Journal of Network Intelligence ©2018 ISSN 2414-8105(Online)
Taiwan Ubiquitous Information Volume **3**, Number **3**, August **2018**

An Adaptive Content-Based Image Retrieval Method Exploiting an Affine Invariant Region Based on a VQ-applied Quadtree Robust to Geometric Distortions

Chin-Feng Lee*, Yi-Jia Wang

Department of Information Management, Chaoyang University of Technology
Taichung 41349, Taiwan
*Corresponding author: lcf@cyut.edu.tw

Shu-Chuan Chu, John F. Roddick

College of Science and Engineering, Flinders University
5Sturt Rd, Bedford Park SA 5042, South Australia
jan.chu@flinders.edu.au, john.roddick@flinders.edu.au

Received March 2018; revised July 2018

Abstract. *The content-based image retrieval (CBIR) is the most acceptable method often used in an image retrieval system because it can manage image database efficiently and effectively. The CBIR methods usually retrieve the images by image features. In this paper, we exploit a region called affine invariant region (AIR) as an image feature to help effectively retrieve the images even when the images have been attacked or processed. Moreover, we use vector quantization (VQ) to reduce the comparison of image features for improving the retrieval efficiency. The experimental results show that the method has a higher recall rate, lower retrieval time, and promising accuracy.*

Keywords: Content-based image retrieval (CBIR), Vector Quantization, Quadtrees, Image features, Canny edge detection.

1. **Introduction.** Social networking is one of the most popular online activities with high user engagement rates. The use of social media sites such as Instagram, Flickr and Facebook is increasing with millions of people sharing their photos. The explosive growth of visual data and the omnipresent accessibility will give a much-needed boost to the image search and retrieval.

Content-based image retrieval (CBIR), which makes use of the representation of visual contents to identify relevant images, has received a steady increased attention in recent two decades. CBIR has become a popular and efficient means for information capturing and sharing among people, so numerous techniques have been developed for content-based image retrieval in the last decade [1-17].

The CBIR systems retrieve the images by image features such as colors, textures, sharps, objects, and so on; moreover, among them, the color feature is most intuitive and easily extracted [7, 8, 11, 17]. However, the color feature used to retrieve images does not have a high recall rate. In 2006, Teng and Lu [2] exploited the vector quantization (VQ) technique [18] to classify images and then used the codeword close to block pixels to represent the original image block, increasing retrieval speeds. Although VQ technology

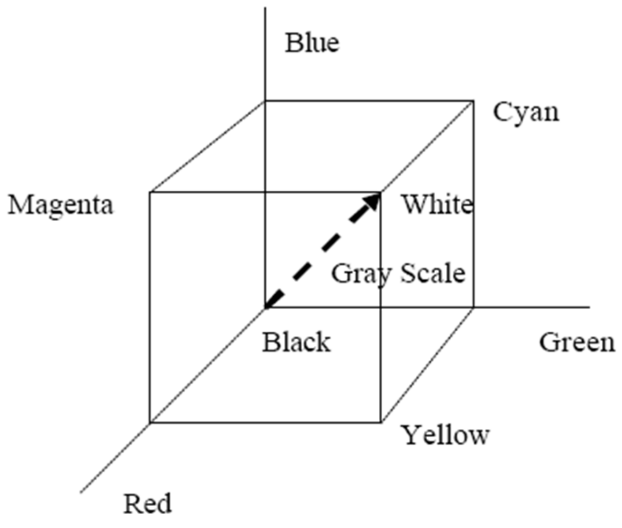


FIGURE 1. Color models transformed from the RGB model

helps speed up image retrieval, the preprocessing cost will also be high. Based on above-mentioned, we propose a feature-based oriented CBIR method. This method is not only efficient and effective, but also can quite pretect image processes.

In this paper, there are three main goals to achieve. First, we combine a VQ technology [18, 19] and 4-ary trees [20, 21, 22] to efficiently and effectively retrieve images. Second, we use an Affine Invariant Region (AIR) [23, 24] and the edge detection [25, 26] to figure out the edge features of an AIR image; afterward, we build these edge features on a Quadtree so that the edge features can be efficiently and effectively rummaged. Third, it is to find out the high-matched images which are most similar to query images. The organization of the paper is as follows. Section 2 will introduce the literature review. Section 3 will describe the proposed method and how to retrieve the images. Section 4 will show the experimental results. The conclusions are remarked in the last section.

2. **Literature Review.**

This section will introduce some methods about our research. These methods include characteristics such as color features, image invariants, tree structures, and edge detections.

2.1. **Color features.**

The color feature is one of most widely used visual features. There are several different kinds of color models, such as gray scale, RGB, CMYK, Lab, etc, where the RGB (Red, Green, Blue) color model is most known and most used every day. It defines a color space in terms of three components as shown in Figure 1 which presents the schematic of the RGB color cube. Points along the main diagonal have gray values from black at the origin (0, 0, 0) to white at point (1, 1, 1). The different colors in the model are points on or inside the cube, and are defined by vectors extending from the origin.

2.2. **Vector quantization.**

Vector Quantization (VQ), one of most popular image lossy compression methods, was proposed by Gray et al. in 1984 [18, 19]. VQ technique initially divides several $W \times H$ grayscale images into non-overlapping and equal-sized small image blocks, and each block is an -dimensional vector of pixels equal to $q \times q$. VQ partitions a large image block set $B = \{b1,\ b2,\ ...,\ b\gamma\}$ into non-overlapping clusters $CB = \{c_1, c_2, ..., c_N\}$, where $<< \gamma$. The partition process is called a training process which assigns an image block $b \in B$ to a dedicated cluster where $i = \arg\min_{1 \le t \le N} d(b, cw_t)$,

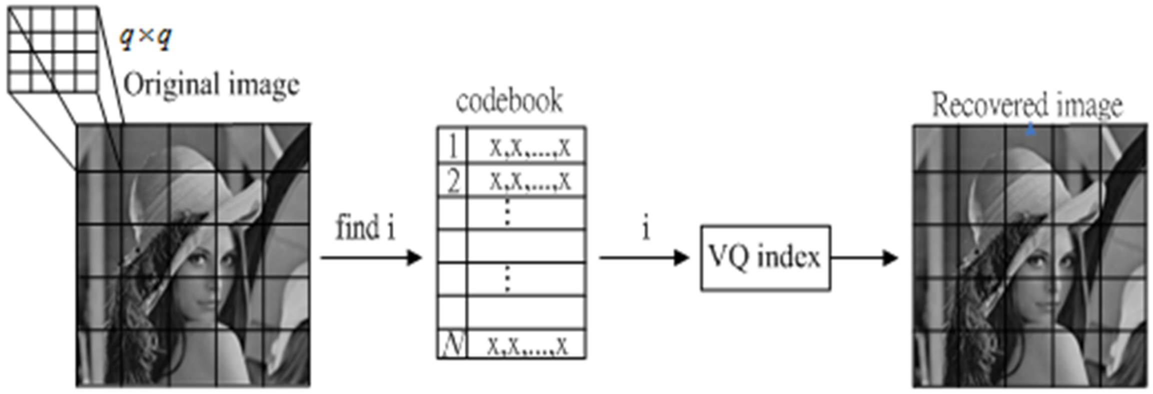


FIGURE 2. VQ encoding and decoding flowchart

$d(b,cw) = \sum_{j=1}^{\ell}(b_j - cw_j)^2$ denote a squared Euclidean distance measured between two vectors b and cw; b_j and cw_j are the-th component of the vectors.

For encoding an image, the sender seeks the closest codeword cw_i through the given codebook CB for the given input vector. For each original image block X , the index i of the selected codeword is compressed and then transmitted; the receiver, after the codeword is decompressed, uses the index i to find the mapping codeword in the codebook to retrieve the closest codeword for recovering that image block. The compression rate of VQ technique is equal to $1/\ell$, thereby reducing the transmission bandwidth. Figure 2 illustrates the image compression based on the VQ coding technique, where each codeword is an ℓ -dimensional vector (ℓ =16).

The advantages of using an index table can reduce the comparison frequency with loss of perceptibility. Furthermore, this method can preserve the positions of objects in the image and substantially increase the accuracy rate.

2.3. **Affine Invariant Region (AIR).**

According to the affine geometry [23], Hung and He [24] proposed a robust watermarking method based on an affine invariant region (AIR for short). The idea is to find a region that will not be changed after the attacks or image-processing, so the information can be embedded in it and will not be destroyed. Therefore, an AIR is an unchangeable characteristic for a corresponding image and can be employed to help image retrieval. The detailed description of this method is as follows:

Step 1: Apply Gaussian Kernel Filtering to an input image.
Gaussian Kernel Filtering is the low-pass filter which can blur the image to reduce noises. The method can remove unimportant information from the image, and reserve the important one. The operation is shown in Eq. 1.

$$g(\sigma) = \frac{1}{2\pi\sigma^2} exp^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

where σ means standard deviation. Another characteristic of Gaussian Kernel Filtering is: no matter how you rotate the images, the result will be symmetry. Therefore, this filter can help us get the feature points in the geometric rotation.

Step 2: Obtain local-based feature points
After applying the Gaussian Kernel Filtering on the image, the feature points help us get the unchanged region. First of all is to calculate the local maximum to get the robust

feature points by Eq. 2

$$MF(x,y) = \max_{x^t,y^t \in \left(N_{b^2-1}(L(x,y,\sigma)) \cup L(x,y,\sigma)\right)} \{L(x^t,y^t,\sigma)\} \tag{2}$$

where the $N_{b^2-1}L(x,y,\sigma)$ denotes b^2-1 pixels which are the area from the center of $L(x,y,\sigma)$. The feature point set is represented by P in Eq. 3.

$$P = \{(x,y)\,|MF\,(x,y) = L(x,y,\sigma)\} \tag{3}$$

Although this method will get a number of feature points, the images may incur geometric or compression attacks so that the feature points will be changed. Therefore, more robust feature points P_A and P_B in the set P can be obtained as follows: P_A and P_B are the most light and dark feature points of set P. It means that calculating the sum of pixel values in every $m \times m$ block with the central part of P_i will get the maximum and minimum. Compared with other feature points, these two points P_A and P_B by Eq. 4 and Eq. 5, respectively, are uneasy to be changed when incurring attacks.

$$P_A = \max\{sum\,(N_{m\times m}\,(P_i))\} \tag{4}$$

$$P_B = \min\{sum\,(N_{m\times m}\,(P_i))\} \tag{5}$$

where $N_{m\times m}\,(P_i)$ represents that feature point $P_i$is the center of block $m \times m$.

