一种基于网络矢量图形 SVG 仿真展示光栅图像的有效方法

袁家政[1,2], 须 德[1], 王育坚[2], 鲍 泓[2]
(1. 北京交通大学计算机与信息技术学院, 北京 100044; 2. 北京联合大学信息技术研究所, 北京 100101)

摘 要: 提出了一种基于网络矢量图形 SVG 展示光栅图像的方法(简称为 SRRI). 首先研究了光栅图像和 SVG 图形的结构组成, 同时依据局部特征(灰度、颜色或者梯度等)相似性将图像分割成若干个互不相交的区域; 然后提取每个区域的边缘, 并对边缘进行合并、压缩; 最后使用 SVG 代码描述分割后的区域边缘形状和颜色特征, 并依据区域同质特征将 SVG 代码进行合并和压缩. 通过数字文物图像集和标准图像数据集进行实验, 表明 SRRI 方法对于粗纹理图像和相似图案较多的文物数字图像, 其存储容量小, 展示效果好.

关键词: 可伸缩矢量图形; 光栅图像; SVG 仿真展示; 图像区域分割
中图分类号: TP391.41 文献标识码: A 文章编号: 0372-2112 (2008) 01-0188-06

An Effective Method for SVG-Based Rendering of Real Images

YUAN Jia-zheng[1,2], XU De[1], WANG Yu-jian[2], BAO Hong[2]
(1. *School of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044, *China*;
2. *Institute of Information Technology, Beijing Union University, Beijing* 100101, *China*)

Abstract: A novel method to convert raster images in a scalable vector graphic(SVG) format using similar features of pixel is presented(call SRRI). SRRI starts image segmentation for raster images with similar features; the edges of image segmentation are obtained and there are rendered by SVG with polygon or Bezier curve. Then the obtained SVG graphics with different region edges for similar features of pixel are merged if the converted SVG is very large size. Experimental results prove the SVG rendering effectiveness of SRRI method and goods performance for converting raster images. The SVG rendering approach is very useful for huge size images and thick texture culture relic images of digital museum.

Key words: SVG(scalable vector graphics); raster images; SVG rendering method; image segmentation for similar features

1 引言

SVG[1](Scalable Vector Graphics 可伸缩矢量图形)是由 W3 公司制定的一个全新基于 XML(eXtensible Markup Language 可扩展的标识语言)特性的二维文本网络图像格式. 目前国内外关于 SVG 的优化结构设计[2~4]、文档分类[5]、SVG 数据库存储设计[6]和软件体系结构[7]等理论和应用方面有了初步的研究; 但是 SVG 处理光栅图像的功能比较弱, 仅提供了内嵌的图像标识〈img〉, 因此关于如何使用矢量图形 SVG 表达光栅图像成为国内外部分学者近期研究的热点. Antoniou etal[8]和 Dr. Lakshman[9]提出根据像素的颜色特征作为区域分割来将光栅图像转换为 SVG 的方法(本文称为 ADS 方法), Battiato etal[10]提出了将光栅图像的每一个区域划分为三角区域, 使用三角测量 DDT 方式将其转换为 SVG 矢量图形, 另外还有一种商用软件 VectorEye[11](本文称为 VE 方法)也可以将图像转换为 SVG 文档. ADS 方法的思想是依据区域内颜色值相同将图像划分为若干区域, 提取每个区域的区域边缘(区域边缘未作任何拟合处理), 使用 SVG 代码描述分割的图像区域边缘和颜色特征, 但是 ADS 方法产生的 SVG 文档数据量大[10], 且放大时会有许多边缘锯齿和色斑效应. DDT 方法是在 ADS 方法的基础上对图像分割产生的区域进行三角形划分, 再依据像素颜色特征相似的多个相邻三角形区域进行合并, 最后使用 SVG 来描述三角区域的形状和像素颜色值, 但是 DDT 方法在处理纹理细节较多的图像时会产生大量的三角形, SVG 文档容量也急剧增大, 当 SVG 文档展示放大到一定倍数时, 三角效应明显. 本文提出一种新的使用基于 SVG 仿真展示实时光栅图像的方法(简称 SRRI 方法), 该方法根据像素邻接关系和像素颜色特征的相似性进行区域填充与合并, 将图像分割成互不相交的区域, 提取每一区域边缘, 对区域边缘进行多边形拟

