

基于 K 均值聚类的快速分形编码方法

陈作平¹⁾ 叶正麟¹⁾ 郑红婵¹⁾ 赵红星^{1),2)}

¹⁾(西北工业大学理学院, 西安 710072) ²⁾(榆林学院数学系, 榆林 719000)

摘 要 针对目前分形图像压缩存在的编码时间过长问题, 提出了使用 K 均值聚类对编码过程进行加速的方法, 其中聚类向量采用图像块的正规化特征向量以保证聚类的精度, 并通过用部分失真搜索来完成传统 K 均值聚类中最耗时的最近邻搜索过程以提高聚类速度。进一步, 通过结合均值图像建库、去平坦块等技巧, 得到了一种快速、可调的分形编码方法。实验结果表明, 相对于全局搜索, 所提方法大幅地提高了编码速度和压缩比, 而解码质量只略有下降。

关键词 K 均值聚类 部分失真搜索 最近邻搜索 分形图像压缩

中图法分类号: TP391.41 **文献标识码:** A **文章编号:** 1006-8961(2007)04-0586-06

Fast Fractal Coding Technique Based on K -mean Clustering

CHEN Zuo-ping¹⁾, YE Zheng-lin¹⁾, ZHENG Hong-chan¹⁾, ZHAO Hong-xing^{1),2)}

¹⁾(School of Science, Northwestern Polytechnical University, Xi'an 710072)

²⁾(Department of Mathematics, School of Yulin, Yulin, 719000)

Abstract Long coding time is the main problem in image compression based on Fractal at present, mainly due to its heavy computation of searching the best-match domain block for each range block. In this paper, a fast K -mean clustering algorithm is proposed firstly using Partial Distortion Search to replace the time-consuming Nearest Neighbor Search process in traditional K -mean clustering algorithm. Then the K -mean clustering algorithm is used to speed up the coding: scheme the domain blocks and search the best-match block for each range block in some nearest neighbors from some nearest clusters. Furthermore, by combining other techniques such as excluding planar blocks and building domain pool from an averaged image, a fast and adjustable fractal coding scheme is obtained. Experimental results indicate that comparing to exhaustive search, the proposed method improves the coding speed and compression ratio greatly with slight quality degradation of decoded image.

Keywords K -mean clustering, partial distortion search, nearest neighbor search, fractal image compression

1 引言

分形图像压缩是近 15 年发展起来的一种图像压缩方法, 以其潜在的压缩比高、解码速度快和解码图像与分辨率无关等特点, 成为了图像压缩领域的一个重要研究课题, 但它存在一个最主要问题: 编码时间太长, 这实际上已成为了该方法走向高效能实用化的最主要障碍, 因此分形编码加速方法已成为了近年来分形压缩的一个研究热点。

1.1 分形编码的时间复杂度

在标准的分形编码方案中, 对待编码图像中每个子块 R (称为值域块, 它们构成了原图像的一种互不重叠的划分), 需求解以下最小化问题:

$$\min_{D \in S_D} \min_i (R, f \circ \tau_i \circ A(D)) \quad (1)$$

其中, S_D 称为定义域块库, 是图像中所有与 R 形状相同但尺寸大于 R (一般取为 2 倍) 的子块 D (称为定义域块) 构成的集合; A 是某种抽样算子, 使 D 的尺寸缩小到与 R 相同; $\tau_i, i = 1, \dots, 8$ 为空域上对图像的 8 种旋转-反射变换; f 为灰度变换: $f(D) = s \cdot$

基金项目: 陕西省自然科学基金项目 (2002A20)

收稿日期: 2005-05-20; 改回日期: 2005-12-20

第一作者简介: 陈作平 (1979 ~), 男, 西北工业大学理学院应用数学专业在读博士研究生。主要研究方向为计算机图像处理和计算机图形学。E-mail: ykhmiou@hotmail.com