Step 3: Center of mass (CI)

The function of $M \times N$ image is $f(x,y)$, and the $(p+q)$ geometric moment is expressed in Eq. 6.

$$m_{pq} = \sum_{x=1}^{M}\sum_{y=1}^{N} x^p y^q f(x,y) \tag{6}$$

, where p, q $\in \{1,2,..,n\}$. The central moment is defined by Eq. 7.

$$\begin{aligned} \mu_{pq} &= \sum_{x=1}^{M}\sum_{y=1}^{N} (x-\overline{x})^p (y-\overline{y})^q f(x,y), \\ \overline{x} &= \frac{m_{10}}{m_{00}}, \overline{y} = \frac{m_{01}}{m_{00}} \end{aligned} \tag{7}$$

where $(\overline{x},\overline{y})$ denotes the center of mass of $f(x,y)$ and can also be called the center of mass(CI), which has the characteristic of unchangeableness like a geometry.

Step 4: Figure out the AIR image

By connecting CI, P_A, and P_B, and extending to the edge, we will segment the image into four regions shown in Figure 3(a), and then find out the CI in each of the four regions in Figure 3(b). Lastly, the CI of four regions are connected so that an AIR image can be obtained as shown in Figure 3(c)

Step 5: Image Normalization:

We divide the image in Figure 3(c) into two triangles shown in Figure 4(a), and then we use Eq. 8 and Eq. 9 to affine the images shown in Figure 4(b). Finally, we combine two triangles into a normalized square image shown in Figure 5.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \left(\begin{bmatrix} a\ b \\ c\ d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \right) + \begin{bmatrix} e \\ f \end{bmatrix} \tag{8}$$

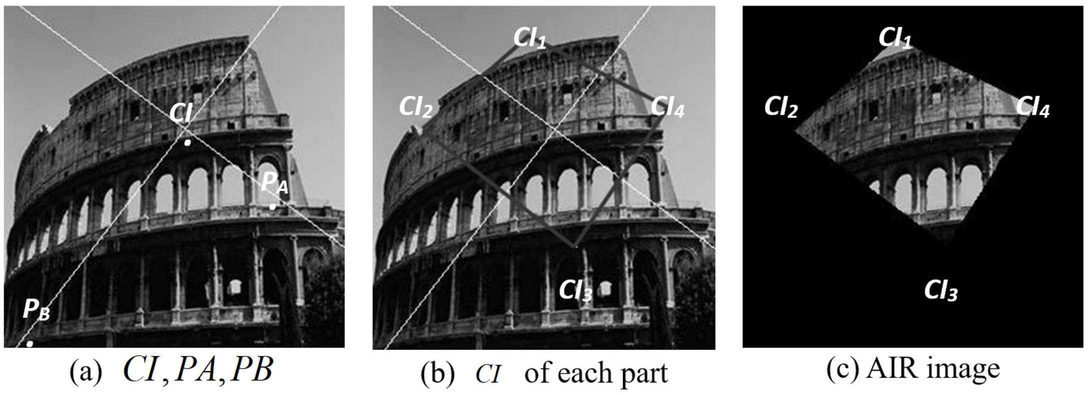


FIGURE 3. Affine Invariant Region transformation

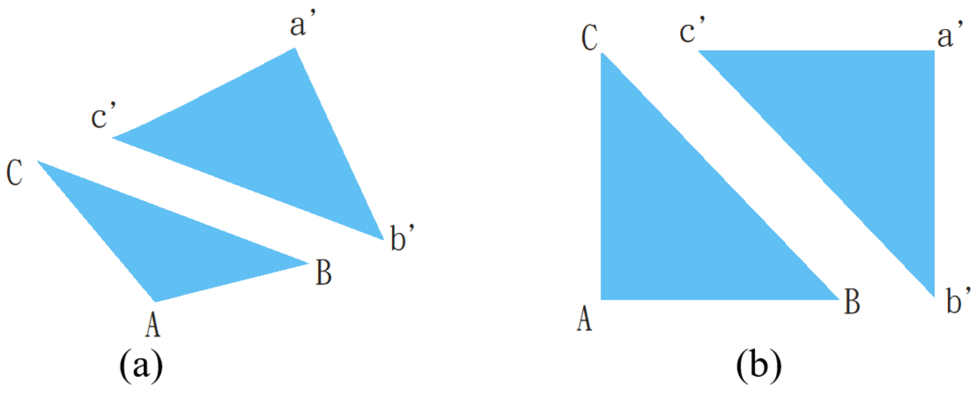


FIGURE 4. Image Normalization between triangles from (a) to (b)

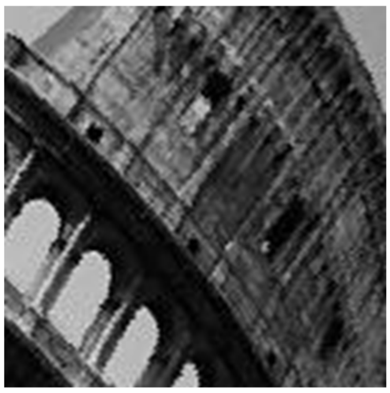
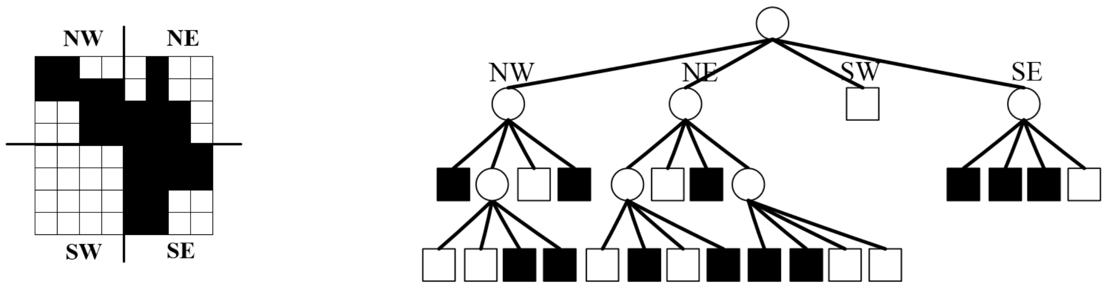


FIGURE 5. Normalized AIR image

$$\begin{cases} x' = ax + by + e \\ y' = cx + dy + f \end{cases} \tag{9}$$

2.4. **Quadtree.** Quadtree [20, 21, 22] is an image process technique, and is proposed to increase recall rates. The process of Quadtree is dividing the image to four parts: NW (northwest), NE (northeast), SW (southwest) and SE (southeast), until the pixel values of the block have the same value. Quadtree is usually used to draw a $2^n \times 2^n$ binary image. The height of Quadtree is n, and the root means the whole image. Figure 6(a) is a block *(B)* of image I, and Figure 6(b) is the Quadtree corresponding to the block B.



(a) Image block (B) of size 8×8

(b) Transform the block B into a corresponding Quadtree

FIGURE 6. Image block and Quadtree mapping

w1	**w2**	**w3**
w4	**w5**	**w6**
w7	**w8**	**w9**

-1	**-2**	**-1**
0	**0**	**0**
1	**2**	**1**

-1	**0**	**1**
-2	**0**	**2**
-1	**0**	**1**

FIGURE 7. Sobel mask

The advantage of this method is easy to differentiate values in each node and the spatial locations of objects in the images. Moreover, it not only can be applied on binary images, but also can be used on color images.

2.5. **Edge detections.** In CBIR, an edge feature is usually used to retrieve images. Two types of commonly used edge detection methods are introduced as follows:

2.5.1. *Sobel edge detection.* Sobel edge detection [25] is used in image processing and computer vision, particularly within edge detection algorithms where an image is created with emphasizing edges. The advantage of this method is simple and quick. It uses Eq. 10 and the mask in Figure 7 to find out the edges. Figures 8 (a) and (b) are the images before/after using Sobel edge detection, respectively.

$$\begin{aligned} E_x &= (w3 + 2w6 + w9) - (w1 + 2w4 + w7) \\ E_y &= (w7 + 2w8 + w9) - (w1 + 2w2 + w3) \end{aligned} \tag{10}$$

So, $E = \sqrt{Ex^2 + Ey^2}$.

Sobel edge detection is a simple detective method, which can figure out most edges in the image. However, it is susceptible to noises and unable to accurately detect step edges interfered by noises and ramp edges.

2.5.2. *Canny edge detection.* Canny edge detection [26] was proposed by John Canny. The Canny edge detector first uses a Gaussian filter to smooth the image, eliminate noises, and then find the image gradient to highlight regions with high spatial derivatives. Canny edge detection in recent years is the most accurate method in accordance with the users needs to adjust the different parameters. Figure 9 is an image applied by the Canny edge detection.