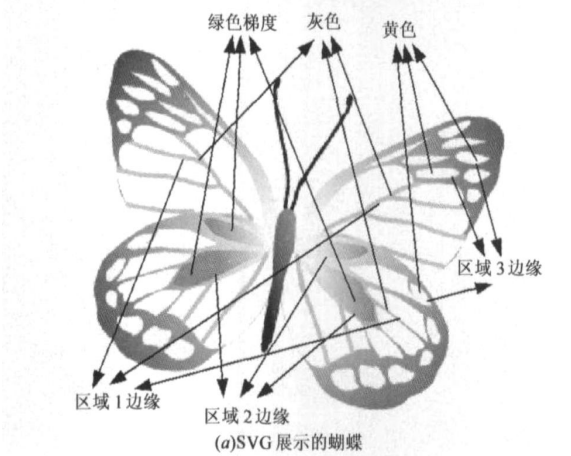
收稿日期: 2006-12-12; 修回日期: 2007-07-06
基金项目: 教育部科学技术研究重点项目(No. 2002KJ124); 北京市优秀人才培养资助项目(No. 20051D0502206)

合、合并、伸展或者闭合, 再使用 SVG 进行描述, 得到了质量比较好的 SVG 矢量图形.

2　光栅图像的区域分割与合并方法

2.1　SVG 的文件组织结构与形式化描述

二进制光栅图像是由像素点构成的位图结构, 当描述高质量实体时, 图像的存储容量比较大. 但是一幅图像的整体或者各个局部的相邻像素点之间一般存在相似特征, SVG 标准正是依据图形的各个区域的形状和颜色特征相似性来描述图形的, 这样描述的图形存储容量小、显示的效果好. 如图 1(*a*) 是使用 SVG 展示的蝴蝶图, 采用的是将整个图形依据局部像素存在相似性划分为彼此互不相交的区域, 每个区域由区域边缘形状坐标和颜色特征描述, 其中颜色特征可以是灰度、单一颜色值、颜色梯度或者复杂颜色特征, 图 1(*b*) 则是蝴蝶图对应的 SVG 代码, 使用元素标识〈path〉描述区域, 属性标识"d = "描述区域边缘形状, "class = "、〈linearGradient〉和"fill"等标识描述区域颜色特征. SVG



(*a*)SVG 展示的蝴蝶

```
<?xml version="1.0" encoding="iso-8859-1"?>                    SVG文件头
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
 "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd" [
      <!ENTITY ns_svg "http://www.w3.org/2000/svg">
      <!ENTITY ns_xlink "http://www.w3.org/1999/xlink">
  ]>
<svg    xmlns="&ns_svg;" xmlns:xlink="&ns_xlink;" ... ...
        xml:space="preserve">
        <style type="text/css">
        <![CDATA[
            .color1{fill:none,stroke:#B3B3B3,stroke-width:0.4,stroke-linecap:round,}
            ... ...
            .color3 {fill:#F8F65F,}                    定义区域单色颜色特征
        ]]>
        </style>
```

```
<defs>                                                         SVG图形体
  <linearGradient id="region2" ... ... x1="-1868.5391" ... ...-7802.6655">
      <stop  offset="0" style="stop-color:#F3F87F" />
... ...
      <a:midPointStop  offset="1" style="stop-color:#2B922A"/>
  </linearGradient>                                  定义区域梯度颜色特征
</defs>
<g  id="Butterfly" width="300" height="300" x="10" y="20">
    <path id="region1" class="color1" d="M7.58-10.72c0,... ... ..."/>
    <path style="fill:url(#region2)" d="M-1.39-3.12c0,...-1.9-1.24-1.9z"/>
    <path id="region3" class="color3" d="M-0.3-12.92c0,...3.63z"/>
    ...
</g>
</svg>
```