$D + o \cdot 1$, 其中 1 是与 R 同阶的全 1 矩阵, s 称为对比度伸缩因子, 满足 $|s| < 1$, o 称为亮度偏移因子; $d(\cdot, \cdot)$ 是某种距离函数, 常取为均方误差 (MSE)。由式 (1) 可知, 每个值域块的编码时间复杂度为 $T = O(N_D)$, 其中 N_D 是 S_D 所含元素的个数, 它往往非常巨大, 这是导致分形编码时间过长的根本原因。

1.2 相关工作

在上述方案中, 每个值域块都对应于一个最优匹配块, 因而从理论上讲, 该值域块与其余 $N_D - 1$ 个定义域块的比较都是不必要的。因此, 降低时间复杂度的一个很自然的想法是尽可能减少不必要的比较的定义域块数目, 而将图像块进行分类则是对这一想法的一种具体实现, 它是根据图像的某些纹理特征将所有图像块 (包括定义域块和值域块, 下同) 分成若干类, 在编码时每个值域块的最优匹配块只在同类或相邻类的定义域块中寻找。代表方法有 Fisher 分类^[1]、Hurtgen 分类^[2,3]、维数分类^[4]、质心分类^[3]、原型分类^[1]和 Kohonen 自组织特征映射分类^[5]等, 其中前两者可看作“离散分类”, 因其分类依据为图像的一些离散特征 (例如图像块的四分子块的灰度均值及方差的大小排列顺序), 存在以下缺点: 一是只能分为固定的类数 (例如 72 和 360), 这使其加速能力有限; 二是类间纹理连续性差, 这使得一旦某一值域块在其所属类中找不到满足匹配精度的定义域块, 无法将搜索扩大到其相邻的类中去继续搜索, 因而其解码图像质量较差。后 4 种方法可视为“连续分类”, 它们通过采用连续的纹理特征弥补了上述缺点, 其中前两者采用了低维特征 (图像的毯子维数和两个灰度质心角), 计算简单, 但随分类数目的增大图像降质较大; 后两者使用高维 (例如 16 维) 特征向量来分类, 相对于前两者能取得更好的解码质量, 但因它们都需经过一个耗时的训练过程, 因而加速效果较前者稍差。本文首先对 K 均值聚类算法进行改进, 得到了一种快速的 K 均值聚类方法, 然后用它对图像块进行分类, 并结合去平坦块等技巧得到了一种快速的分形编码方法。

2 改进的 K 均值聚类算法

2.1 K 均值聚类算法

对实践中经常遇到的数据聚类问题, K 均值聚类算法以其简单有效性得到了较广泛的应用。设有

数据集

$$X = \{x_i\}, x_i \in \mathbf{R}^n, i = 1, 2, \dots, N$$

要将其分为 K 类, 即对数据集作划分: $X = \bigcup_{j=1}^K C_j, C_j \cap C_{j_2} = \emptyset, j_1 \neq j_2$ 。采用 K -均值算法^[7]的聚类过程如下:

Step1 初始化聚类中心 c_j , 实际中常取为 X 的最初 K 个向量, 此时各类 $C_j = \emptyset$;

Step2 使用聚类中心对 X 进行聚类, 即

$$C_j := \{x \mid \|x - c_j\| = \min_{k=1, \dots, K} \|x - c_k\|, x \in X\} \quad (2)$$

$$j = 1, 2, \dots, K$$

Step3 计算新的聚类中心

$$c_j^* := \frac{1}{K_j} \sum_{x \in C_j} x, j = 1, 2, \dots, K \quad (3)$$

Step4 计算新旧聚类中心的差别: $\text{diff} =$

$$\sum_{j=1}^K \|c_j^* - c_j\|, \text{若 } \text{diff} < T, \text{算法停止; 否则 } c_j = c_j^*, j = 1, \dots, K, \text{转至 Step2}.$$