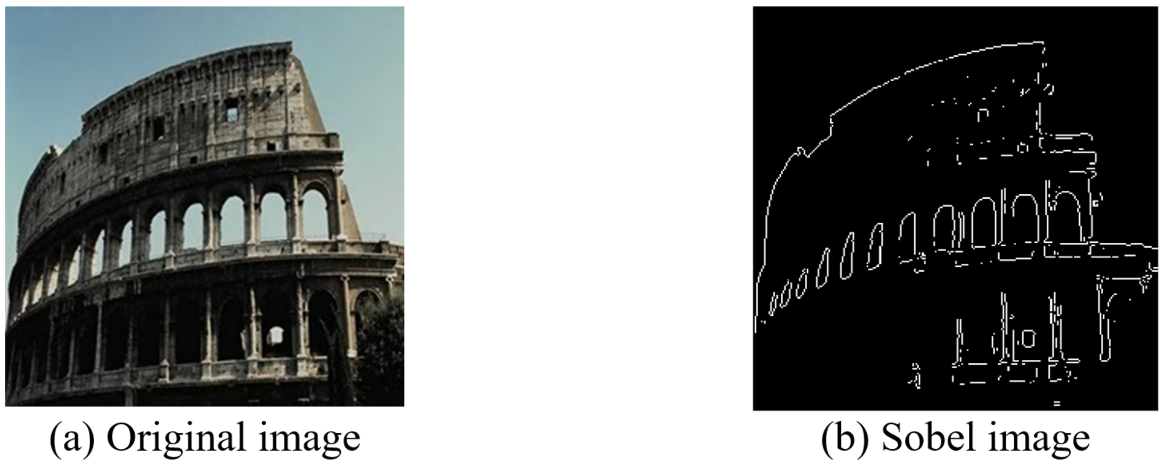


Figure 8. Sobel edge detection

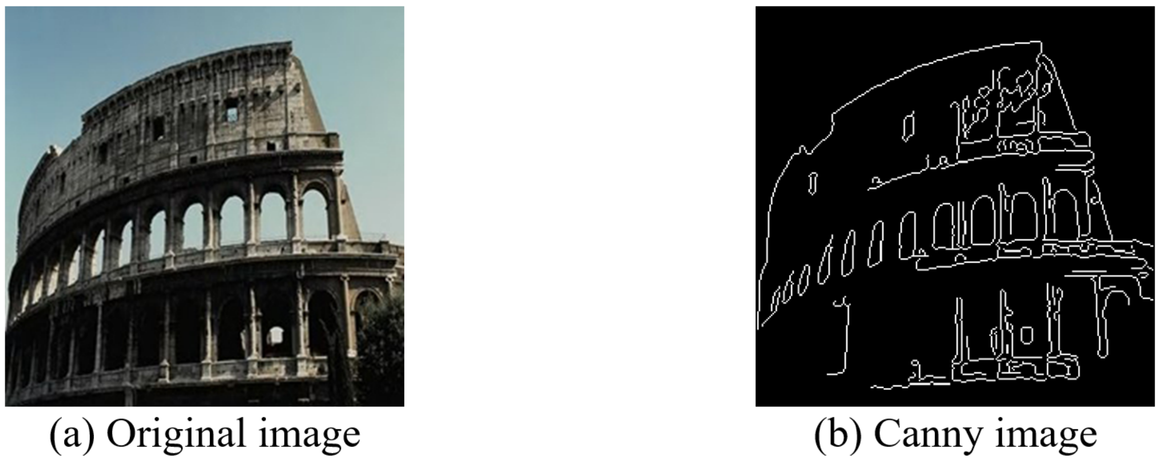


Figure 9. Canny edge detection

3. **Proposed method.** We proposed three modules in our method: VF-retrieval, ACQ-retrieval, and AV-retrieval. These three modules are complementary. A VF-retrieval module exploits the vector quantization (VQ) technique to obtain the image features and then build these features on a quad tree. Using VQ technique can decrease the features in the image, and then building features on a 4-ary tree is to speed up the image retrieval. Before images are saved to an image database, they usually experience some processing. In order to resist a geometric image processing, an ACQ-retrieval module is developed. This ACQ-retrieval module uses the characteristic of Affine Invariant Region (AIR) to avoid being unable to access the processed images. When the ACQ-retrieval module is applied, and the AIR images are obtained, the Canny edge detection is adopted to obtain the edge images. To accelerate the image retrieval with edges, the Quadtree is used to encode the edge information. In the ACQ-retrieval module, although a similar image can be found, this approach will fail to extract the edge information when the target image is ever processed. Therefore, the AV-retrieval module is designed to make up the lack of the ACQ-retrieval module to retrieve more AIR images and improve recall rates. Figure 10 illustrates a visual representation of the sequence of steps for the proposed content-based image retrieval (CBIR) system.

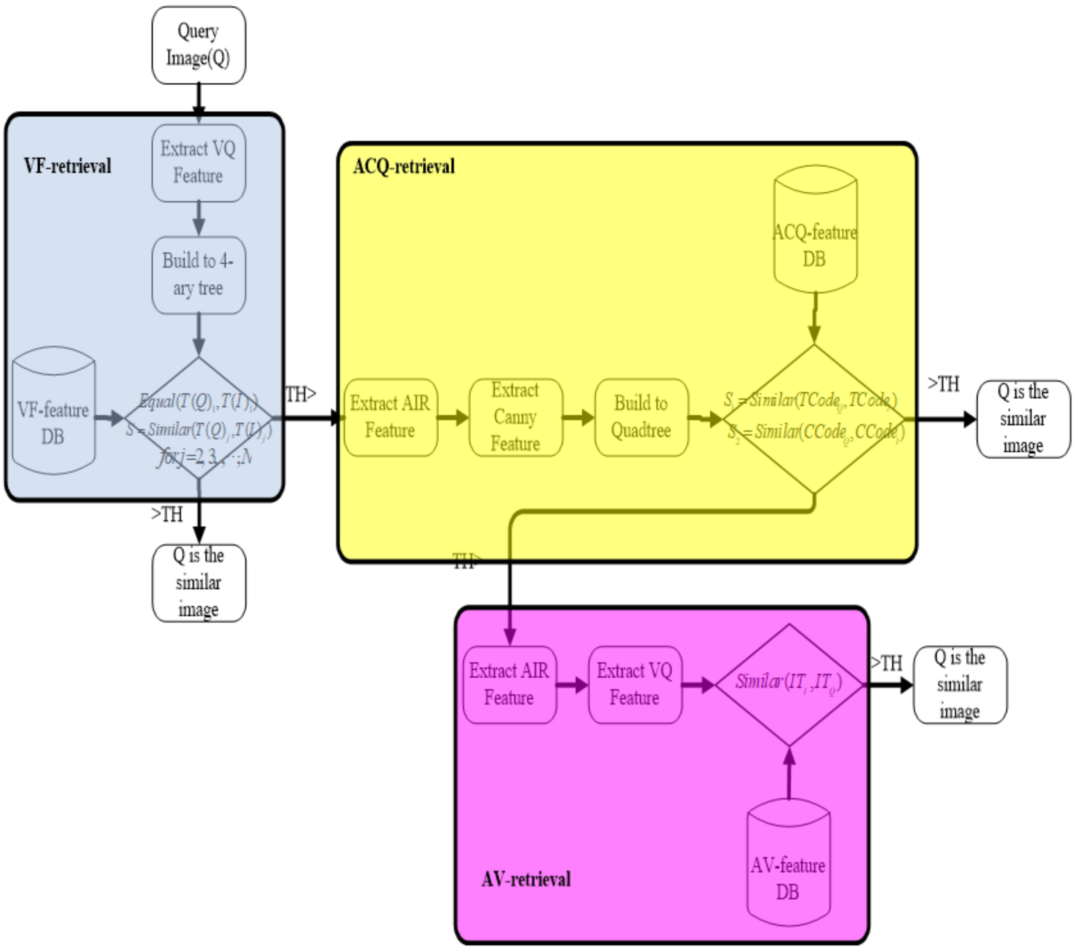


Figure 10. Flowchart of proposed content-based image retrieval (CBIR) system

3.1. **VF-retrieval.** In order to build a fast image retrieval approach, we use the VQ technique to encode image features and build these features on a 4-ary tree. The steps are as follows:

Step 1: Given an image database with each image of $M \times M$ pixels.

Step 2: Train Ncodebooks, CB_i, $= 1, 2, ,;$ for the -th codebook, the codeword number is 4^i ; the length of each codeword is $(M \times M)/4^i$.

Step 3: For each image I, build Nindex tables with Ncodebooks. The size of each index table is $2^i \times 2^i$. $ITB(I)_i$ stands for the i-index table encoded by CB_i codebook.

Step 4: Use $ITB(I)_i$, $i = 1, 2, ..., N$ to build a 4-ary tree $T(I)$ corresponding to the image I. The tree has (N+1) levels, and the 0-th level is a root which is a dummy node. Each node of level i is encoded with $ITB(I)_i$. Therefore, the number of nodes at level i is 4^i.

There are 4 subtrees of level i from left to right to represent 4 locations: NW, SE, SW, and SE of $ITB(I)_i$. Because images have regional continuity of related information, we can keep the correlation of image locations in a spatial domain. With the correlation encoding, the recall rate of image retrieval can be increased. Moreover, the steps for matching features are reduced due to the VQ indexing so that the retrieve time can be speeded up.

