基于分形特征的二值图像检索方法的研究

李粤青　蒋金山　汪国强　张佳温
（华南理工大学应用数学系，广州 510640）
E-mail：yyqqq@163.net

摘　要　论述了基于分形特征的二值图像检索方法。图像的内容由4种特征来描述：图像分形维数、图像分形矢量、边界分形矢量和骨架分形矢量。实验表明该方法计算简单、有效，匹配快速，检索结果比较理想，只要查询图像在图库中，就一定能通过该方法检索出来。另外，该方法还具有一定的鲁棒性，证明这种检索方法是具有较大实用意义的。

关键词　分形维数　分形矢量　图像检索　轮廓　骨架

文章编号 1002-8331-(2004)17-0102-04　**文献标识码** A　**中图分类号** TP391.3

Study of Binary Image Retrieval Method Based on Fractal Behavior

Li Yueqing　Jiang Jinshan　Wang Guoqiang　Zhang Jiawen
(Dept. of Applied Mathematics, SCUT, Guangzhou 510640)

Abstract: In this paper, a novel approach to feature extraction based on fractal theory is presented as a powerful technique in image retrieval. This paper presents a method that can be applied to extract the feature of binary images. The content of image is depicted by four features: fractal dimension of image, fractal vector of image, fractal vector of edge and fractal vector of skeleton. It is constructed by a hybrid feature extraction combining fractal theory, edge detection and skeletonization. Through extensive experiments with a database of 2000 trademark images, the method we demonstrated is robust and flexible and a high recognition rate is reported.

Keywords: fractal dimension, fractal vector, image retrieval, edge, skeleton

1　引言

近年来，多媒体技术快速发展，计算机硬件速度持续提高，大容量存储器和数字化图像设备，如扫描仪、数码相机、数码摄像机等技术不断发展，图像数据的种类、数量也都随之飞速增长，从而图像数据库和图像检索技术逐渐成为国内外研究的热点。早期的图像检索方法是基于文本的，通过对每一幅图像建立关键词等描述信息作为图像索引，这种方法存在多种弊端，如人工处理方法费时费力，特征描述带主观性，随着数据量的增大，难以准确地对图像使用关键字注释，并且检索中不能充分利用各种特征等等[8]。于是，在上个世纪90年代初人们提出了基于内容的图像检索技术（CBIR），其主要思想是根据图像所包含的纹理、形状、色彩、以及它们的空间关系等内容信息，建立图像的特征矢量作为其索引，计算查询图像和目标图像的相似距离，按相似度匹配进行检索。

目前，国内外许多著名大学和研究机构已经开发了多种CBIR原型系统，如OBIC系统、Photobook系统、VisualSEEK系统、MARS系统、Virage系统等等[7]。虽然多种CBIR技术都已取得很好的检索结果，但在此研究方向上仍存在一些有待解决的问题，包括特征提取、图像索引、相似性度量等问题[1]，另外，查询准确率与效率也是检索系统中很重要并且有待提高的两个方面。

基于内容的图像检索主要以图像的视觉特征来区分图像相似与否，纹理和形状是图像内容组成的基本要素，是人识别图像的主要感知特征之一。分形维数可以表示曲线的复杂特性，即图像的纹理特征。该文在求分形维数方法的基础上引入了一个分形矢量的定义，实验证明该矢量用在图像检索上效果比分形维数更好。图像的边界和骨架是对图像形状很好的描述，二值图像的边界容易获取、算法简单，而骨架是图像的一种细化结构，是图像目标重要的拓扑描述。在提取图像的边界及骨架后，再分别计算二者的分形矢量，最后计算图像间的距离实现图像检索。该文着重研究图像的上述分形特征，并比较各个特征的检索特点，最后通过对二值商标图像的实验来检验检索效果。

2　算法描述

2.1　分形维数及分形矢量的计算

分形几何学是由法国数学家Benoit B.Mandelbrot在20世纪70年代创立的，它对自然界出现的一些看似不规则的几何体给出了数学描述，其本质是“自相似”。分形维数是定量描述分形特性的一个重要参数，由于分形维数直观上与物体表面的粗糙程度相吻合，而自然界中的不同纹理粗糙度有很大差别，因此可以用分形维数作为区分不同类别纹理的有效参数。分形维数的定义有多种形式，其中豪斯多夫维数（Hausdorff dimension）是一种较合理的维数，但其缺点是难以计算或估计它的值，因此，实用中通常研究其它一些形式的维数。该文采用“数盒子”算法（box-counting）计算图像的分形维数——盒维，盒维