其中, $\|\cdot\|$ 为某种范数, K_j 为 X 中属于 C_j 的向量数目, T 为给定的很小的实数。在 Step2 中,

$$\|x - c_j\| = \min_{k=1, \dots, K} \|x - c_k\| \quad (4)$$

即是所谓的“最近邻搜索” (nearest neighbor search, NNS) 问题, 算法的绝大部分计算量即在于此; 特别地, 在 2-范数下, 算法的计算量主要花费在乘法上。传统的 K 均值聚类采用穷举法求解式 (4), 即依次计算 x 和每个 c_k 的距离并在其中找一个最小者, 其时间复杂度为 $O(K)$, 称之为“全失真搜索” (full distortion search, FDS)。假设算法经过 m 次迭代后达到收敛, 则其所需乘法次数为 $m \times (N \times K \times n + K \times n)$, 其中 $N \times K \times n$ 是每次迭代中式 (2) 所需乘法次数, 而 $K \times n$ 则是第 4 步所需乘法次数, 因此当数据集规模 (n 或 N) 或聚类数目 K 较大时, 采用 FDS 进行 K -均值聚类是很慢的。为此, 从减少乘法次数的角度出发, 使用下述的部分失真搜索来求解式 (4), 从而降低算法的时间复杂度。

2.2 部分失真搜索

部分失真搜索^[8] (partial distortion search, PDS) 算法是一种简单有效的最近邻搜索算法, 其基本思想是在计算某个 c_k 和 x 之间失真的过程中始终判断累加的部分失真是否已超过目前的最小失真, 一旦超出则终止它们之间的计算: 假定目前最小失真

$$\text{为 } d_{\min} = d(x, c_p), 1 \leq p \leq K, \text{若存在 } \sum_{i=1}^s (x_i - c_{u_i})^2 \geq d_{\min}, 1 \leq s \leq n, \text{则 } d(x, c_i) \geq d_{\min}.$$

扩展的部分失真搜索^[9] (extended partial distortion search, EPDS) 是 PDS 的一种改进形式, 它能最大限度地减少最近邻搜索的乘法次数, 其基本过程可描述如下:

(1) 令 $h_j = 1$, 并计算 x 的第 1 维分量 x_1 和聚类中心 c_j 的第 1 维分量 c_{j1} 间的失真 $D_j = (x_1 - c_{j1})^2$, $j = 1, 2, \dots, K$, 其中 h_j 表示聚类中心 c_j 的第 h_j 维;

(2) 找出 $D_j = \min_j D_j$, $s = \operatorname{argmin} D_j$;

(3) 若 $h_s = K$, 则 c_s 为距离 x 最近的聚类中心, 算法终止; 否则 $h_s = h_s + 1$, 计算 x 与 c_s 间的第 h_s 维失真, 将结果同 D_s 相加, 转步骤 2。

PDS 和 EPDS 的效率皆在于以增加比较运算来换得乘法和加法运算, 例如 PDS 是以 s 次比较换得减少 $n - s$ 次乘法和 $2(n - s)$ 次加法。在空间开销方面, PDS 不需要额外的存储空间, 而 EPDS 也只需 $K \times b$ 字节的额外存储空间。在速度方面, 其加速效率一方面依赖于数据集本身的性质 (越不均匀加速效果越好), 另一方面依赖于所采用的处理器, 因为乘法和比较运算的复杂度大小关系随处理器的不同而不同。目前, 两者都已用于矢量量化编码的加速并取得了较好的效果。

2.3 改进的 K-均值聚类算法

根据上述分析, 对 K 均值聚类提出以下改进: 其一, 在步骤 2 中用 PDS 或 EPDS 来完成距离最近的聚类中心的搜索, 其二, 同样采用 PDS 的思想, 将步骤 4 改为