3.2. **ACQ-retrieval.** We can retrieve similar images quickly at the VF-retrieval method. However, if the image is processed by the geometry processing, it is impossible to be

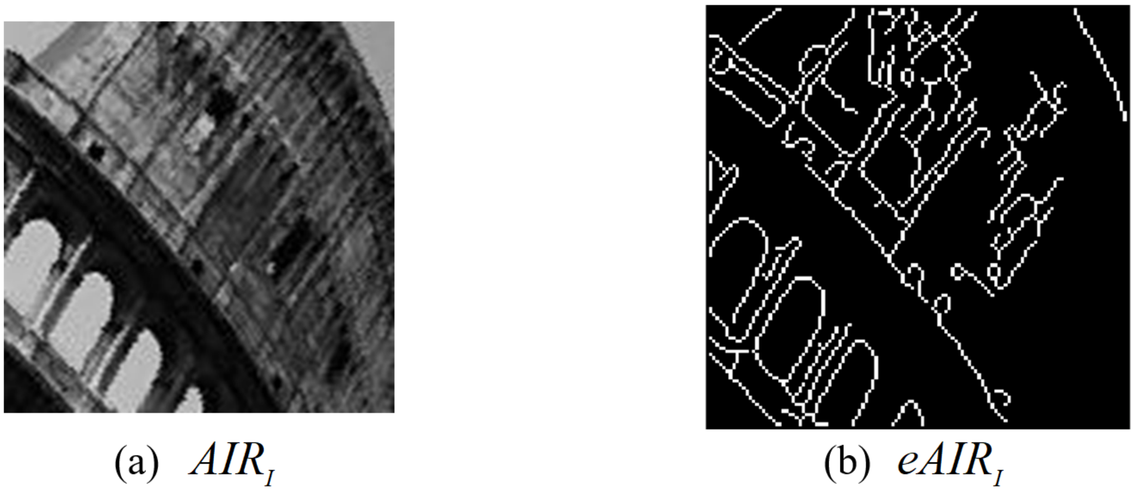


FIGURE 11. Image AIR_I and its edge images$eAIR_I$ by the Canny edge detection

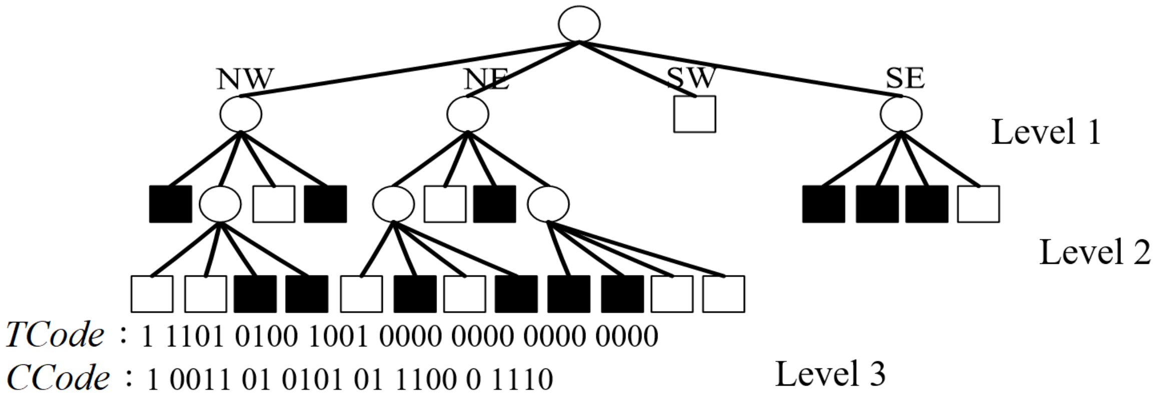

NW
NE
SW
SE
Level 1
Level 2
Level 3
TCode : 1 1101 0100 1001 0000 0000 0000 0000
CCode : 1 0011 01 0101 01 1100 0 1110

FIGURE 12. The Quadtree transform form of Fig.11

retrieved. Therefore, we use AIR to obtain the affine invariant region and use this region to retrieve images even the images are processed, followed by the Canny edge detection to obtain edge images. Since edge images are not easy to retrieve, we construct the features based on the Quadtree structure. The advantage of this structure is that it can retrieve images quickly and protect the color domain and geometry processed. The steps of this method are as follows:

Step 1 Given an image database with image size of $M \times M$ pixels.

Step 2 Apply Gaussian Kernel filtering to each image I to obtain the corresponding Gaussian image.

Step 3 Obtain the Local-based image of feature points.

Step 4 Obtain the center of mass of image I.

Step 5 Obtain the Affine Invariant Region of image I.

Step 6 Transform the Affine Invariant Region into the normalized image AIR_I, the size of AIR_I is $m \times m$, as shown in Figure 11(a).

Step 7 Obtain the edge images $eAIR_I$ of each AIR_I by the Canny edge detection, as shown in Figure 11(b).

Step 8 Transform each $eAIR_I$ image into the Quadtree (QT_I), and record the codes for the tree structure ($TCode_I$) and pixel values($CCode_I$), respectively.

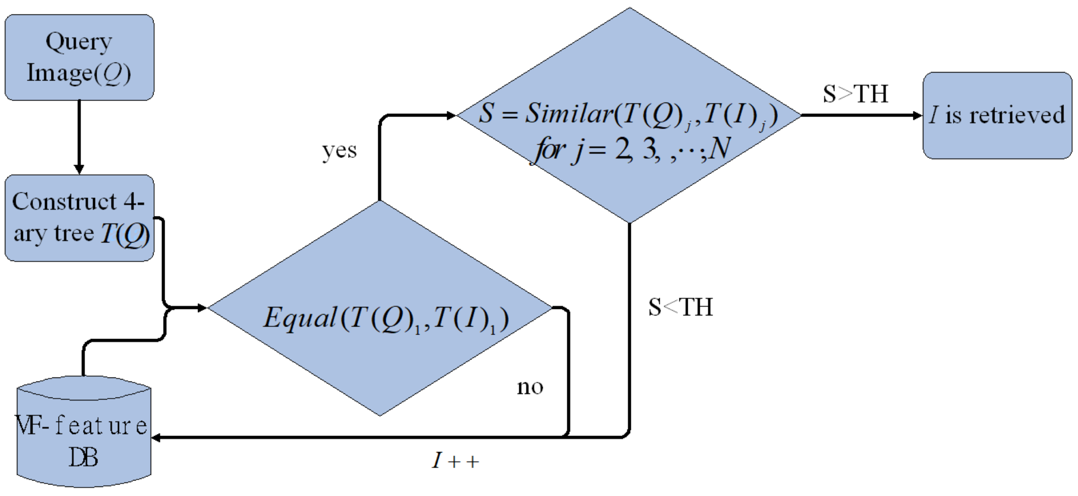


Figure 13. VF-retrieval's retrieve diagram

3.3. **AV-retrieval.** We will create an AIR image for every image in the image database. Several representative AIR images are used to train a codebook (CB) by which the AIR image can be encoded and transformed into a corresponding index table treated as an image feature and used for retrieving images.

Step 1: Use Gaussian Kernel Filtering in Eq. (1) for reducing image noises.

Step 2: Get the local-based maximum and minimum robust feature points P_A and P_B of image I.

Step 3: Get the center of mass (CI) of image I by Eq. (7).

Step 4: Find out the AIR region of image I.

Step 5: Normalize the AIR region into square images (AIR_I). The size of AIR_I is $m \times m$ pixels.

Step 6: Apply a VQ technique to encode the AIR_I images to obtain the corresponding index table IT_I.

To be protected from attacks or image processing, we combine VQ and AIR to help extract the image characteristics by which we can retrieve the most similar images. The affine invariant region related to images is an unchangeable characteristic and can be employed to help image retrievals. The VQ coding of index reduces the complexity of feature comparisons when retrieving images. Besides, VQ encodes the image in the block wisely so to preserve the spatial locations of objects in the image, thereby, increasing the accuracy rate of image retrievals. The proposed scheme consisting of feature extractions and image retrievals is introduced below.

3.4. **Image retrieval.** We use the approaches proposed in Section 3.1 to obtain image features and then retrieve images. At first, the VF-retrieval method finds out those images similar to the query image. Then the ACQ-retrieval or AV-retrieval method is applied to find out more similar images. The procedure steps of VF-retrieval, ACQ-retrieval and AV-retrieval methods for image retrievals are illustrated in the follow diagrams in Figure 13.

Step 1Given a Query image Q.

Step 2Build the 4-ary tree Procedure to construct a 4-ary tree $T(Q)$.

Step 3Take out the feature $T(I)$ of image I in the VF-feature database and compare with $T(Q)$. Our method will not compare the root, because it is dummy node. Then compare $T(I)_j$ and $T(Q)_j$ for j=1, which means to compare all nodes of Level 1. ($T(I)_1$,$T(Q)_1$) means $T(I)_1$ and are corresponding nodes and are fully equal. If $T(I)_1$ and $T(Q)_1$ are equal, then go to Step 4. If $T(I)_1$ and $T(Q)_1$ is not equal, then $I++$, take out next $T(I)$ in VF-feature database, and return to Step 3.

Step 4 Use the Match_Number_of Node Procedures to compare Level j $(j = 2, 3, .., N)$ nodes.

```
Procedure Match_Number_of_Nodes for $j = 2, 3, .., N$
$If((count(T(I)_j.NW(node) == T(Q)_j.NW(node)) + count(T(I)_j.SE(node) == T(Q)_j.SE(node))/(Numberof(NW(node) + SE(node)))) > TH$
$if\ ((count(T(I)_j.SW(node) == T(Q)_j.SW(node)) + count(T(I)_j.NE(node) == T(Q)_j.NE(node))/(Numberof(SW(node) + NE(node)))) > TH$
$j++;$
if $j > N$
Image I is retrieved;
End
End
```

During the processing to compare $T(I)$ and $T(Q)$, Level 1 uses CB_1, so this codebook has a few codewords. Therefore, the Euclidean distance pairs between the index values are huge. If the index values are different, then the image will be not similar. In other Level, if the Procedure Match_Number_of_Nodes($T(I)$, $T(Q)$)¿TH, then this Level is similar. This procedure can speed up comparing nodes and obtain higher recall rates.

Using the VF-retrieval method can find out the similar images quickly, but usually the users will process the images before storing them into the image database. The VF-retrieval method can't find out the images which have been resized, scaled or rotated. To address the issue, we proposed the ACQ-retrieval method to find out this kind of images.

We use the ACQ-retrieval method to obtain the *TCode* and *CCode*, and use these two codes to let the edge images retrieved. Moreover, we use the AIR feature to find out more images that be processed with higher recall rates. Figure 14 shows the retrieve diagram and the retrieval steps.

Step 1: Transform the Query Image(Q) into the normalized image (AIR_Q), the size of normalized image is $m \times m$.

Step 2: Use the Canny edge detection to obtain edge image .

Step 3: Build $eAIR_Q$by Quadtree. Extract the tree structure ($TCode_Q$), and the color value ($CCode_Q$).

