객체 분할과 HAQ 알고리즘을 이용한 내용 기반 영상 검색 특징 추출

김대일, 홍종선, 장혜경, 김영호, 강대성
동아대학교 전자공학과
e-mail : daeilu@empal.com

Feature Extraction Of Content-based image retrieval Using object Segmentation and HAQ algorithm

Daeil Kim, Jong-Sun Hong, Hye-Kyoung Jang
, Young Ho Kim, Dae-Seong Kang
Dept. of Electronic Eng., Dong-A University

Abstract

Compared with other features of the image, color features are less sensitive to noise and background complication. Besides, this adding to object segmentation has more accuracy of image retrieval. This paper presents object segmentation and HAQ(Histogram Analysis and Quantization) algorithm approach to extract features(the object information and the characteristic colors) of an image. The empirical results shows that this method presents exactly spatial and color information of an image as image retrieval's feature.

I. 서론

멀티미디어 데이터는 문서, 음성, 영상, 비디오 등으로 구성되는데, 그 중 영상이나 비디오 등 영상에 관련된 데이터가 많은 용량을 차지하며 중요한 정보를 가지고 있다. 다양한 종류의 영상자료를 사용자의 입장에서 효율적으로 검색할 필요성이 증대하고, 대용량 멀티미디어 정보를 효과적으로 검색하기 위해서는 효율적인 색인 및 관리가 필수적이다. 따라서 이를 효율적으로 검색하기 위한 특징 추출 기법이 요구되며, 이는 영상관련 데이터 뿐 아니라 전체 멀티미디어 데이터의 관리 및 활용에 직접 관련된다. 한편, 기존의 gray level에서의 검색은 한계에 부딪혔는데, gray level에서 추출할 수 있는 feature가 검색의 정확성을 높이는데 한계가 있기 때문이다. Viewing point들의 다변화, 다변하는 image resolution, lighting condition들의 변화와 같은 많은 변화 요소들을 모두 반영하기 위해서는 color 영역이 필요하다. 또, color는 기하학적 변화들을 감안하여 계산할 수 있어서 gray보다 훨씬 정확한 검색을 할 수 있다. 하지만 color만을 이용한 영상 검색은 높은 정확도와 빠른 속도로 영상의 유사도를 산출하고, 회전과 이동에 강건하다는 장점이 있음에도, 복잡하고 유사한 color영상들을 분간하기 힘들고, 공간 정보를 반영하지 못해 홀로 쓰기 힘들다. 그러므로 color feature와 함께 객체 정보를 같이 써서 color feature 만의 한계를 극복한다. 객체 분할 기법은 canny edge detecter를 사용하여, edge를 검출한 후 객체에 해당하는 영역을 분할한다. 여기서, color feature 추출 기법들을 살펴보면 다음과 같다. Swain과 Ballard가 제안한 the histogram intersection method[1]은 각 query image에 대한 3-D histogram bin을 만들기 위해 color histogram을 사용한다. 이 3-D bin은 histogram match value를 사용하여 database image들의 3-D bin과 매치시킨다. 이 기법은 많은 상황에서 유용함에도 불구하고, illumination의 변화에 심각하게 degrade된다. Funt와 Finlayson이 제안한 color constant color indexing[2]는 color ratio의 histogram을 match함으로 객체를 인식하는 기법이다. 고정된 휘도에서는 Swain의 기법보다 성능이 떨어지지만, 변화하는 illumination에서 Swain의 기법보다 낫다. Slater와 Healey가 제안한 local color invariant[3]는 spectral reflectance의 distribution에 관한 정보를 캡쳐하여, 3-D 객체를 인식한다. 이 알고리즘은 객체의 configuration과 scene illumination에 독립적이지만, 계산량이 높은 단점이 있다. Mehtre가 제안한 distance method and reference color table method[4]는 3개의 color component 각각의 1-D histogram의 mean value를 feature로 사용한다. 본 논문에서 제안한 HAQ 알고리즘은 전체 color를 정확히 나타내고, feature의 비교가 수월해서 객체 분할 후 적용하면 뛰어난 검색 성능을 기대할 수 있다.

본 논문에서는 image의 객체 분할을 통해 공간 정보를 구한 후, characteristic color를 사용하여 질의 image에 부합하는 비디오를 정확하고 경제적으로 검색하고자 한다.

II. 객체 분할 알고리즘

Step 1. image의 gray를 입력받는다.