区域引用颜色特征　区域边缘形状特征

图形区域代码

(*b*)SVG 代码

图 1　SVG 图形结构

描述图像时, 一般通过〈image〉元素标识直接嵌入基于 Base64 编码的像素值或文件名链接, 这样描述的图像仍是位图结构, 因而存储容量大、显示效果差, 无法利用 SVG 的矢量性质. 为了解决这个问题, 必须将图像划分为若干区域, 使用形状和颜色特征来描述这些区域, 使其转换能够利用 SVG 描述的基本或复杂的矢量图形. 图像区域划分定义如下:

定义 1　图像的一个区域划分. 令 I 是一个图像空间, $R \in I$ 是一幅光栅二进制图像, R 的一个 K 区域 $R_i\{i = 1, 2, \cdots, k\}$ 的划分, 必须满足以下条件:

$$(1)\ R = \bigcup_{i=1}^{K} R_i \tag{1}$$

其中, $R_i \cap R_j = \varnothing$; $\forall R_i, R_j, i \neq j$; $i = 1, 2, \cdots, K$; $j = 1, 2, \cdots, K$; $\varnothing$ 表示不包含任何像素的空集.

(2) 对于 R 中的每一个区域 R_i 都有一个同质函数 F_i, R_i 中的每个像素 $V(x_{ij}, y_{ij})$ 性质相同, 记作: $R_i = \{(V(x_{ij}, y_{ij}), F_i) \mid i = 1, 2, \cdots k; j = 1, 2, \cdots, m\}$.

2.2　图像的区域分割方法

SRRI 方法首先将图像进行区域分割, 图像区域划分分为两个步骤: 区域增长和区域合并. 区域增长主要采用文献[13] 中的方法进行改进通过计算区域中的一个"种子"像素与其邻接像素的相似特性来扩大区域, 使其扩大到不能再扩大为止. 其步骤如下:

(1) 初始, 根据一定规则随机选取图像 I 的未知区域 R_i 中一个"种子"像素 C, 该像素满足 F_i, 计算其 8 个邻接像素 $V(i, j)$ 与"种子"像素 C 的相似特性: $|V(x_i - y_j) - F_i| \leqslant \Theta$(其中 Θ 为选取的阈值), 如果相似则将邻接像素加入 R_i 中.

(2) 计算 R_i 中新加入的每一个像素的邻接像数, 判断其是否包含在 R_i 中和是否满足 $|V(x_i, y_j) - F_i| \leqslant \Theta$? 如满足则将该像素加 R_i 入中, 重复(2) 直到区域 R_i 的邻接像数都不满足 $|V(x_i, y_j) - F_i| \leqslant \Theta$ 为止, 这样使 R_i 变得最大, R_i 则为 R 中已通过计算求得的一个区域.

(3) 选取 R_i 中的一个未被计算区域的邻接像素 C' 作为新的待计算区域的一个种子像素, 再利用(1) 和(2) 计算新的区域, 直到图像 I 中的所有像素都属于某一个区域划分为止, 这样就得到了一个图像 R 的区域分割划分.

根据文献[2 ~ 4] 和定义 1, 区域 R_i 可以记为: $R_i = (e(R_i), F_i)$, $e(R_i)$ 为 R_i 的边缘, 因此图像 R 可以记为:

$$R = \bigcup_{i=1}^{k} (e(R_i), F_i) \tag{2}$$

其中 $R_i \cap R_j = \varnothing, \forall R_i, R_j, i \neq j, i = 1, 2, \cdots, K$; $j = 1, 2, \cdots, K$.