作者简介：李粤青，硕士生，主要研究方向包括计算机应用、数理统计、数字图像处理等。蒋金山，副教授，博士生，主要研究方向包括计算机应用、数字图像处理与模式识别等。汪国强，教授，硕士生导师。

是迄今在各学科领域中应用得最为广泛的一种维数，这主要是因为它非常容易由计算机求得。盒维的定义如下[5]：

设集 $F\subset R^n$，记 $N_\varepsilon(F)$ 是可以覆盖 F 的、边长为 ε 的 n 维立方体的最少个数，则 F 的上、下盒维分别定义为：

$$\overline{\dim}_B F=\overline{\lim_{\varepsilon\to 0}}\frac{\log N_\varepsilon(F)}{-\log\varepsilon},\underline{\dim}_B F=\underline{\lim_{\varepsilon\to 0}}\frac{\log N_\varepsilon(F)}{-\log\varepsilon}$$

若上、下盒维相等，即 $\overline{\dim}_B F=\underline{\dim}_B F$，则这一公共值称为集 F 的盒维，记为 $D_B(F)$。

根据上述定义，对二值图像计算盒维的方法为：用边长为 ε 的盒子去覆盖二值图像的黑色像素点集合，并统计盒子数 N，然后将边长和盒子数的对数即 $\log(\varepsilon)$ 和 $\log(N(\varepsilon))$ 点绘在坐标上，最后对标度区间内的点，利用线性回归方法确定直线斜率，该斜率就是所求的分形维数[10]。所以“数盒子”算法的关键是如何统计盒子数，传统的方法可能导致求得的分形维数偏小[2]。本系统采用改进的算法，引入一个参数 h 用来衡量某个盒子内的黑色像素点数与盒子像素点总数的相对大小，即：

$$h=\frac{k}{a^2}$$

其中，k 为盒子内的黑色像素点数，a 为盒子的边长，统计盒子数时只取 h 大于某个值的盒子进行统计，而略去那些只覆盖很少分形集点的盒子。这种改进的“数盒子”算法更能体现分形体的本质，因而能够更精确地测量其分形维数，将其用于图像检索，对于提高检索的准确率和效率都是有利的。

该文在计算分形维数方法的基础上再定义一个分形矢量，假设对分形集 F 分别用边长为 $\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_n$ 的盒子去覆盖，统计出的盒子数分别为 $N(\varepsilon_1),N(\varepsilon_2),\ldots,N(\varepsilon_n)$，并设该分形集的分形矢量为 $Vec(F)$，则该矢量的第 i 个分量定义为 $Vec_i(F)=\frac{\log N(\varepsilon_{i+1})-\log N(\varepsilon_i)}{\log(\frac{1}{\varepsilon_{i+1}})-\log(\frac{1}{\varepsilon_i})}$，$i=1,2,\cdots,n-1$。对于上文所述的在坐标上描的点，分形矢量的每一个分量就相当于计算坐标上每对相邻点间直线的斜率，该矢量能更全面地衡量图像的分形特征，而且对图像的变化更加敏感，所以将其作为图像的重要特征值。

综上所述，对二值图像计算分形维数与分形矢量的算法的伪代码如下：

输入 Image_array[n][m]：图像二值矩阵，矩阵中值 1 对应图像上黑色像素点，值 0 对应白色像素点；image_width：图像宽度；image_height：图像高度。

输出 FD：图像的分形维数；Fd_vector[4]：图像的分形矢量。

参数和辅助矩阵 Box_length[5]：盒子的边长矩阵；Box_count[5]：各边长对应的盒子数矩阵；h：某个盒子内像素点数与盒子内格子总数的相对大小；boxlength：当前的盒子边长；boxcount：当前的盒子数。

初始化 Box_length[5]={5,4,3,2,1}；Box_ count[5]各元素赋值为 0；FD_vector[4] 各元素赋值为 0；h:=0.2；boxlength:=0；boxcount:=0；tmp1:=0；tmp2:=0；tmp3:=0；tmp4:=0；tmp5:=0。