Step 4: Take out the $TCode_I$ and $CCode_I$ of image (I) from the ACQ-feature DB.

Step 5: Compare the $TCode_Q$and $TCode_I$. If the $Similar(TCode_I,\ TCode_Q) = S_1 > TH$, then compare the $CCode_Q$ and $CCode_I$. If $Similar(CCode_I,\ CCode_Q) = S_2 > TH$, then think the image Q and image $I_\ell$are similar. If no one succeeds, then go to Step 4 to take out the $TCode_Q$and $TCode_I$ of next image(I).

Although the ACQ-retrieval module can find out a lot of similar images, the image process is going to destroy the edges. This method cannot find out similar images efficiently. Therefore, we proposed the AV-retrieval module to find out more images. The retrieve diagram and steps in Figure 15 are presented as follows:

Step 1: Let the query image Q to be a normalized image AIR_Q, and the size of AIR_Q is $m \times m$.

Step 2: Use the codebook to encode AIR_Q into its related index table.

Step 3: Compare IT_I and IT_Q, if $Similar(IT_I, IT_Q) > TH$, then IT_I and IT_Q is similar. Q is output as a similar query image; otherwise compare with the next image

4. Experimental Results.

This section introduces the experiment design and the experimental results to show recall rates and speeds of the proposed method. In our research,

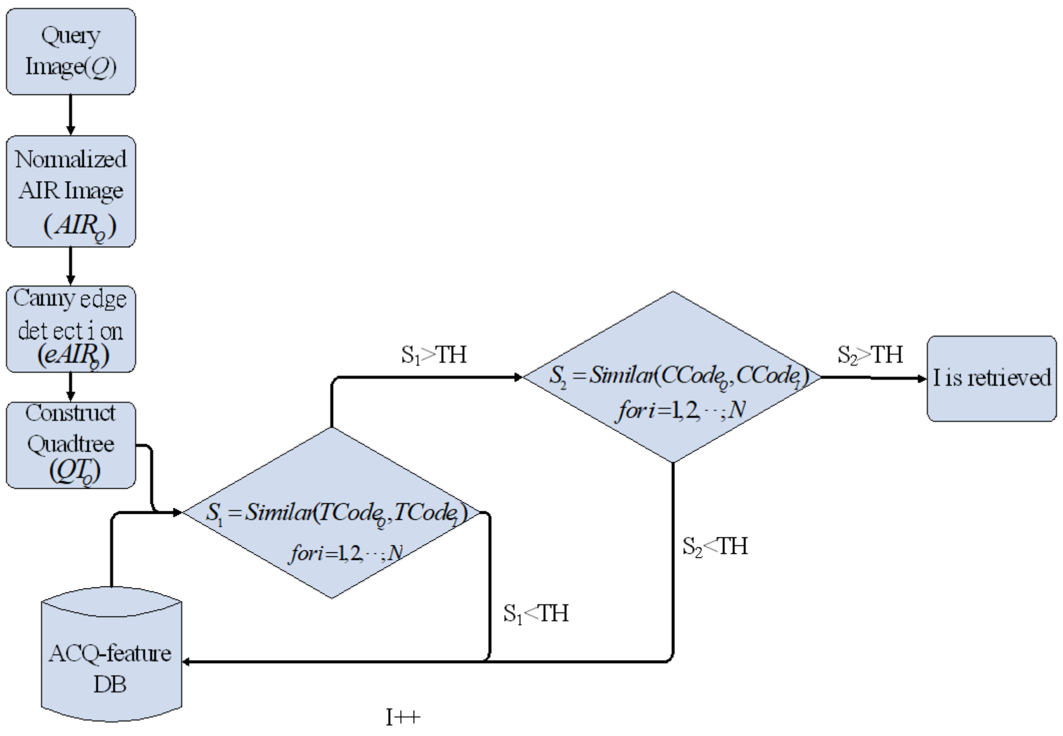

Query Image(Q)
Normalized AIR Image (AIR_Q)
Canny edge detection ($eAIR_Q$)
Construct Quadtree (QT_Q)
ACQ-feature DB
$S_1 = Similar(TCode_Q, TCode_i)$ $for\ i = 1, 2, \cdots, N$
S_1>TH
S_1<TH
$S_2 = Similar(CCode_Q, CCode_i)$ $for\ i = 1, 2, \cdots, N$
S_2>TH
S_2<TH
I is retrieved
I++

FIGURE 14. ACQ-retrieval's retrieve diagram

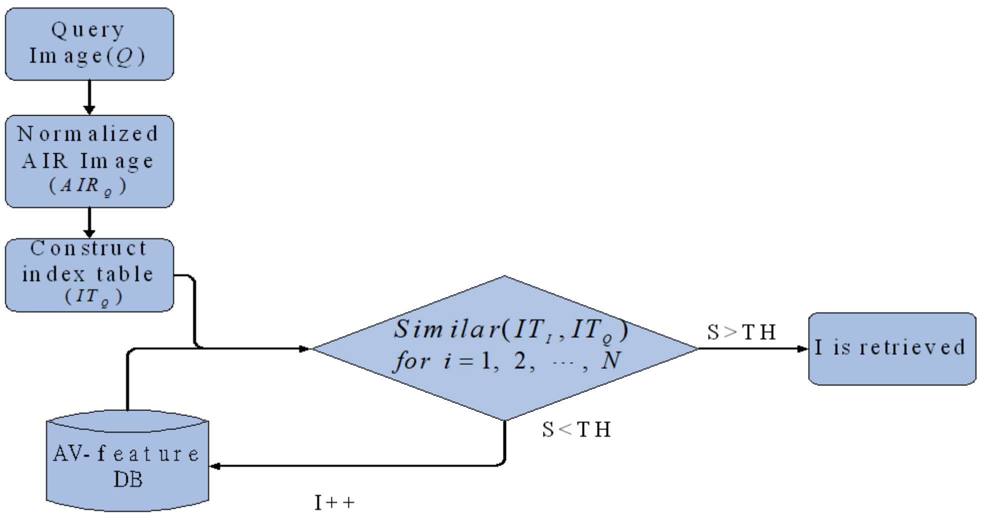

Query Image(Q)
Normalized AIR Image (AIR_Q)
Construct index table (IT_Q)
AV-feature DB
$Similar(IT_i, IT_Q)$ $for\ i = 1, 2, \cdots, N$
S>TH
S<TH
I is retrieved
I++

FIGURE 15. ACQ-retrieval's retrieve diagram

the images are provided by Wangs [27]. This image database has 10 categories, and we use one of them (Building) to find the best parameters of our method, and then use the results in other categories. We use MATLAB 7.1 to design our method run by a hardware system with AMD 3 GHz CPU and 2 GB main memory. Every category has 100 images, and each image in Figure 16 (a) will extend to 9 similar images as shown in Figure 16 (b) to Figure 16 (j). Therefore, we have 1000 images in an image database.

Recall(Rec), *Precision(Pre)*, *False positive*(F_p), and False negative(F_n) are used to estimate our proposed methods. The definitions are described in the following Equations (12)-(15), respectively. The Recall rate (Rec) means how many relevant images of the

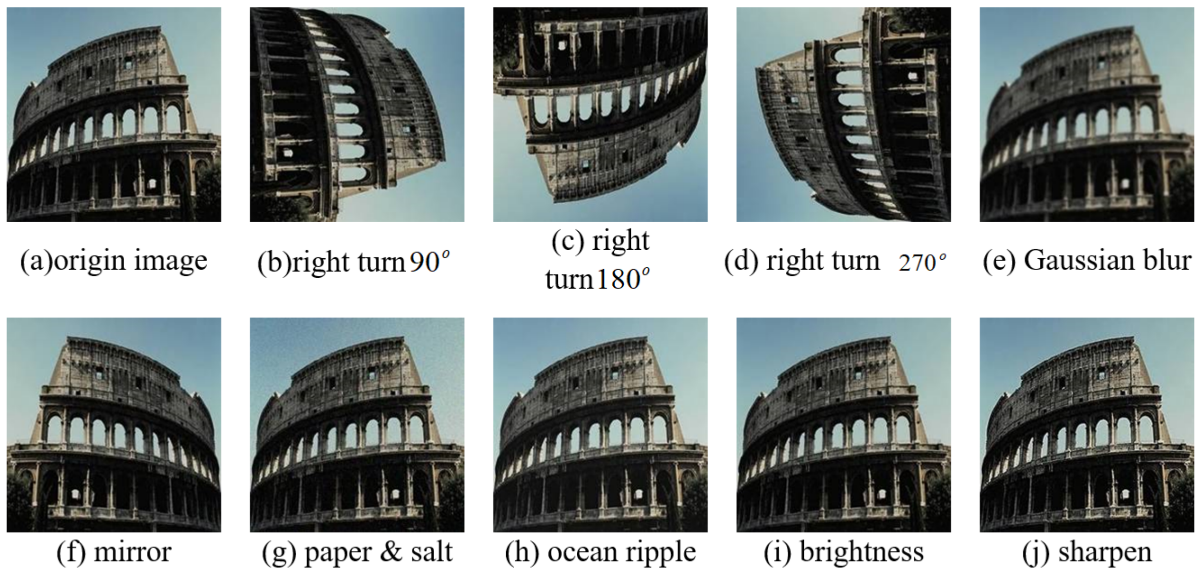


Figure 16. Similar images

total relevant images are retrieved. The Precision (Pre) means how many relevant images of the retrieve images are retrieved. The False positive (F_p) means that the image is not a relevant image, but our method think it is. The False negative (F_n) means that the image is a relevant image, but it cannot be retrieved by the proposed method. Because there are 10 similar images in our method, we will query top 10 similar images. Therefore, the Rec and Pre will be the same.

