the system configuration and the operation of the user's interactive, users can change graphics display angle, range, and display results.

Display of matching points on the three-dimensional visualization contributes to the matching algorithm, and it also provides researchers with a better matching results display platform.

**Key words: Contour matching, segmentation, visualization**

**目 录**

[1.绪论 1](#_Toc358142879)

[1.1背景 1](#_Toc358142880)

[1.2研究动机 1](#_Toc358142881)

[1.3要解决的问题 2](#_Toc358142882)

[2.相关研究 3](#_Toc358142883)

[2.1书法字特征轮廓点匹配 3](#_Toc358142884)

[2.2三维可视化显示 4](#_Toc358142885)

[3.概要设计 7](#_Toc358142886)

[3.1系统框架 7](#_Toc358142887)

[3.2系统流程图 8](#_Toc358142888)

[4.数据获取 9](#_Toc358142889)

[4.1数据获取 9](#_Toc358142890)

[5.书法字特征轮廓点匹配 10](#_Toc358142891)

[5.1二值化 10](#_Toc358142892)

[5.2归一化 12](#_Toc358142893)

[5.3轮廓提取 13](#_Toc358142894)

[5.4特征信息计算 15](#_Toc358142895)

[5.5形状矩阵的构建 16](#_Toc358142896)

[5.5采样去重 16](#_Toc358142897)

[5.6余弦度量算法 17](#_Toc358142898)

[5.7基于拐角的算法 18](#_Toc358142899)

[6.三维可视化显示 19](#_Toc358142900)

[6.1轮廓的三维显示 19](#_Toc358142901)

[6.2三维仿射变换 22](#_Toc358142902)

[6.3点对细节显示 23](#_Toc358142903)

[6.3.1匹配点对的分项显示 24](#_Toc358142904)

[6.3.2点对细节动态显示 24](#_Toc358142905)

[6.4交互控制 25](#_Toc358142906)

[6.4.1鼠标拖拽 26](#_Toc358142907)

[6.4.2键盘控制 26](#_Toc358142908)

[7.系统运行结果 27](#_Toc358142909)

[7.1 系统运行环境 27](#_Toc358142910)

[7.2 系统运行结果 27](#_Toc358142911)

[7.2.1主界面 27](#_Toc358142912)

[7.2.2读取书法字图像 28](#_Toc358142913)

[7.2.3书法字匹配结果 30](#_Toc358142914)

[7.2.4交互控制 32](#_Toc358142915)

[7.2.5具体匹配点对查看 35](#_Toc358142916)

[7.2.6控制说明界面 37](#_Toc358142917)

[8. 结论和展望 38](#_Toc358142918)

[8.1 结论 38](#_Toc358142919)

[8.2 展望 38](#_Toc358142920)

[致谢 39](#_Toc358142921)

[主要参考文献 40](#_Toc358142922)

# 1.绪论

## 1.1背景

大量中国书法作品是一种宝贵的文化遗产。然而，很难采用任何现有的高科技检索书法作品，因为对于中国书法字来说相似匹配算法并不是一个小问题。相较于其他文字，中国书法有着更多的变化，和复杂性，不同作品出自于不同时期，不同大师的手笔，尽管是同一个字，但也是形态各异的个体，如王羲之的《兰亭集序》，其中光是“之”字就有21种不同的变化。书法字的复杂性体现在于：

1）笔画变形：同一作家在不同的心情下可以产生不同的风格相同的字符。有时一个字符为了得到一个更好的艺术效果而故意扭曲变形，如枯笔字。

2）复杂性：据统计，平均每一个字符组成约12.71笔画（见[[1](http://translate.googleusercontent.com/translate_f#8)]）。每一笔画长度取决于该字总笔画数，而每一笔段长度则取决于该笔画总笔段数。此外，在一些书法风格上，笔画的书面连接，使得它甚至难以辨认。

3）模糊性：由于原始作品饱经历史沧桑，受自然因素影响，部分笔画可能模糊不清。

虽然传统的中文字符识别技术（见[[2](http://translate.googleusercontent.com/translate_f#8)]）和书法字检索具有着许多同样的问题和解决方案，但OCR识别是一种精确的匹配过程，无法应用到同一书法字不同风格的检索中。书法字检索的问题的关键是要找到类似的字符根据用户的查询样本的塑造。为了找到形态各异的同一个字，没必要先识别该字，只需以样本自为标本找出形状相似的字。

但是，对于这些字体检索算法来说，我们得到的可能都是一些采样的坐标点及其比对值，相对于图形图像来说，数值较为抽象。通过数值，我们很难分析出算法是否有错，或者对应点的关系，究竟是怎么样的，这种方式相对抽象，对于研究和实现来说，都是很大的障碍。因此，对于这些关键点的匹配做可视化图形显示，给人直观的感受，这对检索来说是很有帮助的。

## 1.2研究动机

书法轮廓点对匹配的三维显示如果开发成功，有以下几个好处：

1. 可以将匹配的点对三维可视化显示出来。
2. 匹配的结果一目了然，对观察和展示来说，提供了一个很好的平台。
3. 有着更为开放的人机交互环境，可以自由的控制图像的显示位置，视角，以及呈现的效果。

本系统旨在开发一个更为友好，更为实用的书法字匹配以及验证程序。本系统一方面便于用户做书法字匹配算法的研究，另一方面，也对算法的展示提供了一个较为开放的平台。

## 1.3要解决的问题

本系统主要解决以下问题：

1. 轮廓点信息获取：通过系统获取到的书法字图像信息，进过处理，提取出其轮廓点的坐标信息。
2. 二维坐标投射到三维空间：由于轮廓点的坐标是二维的，因此要投射到三维空间中，需要设置z坐标的值，这样二维物体就投射到了三维空间。
3. 三维物体投影到二维平面显示：对于显示的屏幕来说，屏幕本身是二维的，将三维空间内的物体显示出来，需要将三维坐标再投影到二维平面上。
4. 轮廓匹配：将轮廓点的信息以及其特征信息进行匹配处理。
5. 匹配点对验证：将匹配的点对结果信息显示出来，并列在列表中，动态显示匹配点对结果。
6. 分段查看匹配结果：对于匹配的点对结果，进行分段查看，可以更直观的显示出匹配的情况。

# 2.相关研究

本系统主要研究的是三维显示书法轮廓点匹配，用于验证和演示检索算法的匹配情况，那么就需要对检索算法进行研究。研究主要从两方面展开，一是形状匹配算法的基本研究，二是三维图形的动态显示。

## 2.1书法字特征轮廓点匹配

我们平常所说的文字识别，指的是OCR文字识别。OCR文字识别[[3](http://translate.googleusercontent.com/translate_f#8)]是指电子设备（例如扫描仪或数码相机）检查纸上打印的字符，然后用字符识别方法将形状翻译成计算机文字的过程；即，对文本资料进行扫描，然后对图像文件进行分析处理，获取文字及版面信息的过程。如何除错或利用辅助信息提高识别正确率，是OCR最重要的课题的友好性，产品的稳定性，易用性及可行性等。

OCR的基本原理[[4](http://translate.googleusercontent.com/translate_f#8)]就是通过扫描仪将一份文稿的图像输入给计算机，然后由计算机取出每个文字的图像，并将其转换成汉字的编码。其具体工作过程是，扫描仪将汉字文稿通过电荷耦合器件CCD将文稿的光信号转换为电信号，经过模拟／数字转换器转化为数字信号传输给计算机。计算机接受的是文稿的数字图像，其图像上的汉字可能是印刷汉字，也可能是手写汉字，然后对这些图像中的汉字进行识别。对于印刷体字符，首先采用光学的方式将文档资料转换成原始黑白点阵的图像文件，再通过识别软件将图像中的文字转换成文本格式，以便文字处理软件的进一步加工。其中文字识别是OCR的重要技术。

文字识别方法[[5](http://translate.googleusercontent.com/translate_f#8)]基本上分为统计、逻辑判断和句法三大类。常用的方法有模板匹配法和几何特征抽取法。

1.模板匹配法：将输入的文字与给定的各类别标准文字（模板）进行相关匹配，计算输入文字与各模板之间的相似性程度，取相似度最大的类别作为识别结果。这种方法的缺点是当被识别类别数增加时，标准文字模板的数量也随之增加。这一方面会增加机器的存储容量，另一方面也会降低识别的正确率，所以这种方式适用于识别固定字型的印刷体文字。这种方法的优点是用整个文字进行相似度计算，所以对文字的缺损、边缘噪声等具有较强的适应能力。

2.几何特征抽取法：抽取文字的一些几何特征，如文字的端点、分叉点、凹凸部分以及水平、垂直、倾斜等各方向的线段、闭合环路等，根据这些特征的位置和相互关系进行逻辑组合判断，获得识别结果。这种识别方式由于利用结构信息，也适用于[手写体](http://baike.baidu.com/view/272589.htm)文字那样变型较大的文字。

本系统所采用的书法字检索算法是属于几何特征抽取法的一种，其基本过程分为：

1.二值化：图像进行平滑处理，并转换成二进制图像。这是因为彩色背景的图像，这可能有益于读者观察，但是并没有益于相似度匹配过程中。

2.书法字符分割：分割仅仅是相对的，因为他们通过毛笔写的和它们之间的空间总是存在的。我们用最小边界矩形区域来分割原始页面转换成隔离的文字，如[[6]中所介绍 。](http://translate.googleusercontent.com/translate_f#8)

3.字符缩放：获得单独的不同大小的字符后，得到所有的字符规格化以保持的缩放不变性，我们的工作是归一化尺度的大小为64×64（以像素为单位）。

4.采样：我们采用边缘检测得到的字符的轮廓。然后，这些轮廓点信息存储在一个数组中，而不是将其存储在一个64×64矩阵。

5.轮廓点特征提取：对于采样出来的轮廓点，我们进行特征提取，对8个方向(上、下、左、右、左上、右上、左下、右下)，以及4段距离形成的空间进行统计，存如数组中，提取情况如图2.1所示:



图2.1. 轮廓点特征表达

6.查找近似点对应：根据之前的轮廓特征提取的变量，进行匹配值运算，可以得出与采样对应的近似点。

OCR是一个相对成熟的技术，有着很长的研究历史，它可以准确的识别文字，但是对于同一个文字不同风格的中国书法字来说，这个技术是不可取的。而另一方面，对于匹配算法来说， 并没有给出有效的可视化显示，相对于数据来说，图形能够更形象、更具体的突出和验证匹配的结果，当然，中间过程也是一样，这对于研究算法来说，有着深远的意义。尽管如此，我们仍可以从中借鉴到很多，如特征提取过程。

## 2.2三维可视化显示

对于文字形状匹配算法来说，虽然缺乏可视化显示，但是我们可以将算法的中间过程、结果数据导出，这还是相对容易的一件事。可是，光有这些数据并不能完成整个显示工作，我们获得的数据是2D坐标信息和匹配数值，为了便于观察，我们需要将2D的信息投射到3D空间中去。

那么我们就需要引入投影法[8]这个概念了。所谓投影法，就是投射线通过物体，向选定的面投射，并在该面上得到图形的方法 根据投影法所得到的图形，称为投影（投影图）。投影法中，得到投影的面，称为投影面，如图2.2所示：

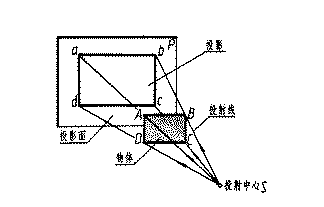


图2.2. 投影法基本特征（以中心投影法为例）

根据投射线的类型（平行或汇交），投影法分为中心投影法和平行投影法两类。而我们所采用的，是平行投影法中正投影法（也成正交投影）。投射线与投影面相垂直的平行投影法。根据正投影法所得到的图形，称为正投影(正投影图)，如图2.3所示：