步骤 4 初始时 $j = h = 1$, $\text{diff} = (c_{j,h}^* - c_{j,h})^2$, 若 $\text{diff} \geq T$, 则 $c_j = c_j^*$, $j = 1, \dots, K$, 转至第 2 步; 否则 $\text{diff} = \text{diff} + (c_{j,h+1}^* - c_{j,h+1})^2$, 转步骤 4 判断语句。

由于 T 一般非常小, 因此改进后的 Step4 能极大的减少乘法次数, 在较好情况下可以用 m 次比较换取 $(m - 1) \times K \times n$ 次乘法。取 $T = 0.001$, 使用 4000 个 16 维均匀随机向量 (各分量皆是 $[0, 110]$ 内的随机数) 对传统 K 均值聚类 (记为 KM-FDS) 和改进的 K 均值聚类 (记为 KM-EPDS 和 KM-EPDS) 做了比较实验, 结果如图 1、2 所示, 其中 M 表示聚类类别数目。实验环境为 P4 2.6GHz/256RAM/Visual C++ .Net (下同), 比较指标为算法达到收敛时所进行的乘法次数和实际时间。

图 1 印证了改进的 K 均值聚类算法能大幅度减少乘法次数的结论, 但在运行时间方面, 由图 2 看出, KM-EPDS 表现很差, 这是由于它增加的比较运

算次数太多的缘故, 而 KM-PDS 则表现最好, 因此在下面的分形编码中, 使用后者来进行聚类。

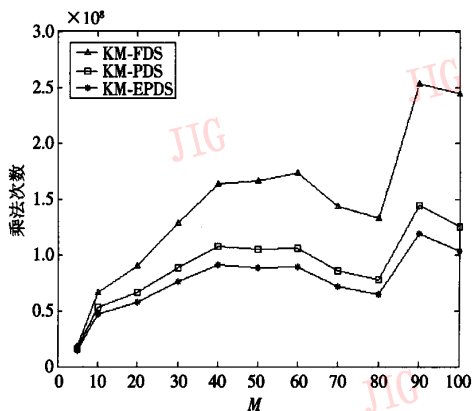
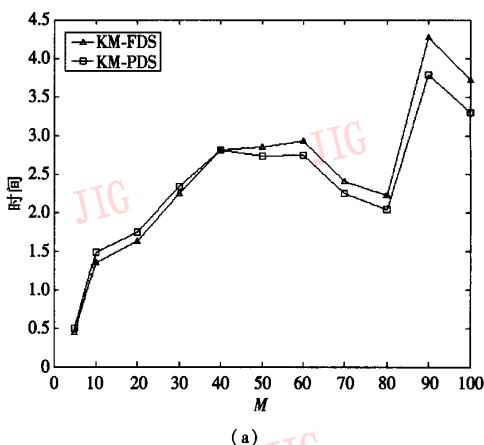
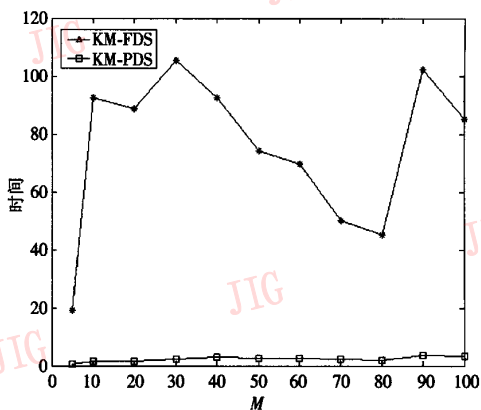


图 1 3 种 K 均值聚类算法的乘法次数比较

Fig. 1 Comparison results on multiplications of three K-means clustering algorithms



(a)



(b)