Step 2. gaussian detector로 edge를 검출한다.(edge image I_e 생성)

$$\nabla^2(x,y)=\frac{\partial g(x,y)}{\partial x^2}+\frac{\partial^2 g(x,y)}{\partial y^2}$$

$$(\nabla^2 G(x,y,\sigma))I(x,y)$$

$$LoG(x,y)=-\frac{1}{\pi\sigma^4}\left[1-\frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Step 3. 검출된 edge의 선을 모두 연결하여, 폐곡선을 생성한다.

$$t_{i,j}=\sum_{k=0}^{5} m_{(i-2)(j+k-2)} \qquad for\ m_{ij}$$

$$t_{(i-1)(j-1)}=m_{ij} \qquad for\ k<2$$

$$t_{(i-1)(j)}=m_{ij} \qquad for\ k=2$$

$$t_{(i-1)(j+1)}=m_{ij} \qquad for\ k>2$$

Step 4. 폐곡선 안을 통합하고, 객체 정보로 추출한다.(object image I_o 생성)

$$O(x,y_k)=the\ \ same\ \ object$$

$$for\ Start_{edge}<k<End_{edge}$$

Step 5. 원 image(I)에 객체 정보를 mask한다.(masking image I_m)

$$I_m(n,m)=I_o(n,m)$$

여기서, (n,m)은 객체 정보 픽셀이다.

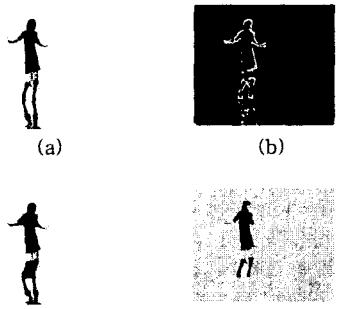
Step 6. 객체의 위치와 크기를 feature로 저장한다.

$$m_o(x,y)=\left(\frac{1}{n_{ox}}\sum_{k=0}^{n_{ox}} O_{xk},\ \frac{1}{n_{oy}}\sum_{k=0}^{n_{oy}} O_{yk}\right)$$

$$S_o=\frac{n_o}{n_u}$$

여기서, $m_o(x,y)$는 객체 영역의 평균 좌표이고, S_o는 객체의 크기이다.

그림 1은 객체 분할 과정을 도시한 것이다.



(a) (b)

(c) (d)

그림 1. 객체 분할 image
(a)original image (b)edge image I_e
(c)object image I_o (d)masking image I_m

III. HAQ 알고리즘

사람의 지각에서는 인식하기 위해 유사한 color들을 서로 grouping한다. 여기에 착안하여, image의 유사한 color들을 grouping하는 것과 검색을 위한 feature로써 characteristic color들을 추출하는 기법을 소개한다. 제안하는 color feature 추출 기법은 image에 대한 color palette[5][6]의 디자인과 유사하다. 이것은 원 color image에서 제한된 color image로의 quantization이 필요하다. HAQ 기법은 image의 color component의 상호 관계를 활용하기 위해 image의 color space를 분할한다. HAQ algorithm은 크게 두 가지로 구성되는데, the bit allocation[7]과 histogram analysis method가 바로 그것이다. image의 전체 좌표 space를 I로 표시하면, I의 color value를 $I=(I_1, I_2, I_3)$로 표시한다. 여기서 I_i는 color component value이다. bit allocation의 목적은 image의 각 color component에 주어진 bit 묶을 자동적으로 할당하기 위함이다. 즉 bit B의 고정된 수를 color component (I_1, I_2, I_3) 사이에서 나누어 놓는다. 일반적으로, 변화가 심한 color component에 더 많은 bit들을, 변화가 적은 쪽에 더 작은 bit들을 할당한다. 만약 데이터가 특정 color component에 퍼져있으면, 그 데이터의 component는 다른 것보다 더 중요하기 때문에 더 조밀하게 grouping한다. 즉, 중요한 component는 quantization error를 줄이기 위해 더 많은 bit들이 주어져야만 하는 것이다. histogram analysis method는 quantization threshold level을 결정짓기 위한 기법이다. 각 color component에 대한 histogram을 분석해 보면, count가 크게 변화하는 구간이 존재하기 마련이다. 이 구간을 grouping하기 위해 threshold level이 필요하고, histogram analysis method로 이 level을 결정한다.

Step 1. 입력 영상들을 RGB space로 표현한다.

Step 2. 각 color component(I_i)에 주어진 bits B의 고정된 수를 할당한다.

$I_i=B$, i=1,2,3

Step 3. I_i의 bit allocation 근사 solution을 구한다.

$$B_i=B+\frac{1}{2}\log_2\frac{\sigma_i^2}{\left[\prod_{j=1}^{3}\sigma_j^2\right]^{\frac{1}{3}}}\ ,\quad j=1,2,3,\ \ldots$$

여기서, $\sigma^2=\frac{1}{n}\sum_{i=0}^{n}(x_i-m)^2$

Step 4. Bit allocation procedure로 I_i의 histogram analysis procedure 반복 회수를 결정한다.

