A fast algorithm for color image segmentation

Liju Dong[1,2,3] Philip Ogunbona[1] Wanqing Li[1] Ge Yu[3] Linan Fan[2] Gang Zheng[1]
[1] University of Wollongong, Australia [2] Shenyang University [3] Northeastern University, China
E-mail liju@uow.edu.au

Abstract

Based on K-means and a two-layer pyramid structure, a fast algorithm is proposed for color image segmentation. The algorithm employs two strategies. Firstly, a two-layer structure of a color image is established. Then, an improved K-means with integer based lookup table implementation is applied to each layer. The clustering result on the upper layer (lower resolution) is used to guide the clustering in the lower layer (higher resolution). Experiments have shown that the proposed algorithm is significantly faster than the original K-means algorithm while producing comparable segmentation results.

1. Introduction

The K-means algorithm has been widely used in the segmentation of color images for many applications, such as color blood corpuscle accounting [1], natural image segmentation [2], face detection of fruit accounting [3], detection of oil overflow [4], and biology micrograph processing [5]. For large images, however, the algorithm takes a considerably large amount of computation time. This paper proposes a fast algorithm by combining effectively a two-layer pyramid structure of an image with an improved K-means with integer look up table implementation. Experiments have shown that the new method is substantially more efficient compared with the original K-means algorithm, whereas both produced comparable segmentation results.

The rest of this paper is organized as follows. A brief description of the K-means algorithm is given in Section 2. The proposed method is presented in Section 3. The experimental results are shown in Section 4. Finally, Section 5 concludes this paper.

2. The K-Means algorithm[6]

Let $X = \{\mathbf{x}_1, \mathbf{x}_2 \cdots, \mathbf{x}_N\} \subset R^p$ be a finite data set where N is the number of data items and R^p is the p-dimensional Euclidean space. Let V_{KN} be the set of $K \times N$ matrices with K, $2 \le K < N$, being the number of clusters. A K partition of X is defined as

$$M_K = \{U \in V_{KN} \mid u_{ik} \in \{0,1\}, \forall i,k; \sum_{i=1}^{K} u_{ik} = 1, \forall k;\ 0 < \sum_{k=1}^{N} u_{ik} < N, \forall i\} \quad (1)$$

where $u_{ik} = 1$ denotes $\mathbf{x}_k$ belongs to cluster i and $u_{jk} = 0$ denotes $\mathbf{x}_k$ is not in cluster j. The objective function is defined as

$$J_K(U,V) = \sum_{i=1}^{K}\sum_{k=1}^{N} u_{ik} d_{ik}^2 \quad (2)$$

where $V = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_K\}$, $\mathbf{v}_i \in R^p, 1 \le i \le K$, denotes the set of the K cluster centers and $d_{ik} = \|\mathbf{x}_k - \mathbf{v}_i\|$ represents the distance between $\mathbf{x}_k$ and $\mathbf{v}_i$. The minimization of $J_K(U,V)$ produces an optimal K partition of X. The iterative optimization algorithm is given as follows.

1. Set the number of clusters K, $2 \le K < N$.
   Initialize $U^{(0)} \in M_K$, and $\mathbf{v}_i^{(0)} = \mathbf{0}$, $1 \le i \le K$.
2. Set initial iteration step $b = 0$.
3. Calculate K cluster centers $\mathbf{v}_i^{(b+1)}, 1 \le i \le K$ with $U^{(b)}$ and:

$$\mathbf{v}_i = \frac{\sum_{k=1}^{N} u_{ik}\mathbf{x}_k}{\sum_{k=1}^{N} u_{ik}}. \quad (3)$$

4. Update $U^{(b)}$ to $U^{(b+1)}$ by

$$u_{ik} = \begin{cases} 1, & d_{ik} = \min\limits_{1 \le j \le K}\{d_{jk}\} \\ 0, & \text{otherwise,} \end{cases} \quad 1 \le i \le K,\ 1 \le k \le N. \quad (4)$$

5. If $\|\mathbf{v}_i^{(b+1)} - \mathbf{v}_i^{(b)}\| < \varepsilon$, $\forall i$, stop; otherwise, set $b \leftarrow b+1$, and go to step 3.

For color image segmentation, we have $p = 3$

Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06)
0-7695-2616-0/06 $20.00 © 2006 IEEE

representing the three components of a color space, e.g. RGB space, in which the colors of pixels are specified. The segmentation result can be obtained from u_{ik} directly. For example, $u_{3k}=1$ denotes that the k pixel belongs to cluster 3.