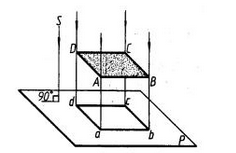


图2.3. 正投影法

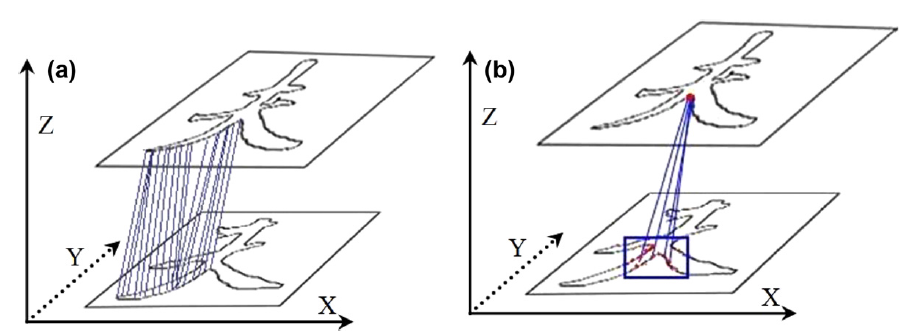
通过正交投影，我们可以将书法字的轮廓显示在3D空间的两个不同平面中，以此作为背景，再通过匹配算法得出的匹配结果，我们可以类似投影方式，将匹配的情况在3D中显示出来，如图2.4所示：

图2.4. 匹配结果显示[9]

# 3.概要设计

## 3.1系统框架

本系统主要划分了六个模块，如下图3.1所示：

图像轮廓提取

归一化

采样去重

轮廓提取

二值化

三维可视化显示

轮廓匹配

匹配输出

三维显示匹配结果

平移、缩放、旋转

余弦度量算法

基于拐角的算法

分段查看

匹配验证

Top 10%、Bottom 10%查看

二维可视化

列表查看

奇数点、偶数点查看

动态显示

图3.1. 模块划分

1. 轮廓提取：将读取的书法字图像数据进行二值化、归一化、轮廓提取、采样去重这四个步骤，然后将轮廓信息保存到内存中。
2. 三维可视化显示： 根据数据采集处理过后得到的轮廓信息，对图像进行初始化，显示在三维空间中。
3. 轮廓匹配：根据用户选择的算法，将两个书法字的轮廓点信息进行匹配计算。
4. 匹配输出：将匹配的结果、匹配的耗时以及匹配点对的情况显示在主界面上。
5. 匹配验证：将匹配的点对信息显示在列表中，并将点对匹配过程，根据用户的选择，动态显示在二维空间里。
6. 分段查看：根据用户的选择，分段查看匹配的点对结果。

## 3.2系统流程图

进入系统过后，首先提示用户读取书法字图像文件，并对两书法字进行轮廓匹配，然后根据其数据信息，通过用户设置显示模式，做三维可视化处理，此时接受用户的分段查看设置，并可以对匹配结果进行验证。

开始

进入程序主界面

选择书法字图像文件

二值化、归一化、轮廓提取、采样去重等处理

匹配并进行三维可视化显示

分段查看

验证匹配结果

结束

图3.2. 系统主框架流程图

# 4.数据获取

## 4.1数据获取

本系统所涉及到的数据主要是单个书法字的图像，这些书法字是经过裁切处理过后的图像，是由章夏芬老师提供的，以下图4.1为例：



图4.1. 书法字图像

本书法字图像格式为JPG，大小从到不等，图片总数11253张。

# 5.书法字特征轮廓点匹配

## 5.1二值化

根据用户操作选择了书法字图像之后，需要将书法字的轮廓信息提取出来，而提取轮廓第一步要做的就是对书法字图像进行二值化处理。

所谓的二值化[10]就是将图像上的像素点的灰度值设置为0或255，也就是将整个图像呈现出明显的黑白效果。将256个亮度等级的[灰度图像](http://baike.baidu.com/view/1549178.htm)通过适当的阈值选取而获得仍然可以反映图像整体和局部特征的二值化图像。在[数字图像处理](http://baike.baidu.com/view/286846.htm)中，二值图像占有非常重要的地位，首先，图像的二值化有利于图像的进一步处理，使图像变得简单，而且数据量减小，能凸显出感兴趣的目标的轮廓。其次，要进行二值图像的处理与分析，首先要把[灰度图像](http://baike.baidu.com/view/1549178.htm)二值化，得到二值化图像。

而在做二值化处理之前，需要将像素点的颜色转换为灰度值，如下面公式：

[11]

然后，二值化处理主要用OTSU法（二值法的一种）.最大类间方差法是由[日本](http://zh.wikipedia.org/wiki/%E6%97%A5%E6%9C%AC)学者[大津展之](http://home.hiroshima-u.ac.jp/tkurita/spr/member/otsu.html)于[1979年](http://zh.wikipedia.org/wiki/1979%E5%B9%B4)提出的，是一种自适应的阈值确定的方法,又叫大津法，简称OTSU[12]。它是按图像的灰度特性,将图像分成[背景](http://zh.wikipedia.org/wiki/%E8%83%8C%E6%99%AF)和目标两部分。背景和目标之间的类间方差越大，说明构成图像的两部分的差别越大,当部分目标错分为背景或部分背景错分为目标都会导致两部分差别变小。因此,使类间方差最大的分割意味着错分概率最小。对于图像，前景(即目标)和背景的分割阈值记作T，属于前景的像素点数占整幅图像的比例记为，其平均灰度；背景像素点数占整幅图像的比例为，其平均灰度为。图像的总平均灰度记为，类间方差记为g。假设图像的背景较暗,并且图像的大小为，图像中像素的灰度值小于阈值T的像素个数记作，像素灰度大于阈值T的像素个数记作,则有:

 　　　　　　　　　　 (1)

 　　　　　　　　　　 (2)

 　　　　　　 (3)

 　　　　　　　　　　 (4)

 　　　　 (5)

 　　　　　　　　　 (6)

将式(5)代入式(6),得到等价公式:

  　　　　　 (7)

采用遍历的方法得到使类间方差最大的阈值T，即为所求。接着逐个遍历图像的像素点，判断其灰度值是否大于阈值，如果大于阈值，则将像素点颜色变成黑色；否则变为白色。

但是，这样的处理过程存在问题，有些书法字背景色是黑色（较暗）,如下图5.1所示，二值化效果正常，底色为白色，字为黑色。

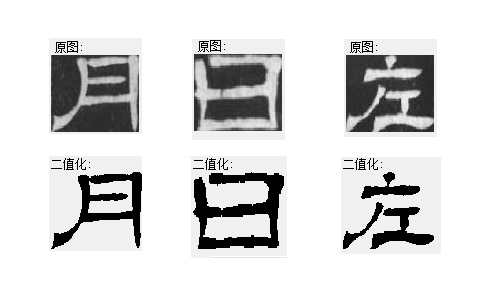


图5.1. 书法字背景为黑色

而相反的，如果书法字背景色是白色（较亮），如下图5.2所示，二值化效果就相反了，底色为黑色，字为白色。

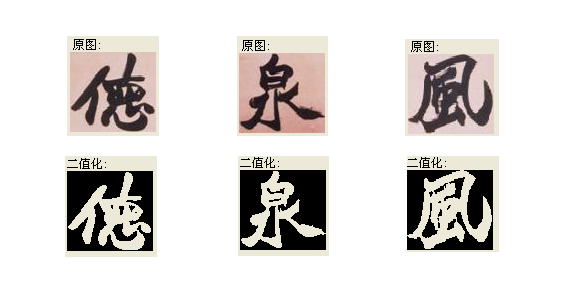


图5.2. 书法字背景为白色

由于书法字背景与字体颜色的关系，二值化的结果也会恰恰相反，因此，为解决问题，这里需要先读取图像（0，0）点的像素值（即左上角），认定这个点像素值与背景色的像素值相近，因此如果此点的灰度值大于阈值，那么背景色也是如此，就可以确定大于阈值的就是背景，应该设置成白色，而字就是黑色了，反之亦然。如下图5.3所示：

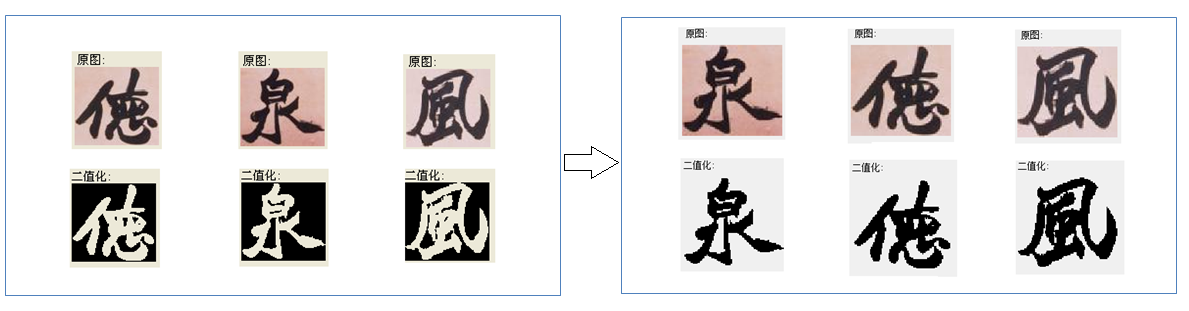


图5.3. 二值化算法翻色后效果

按照上述的处理过程，图像就以白底黑字的形式，存储在内存中。

## 5.2归一化

对于用户所选择的书法字图像来说，大小规格并不是统一的，这对匹配有着很大的影响，此时，需要对二值化过后的图像进行归一化处理。

而归一化，这里简单采用了比例缩放算法，以下图5.4为例，首先读取图像的大小，然后，根据本系统归一化的大小为，因而我们可以算出横纵方向的缩放比例：

宽度缩放系数（width为图像宽度）

高度缩放系数（height为图像宽度）

然后，对图像进行遍历，根据缩放系数，统计每个一个缩放系数区域，即区域（阴影部分）内黑点面积进行统计，如果黑点累计面积超过缩放区域面积的一半()，则将归一化后的图像的相应位置像素点设成黑色，否则设成白色。

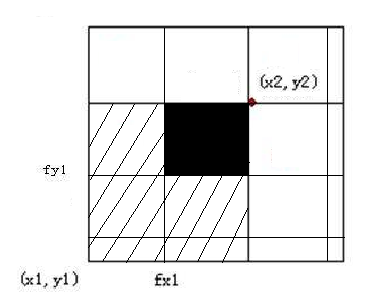


图5.4. 缩放示意图

## 5.3轮廓提取

进行归一化过后，图像大小都被处理成了，对于这些统一规格的图像，这里将进行下一步操作-轮廓提取。

本系统采用的是扫描线算法，由于这里的图像已经经过二值化处理，因此只需要判断像素点颜色不同，即代表是背景与字的分界线，以下图5.5为例，这里扫描线从右至左，遍历书法字图像上的每一行，判断，如果像素值不同，代表到了轮廓分界处，此时分界处的点保存到程序体变量中，遍历过后，轮廓点就提取完成了。

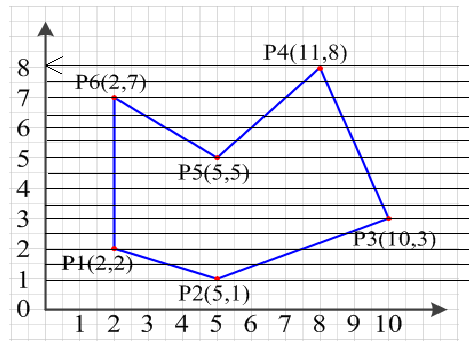
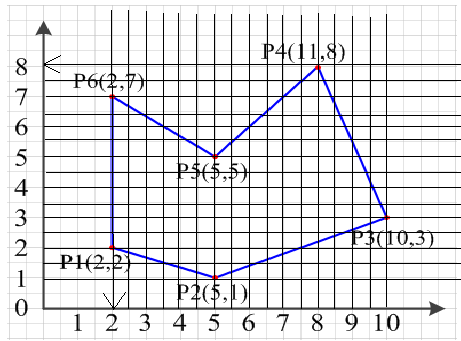