$r_i=2^{B_i}-1$, i=1,2,3

Step 5. histogram I_i을 구한다.

$$H_{ij}=\frac{1}{n}\sum_{k=0}^{L-1} n_{ik}(z_{ik})^j\ ,\quad j=1,2,3,\ \ldots.$$

여기서, n은 총 입력 픽셀 수, L은 histogram I_i의 총 분포 level, n_{ik}은 histogram I_i 내 z_{ik} 값을 가지는 픽셀 수이다.

Step 6. 다음 과정을 r_i번 반복한다.

I_i의 threshold level $t_i(m_{r_i})$을 찾는다.

$t_i(m_{r_i})$=MIN(n_{ik})

for $S_i \le n_{ik} \le E_i$, k=0,1,2, ... ,L-1

여기서, S_i는 histogram I_i내의 첫 maximum n_{ik}이고, E_i는 histogram I_i내의 마지막 maximum n_{ik}이다.

Step 7. Step 6에서 구한 threshold level $t_i(m_{r_i})$에 따른 각 color point들의 중심을 color palette의 특징 color($C_{t_i(m_{r_i})}$)로 지정한다.

$$C_{t_i(m_{r_i})} = \frac{\sum_{k=0}^{t_i(m_{r_i})} n_{ik} z_{ik}}{\sum_{k=0}^{t_i(m_{r_i})} n_{ik}} \quad , \quad i=1,2,3$$

그림 2는 각 color component의 histogram에서 threshold level을 도시한 그림이다.

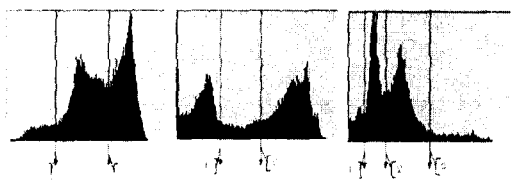
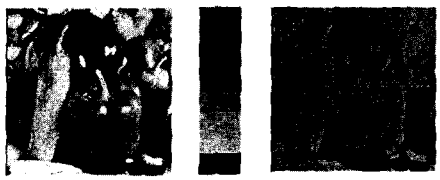


그림 2. 각 color component의 threshold level

그림 3은 pepper 영상에 HAQ algorithm을 적용하여 characteristic color들로 이루어진 color palette와 복원된 영상을 도시한 그림이다.



(a) (b) (c)

그림 3. HAQ 알고리즘을 이용하여 추출한 color feature
(a)original image (b)color palette
(c)reconstructed image

Ⅳ. 검색 알고리즘

Step 1. 질의 image의 객체 위치 feature(F_{ql})를 입력받는다.

Step 2. F_{ql}의 객체 위치 threshold(T_l) 내 모든 image를 DB에서 검색한다.

$I(F_{DBl})$, $\quad -T_l < F_{DBl} < T_l$

여기서, $I(F_{DBl})$ 는 검색된 image이고, F_{DBl}은 DB내의 객체 위치 feature이다. T_l은 실험값이다.

Step 3. 질의 image의 객체 비율 feature(F_{qr})를 입력받는다.

Step 4. F_{qr}의 객체 비율 threshold(T_r) 내 모든 image를 $I(F_{DBl})$에서 검색한다.

$I(F_{DBr})$, $\quad -T_r < F_{DBr} < T_r$

여기서, $I(F_{DBr})$는 검색된 image이고, F_{DBr}은 $I(F_{DBl})$내의 객체 비율 feature이다. T_r은 실험값이다.

Step 5. $I(F_{DBl})$내의 color feature를 histogram내 비중이 높은 color부터 차례로 가중치(W_n)를 두어 비교한다.

Step 5-1) 다음을 n번 반복하여 W_n을 결정한다.

$W_n = B_c - n$,

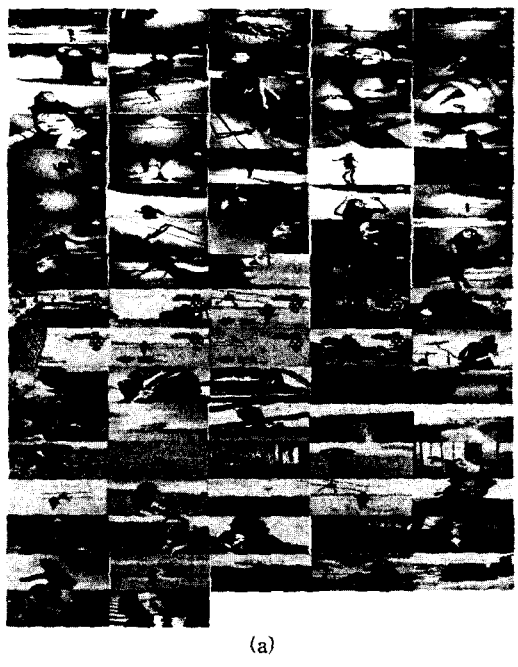
B_c = # *of color palette's bits*

Step 5-2) W_n가 가장 높은 순서대로 각 color threshold(T_c)내의 image를 선택한다.