图 2 3 种 K 均值聚类算法的运行时间比较

Fig. 2 Comparison results on time of three K-means clustering algorithms

3 基于改进K均值聚类的快速分形编码方法

3.1 Saupe 特征K均值聚类

使用 Saupe 提出的正规化投影向量^[10] (简称 Saupe 特征)来对定义域块进行K均值聚类,因为该特征的引入将式(1)等价于特征空间中值域块与定义域块对应特征向量间的最近邻搜索问题。显然,这种等价性极大地提高了分类准确性,即视觉上相似的图像块被最大限度地分在了一个类中;然而另一方面,该等价性是在灰度变换参数未经量化的基础上取得的,这些参数被量化后将无法保证所对应定义域块仍是(在MSE下)最优的,为此,在为值域块编码时,可搜索与其同类及少数邻近类中的少数(不再是一个)Saupe特征最近的定义域块。另外,去掉反射-旋转变换,因为在率失真意义下它们是不必要的^[11],而且从加速的角度看,这样做还可减少参与聚类的向量数目和7倍的搜索量。

3.2 灰度变换的选取

考虑到标准的分形编码方案中,参数 s 和 o 之间所具有的强相关性以及 o 的动态范围的不确定性给它们的量化带来的不便,采用如下的灰度变换^[6]:

$$f(D) = s \cdot D + o \cdot 1 = s(D - \mu_D \cdot 1) + \mu_R \cdot 1$$

其中, μ_D, μ_R 分别表示定义域块和值域块的灰度均值。它有如下优点:其一,参数 s 和 μ_R 几乎是不相关的;其二,减少计算量,因为编码时对每个值域块,在与定义域块比较时只需要计算一个参数 s 就可以了;其三,提高压缩比,因为参数 μ_R 的动态范围是 o 的动态范围的1/2;其四,提高解码速度,因为可使用量化后的 μ_R 来构造初始解码图像,由于后者已是原图像的低分辨率近似,根据拼贴定理,迭代解码的收敛速度必然提高。

3.3 算法的实现

第1步 建立定义域块库。首先得到原始图像的均值图像,该图像将被用来建立定义域块库,这样可在编码阶段避免对定义域块的重复抽样。其次,去掉平坦定义域块,即灰度均方差小于某个给定阈值 T_D 的块,因为平坦块完全可用灰度值等于值域块均值的常数块来代替,这显然减小了聚类数目和搜索空间,另外由于此时只需要记录一个编码参数 μ_R ,也提高了压缩比;对非平坦定义域块,计算它们的Saupe特征,并以此为聚类向量用KM-PDS将其聚为 M 类。

第2步 为每个值域块搜索其最优匹配块。将

灰度均方差小于某个阈值 T_R 的值域块视为平坦块,直接用灰度值等于值域块均值的常数块来逼近,这提高了编码速度,也提高了压缩比。对非平坦值域块,计算其Saupe特征,首先用PDS在 M 个类中(通过与聚类中心进行比较搜索)找到 M_0 个最近的类,然后在每个最近类返回 N_0 个最近的定义域块,最后在此 $M_0 N_0$ 个定义域块中找一个MSE最小者。

4 数值实验及结果分析

实验均使用标准的 $256 \times 256 \times 8$ 标准灰度测试图像Lena。编码性能指标有:编码时间Time(s)、解码图像质量PSNR(dB)、压缩比CR。对图像采用 4×4 均匀划分,定义域块尺寸取为 8×8 ,相邻定义域块的水平和垂直间隔都取为4, s 和 μ_R 分别用5和7比特来均匀量化,定义域块位置采用12比特量化。在此设置下,对式(1)采用全局搜索取得的编码时间、解码质量、压缩比(未经熵编码,下同)分别为12.27、29.02、5.33。

固定 $T_R = 30, T_D = 50, M_0 = N_0 = 1$,此时压缩比为7.06,其余性能指标随 M 的变化情况如图3所示。将整个编码时间分为两部分:建立定义域块库(包括聚类)和为每个值域块搜索最优匹配块所需时间,从图3(a)可看出,随着 M 的增加,前者将在 $M = 30$ 左右超过后者,这使得当其他参数固定时,不能单纯依靠增加聚类数目来进一步加速编码;另外从图3(b)看出,只搜索最近类($M_0 = 1$)及其中的最近邻($N_0 = 1$)无法取得好的解码质量。从理论上, M_0, N_0 越大,解码质量就越好,当然编码时间也将增加,这就存在一个合理选择 M_0, N_0 的问题。