图5.5. 单方向扫描线算法提取轮廓

但是，程序运行结果验证了，这样单方向的扫描线提取轮廓方式，会忽略了与扫描线相平行的点，这样的提取的轮廓点是不全的，因此，需要对纵向继续一次扫描，如下图5.6所示：

  
图5.6. 双方向扫描线算法提取轮廓

对于这种改进过后的横纵双向扫描线算法，数据会有一定的重复，在数据存储时需要做去重处理，这样，整个书法的轮廓提取操作就完成了。

## 5.4特征信息计算

轮廓信息获取之后，需要计算每个轮廓点的特征信息，将其存储于于轮廓点坐标相关联的数组中。

对于特征信息，本系统是这样描述的：特征就好比是地理位置，在人的大脑里，对于世界上的每个地方，似乎并没有其他地方会与之相同，因为在人的思想中，描述一个地点信息，不光光是通过地点的位置特性，而是透过此地点的各个方位有什么建筑（这里理解为其他点），这样人就知道这个地点区别于其他地点的特性了，这里的地点和轮廓坐标点类似。

特征信息，这里对8个方向(上、下、左、右、左上、右上、左下、右下)，以及4段距离（2，4，8，16）形成的32位空间进行统计如图5.7所示，提取情况如图5.8所示:

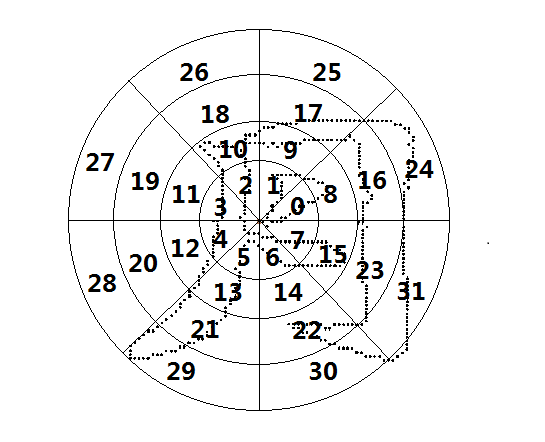


图5.7. 特征信息32位空间定义

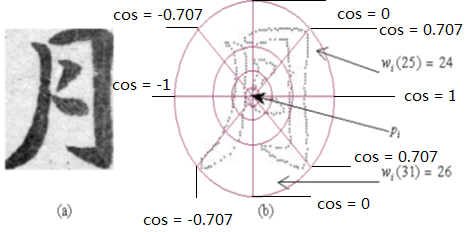


图5.8. 特征信息提取

从上图中可以看到，对于同一距离段的8个方向的特征信息提取，可以通过与单位向量（1，0）的夹角余弦值区分出来，因此，这里仅需遍历图像的轮廓点，判断点距离是否满足特征信息提取的4段距离区间，然后根据夹角余弦值来确定具体点在哪个方向上，这样，每个轮廓点的32位特征信息，就被提取出来了。

## 5.5形状矩阵的构建

对于系统中，对每个轮廓点的特征信息计算，每个轮廓点会产生32位的特征信息，这些特征信息在系统中存储在形状矩阵中，如下形状矩阵所示：

第1个点的特征信息

第n个点的特征信息

对于每个书法字图像来说，都会产生一个形状矩阵，其中，每一行代表一个点的32位形状特征信息。

## 5.5采样去重

尽管轮廓点以及其特征信息已经提取出来了，但是对于匹配来说，所有轮廓点都去匹配计算耗时较大，并且，有些轮廓点本身特征就很相近，区别并不明显，这样的点重复匹配一方面加大了计算量，一方面也对匹配的结果影响不大。因此，这里有必要对轮廓点进行采样去重处理。

这里要引入，特征相近的概念，本系统是这样定义特征相近的：对于图像中每个点的32位特征信息来说，这些特征信息用于描述一个轮廓点，这里的轮廓点相当于一个向量，对于一个向量来说，向量类似则代表其夹角为0，即Cos值为1，这也就代表两个轮廓点的特征相似。

本系统采用余弦度量（类空间向量夹角[13]）匹配值，以点p，q为例，将其32位特征信息代入以下公式计算：

(1)

遍历图像的形状矩阵，计算每两个轮廓点的余弦值即匹配系数，根据系统所设定的采样去重标准，将余弦值高于标准（较相似）的轮廓点都去除掉，如下图5.9所示：

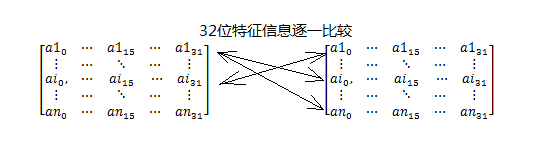


图5.9. 形状矩阵采样去重

上面主要是去重过程，除此之外，一些轮廓点必须被保留，如在距离为1范围内的空间里，与此轮廓点相邻的只有1个点，或者只有两个点，并且这两个点与参照点连线夹角为90度，这些都是端点、拐角点或者说是比较具有特征的点。

## 5.6余弦度量算法

根据以下公式，遍历整个轮廓点，对每一个轮廓点，计算此点和其他轮廓点的Cos值，Cos值越接近1，代表两个点越相似，取与参考点Cos值最大的点，认定这两个点匹配。

(1)

空间向量值参考如下图5.10所示：

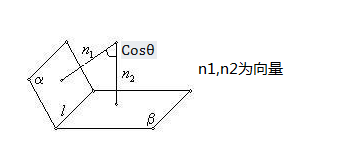


图5.10. 空间向量值

最后，将所有轮廓点的匹配值信息取均值，显示在界面中。

## 5.7基于拐角的算法

由于如果每个点很相似，那么他们的32位值变化量必定很小，因此，我们采用类似方差的计算方式，为了突出变化的程度，这里将公式变为（以点p，q为例）[9]：

(1)

依照上面的公式，遍历整个轮廓点，对每一个轮廓点，计算此点和其他轮廓点的匹配值，匹配值越接近0，代表两个点越相似，取与参考点匹配值最小的点，认定这两个点匹配。

最后，将所有轮廓点的匹配值信息取均值，显示在界面中。

# 6.三维可视化显示

## 6.1轮廓的三维显示

前面所提取的轮廓点信息，都是二维的，因此需要对书法字的第三个维度Z坐标进行初始化，本系统默认书法字一为背景，Z坐标为0，而书法字二的坐标，根据系统设置界面的参数，相对于书法字一进行三维坐标初始化，这样，二维坐标点就转变成了三维坐标点了。

对于窗体来说，图像显示窗口本质上二维的，这里需要将三维坐标投影成像到二维平面上，本系统采用正等测投影[14]。

这里，引入正等测的定义：将形体放置成使它的三条[坐标轴](http://baike.baidu.com/view/547315.htm)与轴测投影面具有相同的(约35°16′)[夹角](http://baike.baidu.com/view/132356.htm)，然后向轴测投影面作正[投影](http://baike.baidu.com/view/550991.htm)。用这种方法做出的[轴测图](http://baike.baidu.com/view/85218.htm)称为正等测图。

轴测图是一种简单的用二维图形来模拟三维立体的表示法。由于它的作图方法比较简单，且具有一定的立体感，所以常用作设计绘图的辅助手段。轴测图有多种类型，我们在此介绍常用的正轴测。

正轴测投影的形成过程和矩阵表示

（1）先将立体绕Z轴正向旋转θ角；（如图6.1所示）

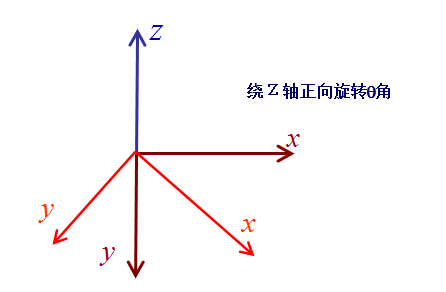


图6.1. 饶Z轴正向旋转θ角

　（2）然后再将立体绕X轴反向旋转φ角；（如图6.2所示）

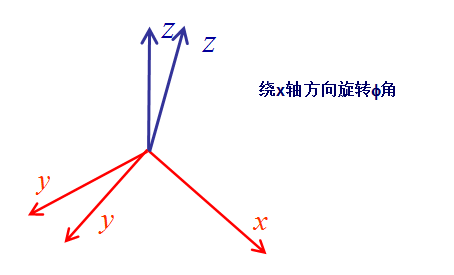


图6.2. 饶X轴反向旋转φ角

　（3）最后将经过两次旋转后的立体

　　　　向XOZ坐标平面作正投影。（如图6.3所示）

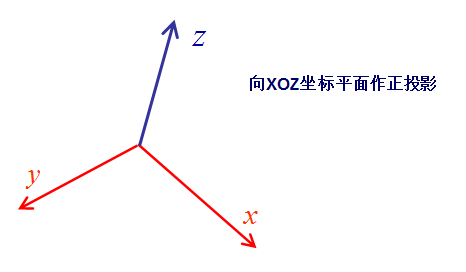


图6.3. 向XOZ坐标平面作正投影

最后得到变换矩阵为：



正等测的特点是：三轴上的变形系数均相等，即

　ηx = ηy = ηz

　∴ cos2θ+sin2θsin2φ

= sin2θ+cos2θsin2φ = cos2φ

解以上联立方程可得：θ＝45°

　　　　　　　　　　　φ＝35°16'

将以上求得的θ和φ角代入矩阵 T，可得到正等测的变换矩阵为：



正等测的轴向变形系数

ηx = ηy = ηz＝cos35°16’= 0.816

正等测的轴间角

tgαx = tg45°sin35° = 0.5774

tgαy = ctg45°sin35° = 0.5774

∴　 αx = αy = 30°

投影中间结果如下图6.4所示：

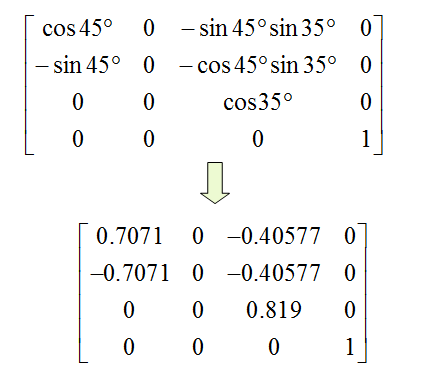


图6.4. 投影中间结果

投影最终结果如下图6.5所示：

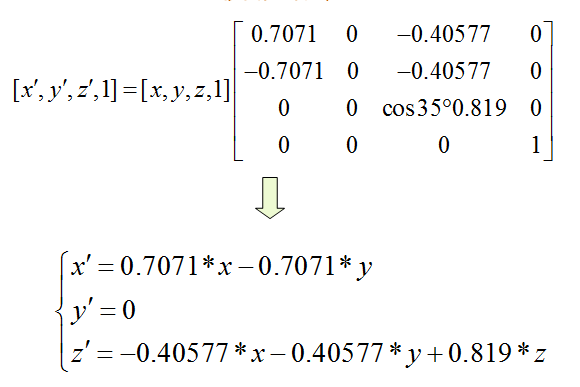


图6.5. 投影最终结果

这样，按照上面的公式，可以将轮廓点的三维坐标，转换为x,z的二维坐标，显示在图像显示窗口中。

## 6.2三维仿射变换

对于三维图像来说，往往固定的视角大小、范围、角度无法满足用户的需求，因此，这里就需要引入仿射变换[15]。仿射变换，又称仿射映射，是指在[几何](http://zh.wikipedia.org/wiki/%E5%87%A0%E4%BD%95)中，一个[向量空间](http://zh.wikipedia.org/wiki/%E5%90%91%E9%87%8F%E7%A9%BA%E9%97%B4)进行一次[线性变换](http://zh.wikipedia.org/wiki/%E7%BA%BF%E6%80%A7%E5%8F%98%E6%8D%A2)并接上一个[平移](http://zh.wikipedia.org/wiki/%E5%B9%B3%E7%A7%BB)，变换为另一个向量空间。