4.1. **Performance of VF-retrieval.** In this experiment, we use our proposed VF-retrieval method to find out the similar images quickly and have fair recall rates. We adjust 3 parameters: *TH*, *CB* size and block size, where *TH* is the threshold of the similarity, and *CB* denotes the size of codebook. Table 1 is the recall rates and retrieval time (in second) using a two-level 4-ary tree coding to encode images. The highest recall rate is 45.89%, when the block sizes are set as 32 $\times$32 (Level 1) and 16 $\times$16 (Level 2), TH=0.4 and CB size is 16/64, respectively. We fixed the block sizes as 16 $\times$16 and 8 $\times$8 as shown in Table 2 and discovered that the CB size set as 8 and 16 has a better recall rate.

4.1.1. *Comparison of VF-retrieval and VQ.* We use two image databases to test the recall rate by VF-retrieval and VQ, respectively. One is without geometry images, and another is with geometry images. Then the results are shown in Tables 3 and 4. In Table 3, we use the image database without geometry images, and discover that the *Rec* is very close. The retrieve times of the VQ the VF-retrieval are 0.69 sec and 0.918 sec, respectively. However, when we use the image database with geometry images, the recalls of our proposed method and VQ will be substantially lower. We also find that our proposed method can compress an image from 92655.925 bytes into 799.364 bytes, and the retrieval time is 1 sec. Let Data compression ratio be defined as the ratio between the uncompressed size and compressed size. Therefore, the compression ratio of our proposed method is 92655.925/799.364=115.912, and the relative data redundancy defined as the reduction in size relative to the uncompressed size: 1- 1/115.912= 1-0.0086=99.13%.

Table 5 shows that the False positive (F_p) of our proposed method is quite low, which implies the VF-retrieval is a good method. Figure 17 is similar to Figure 16 (a) using VF-retrieval, and the geometry images cannot be retrieved by the VF-retrieval method.

Table 1. The recall rates (Rec) and retrieval time (sec) of different CB sizes when block size is fixed as 32 ×32 and 16×16

CB size	16/64		64/128		8/16		16/32		32/64	
TH	*Rec*	Time	*Rec*	Time	*Rec*	Time	*Rec*	Time	*Rec*	Time
0.4	**0.4589**	1.5421	0.4236	1.2052	0.4286	0.593	0.3304	0.607	0.3064	0.647
0.5	0.4104	1.5078	0.389	1.3447	0.3564	0.626	0.3096	0.606	0.2732	0.645
0.6	0.3762	1.5735	0.3378	1.4939	0.3216	0.672	0.2922	0.6	0.234	0.651
0.7	0.3206	1.5212	0.3084	1.513	0.312	0.595	0.2454	0.609	0.2214	0.692
0.8	0.3042	1.53	0.303	1.5351	0.2916	0.579	0.2242	0.62	0.2102	0.635
0.9	0.29	1.5355	0.2642	1.5558	0.229	0.597	0.2182	0.625	0.3064	0.647

Table 2. The recall (Rec) and retrieval time (sec) of different CB sizes when block size is fixed as 16 ×16 and 8 ×8

CB size	16/64		32/64		8/16		16/32	
TH	*Rec*	Time	*Rec*	Time	*Rec*	Time	*Rec*	Time
0.4	0.4070	0.850	0.4068	0.955	**0.4460**	0.922	0.4208	0.799
0.5	0.3718	0.815	0.3718	1.003	0.4192	1.026	0.4000	0.797
0.6	0.3242	0.897	0.3242	1.024	0.4046	0.943	0.3682	0.864
0.7	0.3032	0.834	0.3032	1.0459	0.3556	0.808	0.3176	0.897
0.8	0.288	1.040	0.2880	1.010	0.3082	0.871	0.2994	0.931
0.9	0.2134	1.089	0.2134	1.003	0.2830	0.926	0.2390	0.872

Table 3. Compare VQ and VF-retrieval (without geometry images)

Images	*Rec* (VQ)	*Rec* (VF-retrieval)
Buildings	0.9589	0.9612

For this reason, we replace the VF-retrieval method with the ACQ-retrieval based on AIR-based features.

4.2. **Performance of VF-retrieval.** Since the VF-retrieval has low *Rec* when the image database has geometric images, we propose the ACQ-retrieval based on AIR-based features and demonstrate that the adaptive method indeed has a good *Rec*.

In the experiment, we detect edges using the Canny method since Canny edge detection filters out useless data, noises and frequencies while preserving the important structural properties in an image. The Canny method finds edges by looking for local maxima of the gradient of an image. The edge function calculates the gradient using the derivative

TABLE 4. Compare VQ and VF-retrieval (with geometry images with *TH=0.4*)

Images	*Rec* (VQ)	*Rec* (VF-retrieval)
Beach	0.3186	0.4108
Building	0.3028	0.4460
Bus	0.3112	0.4028
Dinosaur	0.3272	0.4317
Elephant	0.3710	0.4286
Food	0.3262	0.4152
Horse	0.3850	0.4131
Mountain	0.3450	0.4218
People	0.3212	0.4022
Rose	0.3380	0.4040

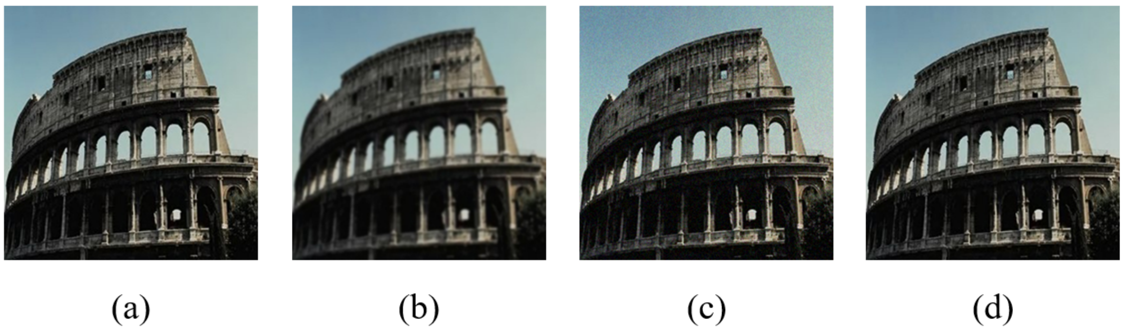


FIGURE 17. Similar images of I_i by VF-retrieval method

TABLE 5. The Rec, F_n, and F_p of VF-retrieval mehtod

Images	*Rec*	F_n	F_p
Beach	0.4108	0.5892	0.005952
Building	0.4460	0.5540	0.005596
Bus	0.4028	0.5972	0.006032
Dinosaur	0.4317	0.5683	0.00574
Elephant	0.4286	0.5714	0.005772
Food	0.4152	0.5848	0.005907
Horse	0.4131	0.5869	0.005928
Mountain	0.4218	0.5782	0.00584
People	0.4022	0.5978	0.006038
Rose	0.4040	0.5960	0.00602

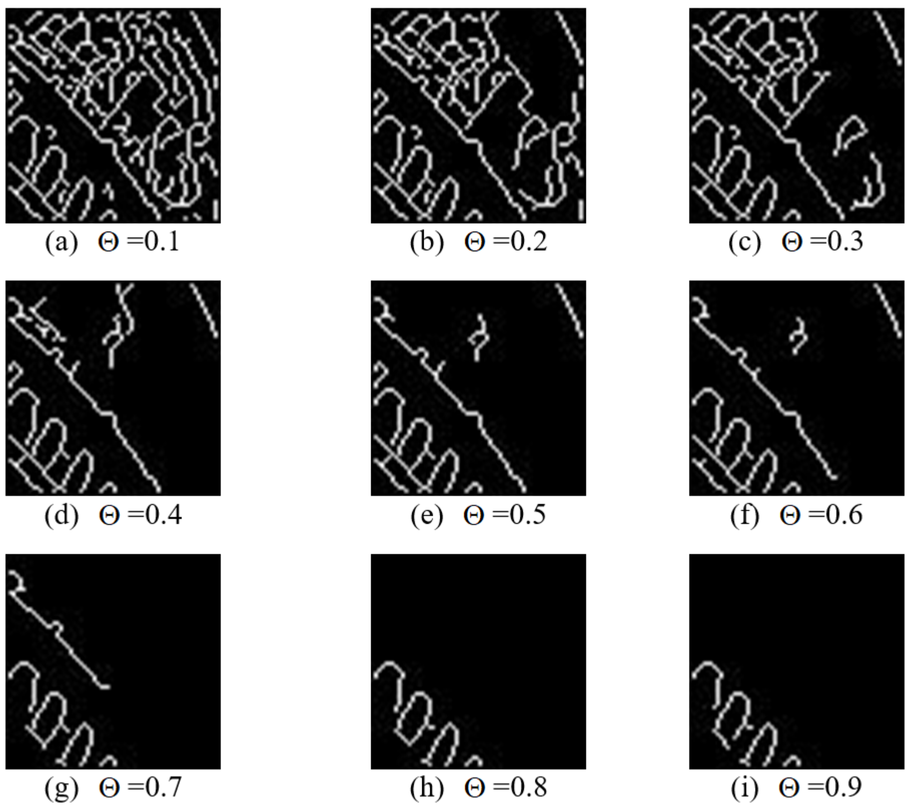


FIGURE 18. The results of Canny edge detection with various Θ

of a Gaussian filter. This method uses two thresholds to detect strong and weak edges, including weak edges in the output if they are connected to strong edges. Precisely speaking, any pixel in the image that has a value greater than $hT$is presumed to be an edge pixel, and is marked as such immediately. Then any pixels connected to this edge pixel with values greater than lT are also selected as edge pixels. By using two thresholds hT and lT, the Canny method is less likely fooled by noises than the other methods, and more likely to detect true weak edges. We set $hT= \Theta$ and $lT= \Theta \times 0.4$ where $\Theta$is the angle of gradient at each point and can be obtained by Eq. 11.

$$M_x\left(x,y\right)=G'_x\bullet I\left(x,y\right),\ M_y\left(x,y\right)=G'_y\bullet I\left(x,y\right),\ \Theta=\arctan\left(\frac{G_y}{G_x}\right) \tag{11}$$

Figure 18 depicts that the results of the Canny edge detection with various sorts of Θ. From Figure 18, we know that the smaller the angle of gradient Θ.is, the more edges are preserved by the Canny edge detection. Moreover, Table 6 shows that the smaller the Θ, the higher the recall rate. In the following experiment, we use Θ =0.2 as the parameter.