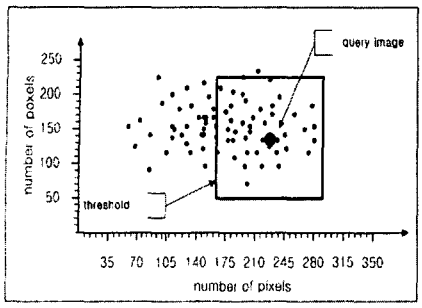
Step 6. 가장 근접한 color palette를 가진 image를 3개 검색하여 출력한다.

Ⅴ. 실험 결과

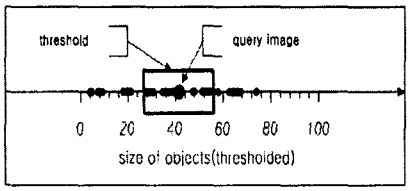
본 논문은 2개의 뮤직비디오(실험영상 1, 2)에서 총 82개의 key frame으로 database를 구성하고, 총 100개의 key frame이 아닌 임의의 frame을 query image로 지정하였다. 그림 4는 database의 key frame과 객체 정보의 위치, 크기를 검색하는 과정을 도시한 것이다.



(a)



(b)



(c)

그림 4. database의 key frame들과 객체 정보의 위치, 크기 검색

(a) database의 key frame들
(b) 객체 정보의 위치 검색 (c) 객체 정보의 크기 검색

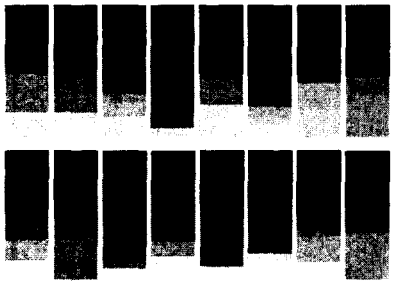


그림 5. 검색된 image들의 대표 color

그림 5는 객체 정보에서 검색된 image들의 대표 color를 나타낸다. 본 실험에서는 대표 color가 가장 유사한 순서대로 3순위까지 검색하였다. 이렇게 하여 검색된 300개의 image들에 대해, 질의 image와 동일한 shot 내의 key frame을 1순위에서 찾은 경우가 81%, 2순위 9%, 3순위 7%, 총 97%의 검색률을 보였다. 3%의 검색 결과는 질의 image와 유사한 다른 key frame을 검색하였다. 위 실험에서 알 수 있듯이 실험적인 결과를 바탕으로 검색의 threshold 값들을 조절하면, 보다 정확한 검색을 할 수 있다.

VI. 결론 및 향후 연구 방향

본 논문에서는 image의 객체 분할 통하여, 검색에 필요한 공간 정보를 파악하고, HAQ 알고리즘을 이용하여 대표 color를 추출함으로써 내용 기반 검색에 필요한 정확한 영상 특징을 추출하였다. 앞으로 더욱 정확한 검색을 위해 객체의 형태 정보를 moment method를 이용하여 파악한 후, 특징을 추출할 것이다.

참고문헌

[1] M. J. Swain and D. H. Ballar, "Color index", Int. J. Computer Vision, vol. 7, no. 1, pp. 11-32, July 1991.

[2] B. V. Funt and G. D. Finlayson, "Color constant color indexing", IEEE Trans. Pattern Anal. Machine Intell., vol. 17, pp. 522-529, May 1995.

[3] D. Slater and G. Healey, "The illumination-invariant recognition of 3D objects using local color invariants", IEEE Trans. Pattern Anal. Machine Intell., vol. 18, pp.206-210, Feb 1996.

[4] B. M. Mehtre, M. S. Kankanhalli, A. D. Narasimhalu, and G. C. Man, "Color matching for image retrieval", Pattern Recognition Lett., vol. 16, pp. 325-331, Mar 1995.

[5] P. Heckbert, "Color image quantization for frame buffer display", Comput. Graph., vol 16, no. 3, pp. 297-307, July 1982.

[6] M. T. Orchard and C. A. Bouman, "Color quantization of images", IEEE Trans. Signal Processing, vol. 5, pp. 124-139, 1995.

[7] A. Segall, "Bit allocation and encoding for vector sources", IEEE Trans. Inform. Theory, vol. IT-22, pp.162-169, Mar 1976.