为了表示[仿射变换](http://zh.wikipedia.org/wiki/%E4%BB%BF%E5%B0%84%E5%8F%98%E6%8D%A2)，需要使用[齐次坐标](http://zh.wikipedia.org/wiki/%E9%BD%90%E6%AC%A1%E5%9D%90%E6%A0%87)，即用四维向量 表示三维向量，对于高维来说也是如此。按照这种方法，就可以用矩阵乘法表示变换。

本系统所使用到的仿射变换一共有三种：平移、缩放、旋转。

平移变换,将每一个点移动到，变换矩阵为：

其中，tx、ty、tz为对应的xyz坐标系的平移分量。

旋转变换，以x轴为轴心顺时针旋转弧度，变换矩阵为：

其中，为对应的旋转弧度。

以y轴为轴心顺时针旋转弧度，变换矩阵为：

其中，为对应的旋转弧度。

以z轴为轴心顺时针旋转弧度，变换矩阵为：

其中，为对应的旋转弧度。

缩放变换，将每一个点的横坐标放大（缩小）至sx倍，将每一个点的纵坐标放大（缩小）至sy倍，将每一个点的竖坐标放大（缩小）至sz倍，变换矩阵为：

其中，sx、sy、sz为对应的xyz坐标系的缩放分量。

## 6.3点对细节显示

本系统根据之前的匹配结果，以及用户分段显示的设置，将匹配的点对连接起来，动态的显示连接情况。由于三维显示在某种程度上来说并不是很清晰，二维的点对连线和动态显示则能够突出、直接的显示出匹配的实际情况。

## 6.3.1匹配点对的分项显示

相较于纯粹的图像可视化显示，文字数据加图像的表示对于算法的研究更有帮助，两种数据格式的结合展现能更有利的把匹配的效果的突显出来，对于研究、验证和观察三方面都有很大好处。

因而，本系统提供了匹配点对的分项显示，在列表控件中，将匹配的点对的信息陈列出来，并将其按照匹配值排序，如下图6.6所示：

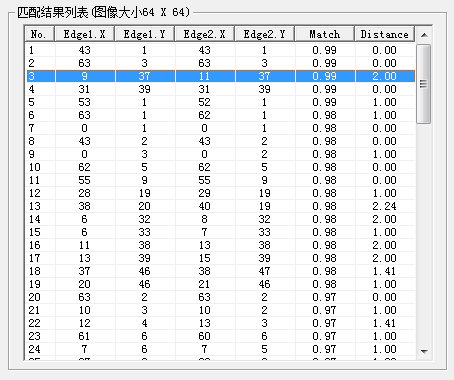


图6.6. 列表控件显示匹配点对信息

## 6.3.2点对细节动态显示

尽管点对信息在列表中显出来了，轮廓线和采样点也在二维平面上显示了，但是对于点对匹配的细节，还是很难观察出来。因此，本系统采用了，列表控件点击匹配点对的方式，来获取匹配点对的信息，通过这些点对坐标、匹配值信息，将匹配点对的匹配情况动态显示在二维平面上。

实现过程：

1.捕获鼠标选中项：在系统处理列表选中项的消息中，获取选中项的信息，如果选中项有匹配点对信息，代表此时系统需要突出显示匹配细节，否则，代表系统不需要显示匹配。

2.动态显示点对细节：此时已经通过捕获选中项获取到了匹配点对的信息，假设匹配的点对坐标分别为，那么用黄线将点相连，将点相连，将点相连，这样就会出现一条连贯的折线，将点连接起来。此时，只需要设置一个变量step，从0开始，到(总长)，每次渲染二维图像自增1，在相应位置显示箭头，就会出现动画的效果，在连线通道上动态显示箭头，如下图6.7所示：

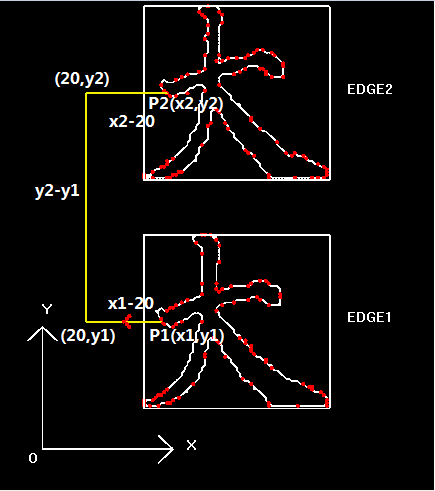


图6.7. 根据step值不同在不同位置显示箭头

## 6.4交互控制

对于图像显示的控制来说，本系统提供了两种方式：鼠标拖拽和键盘控制。

## 6.4.1鼠标拖拽

鼠标拖拽主要是通过捕获鼠标点按键的位置，以及鼠标移动位置的变化。

首先，捕捉鼠标按住的位置，如果位置在显示窗体内，那么这个操作是成功的，记录下当前点的位置。在OnLButtonDown事件里，捕获当前鼠标按下的位置，然后判断坐标位置是否在显示窗体的Rect区域内。

在OnMouseMove事件中，捕获鼠标移动位置，根据位置差来计算图像仿射变换的偏移量。

在OnLButtonUp事件中，捕获鼠标松开的位置，那么和的偏移量就决定了图像的位移、旋转的度量。左键控制位移，右键控制旋转。对于鼠标左键的时间来说，代表x轴方向上的位移，代表y轴方向上的位移;而对于鼠标邮件来说，代表z轴方向上的旋转，代表x轴方向上的旋转。

在OnMouseWheel事件中，捕获鼠标中键的滚动，朝一个方向滚动一次，图像缩放系数，这样，整个图像就随鼠标控制而改变。

## 6.4.2键盘控制

键盘控制也是这样的原理，捕获按下的状态，偏移位置递减。

键盘控制需要捕获PreTranslateMessage(MSG\* pMsg)消息，在这个消息处理的代码段中，通过传递进来的形参pMsg这个指针，判断pMsg->wParam这个参数，根据wParam来判断键盘具体按下的是哪个键，从而根据键位递增/递减相应的仿射变换值，以达到控制图像显示的目的。

具体捕获的键位为：

VK\_UP 沿y轴平移增加0.5 VK\_DOWN 沿y轴平移减少0.5

VK\_LEFT 沿x轴平移增加0.5 VK\_RIGHT 沿x轴平移减少0.5

VK\_NUMPAD2 沿z轴平移增加0.5 VK\_NUMPAD0 沿x轴平移减少0.5

'Q' 饶x轴顺时针旋转增加2度 'W' 饶x轴逆时针旋转增加2度

'A' 饶y轴顺时针旋转增加2度 'S' 饶y轴逆时针旋转增加2度

'Z' 饶z轴顺时针旋转增加2度 'X' 饶z轴逆时针旋转增加2度

对于上述此类的变换，代入仿射矩阵后，图像就更新变换了。

# 7.系统运行结果

## 7.1 系统运行环境

本系统运行的硬件环境是：

CPU：Intel® Core™ i5 CPU M520 @ 2.40 GHz

内存：2 GB

该程序在 Windows 7下运行，通过Microsoft Visual C++ 6.0编译

## 7.2 系统运行结果

## 7.2.1主界面

系统刚开启时的主界面，如下图7.1所示。

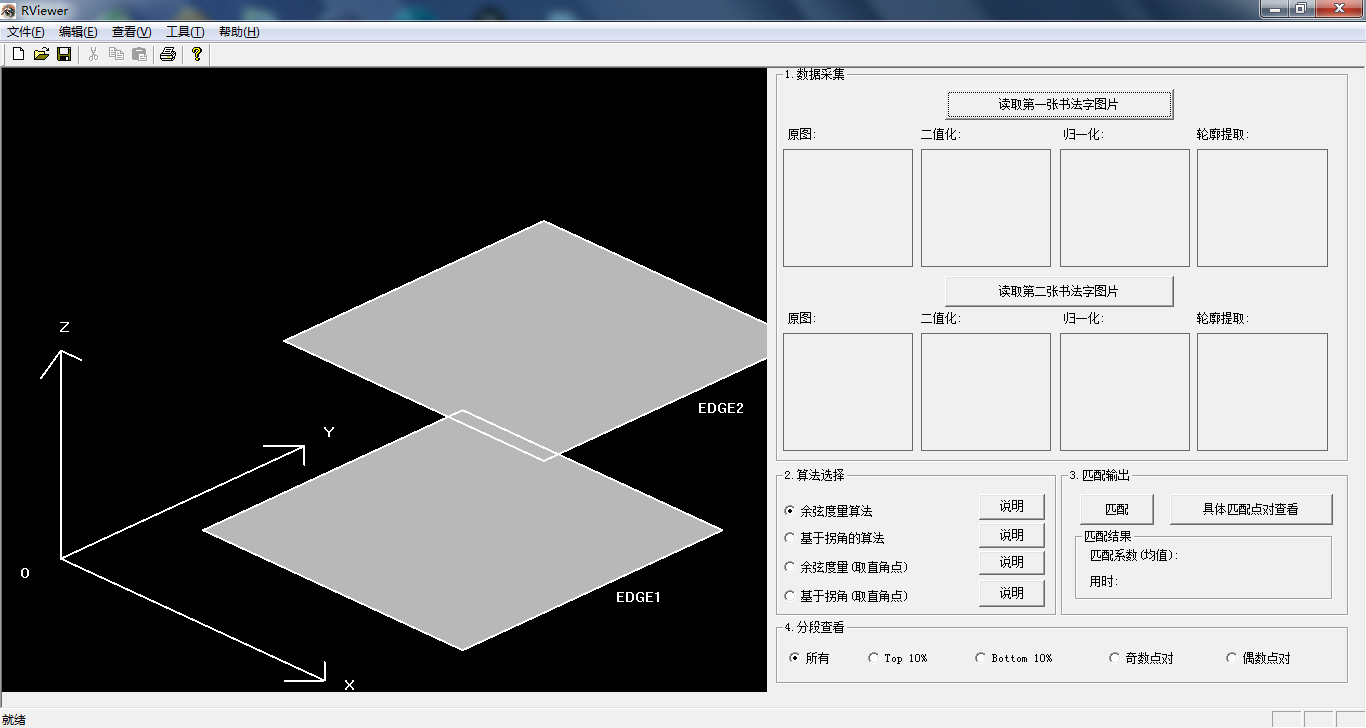


图7.1. 系统主界面

最左边的窗口用于显示三维画面，右边用于控制程序流程，通过点击按钮选择书法字图像，然后选择匹配算法，点击按钮进行匹配，在下方分段显示匹配结果，点击匹配结果验证进行算法研究和验证，在上方菜单栏点击出来有“工具-设置”进行相关设置，帮助中有控制说明。