图4是 $T_R = 30, T_D = 50, M = 20$,值域块比较的定义域块数目固定为 Num 时,搜索 Num 个最近类中的最近邻(用 $(Num, 1)$ 表示)和搜索最近类中的 Num 个最近邻(用 $(1, Num)$ 表示)时各性能指标的变化情况。可以看出,使用第1种方案,编码时间依 Num 增加呈线性增长,但解码质量只在 Num 从1变为2时改善较大,而使用第2种方案的编码时间随 Num 增加只有很小的变化,并在 $Num = 5$ 处取得最小值,而此后的解码质量改变也很小;综合起来看,第2种方案占优。

取经验值 $M_0 = 2, N_0 = 5$,同样令 $T_R = 30, T_D = 50$,考察编码时间和解码质量随聚类数目的变化情况见图5,由图5(b)可见,相对于全局搜索,本文方法能大幅度减少编码时间且提高了压缩比,而解码

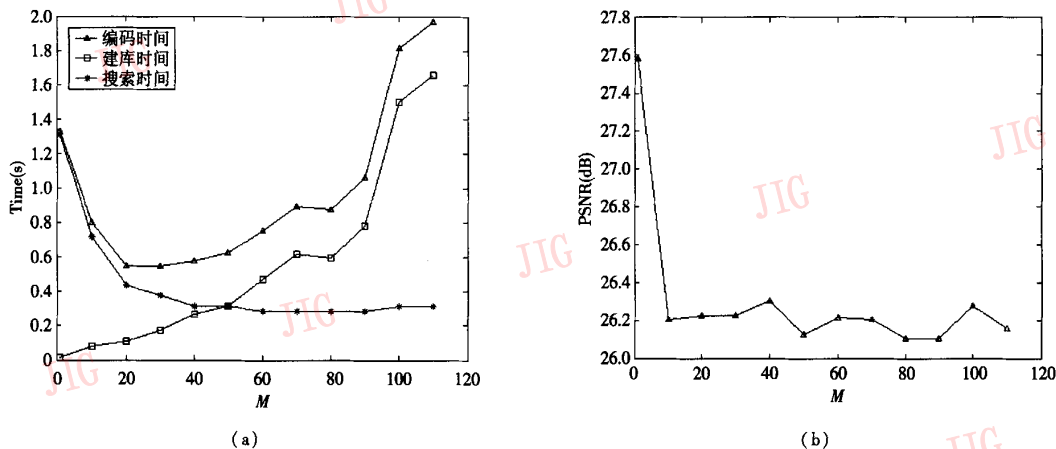


图 3 本文方法的最近邻搜索结果
Fig. 3 Results of nearest neighbor search using our method