统计盒子数

```
for(k := 1 to 5)
    {boxlength :=Box_length[k]; i :=1; j :=1; boxcount :=0; i_step :=
boxlength-1;
    do {do {if(在 Image_array 矩阵的行坐标由 i~i+ i_step,列坐标由
j ~j +boxlength -1 的元素中，值为 1 的元素的个数>boxlength*
boxlength*h){boxcount :=boxcount+1 ;}
    j :=j+boxlength ;
    if(j+ i_step >image_width 且在 Image_array 矩阵的行坐标由 i~i+ i_
step,列坐标由 j~image_width 的元素中,值为 1 的元素的个数>boxlength*
boxlength*h){boxcount := boxcount +1 ;}
        }while(j<=box_width)
      i :=i+boxlength ; j :=0 ;
      if(i+ i_step >image_height){i_step :=image_height-i ; }while
(i<=image_height)
    Box_ count [k] := boxcount ;}
    计算分形维数
    for(k :=1 to 5)
      {tmp1 :=tmp1+ln(1/Box_length[k])*ln(Box_ count[k]) ;
      tmp2 :=tmp2+ ln(1/Box_length[k]) ;
      tmp3 :=tmp3+ln(Box_ count[k]) ;
      tmp4 :=tmp4+ln(1/Box_length[k])*ln(1/Box_length[k]) ;
      tmp5 :=tmp2*tmp2 ;FD :=(5*tmp1-tmp2*tmp3)/(5*tmp4-tmp5) ;
    计算分形矢量
    for(k :=1 to 4)
      {Fd_vector[k] :=(ln (Box_count[k+1]) -ln (Box_count[k]))/(ln
(1/Box_length[k+1])-ln(1/Box_length[k]))}
```

用 Koch 曲线和 Sierpinski 曲线检验该算法的效果，这两种曲线的理论分形维数分别为 1.262 和 1.585，而用本算法求得的分形维数分别为 1.266 和 1.601，误差分别为 0.317% 和 1.010%，说明此算法是较准确的。

作为例子，分别对 3 幅二值商标图像计算分形维数和分形矢量，图像见图 1，第 1 列显示了 3 幅不同的二值商标图像，其中图(a)与图(b)在视觉感受上比较相似，图(c)则与它们相差很远，本例考察相似的图像是否具有相近的分形维数与分形矢量，计算结果见表 1。从表 1 第二、三列数据可知，相似的图(a)与(b)确实是具有较之图(c)相近的分形维数与分形矢量，说明使用这两个分形特征来检索图像是合理的。

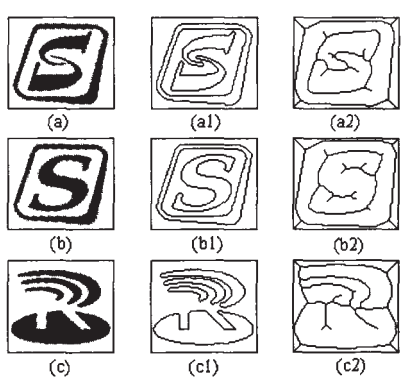


图 1　图像边界、骨架提取的例子

2.2　图像边界的提取

表 1　图像分形维数结果表

标号	原图的分维	原图的分形矢量	边界的分维	边界的分形矢量	骨架的分维	骨架的分形矢量
a	1.701	1.661,1.822,1.471,1.965	1.187	1.175,1.201,1.199,1.180	1.191	1.183,1.205,1.171,1.201
b	1.690	1.668,1.710,1.652,1.804	1.201	1.189,1.203,1.182,1.211	1.194	1.199,1.182,1.206,1.178
c	1.781	1.786,1.804,1.788,1.660	1.275	1.281,1.278,1.265,1.288	1.207	1.218,1.212,1.198,1.185

在二值图像中，所谓的对象就是值为 1 的连接在一起的像素集合。二值图像中对象的识别依赖于确定图像中相邻像素是否连接的约定方式，该系统使用的连接方式是 8-连接，即所有与用户期望像素点接触的 8 个像素点（垂直方向上的 4 个像素点和左上、右上、左下、右下等 4 个像素点）均为可能连接的像素。确定图像中某像素为边界像素的标准是该像素的值为 1，其领域中至少有一个像素是非零的。该文利用传统的微分算子中的梯度算子来提取图像边界[6]，由于是针对二值图像，所以使用简化的算法，该算法的伪代码如下：