## 7.2.2读取书法字图像

点击“读取第一张书法字图片”，出现打开窗口，如下图7.2所示：

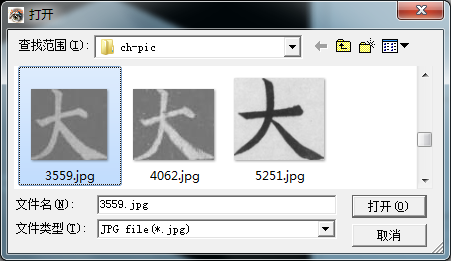


图7.2. 读取书法字结果

选择书法字图像，点击当打开，系统会自动进行二值化、归一化、轮廓提取、特征信息采集以及采样去重操作，如下图7.3所示：

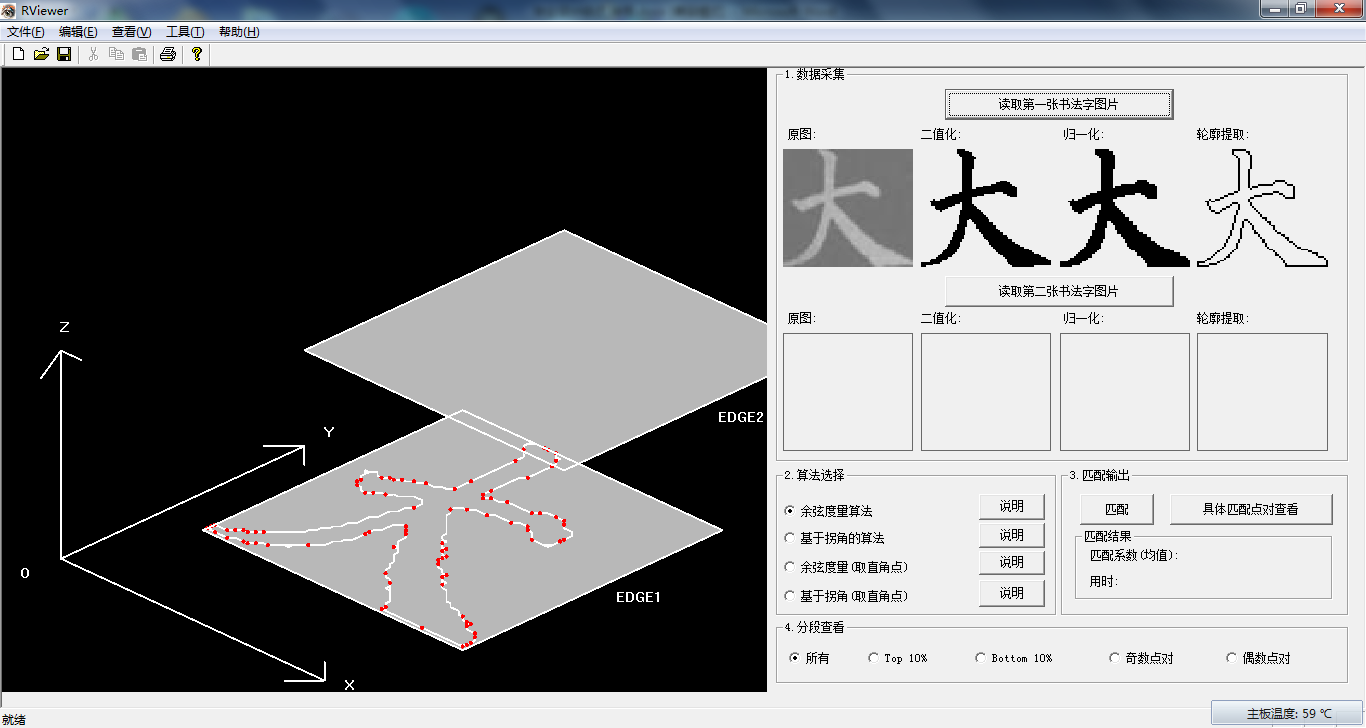


图7.3.读取第一张书法字图像结果

类似地，点击“读取第二张书法字图片”，会出现打开窗体，选择书法字图像，选择打开后，结果如下图7.4所示：

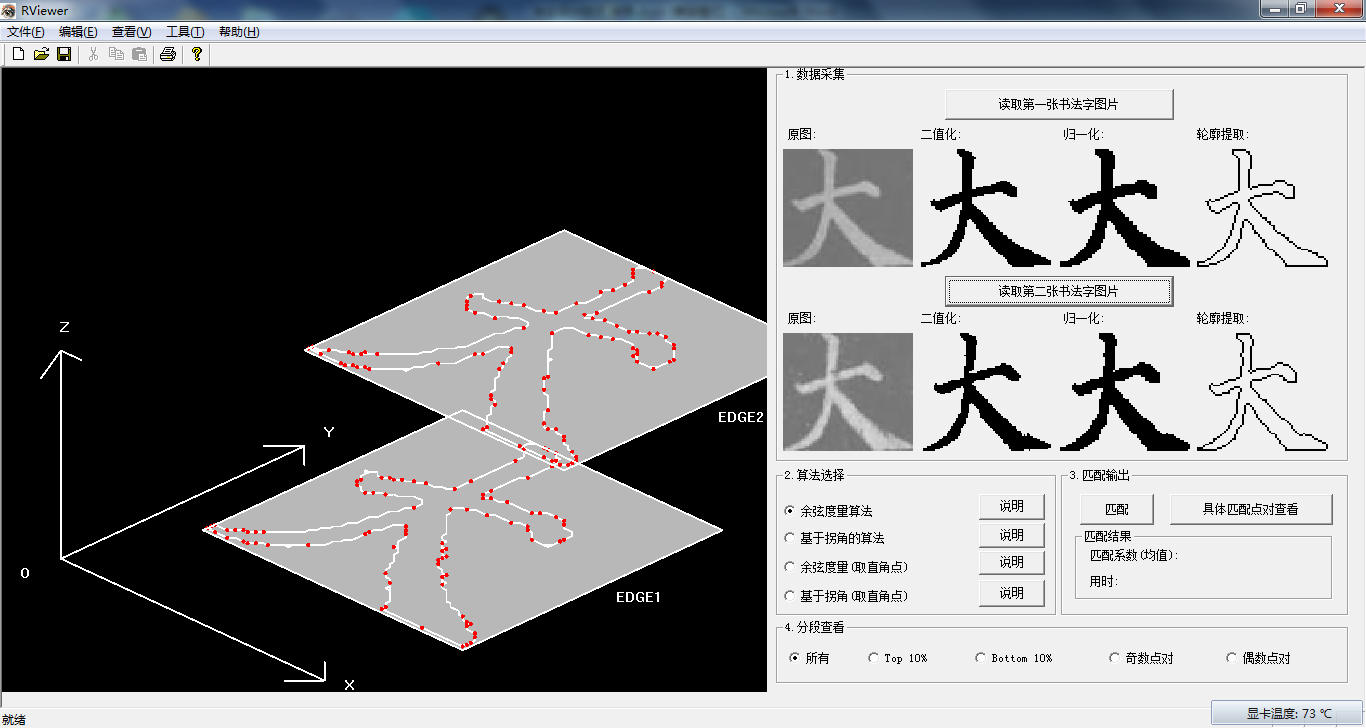


图7.4. 读取第二张书法字图像结果

三维显示以及采样去重标可以通过点开菜单栏中的“工具-设置”，在设置界面配置，如下图7.5所示：

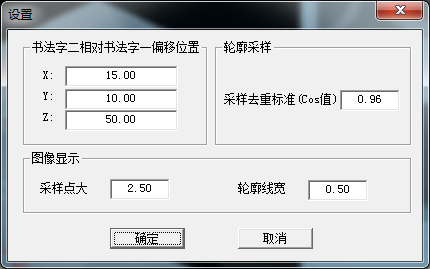


图7.5. 设置界面

改变“采样去重标准”为1后，系统就基本不去重了，重新选择书法字图像或者匹配，效果如下图7.6所示：

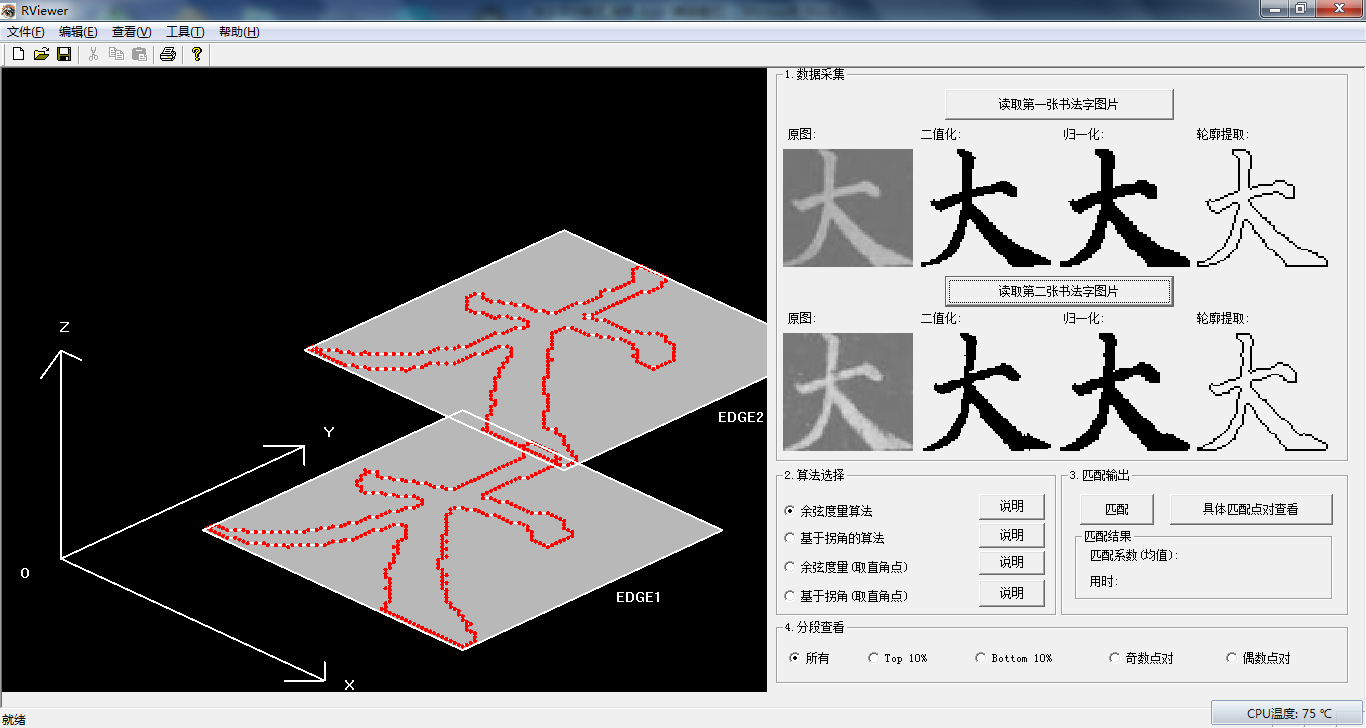


图7.6. 改变“采样去重标准”为1后效果

## 7.2.3书法字匹配结果

在“算法选择”区域，点选“余弦度量算法”，再点击“匹配”按钮，匹配结果如下图7.7所示：

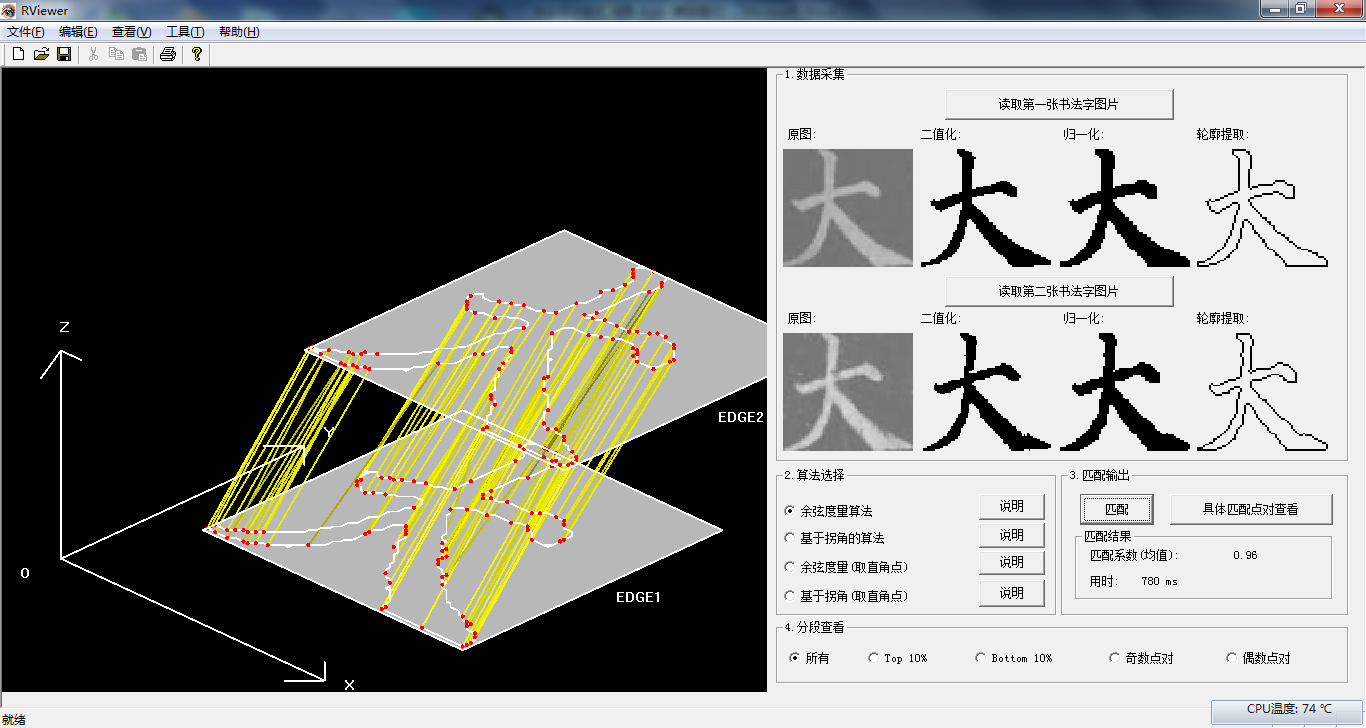


图7.7. “余弦度量算法”匹配结果

图中连线颜色代表匹配值强弱，越接近黄色匹配值越高，越接近黑色，匹配值越低。

在“算法选择”区域，点选“基于拐角的算法”，再点击“匹配”按钮，匹配结果如下图7.8所示：

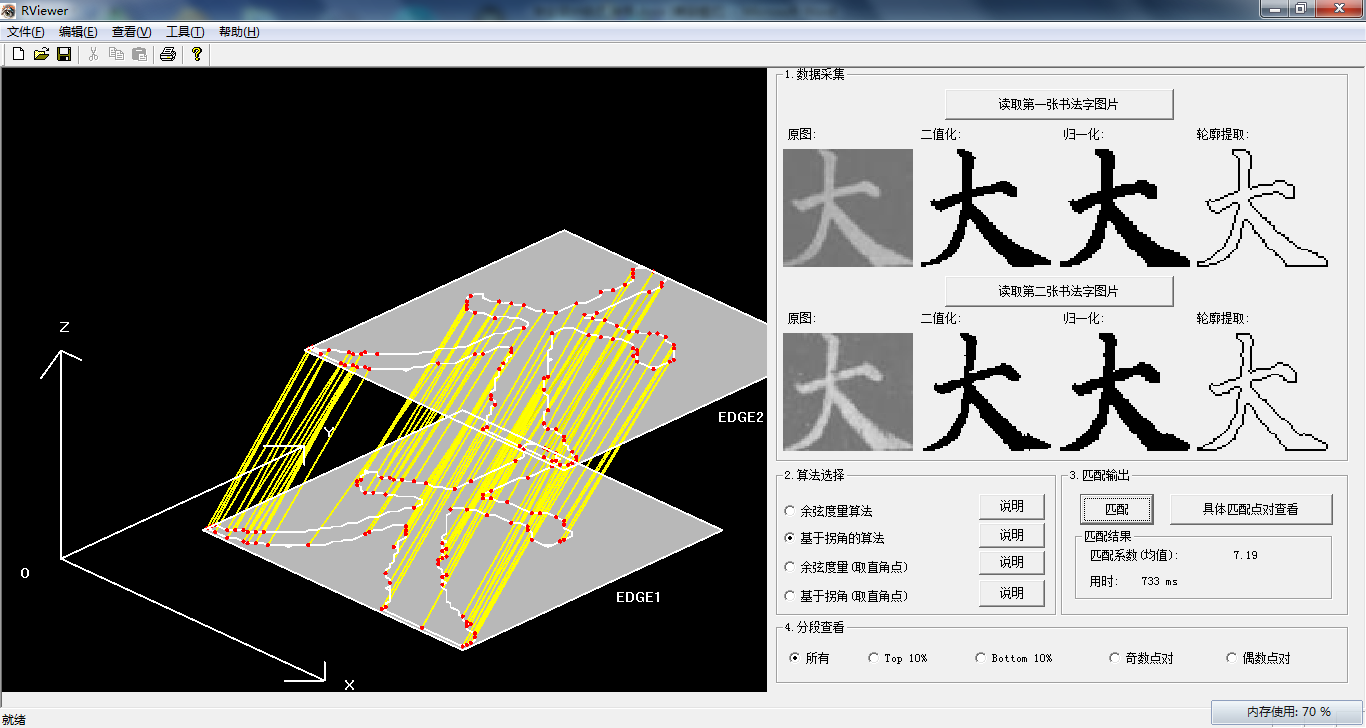


图7.8. “基于拐角的算法”匹配结果

在“算法选择”区域，点选“余弦度量（取直角点）”，再点击“匹配”按钮，匹配结果如下图7.9所示：

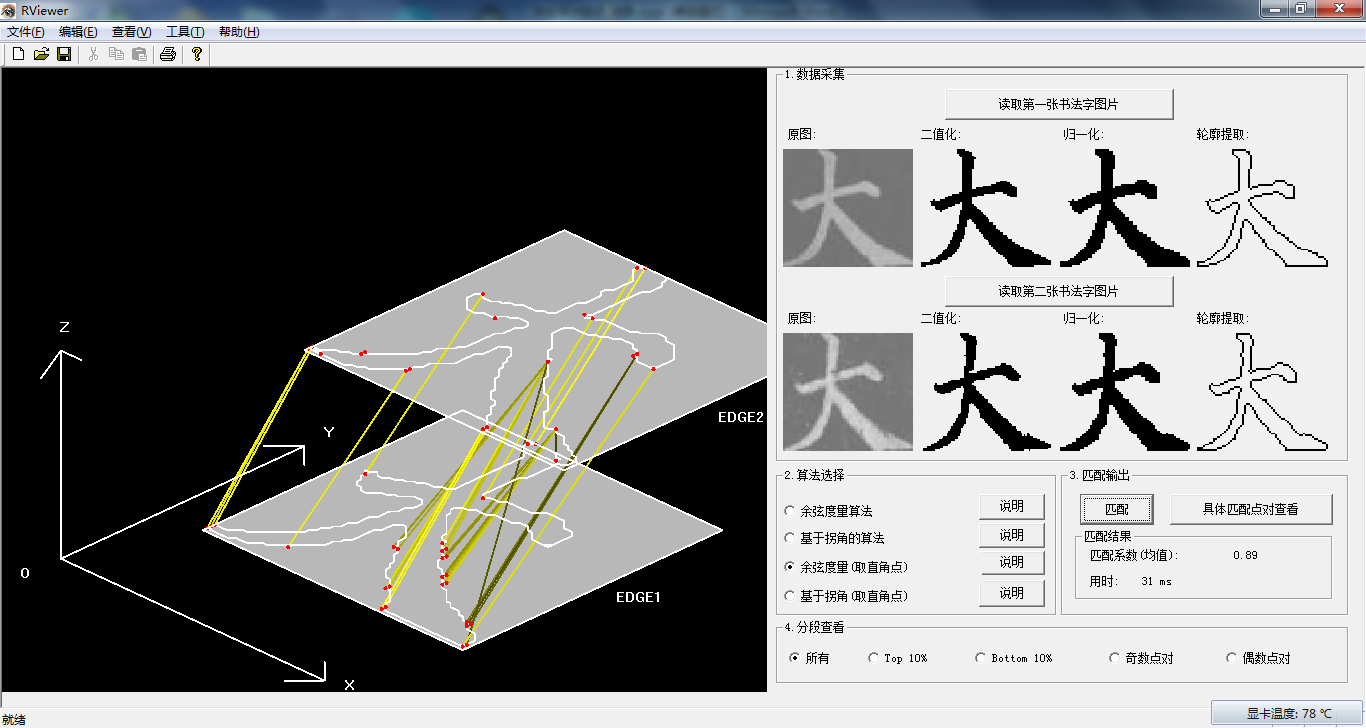


图7.9. “余弦度量（取直角点）”匹配结果

在“算法选择”区域，点选“基于拐角（取直角点）”，再点击“匹配”按钮，匹配结果如下图7.10所示：

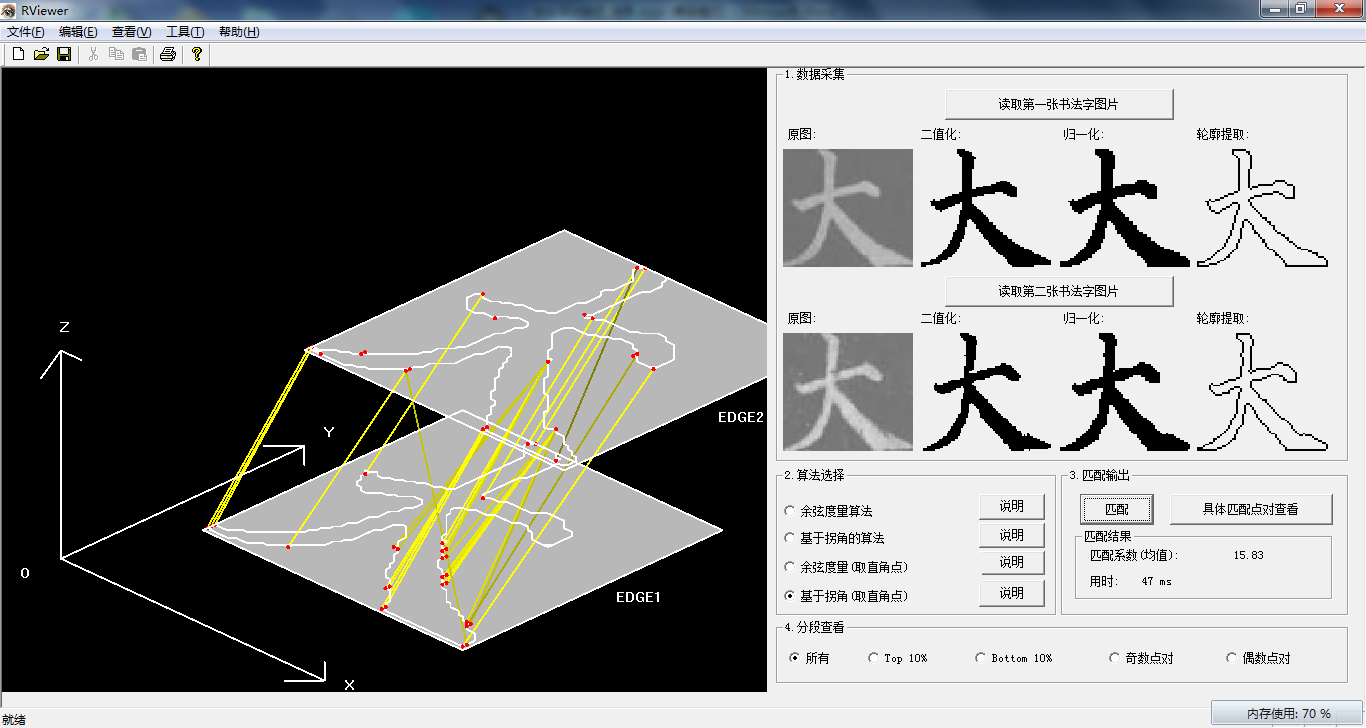


图7.10. “基于拐角（取直角点）”匹配结果

对于匹配结果，在“分段查看”区域，默认选择“无”，点击“Top 10%”可以查看匹配值最高的10%，如下图7.11所示：

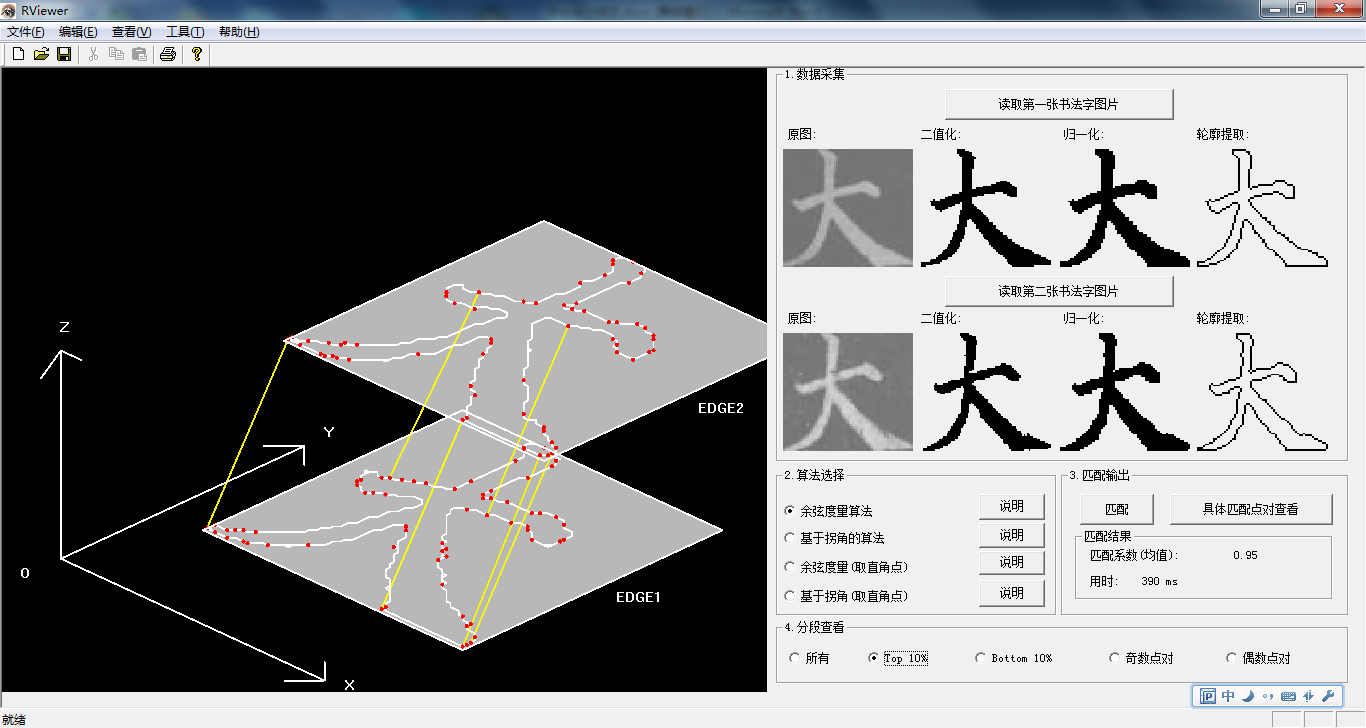


图7.11. 分段显示