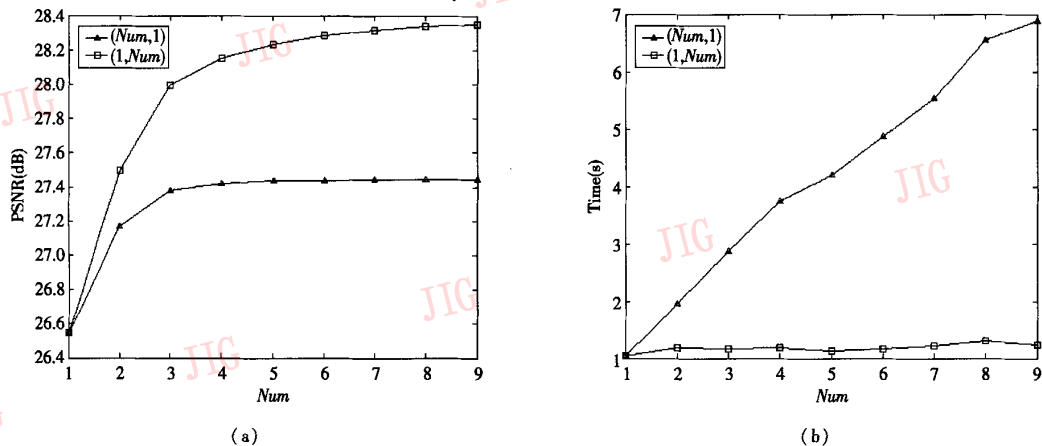


图 4 编码时间和解码质量随最近类数目和最近邻数目的变化情况
Fig. 4 Coding results under different number of nearest clusters and nearest neighbors

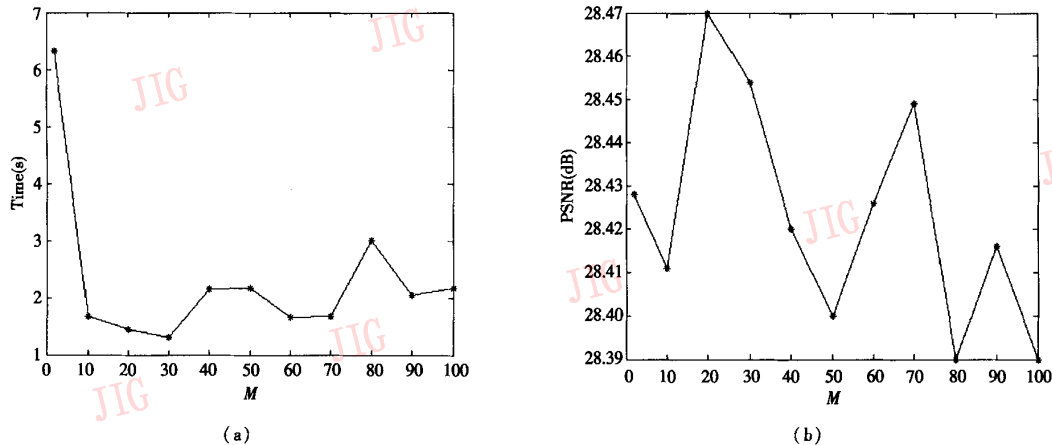
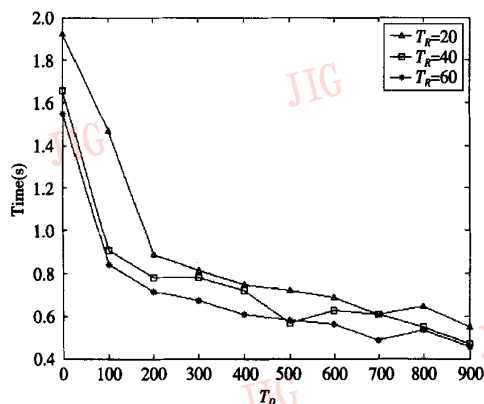


图 5 编码时间和解码质量随聚类数目的变化情况
Fig. 5 Coding results under different cluster numbers

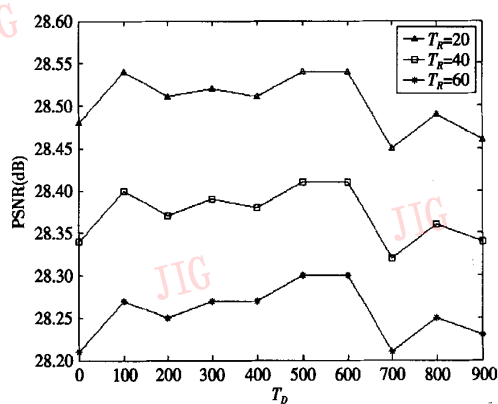
质量只是相对于全局搜索下降了 0.5dB;另外,编码时间和解码质量随 M 的增加表现出一定的波动,这是由聚类需要花费的时间并不与 M 成正比且聚类后各类中的定义域块数目不均匀所致。