输入 Image_array[n][m]：图像二值矩阵；image_width：图像宽度；image_height：图像的高度。

输出 Edge_array[n][m]：图像的边界矩阵。

参数 flag：若当前扫描的对象已出现第一个 1 元素且未出现最后一个 1 元素，则 flag=true，否则 flag=false。

初始化 Edge_array 各元素赋值为-1；flag：=false。

提取图像边界

```
for(i:=1 to image_height)
    {flag:=false;
    for(j:=1 to image_width)
        {if(当前对象还没出现第一个 1 元素){Edge_array[i,j]:=0;
continue;}
        if(Image_array[i,j]=1){if(当前对象第一次出现 1 元素)
{Edge_array[i,j]:=1;flag:=true;}
            else{Edge_array[i,j]:=0;}
        if(已扫描完一个对象){Edge_array[i,j-1]:=1;flag:=false;}}
    if(flag=true){Edge_array[i,j-1]:=1;} }
    for(j:=1 to image_width)
    {flag:=false;
    for(i:=1 to image_height)
        {if(当前对象还没出现第一个 1 元素){Edge_array[i,j]:=0;
continue;}
        if (Image_array [i,j]=1){if (当前对象第一次出现 1 元素)
{Edge_array[i,j]:=1;flag:=true;}
            else{Edge_array[i,j]:=0;}
        if(已扫描完一个对象){Edge_array[i-1,j]:=1;flag:=false;} }
    if(flag=true){Edge_array[i-1,j]:=1;} }
```

用本算法提取图像边界的效果如图 1 第二列所示，图(a1)、(b1)、(c1)分别是图(a)、(b)、(c)提取边界后的结果。在此基础上计算图像边界的分形矢量，作为图像的另一个检索特征值，计算结果见表 1。作为参考，表 1 也给出了图像边界的分形维数。从表 1 第四、五列数据可知，图像的边界同样具有较相似的图对应较相近的分形维数与分形矢量这一特点，从而也证明了使用边界分形矢量作为图像特征的合理性。

2.3 图像骨架的提取

骨架化算法起源于 Pfaltz 和 Rosenfeld 的骨架定义，他们把骨架定义为带状图像内最大圆盘的圆心集合。抽骨架也称为中轴变换或焚烧草地技术，一般来说，骨架主要具有 3 个特征：连续性、最小宽度为 1 和中心性。骨架化是将图像中的二维目标变换为其一维中线表示的过程，算法的思想是根据某些中心属性来直接定义骨架的像素，这种算法获取骨架的最常用技术是利用图像距离的变化来求得可能组成骨架的像素，然后根据上面骨架的 3 个标准特征来选择其中的子集。该系统在提取骨架时采用在文献[9]的算法基础上针对二值图像简化的算法，算法的伪代码如下：

输入 Image_array[n][m]：图像二值矩阵；image_width：图像宽度；image_height：图像的高度。

输出 Thin_array[n][m]：图像的骨架矩阵。

参数和辅助矩阵 Dist_array[ ]：局部距离矩阵；Flag_array[] 标号矩阵；B[2][]：像素桶；p：像素点；N(p)：像素 p 的 8 邻域；F：先进先出队列，其中 F.Add(p)表示将像素 p 加入到队列的尾部，q：=F.First()为取出队列的第一个元素 q，F.IsEmpty()为判断队列是否为空，若是则返回 TRUE，否则返回 FALSE。