## 7.2.4交互控制

本系统提供两种交互控制图像的方式，鼠标拖拽和键盘控制。

按住鼠标左键不放，向左拖拽图像后放开，效果如图7.12所示：

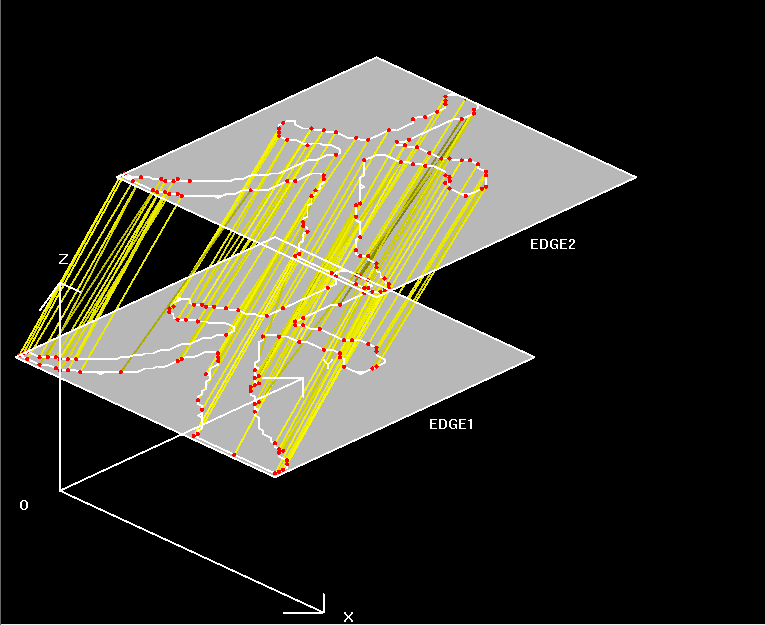


图7.12. 鼠标左键按住向左拖拽图像

按住鼠标左键不放，向上拖拽图像后放开，效果如图7.13所示：

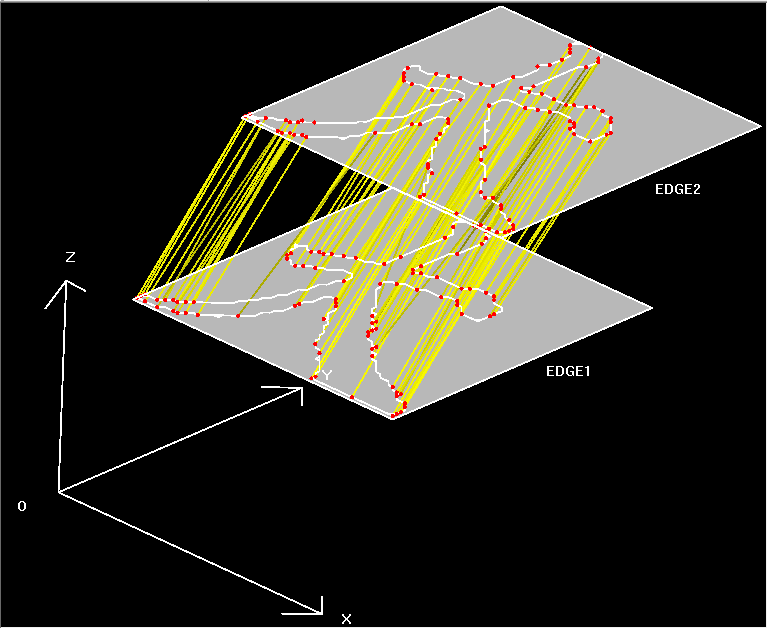


图7.13. 鼠标左键按住向上拖拽图像

按住鼠标右键不放，向右拖拽图像后放开，效果如图7.14所示：

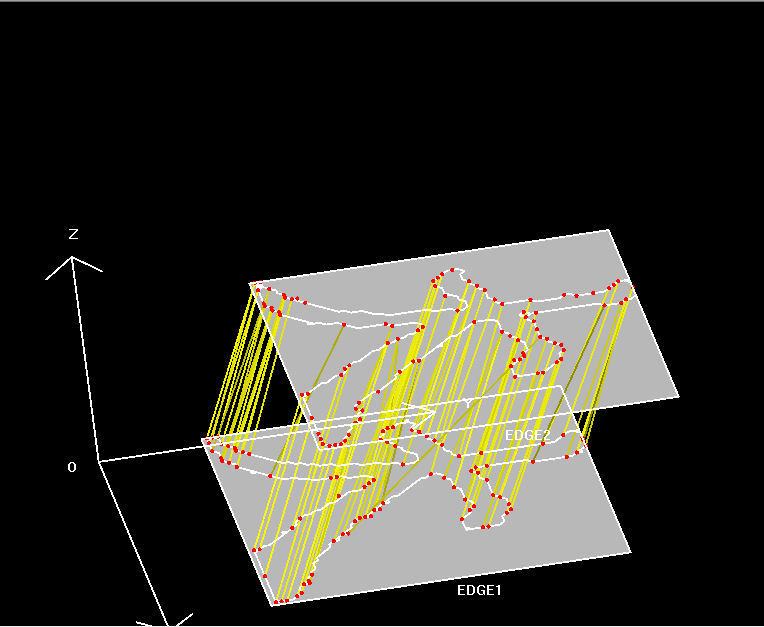


图7.14. 鼠标右键按住向右拖拽图像

按住鼠标右键不放，向下拖拽图像后放开，效果如图7.15所示：

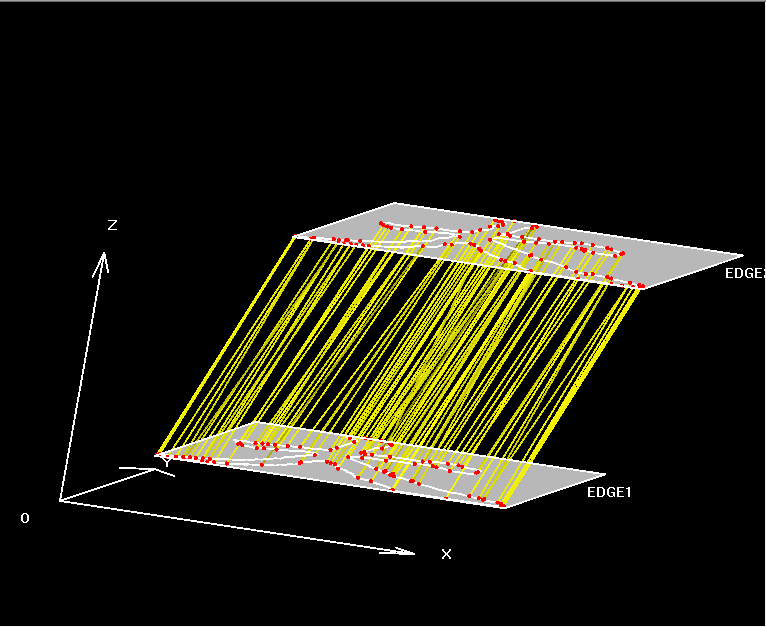


图7.15. 鼠标右键按住向下拖拽图像

鼠标中键滚轮向后滚动4次，效果如图7.16所示：

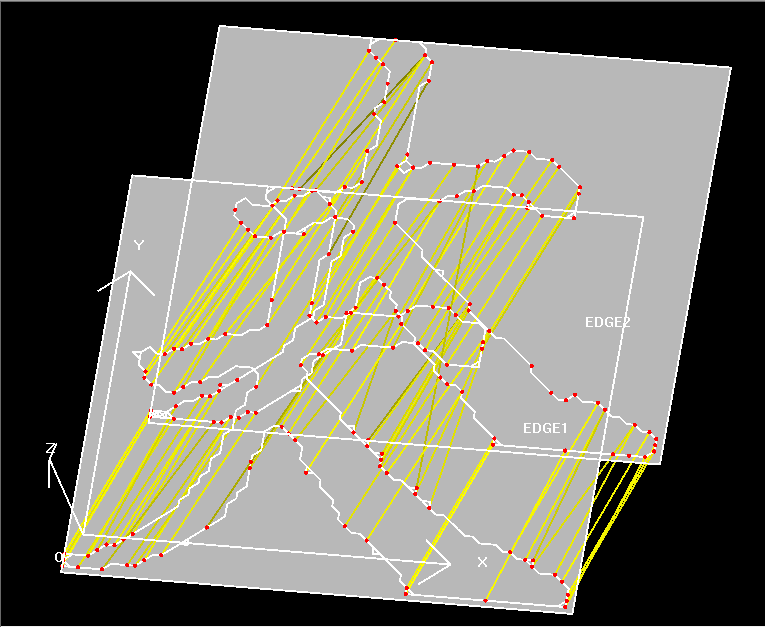


图7.16. 鼠标中键滚轮向后滚动6次

类似的，键盘按键和鼠标拖拽效果相雷同。

## 7.2.5具体匹配点对查看

有了匹配结果之后，就可以进行具体匹配点对查看，点击“具体匹配点对查看”，出现验证界面，如下图7.17所示：

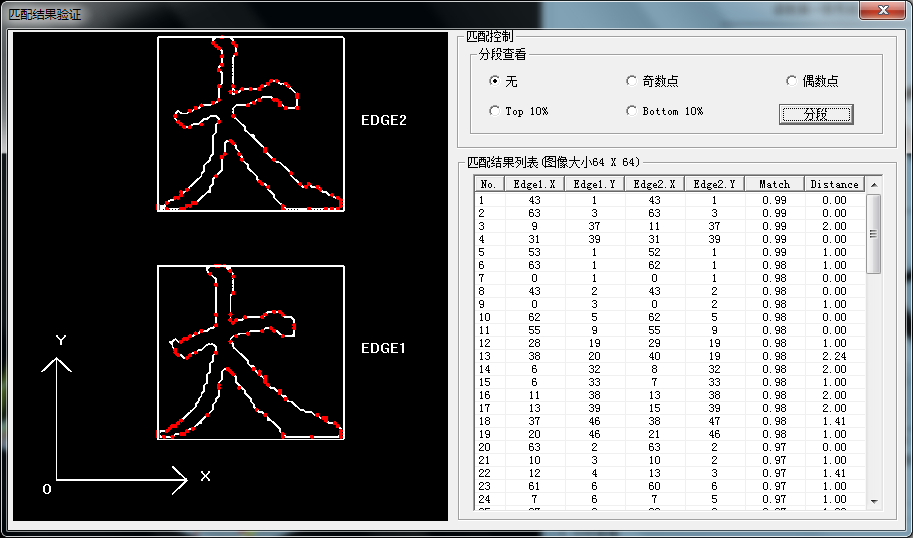


图7.17. 具体匹配点对查看

在下面列表中点击具体点对，会出现动画演示匹配点对情况，如下图7.18所示：

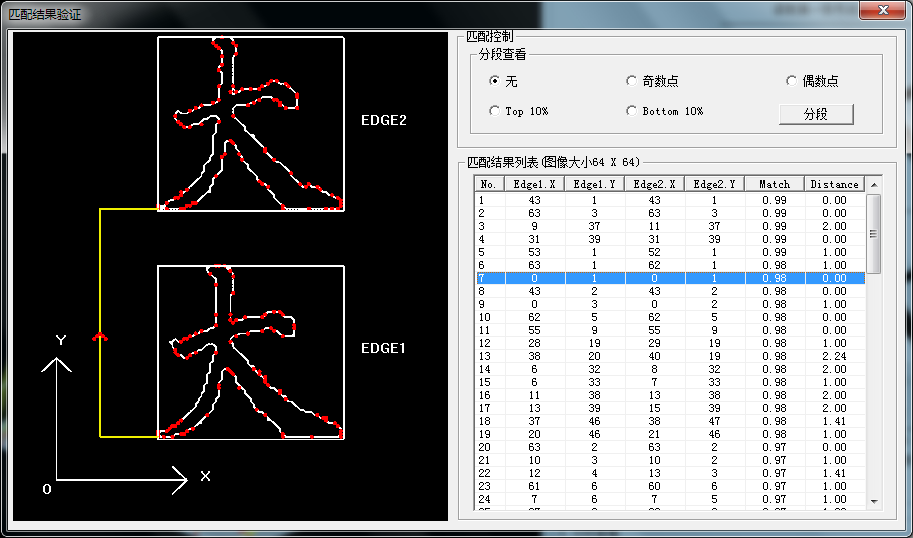


图7.18. 动画演示具体匹配

在匹配结果的点对列表中有些点对匹配的并不是准确，因为算法并没有加入欧式距离，因而可能会与较远的值匹配，这就需要研究者去修正算法了，如图7.19所示：

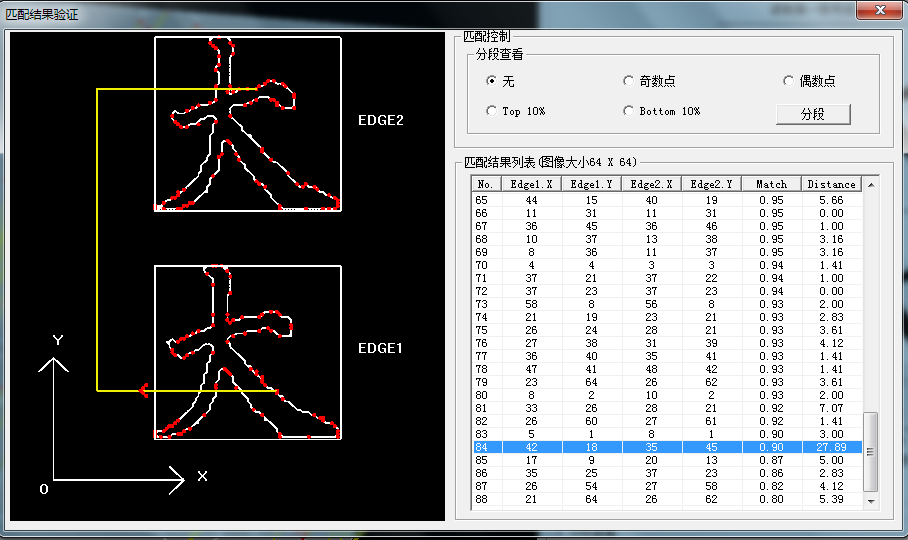


图7.19 匹配错误的情况

在匹配结果验证界面中，可以进行分段显示，具体体现在表格中，在“分段查看”区域选择“Top 10%”，这里默认选择“无”，然后点击分段，表格里的匹配点对就筛选出匹配值最高的10%个点，如下图7.20示：

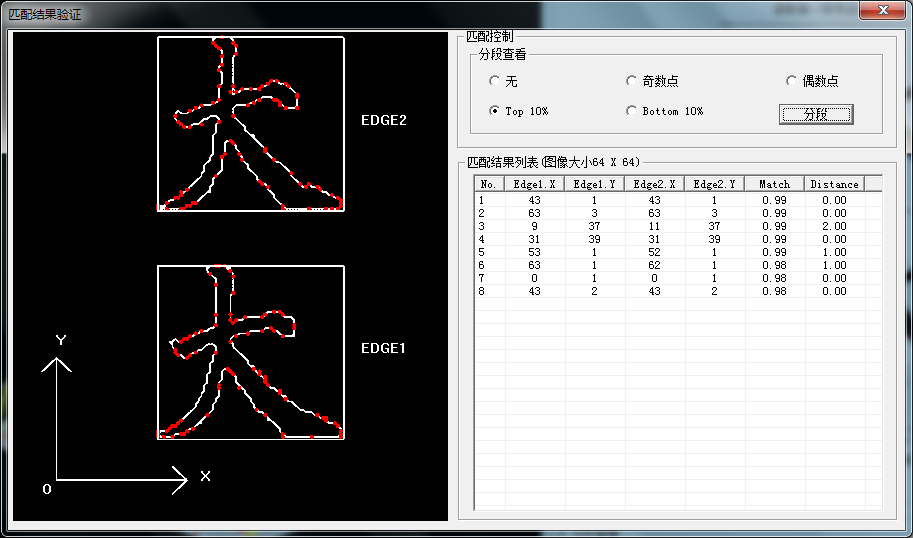


图7.20 分段显示匹配点对结果

## 7.2.6控制说明界面

本系统在菜单栏的“帮助”—“控制说明”里提供了对于图像的显示控制说明，说明界面如下图7.21所示：



图7.21. “控制说明”界面

# 8. 结论和展望

## 8.1 结论

本系统着重于匹配算法的显示，对于图形显示来说有着一定的开放性和优越性，对与匹配算法的研究有着极其重大的意义。