3. The Proposed method

The algorithm employs two strategies. Firstly, a two-layer structure of a color image is established. Then, an improved K-means with integer based lookup table implementation is applied to each layer. A typical size of the upper layer is 1/16 of the size of its lower layer. Therefore, the clustering in the upper layer can be completed quickly, and the approximate cluster centers can be obtained. These approximate centers serve as the initial centers for the clustering in the lower layer, reducing the number of iterations of the clustering significantly.

3.1. Two-layer pyramid data structure

Fig.1 shows the two-layer pyramid data structure of a color image. The value of a pixel in the upper layer is the average value of a block of connected pixels in the lower layer. The lower layer has a higher resolution than the upper layer.

Let $x^R(i,j)$, $x^G(i,j)$, and $x^B(i,j)$ be the three RGB components of a pixel $\mathbf{x}(i,j)=(x^R(i,j),x^G(i,j),x^B(i,j))$ in the lower layer. Also let $x^{R'}(i,j)$, $x^{G'}(i,j)$, and $x^{B'}(i,j)$ be the three RGB components of a pixel in the upper layer, the size of the lower layer be $M_1\times M_2$, and the size of the upper layer be $M_1'\times M_2'$. Without loss of generality, suppose $M_1=4M_1'$ and $M_2=4M_2'$, then

$$x^{R'}(i,j)=\frac{1}{16}\sum_{m=0}^{3}\sum_{n=0}^{3}x^R(4i+m,4j+n),$$
$$0\le i\le M_1'-1, 0\le j\le M_2'-1, \quad (5)$$

$$x^{G'}(i,j)=\frac{1}{16}\sum_{m=0}^{3}\sum_{n=0}^{3}x^G(4i+m,4j+n),$$
$$0\le i\le M_1'-1, 0\le j\le M_2'-1, \quad (6)$$

$$x^{B'}(i,j)=\frac{1}{16}\sum_{m=0}^{3}\sum_{n=0}^{3}x^B(4i+m,4j+n),$$
$$0\le i\le M_1'-1, 0\le j\le M_2'-1. \quad (7)$$

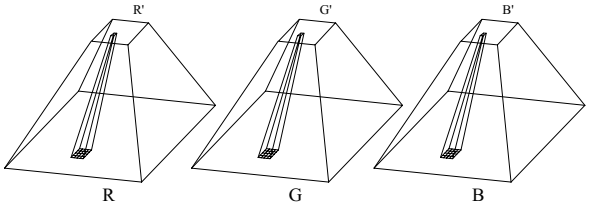


Fig. 1. Two-layer pyramid data structure.

We apply an improved version of K-means (as described below) to the upper layer. Because the number of pixels at this layer is only a small proportion of the pixels in the original images the algorithm will converge quickly. The obtained cluster centers are then used to guide the further clustering in the lower layer in an attention to reduce significantly the number of iterations.

3.2. An improved K-means with integer lookup tables

The conventional K-means algorithm described in Section 2, usually has many repetitive operations in computing the distances, d_{jk}, for the pixels with the same color or RGB values. The repetition becomes prominent for large size images, Here, we design an integer lookup table, LUT, to eliminate the repetition. LUT is three-dimensional and defined as follows:

$$LUT[i][p][x]=\text{Round}(100.0\times(x-v_{ip})^2), \quad (8)$$

where $1\le i\le K$, $1\le p\le 3$, and $0\le x\le 255$. Establishing such a lookup table takes $K\times 3\times 256$ times of computation. If $K=4$, then $K\times 3\times 256=3072$, which is much less than the number of pixels in a large image. The maximum value of LUT is $100\times(255-0)^2=6502500$, which is within the range of a 4-byte integer. In (8), the real number $(x-v_{ip})^2$ is multiplied by 100.0 and rounded, which is equivalent to using the real number v_{ip} for the computation with a precision of one decimal place.

3.3. The new algorithm

With the two-layer pyramid structure and the improved K-means, the proposed algorithm for segmenting a color image is described as follows.

1. Set the number of clusters K , $2\le K<N=M_1'\times M_2'$. Initialize $U^{(0)}\in M_K$, and $\mathbf{v}_i^{(0)}=\mathbf{0},\ 1\le i\le K$.
2. Set initial iteration step $b=0$.
3. Calculate K cluster centers $\mathbf{v}_i^{(b+1)}, 1\le i\le K$, with $U^{(b)}$ and (3).
4. Establish lookup table with $\mathbf{v}_i^{(b+1)}, 1\le i\le K$, and (8).