初始化 Thin_array 各元素赋值为-1；Dis_array 各元素赋值为 0；flag：=0；dist：=0；MASK：=-2；SKE：=0。

提取图像骨架

```
将值为 0 的像素放入 B[0][]中，将值为 1 的像素放入 B[1][]中；
for(i := 0 to 1)
  {for(B[i][]中的任意像素 p)
    {Flag_array[p]:=MASK;
    if(存在 q∈N(p)满足 Flag_array[q]>0 或 Flag_array[q]=SKE)
{Dist_array[p]:=1;F.Add(p);} }
  dist:=1;
  设 w=(-1,-1)为假想像素，A.Add(w);
  while(TRUE){p:=F.First();
      if(p=w){if(F.IsEmpty())break;else{F.Add(w);p:=F.First
();dist:=dist+1;} } }
  for(任意 q∈N(p))
    {if(Dist_array[q]<dist and(Flag_array[q]>0 or Flag_array[q]=SKE))
      {if(Flag_array[q]>0){if(Flag_array[p]=MASK or Flag_array[p]=SKE)
          {if(p 既非端点也非脊点)Flag_array [p]:=Flag_array[q];
          else Flag_array [p]:=SKE;}
          else if(Flag_array [p]!=Flag_array [q])Flag_array[p]:=SKE;}
        else if(Flag_array[p]=MASK)Flag_array[p]:=SKE;}
     else if(Flag_array [q]=MASK and Dist_array[q]=0){Dist_array
[q]:=dist+1;F.Add(q);} }
  for(B[h]中的任意像素 p)
    {Dist_array[p]:=0;
    if(O[p]=MASK){flag:=flag+1;F.Add(p);O[p]:=flag;
        while(F.IsEmpty()=FALSE)
            {q=F.First();
            for(任意 s∈N(p))
                {if(Flag_array[s]=MASK){F.Add(s);O[s]:=flag;}}}}}
  if(Flag_array[p]=SKE){p 为骨架像素；Thin_array 中对应 p 的元素为 1，其余为 0;}
```

用该算法提取图像骨架的效果如图 1 第三列所示，图(a2)、(b2)、(c2)分别是图(a)、(b)、(c)提取边界后的结果。在此基础上计算图像骨架的分形矢量，作为图像的第四个检索特征值，计算结果见表 1 第七列数据。结果同样证明了使用图像骨架的分形矢量来衡量图像间的相似程度是合理的。

3 图像相似度的定义

图像特征提取之后，使用特征向量的距离来定义两幅图的相似度，距离越大表示相似度越小，该文采用欧几里德公式来计算距离。每个特征对应一个距离，图像总距离的计算使用加权求和的方法，较重要的特征对应较大的权重。令 Fd1、Fd2、Fd_v1[4]、Fd_v2[4]、Efd_v1[4]、Efd_v2[4]、Sfd_v1[4]、Sfd_v2[4]分别表示图(a)、图(b)的分形维数、分形矢量、边界分形矢量、骨架分形矢量的值，且设这 4 个特征对应的权重分别为 w_1，w_2，…，w_4 对应的图像距离分别为 d_1，d_2，…，d_4，则图 a 与图 b

的距离计算公式为：$Dist=\sum_{i=1}^{4} w_i d_i$，其中 $d_1=|Fd1-Fd2|$，$d_2=\sqrt{\sum_{i=1}^{4}(Fd_v1[i]-Fd_v2[i])^2}$，$d_3=\sqrt{\sum_{i=1}^{4}(Efd_v1[i]-Efd_v2[i])^2}$，$d_4=\sqrt{\sum_{i=1}^{4}(Sfd_v1[i]-Sfd_v2[i])^2}$。

检索图像时按图像总距离从小到大排序，最后显示与查询图像距离最小的即最相似的若干幅图像作为检索结果。

4 实验结果

为了检验文章提出的检索方法的有效性，对一个含有2000幅二值商标图案的库进行两个实验，首先从中随机选取500幅图案进行平移、旋转、缩放或挤压变换，分别将原图和变换后的图作为查询图像，然后通过不同的方法进行检索，看能否检索出库中相应的原图(目标图像)。第一个实验是为了比较各个特征的检索效果，所以单独使用各个特征进行检索，统计结果见表2，表中的数据表示目标图像能出现在检索结果前20位的图像所占的比率，由表2数据知，使用原图与平移后的图像作为查询图像时，检索效果是比较好的，而另外三种查询图像由于其像素分布发生了不同程度的改变，所以检索效果也有所下降。另一方面，使用图像分形矢量的检索效果是最好的，图像分形维数的检索率则相对较低，而边界分形矢量与骨架分形矢量的效果差不多。总的来说，单独使用一种特征来检索图像，检索率都是不够高的，检索率不高并不是检索不到库中的目标图像，而是目标图像在检索结果中所处的位置偏后(超过20)，这是因为完全不同的图像也可能具有很相近的分形维数或分形矢量。但由于相似的图像必定会具有相近的分形维数与分形矢量，所以若综合使用多种特征来检索图像，采用“逐步筛选”的方式，即对每一图像特征都筛选掉与查询图像相似度较小的一部分图像，逐步减少目标集，则必能提高检索率。