验证了匹配算法的结果，并对图像进行三维显示，对匹配的结果进行进一步的二维清晰定位，有着极高的研究价值。

本系统主要实现了：

1. 将书法字轮廓点信息以及其特征信息提取。
2. 匹配书法字轮廓，并将其三维可视化显示于界面中。
3. 提供算法选择、分段显示以及匹配验证，供研究和展示。
4. 给出了相应的操作说明和算法介绍，同时，也提供了图像显示的参数设置，使用户能够更方便、更快捷的调整图像。

## 8.2 展望

在本系统开发的过程中，对于匹配的每个过程，包括二值化、归一化、轮廓提取、特征收集、采样去重以及匹配算法本身，都有着很大的优化空间，对于三维的显示，更可以进行进一步的美化。因而，有以下展望：

1.对于轮廓提取进一步优化，对于二值化、归一化以及轮廓提取过程做进一步的细致研究。

2.对匹配算法进行改良，在速度和匹配精度上都有所突破。

3.对于三维图像显示，有着更美观的显示效果，对于控制方面更友好、更便捷。

本系统的可塑造性是极强的，可以进一步将书法字匹配提升为图像匹配，而要做的仅仅是改变算法。

# 致谢

时间如梭，转眼毕业在即。回想在大学求学的两年，心中充满无限感激和留恋之情。感谢母校为我们提供的良好学习环境，使我们能够在此专心学习，陶冶情操。谨向我的论文指导老师章夏芬老师致以最诚挚的谢意！章老师不仅在学业上言传身教，而且以其高尚的品格给我以情操上的熏陶。本文的写作更是直接得益于她的悉心指点，从论文的选题到体系的安排，从观点推敲到字句斟酌，无不凝聚着她的心血。

在这两年时间里，我学到了很多，也收获了很多。非常感谢章夏芬老师的图形学课程，引发了我对图形学浓厚兴趣。她严肃的科学态度，严谨的治学精神，精益求精的工作作风，深深地感染和激励着我。在学习过程中，同寝室的易毅、张纯、吴骁俊，作为室友和同学，他们更是我生活中的朋友，我们一起生活、一起学习、一起进步，感谢他们两年来的陪伴。

最后，我必须感谢我的父母。焉得谖草，言树之背，养育之恩，无以回报。作为他们的孩子，我秉承了他们朴实、坚韧的性格，也因此我有足够的信心和能力战胜前进路上的艰难险阻；也因为他们对我学业的精神以及物质上的支持，我才有机会如愿完成自己的大学学业，进而取得进一步发展的机会。

徐培

2013年5月4日

# 主要参考文献

1. 张锡中，中文字符识别技术。 北京：清华大学 统一大学出版社，1992年。
2. 吴守，丁小清，中文字符识别的原则的实现。 北京：高等教育出版社，1992年。
3. 百度百科-文字识别，<http://baike.baidu.com/view/852619.htm>
4. 百度百科-OCR文字识别，<http://baike.baidu.com/view/3383779.htm>