Table 7 shows the recall (Rec), False positive (F_p) and False negative (F_n) in a variety of image categories using the ACQ-retrieval method to enhance our previous VF-retrieval method which cannot effectively extract the geometric images. Figure 19 uses Figure 16(a) as the query image, and ACQ-retrieval to retrieve the top 10 similar images. In Figure 19, we discover our method can actually find out the images which are applied by special image preprocessing techniques. In Table 7, we can see the ACQ-retrieval method indeed achieves higher recall rates than the VF-retrieval method. Due to the low speed by the ACQ-retrieval method, we continue to propose an AV-retrieval method to deal with this problem.

TABLE 6. The Rec F_n, and F_p with various sorts of Θ.

Θ	*Rec*	F_n	F_p	time
0.1	0.874	0.126	0.001273	21.84
0.15	0.873	0.127	0.001283	19.28
0.2	**0.871**	**0.129**	**0.001303**	**15.00**
0.25	0.868	0.132	0.001333	15.001
0.3	0.867	0.133	0.001343	16.16
0.4	0.831	0.169	0.001707	13.17
0.5	0.824	0.176	0.001778	12.44
0.6	0.798	0.202	0.00204	10.39
0.7	0.784	0.216	0.002182	12.1
0.8	0.775	0.225	0.002273	11.48
0.9	0.762	0.238	0.002404	12.45

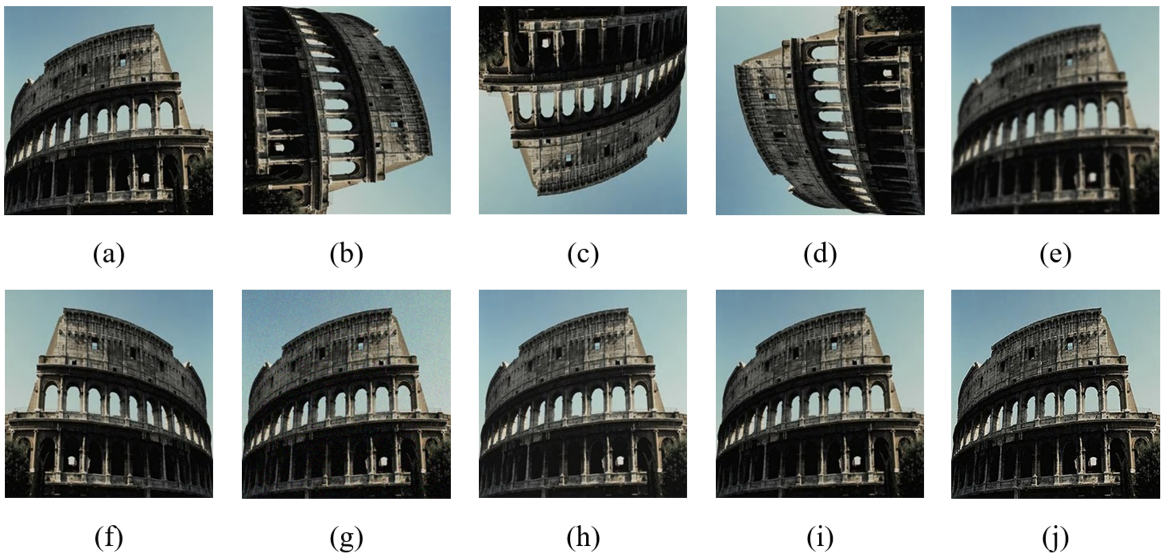


FIGURE 19. The Top 10 similar images of I_1

TABLE 7. The Rec, F_p, and F_n using ACQ-retrieval(Θ=0.2)

Images	*Rec*	F_n	F_p
Beach	0.724	0.276	0.002788
Building	0.871	0.129	0.001303
Bus	0.701	0.299	0.00302
Dinosaur	0.712	0.288	0.002909
Elephant	0.743	0.257	0.002596
Food	0.688	0.312	0.003152
Horse	0.715	0.285	0.002879
Mountain	0.709	0.291	0.002939
People	0.694	0.306	0.003091
Rose	0.682	0.318	0.003212

4.3. **Performance of AV-retrieval.** In the above experimental results, we present that the VF-retrieval method provides good retrieve time, and the ACQ-retrieval method provides good recall rates. Therefore, we proposed the AV-retrieval method to combine the advantages of these two methods. The parameters used in this experiment are both block sizes and CB sizes because block sizes affect the number of features required for matching, and the CB size affects whether the blocks are similar to the original image. Table 8 shows that the smaller the block size, the higher the recall rate. However, it needs more retrieval time. Different CB sizes also effect the recall rates. In this table, when the block size is 8×8, and the CB size is 16, the AV-retrieval method has the best recall rate, 0.9248, and acceptable retrieval time.

Table 8. Rec and retrieve time of the AV-retrieval method with different block sizes and CB sizes

Block size	64×64		32×32		16×16		8×8	
CB size	*Rec*	Time	*Rec*	time	*Rec*	time	*Rec*	time
4	0.1542	3.8776	0.625	3.42	0.886	3.5742	0.823	3.513
8	0.1823	3.9328	0.7462	3.9688	0.8999	4.395	0.9216	5.823
16	-	-	0.8102	4.7995	0.9093	5.212	**0.9248**	**6.7937**
32	-	-	0.7076	5.2417	0.8932	5.662	0.9025	8.5618
64	-	-	-	-	-	-	0.88	9.451

Therefore, we use the block size, 8 ×8, and the CB size, 16, as the parameters to show the Rec, F_n, and F_p for various image categories. In Table 9, we can see the AV-retrieval method has a better recall rate and less retrieval time compared with ACQ-retrieval method.

Table 9. Rec, F_n, and F_p of AV-retrieval

Images	*Rec*	F_n	F_p
Beach	0.8648	0.1352	0.001366
Building	0.9248	0.0752	0.00076
Bus	0.8075	0.1925	0.001944
Dinosaur	0.8271	0.1729	0.001746
Elephant	0.7729	0.2271	0.002294
Food	0.7416	0.2584	0.00261
Horse	0.7918	0.2082	0.002103
Mountain	0.7584	0.2416	0.00244
People	0.7638	0.2362	0.002386
Rose	0.7585	0.2415	0.002439

In Figure 20, Figure 16 (a) are used as the query image, and the AV-retrieval method is used to retrieve the top 10 similar images. In this figure, we can discover our method can actually find out the geometric images effect the retrieval time. Moreover, according to the experiment, the AV-retrieval method can compress the images from avg.

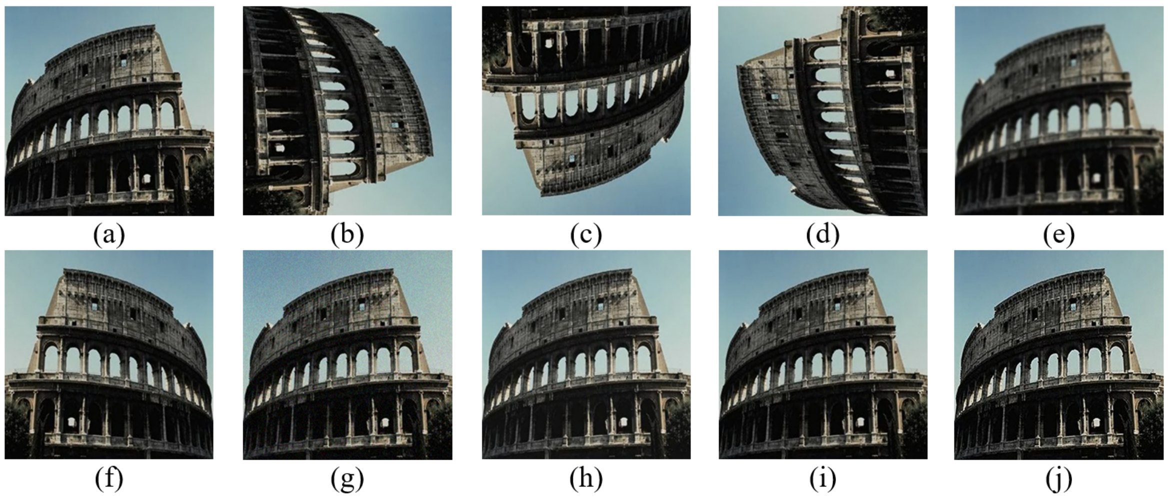


FIGURE 20. The Top 10 similar images of I_1

92655.925 bytes to 209.206 bytes, which indicates that the compression ratio reaches 92655.925/209.206=442.89, and the relative data redundancy is 1- 1/442.89= 99.77%.

4.4. **Comparison of the ACQ-retrieval and AV-retrieval methods.** According to the experiments 4.2 and 4.3, the Av-retrieval method has a higher Rec than that of the ACQ-retrieval method with less retrieval time. All the images are greyscale images. In this section, we want to explore the performance in different hues and saturations. The images of the experiment are shown in Figures 21(a)-(f). We use Adobe Photoshop to modify the color domain. Figure 21(a) comes from one of the Building category image database with 100 images in it. Figure 21(b) uses the default color of the Hue/Saturation in Photoshop, and then we apply the ACQ-retrieval and the AV-retrieval methods on these images. The results are shown in Table 10(a). In Figure 21(c) with Hue= -180, the result is shown in Table 10(b). In Figure 21(d) with Saturation=100, the result is shown in Table 10(c). In Figure 21(e) with Hue=180 and Saturation=100, the result is shown as Table 10(d). In Figure 21(f) with Invert image, the result is shown as Table 10(e).