5. Update $U^{(b)}$ to $U^{(b+1)}$:

from $k=1$ to N

a. $D_{ik} = LUT[i][1][x_k^R] + LUT[i][2][x_k^G] + LUT[i][3][x_k^B]$ (9)

b. $u_{ik} = \begin{cases} 1, & D_{ik} = \min\limits_{1\le j\le K}\{D_{jk}\} \\ 0, & \text{otherwise,} \end{cases} \quad 1 \le i \le K$ (10)

$k \leftarrow k+1$.

6. If $\left\|\mathbf{v}_i^{(b+1)} - \mathbf{v}_i^{(b)}\right\| < \varepsilon$, $\forall i$, go to step 7;

otherwise, set $b \leftarrow b+1$, and go to step 3.

7. Let $N = M_1 \times M_2$. Denote the centers obtained from the clustering in the upper layer by $\mathbf{v}_i^{(0)}, 1 \le i \le K$. Set initial iteration step $b = 0$.

8. Establish lookup table with $\mathbf{v}_i^{(b+1)}, 1 \le i \le K$, and (8).

9. Update $U^{(b)}$ to $U^{(b+1)}$:

from $k=1$ to N

a. Calculate D_{ik} with the lookup table as in (9).

b. Update u_{ik} as in (10).

$k \leftarrow k+1$

10. Calculate K centers $\mathbf{v}_i^{(b+1)}, 1 \le i \le K$, with $U^{(b+1)}$ and (3).

11. If $\left\|\mathbf{v}_i^{(b+1)} - \mathbf{v}_i^{(b)}\right\| < \varepsilon$, $\forall i$, go to step 12; otherwise, set $b \leftarrow b+1$, and go to step 8.

12. Segment the original image by u_{ik} of the matrix U, $1 \le i \le K$, $1 \le k \le M_1 \times M_2$.

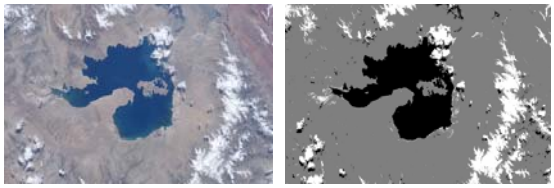
In the above algorithm, steps 1–6 are performed upon the upper layer and the rest steps are performed on the lower layer. In steps 5(a) and 9(a) the relative distances from a pixel to the cluster centers are obtained directly from the lookup tables, therefore, eliminating the repetitive operations. By relative distance it is meant that the integral part of the scaled real distance. Steps 5(b) and 9(b) are also simplified due to the use of the integers.

In the algorithm, cluster centers obtained from the upper layer clustering serve as the initial centers $\mathbf{v}_i^{(0)}, 1 \le i \le K$, in the lower layer clustering. This allows an efficient reduction in the number of iterations in the lower layer clustering where there are a large number of pixels involved. In steps 1–6, the $\mathbf{x}_k$ denotes the pixels in the upper layer, and in steps 7–12, the $\mathbf{x}_k$ denotes the pixels in the lower layer.

4. Experimental results

In the experiments, over 30 large color images obtained from two web sites [7,8] were used to compare the proposed algorithm and the conventional K-means algorithm. The algorithms are implemented with Visual C++ running on a Pentium 4 PC. The terminating condition ε was set to 0.1 for both algorithms.

Due to the limitation of space, here we only give two representative examples. Fig. 2(a) is a 2000×3000 color image, in which the white, blue and other regions represent cloud, water, and mountain, respectively. Fig. 2(b) gives the identical segmentation results obtained by the two algorithms.



(a) (b)

Fig. 2. (a) A color remote sensing image. (b) Three-cluster segmentation result by the K-means algorithm or the proposed algorithm. The three clusters are represented with three gray levels 0, 127, and 255.