5. 百度文库-OCR识别， [http://wenku.baidu.com/view/82df13d426ff f705cd170a07.html](http://wenku.baidu.com/view/82df13d426ff%20f705cd170a07.html)
6. R. Manmatha and N. Srimal. Scale space technique for character segmentation in and-written manuscripts. In Pro. 2nd Int’l Conf. On Scale-Space Theories in Computer Vision, 1999, pp. 22-33.
7. Retrieval of Chinese Calligraphic Character Image Yueting Zhuang, Xiafen Zhang, Jiangqin Wu, and Xiqun Lu The Institute of Artificial Intelligence, Zhejiang University, Hangzhou, 310027, P.R.China
8. 网易博客-三视图，[http://wwwhaiercom.blog.163.com/blog/static/ 38693400200911810454330/](http://wwwhaiercom.blog.163.com/blog/static/%2038693400200911810454330/)
9. Dynamic Time Warping for Chinese calligraphic character matching and recognizing Xiafen Zhang , Yueting Zhuang
10. 百度百科-图像二值化，<http://baike.baidu.com/view/1532602.htm>
11. 百度百科-灰度值，http://baike.baidu.com/view/2796249.htm
12. 维基百科-大津算法，[http://zh.wikipedia.org/ wiki/%E5%A4%A7%E6%B4%A5%E7%AE%97%E6%B3%95](http://zh.wikipedia.org/%20wiki/%E5%A4%A7%E6%B4%A5%E7%AE%97%E6%B3%95)
13. 空间向量夹角，<http://www.cnblogs.com/crazyac/articles/1991957.html>
14. 百度百科-正等轴测图，<http://baike.baidu.com/view/1657001.htm>
15. 百度百科-仿射变换，http://baike.baidu.com/view/954621.htm