最后,考察 T_R, T_D 对各项编码性能的影响。固定 $M=20, M_0=2, N_0=5$, 分别令 $T_R=20, 40, 60$ 时



(a)

编码时间和解码质量随 T_D 的变化情况如图 6 所示,此时的压缩比分别为 6.49、7.45 和 8.02。可见, T_R 的增加能较大的提高压缩比和编码速度,例如当 $T_R=60, T_D=600$ 时,速度是全局搜索法的 $12.27 \div 0.56 \approx 22$ 倍,压缩比提高了 $(8.02 - 5.33) / 5.33 \approx 50\%$,质量只下降了 $29.02 - 28.30 = 0.7\text{dB}$ 。



(b)

图6 平坦性阈值对各项编码性能的影响

Fig. 6 Coding results under different smoothness thresholds

5 结论

本文提出的快速 K 均值聚类方法,在计算量和实际时间上都优于传统 K 均值聚类方法,它适用于任何需要数据分类的场合。将它用于分形编码的加速中,大幅度地提高了分形编码的速度和压缩比,且较好地保证解码图像的质量。为便于分析本文方法采用了均匀划分,这使得它的压缩比不高,为此可以采用变尺寸划分例如四叉树等;另外,对定义域块进行聚类后,可为每个类建立某种树结构如KD-Tree或R-Tree等以进行最近邻搜索,以进一步提高编码速度。

参考文献 (References)

- 1 Fisher Y. Fractal Image Compression-Theory and Application[M]. New York, Springer-Verlag, 1994.
- 2 Hurtgen B, Stiller C. Fast hierarchical codebook search for fractal coding of still images[A]. In: Visual Communications and PACS for Medical Applications [C], Berlin, Germany, 1993, 1977: 397~408.
- 3 Mario Polvere, Michele Nappi. Speed-up in fractal image coding: comparison of methods[J]. IEEE Transactions on Image Processing, 2000, 9(6): 1002~1009.

- 4 Li Jie, Fu Ping, Liu Jin-guo. Fractal image coding based on classification and clustering[J]. Journal of Computer-Aided Design & Computer Graphics, 2002, 14(4): 348~350. [李杰,付萍,刘金国. 基于复合分类的快速分形图像压缩编码[J]. 计算机辅助设计与图形学学报, 2002, 14(4): 348~350.]
- 5 Lototskiy R V. Images fractal compression optimization by means of artificial Kohonen neural networks[J]. Journal of Automation and Information Sciences, 2003, 35(1): 50~60.
- 6 Tong C S, Pi M. Fast fractal image encoding based on adaptive search[J]. IEEE Transactions on Image Processing, 2001, 10(9): 1269~1277.
- 7 Yahagi T, Hagiwara M, Yamaguchi T. Neural Network and Fuzzy Signal Processing [M]. Beijing: Science Press, 2003: 38~39. [(日)获原将文,山口亨,古获隆嗣著. 人工神经网络与模糊信号处理[M]. 马炫译,北京:科学出版社,2003: 38~39.]
- 8 Bei C D, Grey R M. An Improvement of the minimum distortion encoding algorithm for vector quantization[J]. IEEE Transaction on Communications, 1985, 33(10): 1132~1133.
- 9 Chen S H, Pan J S. Fast search algorithm for VQ-based recognition of isolated word[J]. IEE Proceedings-I, 1989, 136(6): 391~396.
- 10 Saupe D. Fractal image compression via nearest neighbor search[A]. In: Proceedings of NATO ASI on Fractal Image Encoding and Analysis[C], Trondheim, Norway, 1995: 1~25.
- 11 Saupe D. The futility of square isometries in fractal image compression[A]. In: Proceedings of IEEE International Conference on Image Processing [C], Lausanne, Switzerland, 1996, 1: 161~164.