Table 1 summarizes the results of the two algorithms on the clustering of the image shown in Fig. 2(a), where $\mathbf{v}$, b, t, and χ denote the cluster centers, number of iterations, time used (in second), and rate of the same classified pixels, respectively. Here χ is defined by

$$\chi = \frac{M_1 \times M_2 - M_3}{M_1 \times M_2} \times 100\% \quad (11)$$

Table 1. Summary of the results on the image shown in Fig. 2(a)

	K-means	Ours
$\mathbf{v}$	(151.05,148.34,157.58) (44.10,78.78,122.91) (211.99,216.86,225.72)	Upper layer: (150.63,147.96,157.26) (42.69,78.06,122.69) (210.47,215.46,224.35) Lower layer: (151.04,148.33,157.57) (44.09,78.78,122.91) (211.95,216.81,225.67)
b	9	Upper layer: 10 Lower layer: 3
t	58s	5s
χ	100%	

where $M_1 \times M_2$ is the number of pixels of the original image, and M_3 is the number of the pixels classified differently by the two algorithms.

Fig. 3 shows another example. Fig. 3(a) is the original color image of size 4000×2600 . The segmentations obtained by the two algorithms are given in Figs. 3(b) and (c). Table 2 summarizes the results. In this example, although χ is not 100% (but very close to 100%), there is no perceptual difference between the two segmented images shown in Figs. 3(b) and (c).

From all our experiments, the new algorithm is significantly faster than the conventional K-means algorithm, and the two algorithms produce almost identical results.

5. Conclusion

The K-means clustering algorithm has been widely used in the segmentation of color images. There is an increasing desire for a fast version of the K-means algorithm for newly developed imaging devices with high resolution. In this paper, a fast algorithm is proposed by effectively combining a two-layer pyramid structure and an improved implementation of the K-means algorithm. The proposed algorithm performs significantly faster without noticeable degradation of the results in comparison to conventional the K-means algorithm.

6. References

[1] N. Sinha and A.G. Ramakrishnan, "Automation of differential blood count", Proc. of International Conference on Convergent Technologies for the Asia Pacific Region, 2003, pp. 547-551.

[2] J. Xu and P.F.Shi, "Natural color image segmentation", Proc. of International Conference on Image Processing, 2003, pp.14-17.

[3] A.R. Weeks, A. Gallagher, and J. Eriksson, "Detection of oranges from a color image of an orange tree", Proc. of International Society for Optical Engineering, 1999, pp. 3808: 346-357.

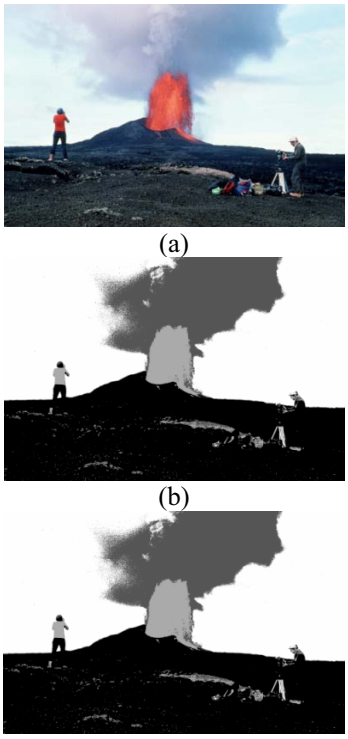
[4] O. Demirors, E. Demirors, Y. Ozturk, and H. Abut, "Application of image segmentation and classification", Proc. of International Symposium on Computer and Information Sciences, 1989, pp. 383-388.

[5] P. Lescure, Y. V. Meas, H. Dupoisot, and G. Stamon, "Color segmentation of biological microscopic images", Proc. of International Society for Optical Engineering, 1999, pp. 3647:182-193.

[6] J. C. Bezdek, Pattern recognition with fuzzy objective function algorithms. Plenum, 1981.

[7] Robotics Institute of Carnegie Mellon University. Computer vision test images. http://www-2.cs.cmu.edu/~cil/v-images.html.

[8] SolorView Company. http://www.solarviews/cap/.



(a) (b) (c)

Fig. 3. (a) A volcano eruption image. (b) Four-cluster segmentation result by the K-means algorithm. (c) Four-cluster segmentation result by the proposed algorithm. The four clusters are represented with gray levels 0, 85, 170, and 255.

Table 2. Summary of the results on the image shown in Fig. 3(a).

	K-means	Ours
v	(149.77,171.85,200.38) (223.30,238.90,241.97) (183.21,95.63,94.56) (33.50,45.91,56.75)	Upper layer: (150.12,172.14,200.78) (222.82,238.43,241.41) (183.83,95.98,94.64) (33.55,46.04,56.87) Lower layer: (149.75,171.82,200.35) (223.29,238.90,241.97) (183.48,95.57,94.47) (33.51,45.92,56.75)
b	16	Upper layer: 11; lower layer: 1
t	144s	7s
χ	99.99%	