3 图像区域边缘处理与 SVG 仿真展示方法

在 SRRI 方法中, 经过初步分割的图像区域还必须经过: 区域边缘形状提取、边缘处理和边缘曲线拟合、SVG 代码仿真转换和 SVG 代码压缩合并等步骤, 图像区域处理方法如下:

Step1 区域边缘提取: 令 R_i 为图像 R 上的一个分割区域, 则提取 R_i 的边缘轮廓 $e(R_i)$, 记为: $e(R_i)=P_{i_1}P_{i_2}\cdots P_{i_k}P_{i_1}$, 其中 $P_{ij}=V(x_{ij}, y_{ij})$, $|V(x_{ij}, y_{ij})-F_i|\leqslant\Theta$, P_{ij} 为 R_i 的边缘轮廓 $e(R_i)$ 的像素顶点位置, F_i 为 R_i 的同质函数; 保存 $R_i=(e(R_i), F_i)$.

Step2 $e(R_i)$ 边缘处理:

(1) 由于 $e(R_i)$ 是一些点构成的折线或者折线多边形, 假设 B_i 与 B_j 是两个开放式区域边缘, $B_i\cap B_j=\varnothing$, 如果 B_i 与 B_j 的距离 $distance(B_i, B_j)\leqslant\Theta_1$(其中 Θ_1 为选取的阈值), 并且 $F_i=F_j$, 如图 2(a) 和 (b) 所示, 则延长 B_i 与 B_j 中距离最近的两个端点, 使其合并为 T_i.

(2) 假定 $T=P_1P_2\cdots P_n$(其中 $P_i(i=1, 2, \cdots, n)$ 为边缘顶点) 是区域 R_i 的边缘, $T'=P_{i_1}P_{i_2}\cdots P_{i_k}\cdots P_{i_l}(i=1, 2, \cdots, n; k=1, 2, \cdots, l)$ 是 T 中的部分边缘, 如果顶点 P_{i_k} 到直线 $P_{i_1}P_{i_l}$ 的最大距离满足 $d_{max}=\max(distance(P_{i_k}, P_{i_1}P_{i_l}))\leqslant\Theta_2$(其中 Θ_2 为选取的阈值), 则如图 2(c) 所示, 让 $T'\approx P_{i_1}P_{i_l}$.

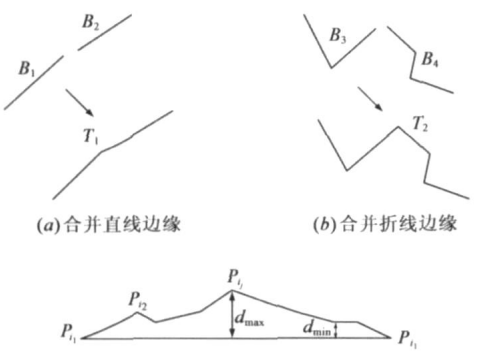


图 2 边缘处理

(3) 经过 (1)、(2) 方式处理后, 可以得到三种不同的图像区域边缘: 第一种如图 3(a) 所示, 由几个像素组成的开放式区域边缘, 这些边缘可以根据需要作为垃圾碎片丢弃; 第二种如果 3(b) 所示, 是一系列顶点构成的封闭区域边缘, 这些边缘构成了多边形、正/长方形、圆形和椭圆形等; 第三种如果 3(c) 所示, 是一系列顶点构成的开放区域边缘, 这些边缘构成了直线、折线和曲形等.

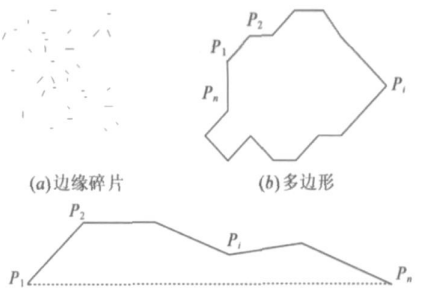


图 3 三种不同类型的边缘

Step3 边缘曲线拟合:

经过 Step2 处理后的边缘, 如果 $T=P_1\cdots C_1\cdots C_2\cdots P_2(C_i(i=1, 2)$ 是 T 中离直线 P_1P_2 距离最大的两个顶点) 是区域 R 的边缘部分, $d_{min}=\min(distance(C_i, line(P_1P_2)))>\Theta_1$, $d_{max}=\max(distance(C_i, line(P_1P_2)))<\Theta_2$, 则使用贝叶斯曲线 $P_1C_1C_2P_2$ 代替 T, 如图 4(a) 所示; 否则如果 $d_{max}=\max(distance(C_i, line(P_1P_2)))>\Theta_2$, 保留 T 作为多边形 $P_1P_2\cdots P_n$ 折线, 如图 4(b) 所示.

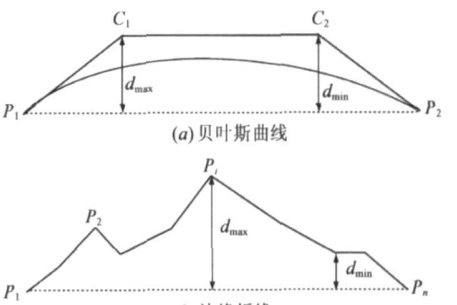


图 4 边缘曲线拟合

Step4 SVG 仿真展示转换方法:

经过以上区域边缘处理方法处理的图像区域 R_i 就很容易转换为 SVG 图像代码, 具体如下:

(1) 对于分割区域边缘是如图 3(b) 所示的封闭多边形, 可以使用如下 SVG 代码代替:

〈path d="M x_1, y_1 L x_2, y_2 L … L x_n, y_n Z" style="fill:(#FuctionI;)"/〉.

(2) 对于分割区域边缘是如图 3(c) 或者图 4(b) 所示的开放式多边形折线, 可以使用如下 SVG 代码代替:

〈path d="M x_1, y_1 L x_2, y_2 L … L x_n, y_n" style="fill:(#FuctionI;)"/〉.

(3) 对于分割区域边缘是如图 3(a) 所示的贝叶斯曲线, 可以使用如下 SVG 代码代替:

〈path d="M x_1, y_1 Q cx_1, cy_1 cx_2, cy_2 … L x_n, y_n" style="fill:(#FuctionI;)"/〉.

在以上 SVG 代码中 (x_1, y_1)、(x_2, y_2)、…、(x_n, y_n)

是区域边缘顶点坐标, Z 代表区域是封闭的, 而 FuctionI 描述的是区域同质函数 F_i 的 SVG 代码, FuctionI 代码通常描述如下:

〈! DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20001102//EN" "http://www.w3.org/TR/2000/CR-SVG-20001102/DTD/svg-20001102.dtd" [〈!

… …

.FuctionI "fill: # FFFFFF; stroke: # FFFFFF;" 〉//单一颜色定义代码

… …]〉

在以上 SVG 代码中# FFFFFF 是一个区域相同颜色性质描述, 如果是单一颜色性质, 则使用# RRGGBB 代替# FFFFFF, 其中# RRGGBB 是一个区域的平均颜色 RGB 色彩表示, 可以通过公式 $\left(\sum_{\text{region}_i} C_j\right)$ /Pixels number of region_i 计算.

当然如果区域颜色性质复杂, 同质函数 F_i 需要用 SVG 颜色代码段来定义 FuctionI 代码, 具体如下:

梯度颜色定义 FuctionI 代码	复杂颜色定义 FuctionI 代码
〈defs〉 〈linearGradient id = "FunctionI" …〉 … … 〈/linearGradient〉 〈/defs〉	〈defs〉 〈g id = "FunctionI" …〉 〈linearGradient id = "subcolor1" …〉 … … 〈/linearGradient〉 〈linearGradient id = "subcolor2" …〉 … … 〈/linearGradient〉 … … 〈/g〉 〈/defs〉

Step5　SVG 代码的压缩与合并:

在同一个 SVG 文件中, 如果有多个〈path〉代码的 FuctionI 值相同或相似, 则需要将它们合并以压缩 SVG 代码. 由于 SVG 是基于 *XML* 语法, 可以采用文献[5] 中的 XML 文档分类算法对 SVG 代码进行处理, 处理过程是通过提取图像中的每一个区域 SVG 代码的文档树[5] 进行结构、形状和颜色等特征的选取, 计算特征权重, 通过特征权重和支撑向量机[5] 获取 SVG 文档代码分类器, 利用分类器将图像各区域 SVG 文档代码进行归类, 对于属于同一类的图像区域 SVG 代码进行压缩和合并, 比如:

区域 1 对应的 SVG 代码 path 1: 〈path d = "M x_1, y_1 L x_2, y_2 …Z" style = "&FuctionI;"/〉

区域 2 对应的 SVG 代码 path 2: 〈path d = "M a_1, b_1 L a_2, b_2 …Z" style = "&FuctionI;"/〉

可以使用 path 3 代替 path 1 和 path 2 进行合并压缩, path 3 为: 〈path d = "M x_1, y_1 L x_2, y_2 …M a_1, b_1 L a_2, b_2 …Z" style = "&FuctionI;"/〉

4　数据分析与实验

在实验中, 使用了 1560 幅真实的数字图像进行实验, 其文件格式为 JPEG, 其中 520 幅为中国古代文物数字图像, 1040 幅图像取自标准实验图像库. 整个实验数据集使用划分为细纹理文物图像(Thin texture, 如铜鼎)、粗纹理物图像(Thick texture, 如玉杯)、人物画(People picture)、风景画(Scene picture)、Lena 图像、Tiger 图像和大明崇祯玖年潞国制铜鼎图像(简称 Ding) 等六类.

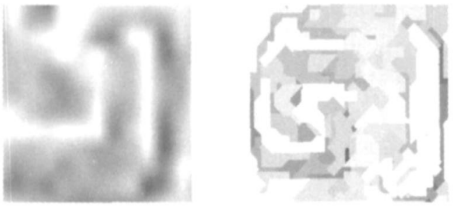
图 5(*a*) 描述的由手持式数码相机拍摄的大明崇祯玖年潞国制铜鼎照片, 图 5(*b*) 是使用 SRRI 方法进行分割图像的区域边缘图. 图 6(*a*) 和(*b*) 分别是图 5(*a*) 细线圈起来的细节部分经过普通插值算法和 SRRI 方法转换为 SVG 文档在 Internet Explorer+ Adobe SVGview 插件中显示分别放大 25 倍的结果. 从图 6 的对比中可以看出转换为 SVG 代码的仿真图像显示效果更好, 纹理细节更清晰, 并且放大不变形, 缺点是光栅二进制图像转换为 SVG 代码过程中在图像区域分割后的进行边缘处理时会丢失部分图像细节信息.

(*a*) 大明崇祯玖年潞国制铜鼎原图 (160×190)　　(*b*) 铜鼎区域分割边缘图

图 5　边缘提取实验数据

在标准的实验图像数据集中, 我们选用"Lena" 图像作为典型的光栅二进制图像例子, 使用普通插值算法、ADS 算法、Vecotr-Eye 软件、DDT 算法和 SRRI 算法分别将"Lena" 图像转换为 SVG 图形和眼睛部分进行 4 倍放大处理, 其实验结果如图 7 所示. 从图 7 可以看出普通插值放大图像有锯齿, 而 Vector Eye 商业软件处理的 SVG 图像区域斑纹明显, 效果很差; ADS 方法效果比 Vector Eye 商业软件稍好, 但仍有区域斑纹; 而 DDT 方法效果比 ADS 算法和 Vector Eye 商业软件效果好得多, 但是还是有少量三角效应, 尤其放大倍数越大越明显; 而 SRRI 方法的显示效果最好.