In Table 10, the image quality is measured by the Peak Signal to Noise Ratio (PSNR), and the accuracy is measured by the recall (Rec) and precision (Pre). In dealing with the color of the domain, when the PSNR of image quality decreases, the Rec and Pre of retrieval accuracy using the ACQ-retrieval method can be maintained to a certain extent, but those of retrieval accuracy using the AV-retrieval method decrease significantly. Therefore, using the ACQ-retrieval or the AV-retrieval method will determine the quality of the image required. If the captured images are similar in visual quality, you can use AV-retrieval method to quickly retrieve images. However, when the image quality is poor, you can get better results by using the ACQ-retrieval method.

TABLE 10. The comparisons of image quality (PSNR), recall rates and retrieval precisions between AV-retrieval and ACQ-retrieval methods

	(a)		(b)		(c)		(d)		(e)	
	AV	ACQ	AV	ACQ	AV	ACQ	AV	ACQ	AV	ACQ
PSNR	20.6722		17.8518		13.9823		10.032		5.2673	
Rec	0.905	0.795	0.745	0.795	0.6525	0.615	0.3725	0.5775	0.52	0.56
Precision	0.181	0.159	0.149	0.159	0.1305	0.123	0.0745	0.1155	0.104	0.112

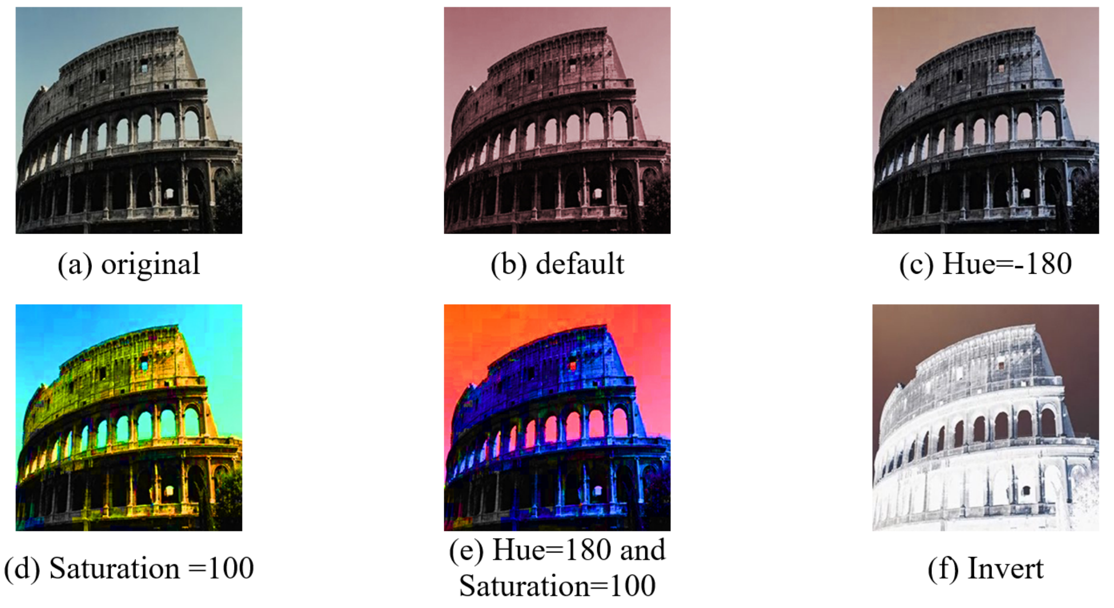


FIGURE 21. Images after color domain processed

In this paper, we proposed 3 different methods: VF-retrieval, ACQ-retrieval, and AV-retrieval. In VF-retrieval method, we can quickly find out the similar images, but it cant retrieve geometric images. Therefore, we proposed the ACQ-retrieval and AV-retrieval methods to increase the recall rates. According the experiments, we discovered that the AIR block is an effect method to assist image retrieval. In the experiment of our proposed method, ACQ-retrieval and AV-retrieval methods have higher Rec when retrieving the top 10 similar images. As to using which kind of methods, it is up to the query image. Our proposed method can retrieve the images processed, and has higher recall rates and lower retrieve time.

5. **Conclusions.** The target of a CBIR system is to search images in a large image database system based on derived image features such as colors, textures and shapes. We proposed three methods: VF-retrieval, ACQ-retrieval, and AV-retrieval to extract the features of the images. The VF-retrieval method can find out the similar images quickly, but the ACQ-retrieval and AV-retrieval methods can find out the images with higher recall rates. Compared with these three methods, our method can lower the retrieval time, increase recall rates, and achieve higher accuracy.

Acknowledgment. This research was partially supported by the Ministry of Science and Technology of the Republic of China under the Grant MOST106-2221-E-324-006 -MY2.

REFERENCES

[1] R. C. Veltkamp and M. Tanase, Content-based image retrieval systems: A survey, *Technical Report UU-CS-2000-34*, Dept. of Computing Science, Utrecht University, October 2000.
[2] S. W. Teng and G. Lu, Efficient implementation of vector quantization for image retrieval, *Proceedings of the 12th International Multi-Media Modelling Conference*, Beijing China, pp. 280-287, 2006.
[3] Y. Liu, D. Zhang, Guojun Lu, and W.Y. Ma, A survey of content-based image retrieval with high-level semantics, *Pattern Recognition*, vol. 40, no. 1, pp. 262-282, January 2007.
[4] G. Carneiro, A.B. Chan, P. J. Moreno, and N. Vasconcelos, Supervised learning of semantic classes for image annotation and retrieval, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 394-410, March 2007.

[5] R. Datta, D. Joshi, J. Li, and J.Z. Wang, Image retrieval: Ideas influences and trends of the new age, *ACM Computing Surveys*, vol. 40, no. 2, April 2008.

[6] S. Arora, D. Bhattacharjee, M. Nasipuri, D.K. Basu, and M.Kundu, Complementary features combined in a MLP-based system to recognize handwritten Devnagari character, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 2, no. 1, pp.71-77, January 2011.

[7] F. Alamdar and M.R. Keyvanpour, A new color feature extraction method based on QuadHistogram," *Procedia Environmental Sciences*, vol. 10, part A, pp. 777-783, 2011.

[8] J. Wang and X.S. Hua, Interactive image search by color map, *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 1, pp. 1-23, October 2011.

[9] Z. Liu, H. Li, L. Zhang, W. Zhou, and Q. Tian, Cross-indexing of binary SIFT codes for large-scale image search, I*EEE Transactions on Image Processing (TIP)*, vol. 23, no. 5, pp. 2047-2057, May 2014.

[10] X. Wang and Z. Wang, The method for image retrieval based on multi-factors correlation utilizing block truncation coding, Pattern Recognition, vol. 47, no. 10, pp. 3293-3303, 2014.

[11] X.Y. Wang, B.B. Zhang, and H.Y. Yang, Content-based image retrieval by integrating color and texture features, *Multimedia Tools and Applications (MTA)*, vol. 68, no. 3, pp. 545-569, 2014.

[12] L. Xie, Q. Tian, W. Zhou, and B. Zhang, Heterogeneous graph propagation for large-scale web image search, *IEEE Transactions on Image Processing (TIP)*, vol. 24, no.11, pp. 4287-4298, Nov. 2015.

[13] S. Sun, W. Zhou, Q. Tian, and H. Li, Scalable object retrieval with compact image representation from generic object regions, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM),* vol. 12, no. 2, article 29:1-21, March 2016.

[14] A. ElAdel, R. Ejbali, M. Zaied, and C. B. Amar, A hybrid approach for content-based image retrieval based on fast beta wavelet network and fuzzy decision support system, Machine Vision and Applications, vol. 27, no. 6, pp.781-799, 2016.

[15] G. Raghuwanshi and V. Tyagi, Texture image retrieval using adaptive tetrolet transforms, *Digital Signal Processing*, vol. 48, pp. 5057, 2016.

[16] S. Fadaei, R. Amirfattahi, and M.R. Ahmadzadeh, New content-based image retrieval system based on optimized integration of DCD, wavelet and curvelet features, I*ET Image Processing*, vol. 11, no. 2, pp. 89-98, 2017.

[17] P. Liu, J.M. Guo, K. Chamnongthai, and H. Prasetyo, Fusion of color histogram and LBP-based features for texture image retrieval and classification, *Information Sciences*, vol. 390, pp. 95111, June 2017.

[18] R.M. Gray, Vector quantization, IEEE ASSP Magazine, vol. 1, no.2, pp. 4-29, 1984.

[19] Y. Linde, A. Buzo, and R. M. Gray, An algorithm for vector quantization design, *IEEE Transactions on Communications*, vol. 28, pp. 84-95, 1980.

[20] T. W. Lin, Set operations on constant bit-length linear quadtrees, Pattern Recognition, vol. 30, no. 7, pp. 1239-1249, 1997.

[21] I. Gargantini, An effective way to represent quadtrees, *Communications of ACM*, vol. 25, no. 12, pp. 905-910, 1982.

[22] C. Dyer, The space efficiency of quadtrees, Computing Graphics Image Process, vol. 19, no. 4, pp. 335-348, 1982.

[23] G. Birkhoff and S. Mac Lane, Affine geometry, *A Survey of Modern Algebra,* 5th ed. New York: Macmillan, pp. 268-275, 1996.

[24] K.L. Hung and S.W. He, Feature based affine invariant watermarking robust to geometric Distortions, *Fundamenta Informaticae*, vol. 92, no. 1-2, pp. 131-143, 2009.

[25] Irwin Sobel, History and definition of the Sobel operator, 2014. Available from: https://www.researchgate.net/publication/239398674_An_Isotropic_33 Image Gradient_Operator [accessed on Feb. 2018].

[26] D. Lijun and G. Ardeshir, On the Canny edge detector, Pattern Recognition, vol. 34, no. 3, pp.721-725, 2001.

[27] J. Z. Wang, Wang Research Group, Penn State, University Park, PA 16802. Available from: http://wang.ist.psu.edu/docs/home.shtml [accessed on Feb. 2018].