表2 实验1检索结果统计表(%)

查询图像	图像分形维数	图像分形矢量	边界分形矢量	骨架分形矢量
原图	100.0	100.0	100.0	100.0
平移后的图像	99.2	99.6	98.4	98.0
旋转后的图像	93.2	97.8	94.0	96.2
缩放后的图像	84.2	90.6	87.8	86.4
挤压后的图像	88.2	93.8	90.0	92.4

已做的第二个实验就是检验综合使用4个特征进行检索的效果，检索结果见表3，其中n表示目标图像在检索结果中所处的位置。由表3的结果数据知，同时使用该文提出的4个图像特征进行检索，能取得很好的检索效果，并且只要目标图像在图库中，就一定能通过该方法检索出来。另外，对图像作局部的改变后，仍能通过以原图作为查询对象而检索出改变后的图像，说明该检索方法还具有一定的鲁棒性，这在商标的防伪、防侵权等方面有很大的使用价值。图2给出部分例子。

表3 实验2检索结果统计表(%)

查询图像	n=1	n≤3	n≤6	n≤20	未检索到
原图	100.0	100.0	100.0	100.0	0
平移后的图像	98.4	99.2	100.0	100.0	0
旋转后的图像	98.8	99.2	99.6	100.0	0
缩放后的图像	93.2	96.2	97.8	100.0	0
挤压后的图像	96.4	97.8	98.4	100.0	0



(a)待查询的商标图像　(b)图(a)的检索结果

图2 检索结果

5 结论

该文论述了基于分形特征的二值图像检索方法，实验表明该方法计算简单、有效，匹配快速，检索结果比较理想，同时还具有一定的鲁棒性，证明这种检索方法是具有较大实用意义的。但该方法仍存在不足之处，笔者进一步的努力方向是把这一方法应用在对灰度图、彩色图的检索上，并可进一步提高检索精度，目前已取得了初步的成果。(收稿日期：2003年3月)

参考文献

1.P W Huang,S K Dai.Image retrieval by texture similarity[J].Pattern Recongnition,2003;36:665~679

2.Yuan Y Tang,Yu Tao,Ernest C M Lam.New method for feature extraction based on fractal behavior[J].Pattern Recognition,2002;35:1071~1081

3.Zhang Ping,Chen Lihui.A novel feature extraction method and hybrid tree classification for handwritten numeral recognition[J].Pattern Recognition Letters,2002;23:45~56

4.Ajay Kumar Bisoi,Jibitesh Mishra.On calculation of fractal dimension of images[J].Pattern Recognition Letters,2001;22:631~637

5.陈守吉,张立明.分形与图像压缩[M].上海科技教育出版社:1(40)

6.章毓晋.图象处理和分析图象工程上册[M].北京:清华大学出版社,181~195,218~251

7.李向阳,庄越挺,潘云鹤.基于内容的图像检索技术与系统[J].计算机研究与发展,2001;3

8.毛力,张晓林.基于内容的图像检索技术与系统[J].信息检索技术1999;5

9.刘俊义,王润生.基于冲刷模拟的灰度模式骨架算法[J].电子学报,2001;9

10.梁东方,李玉梁,江春波.测量分维的“数盒子”算法研究[J]中国图象图形学报,2002;3

(上接55页)

10.Andrew Fitzgibbon,Maurizio Pilu,Robert B Fisher.Direct least Square Fitting of Ellipses.1998

11.M J Black,Y Yacoob.Tracking and Recognizing rigid and nonrigid facial motion using local Parametric Model of Image Motion[C].In:proc ECCV,1995:374~381

12.Jorgen Ahlberg.Face Tracking for Model-based Coding and Face Animation[C].In:LiTH-Isy 2002,2002

13.Fu Jie Huang,Tsuhan Chen.Tracking of Multiple Faces for Human-Computer Interfaces and Virtual Environments.2000