整个实验图像数据集分别使用 ADS 算法、DDT 算法、Vector-Eye 软件和 SRRI 方法转换图像, 按图像类别对每类图像计算反映生成 SVG 图像质量的峰值信噪比 PSNR(其中人物画和风景画使用的是 PSNR 的均值) 和图像文件存储大小, 所得数据如图 8 和图 9 所示. 通过



(a)普通插值方法 (b)SRRI方法

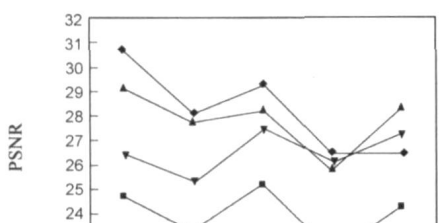
图6 铜鼎的细节25倍放大

(a)“Lena”原图及插值算法 (b)ADS算法

(c)Vector-Eye 软件 (d)DDT 算法 (e)SRRI方法

图7 图像及局部放大效果比较

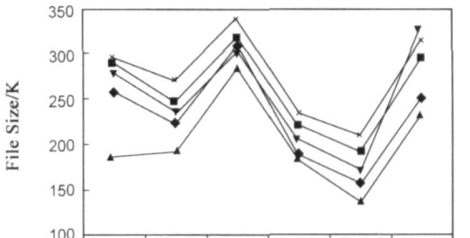
对比发现使用VE软件生成的SVG图像质量最差, ADS次之, DDT算法的质量比较好, 而SRRI方法除Tiger图像比ADS和DDT算法稍差外, 其他处理都要好于这两种算法. 比较JPEG格式的图像和四种算法生成SVG格式图像的文件大小, JPEG图像文件格式最小, SVG格式比JPEG格式的图像大了约5%~20%左右, 而在四种算法中, 采用VE方法生成的SVG文件最小(由于VE软件丢弃了大量细节), SRRI方法相对于ADS算法和DDT算法, 生成的图像质量不但好, 而且存储容量小.



	Ding	Lena	People picture	Scene picture	Tiger
-▼-ADS	26.4	25.3	27.5	26.1	27.2
-■-VE	24.7	23.3	25.2	22.4	24.3
-▲-DDT	29.1	27.7	28.2	25.8	28.3
-◆-SRRI	30.7	28.1	29.3	26.5	26.5

图8 算法的峰值信噪比PSNR

比较ADS、DDT和SRRI三种算法的发现: ADS算法采用SVG代码描述图像区域, 区域中的像素点颜色单一, 对于梯度颜色和复杂颜色则没有描述, 其形状边缘未作拟合处理, 图像质量较差; DDT算法利用三角形描述图像区域; SRRI算法利用边缘拟合的不规则区域描述具有相似特征(包括单一颜色、梯度颜色和混合颜色)的像素区域, 图像效果最好.



	Thin texture	Thick texture	Scene piture	People picture	Lena	Tiger
-◆-JPEG	257	223	308	190	156	251
-■-ADS	289	245	320	220	190	295
-▲-VE	185	191	283	182	137	232
-×-DDT	295	269	337	235	210	315
-▼-SRRI	278	235	301	205	173	327

图9 算法生成SVG图像的文件大小

5 结论

SVG对于处理光栅二进制图像功能较差, 本文针对图像中局部各区域本身是由相似特征的像素构成, 具有一定的固有特征, 将图像按固有特征分割成许多区域, 这些区域可以用形状和颜色相似特征描述, 提出了有利于转为SVG仿真描述的SRRI方法, 该方法应用到了虚拟文物数字化博物馆的文物图像展示, 为SVG展示传统图像提供了较好的方法和理论基础. 但是SRRI方法仍有局限性, 比如转换方法的最优性和复杂度, 我们没有深入研究. 另外该方法虽然对于粗纹理图像和相似图案较多的细纹理图像效果较好, 但对于纹理杂乱的细纹理图像转换为SVG的文档容量仍然较大.

参考文献:

[1] Scalable Vector Graphics(SVG) 1.2 Specification[EB/OL]. http://www.w3.org/TR/2004/WD-SVG12-2004-10-27/2004-10-27.

