Journal of Engineering and Applied Sciences 14 (7): 2203-2207, 2019
ISSN: 1816-949X

Suggested Method to Create Color Image Features Victor

Belal Ayyoub, Jihad Nader, Ziad Al-Qadi and Bilal Zahran
Department of Computer and Network Engineering, Faculty of Engineering Technology,
Al-Balqa Applied University, Salt, Jordan

Abstract: Digital color images are the most widely used data types and are used in many vital applications such as computer recognition systems and image retrieval system, so, creating an image features victor is an important issue, this victor can be used as a signature or key to identify or retrieve the image from a color image database. Two methods of color image features extraction will be proposed, implemented and analyzed, local contrast method and a proposed method which will be based on local binary pattern. The obtained experimental results will be compared in order to raise some recommendations which will be useful when building power image recognition systems.

Key words: Histogram, local contrast, LBP, CSLBP, feature victor, extraction time, speedup

INTRODUCTION

T digital color images are the most widely used data types and are used in many vital applications such as computer recognition systems and image retrieval system, color image size is too big, that's why identifying the image pixel by pixel is an expensive by mean of time task. Digital RGB color image is constructed by merging 3 2D matrices as shown in Fig. 1, the first one for the red color, the second for the green color and the third one for the blue color (Qaryouti *et al*., 2016, 2017; Zneit *et al*., 2017; Moustafa and Alqadi, 2009). Color image size equal the size of the three matrices and it is usually big, so, we refer to the image histogram to represent the image. Image histogram is a one column victor and each element value in this victor represents the number of repetition of pixel intensity (from 0-255), so, using this victor of smaller size we can identify the image. Reducing the elements in histogram victor will reduce the efforts of color image recognition or retrieving (Qaryouti *et al*., 2017).

Local Constract Method (LCM): Image Local Contrast (LC) can be used to generate local contrast array of different elements (Hunt, 1992) (in our study, we will use 4 levels of resolution). LC is an average difference between neighboring pixels (the horizontal neighbors and the vertical neighbors) and to calculate LC we have to apply the following steps (Matkovic *et al*., 2005):

- Get the color image
- Reshape the color 3D-2D matrix

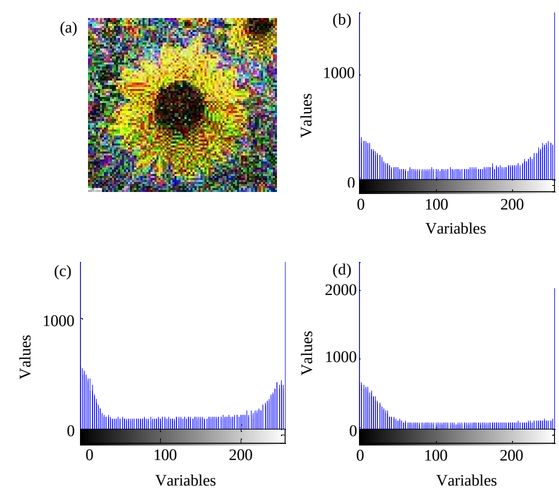

Fig. 1: Color image and histograms

- Calculate the scaled and corrected values of linear Luminance l:

$$l = \left(\frac{k}{255}\right)^{\gamma} \quad (1)$$

Where:

k = The pixel value (0-255)
γ = 2.2

Calculate perceptual Luminance L using Eq. 2:

$$L = 100 * \sqrt{l} = 100 * \sqrt{\left(\frac{k}{255}\right)^{\gamma}} \quad (2)$$

Corresponding Author: Belal Ayyoub, Department of Computer and Network Engineering, Faculty of Engineering Technology, Al-Balqa Applied University, Salt, Amman, Jordan