[2] 袁家政, 须德, 鲍泓. 基于XML的矢量图形SVG的结构设计与实现[J]. 计算机研究与发展, 2005, 42(Suppl A): 129-136.
Yuan Jiazheng, Xu De, Bao Hong. Architecture design & implementation of XML-based SVG[J]. Journal of Computer Research and Development, 2005, 42(Suppl A): 129-136. (in Chinese)

[3] 袁家政, 须德, 沈洪, 鲍泓. 一种有效的矢量图形SVG的结构化方法[J]. 系统仿真学报, 2006, 18(S1): 5-7+9.
Yuan Jiazheng, Xu De, Shen hong, Bao Hong. Effective structure method for SVG[J]. Journal of System Simulation. 2006, 18(S1): 5-7+9. (in Chinese)

[4] Jiazheng Yuan, De Xu, hong Sheng, Hong Bao. Structured method for XML-based SVG[J]. Journal of Computational Information System, 2007, 3(2): 519-525.

[5] 袁家政, 须德, 鲍泓. 基于结构与文本关键词相关度的 XML 网页分类研究[J]. 计算机研究与发展, 2006, 43(8): 1361- 367.
Yuan Jiazheng, Xu De, Bao Hong. An efficient XML documents classification method based on structure and keywords frequency[J]. Journal of Computer Research and Development, 2006, 43(8): 1361- 367. (in Chinese)

[6] 袁家政, 须德, 鲍泓. 基于 XML 的矢量图形 SVG 的数据库模型与存储研究[J]. 计算机研究与发展, 2006, 43(Suppl): 444- 450.
Yuan Jiazheng, Xu De, Bao Hong. Study of SVG database model & storage based on XML[J]. Journal of Computer Research and Development, 2006, 43(Suppl): 444- 450. (in Chinese)

[7] 袁家政, 须德, 鲍泓. 基于 XML 的矢量图形 SVG 应用的软件体系结构研究[J]. 中国图象图形学报, 2007, 12(4): 718- 725.
Yuan Jiazheng, Xu De, Bao Hong. Study of SVG software architecture based on XML[J]. Journal of Image and Graphics. 2007, 12(4): 718- 725. (in Chinese)

[8] B Antoniou, L Tsoulos. Converting raster images to XML and SVG[A]. Proceedings of SVG Open 2004[C]. Tokyo, Japan, 2004.

[9] Dr. Lakshman Prasad. Raster to vector conversion of images for efficient SVG representation[A]. Proceedings of SVG Open 2005[C]. Enschede, Netherlands, 2005.

[10] S Battiato, G Gallo, G Messina. SVG rendering of real images using data dependent triangulation[A]. Proceedings of ACM/SCCG 2004 Spring Conference on Computer Graphics[C]. Slovakia, 2004.

[11] Vector Eye. Raster to Vector Converter[EB/OL]. http://www.siame.com/index.html, 2003.

[12] Eugenio Di Sciascio, Francesco M Donini, Marina Mongiello. A knowledge based system for content-based retrieval of scalable vector graphics documents[A]. Proceedings of the 2004 ACM symposium on Applied computing[C]. New York, NY, USA: ACM Press, 2004. 1040- 1044.

[13] Chiou-Shann Fuh, Shun-Wen Cho, Essig K. Hierarchical color image region segmentation for content-based image retrieval system[J]. IEEE Transactions on Image Processing, 2000, 9(1): 156- 162.

作者简介:

袁家政　男, 1971 出生于湖南省隆回县, 北京联合大学副教授, CCF 会员, 2004 年起在北京交通大学攻读博士学位, 主要从事 XML、多媒体数据库和网络与信息安全等方面的研究.
E-mail: xxtjiazheng@buu.com.cn; jzyuan@sohu.com

须　德　男, 1944 出生于江苏常州, 北京交通大学教授, 博士生导师, 主要研究领域为数据库系统及应用和多媒体信息处理.
E-mail: xd@computer.njtu.edu.cn