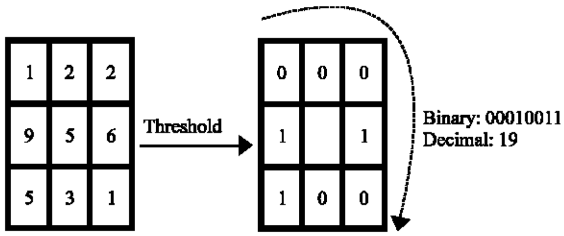

Fig. 2 : LBP method operations

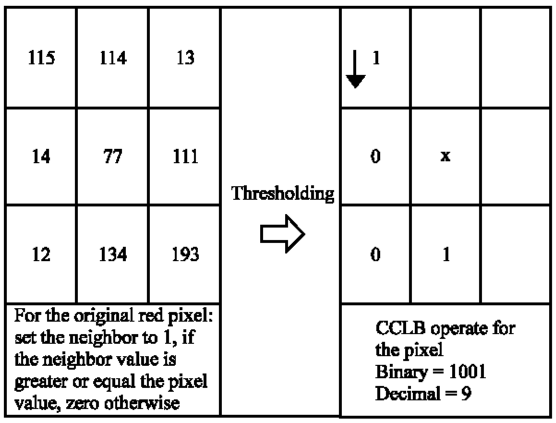

Fig. 3: CSLBP method operations

For each of the 4 resolution levels. Calculate LC_i for each pixel using Eq. 3:

$$l_{c_i} = \frac{|L_i - L_{i-1}| + |L_i - L_{i+1}| + |L_i - L_{i-w}| + |L_i - L_{i+w}|}{4} \quad (3)$$

Find the average local contrast for each resolution level applying Eq. 4:

$$C_i = \frac{1}{w*h} * \sum_{i=1}^{w*h} lc_i \quad (4)$$

Where:
w = The number of rows
h = The number of columns in the image matrix

Local binary patterns: Local Binary Patterns (LBP) is used to recalculate the image histogram as illustrated in Fig. 2, this recalculation leads to generate a histogram with 256 elements, thus it does not reduce the features victor elements (Heikkila *et al*., 2006; Petruk, 2011), a modification of LBP is Center-Symmetric Local Binary Pattern (CSLBP) method which can be used to reduce the number of elements in the features victor to 16, Fig. 3 illustrates how CSLBP method works (Gehler and Nowozin, 2009; Uma and Srujana, 2015; Al-Zudool *et al*., 2017; Khawtreh *et al*., 2018).

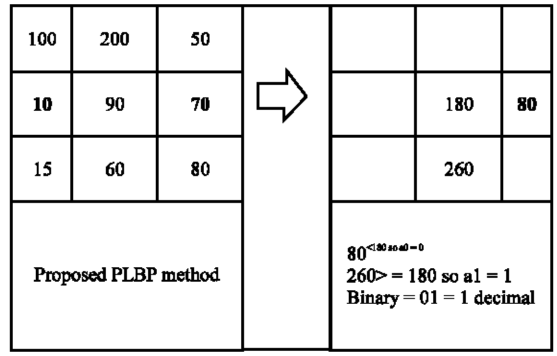

Fig. 4: Calculating PLBP

MATERIALS AND METHODS

The proposed method uses the pixel 4 neighbors as illustrated in Fig. 4 to generate a 2 digit binary number, thus, the number of elements in features victor will be reduced to 4. The proposed method (PLBP) can implemented applying the following steps:

- Get the original color image
- Initialize the 4 elements features victor to zeros
- Reshape the 3D color matrix to 2D matrix
- For each pixel in the 2D image matrix calculate the binary pattern and get the equivalent decimal number d
- Add 1 to the element with d index in the features victor

RESULTS AND DISCUSSION

Implementation and experimental results: The following matlab code was written to implement LC method of features victor generation.

Implementation and experimental results: The following matlab code was written to implement LC method of features victor generation.

Algorithm 1; LC method of feature victor generation:

```
clc; clear all
a = imread('a20.jpg')
[rr1 rr2 rr3] = size(a)
rr1*rr2*rr3
tic
im = reshape(a,rr1*rr3, rr2)
  LC = zeros(4, 1)
  W = size(im, 2)
  H = size (im, 1)
  rIm = im
  for i = 1:4
        %attempt at resizing as in the paper
    if i>1
     rIm = imresize(im, 1/(2^(i-1)), 'bilinear')
```

```
        end
        W = size (rIm, 2)
        H = size (rIm, 1)
        rL = zeros (size (rIm))
        % compute linear luminance l
l = (double(rIm(:, :))/255)*2.2
% compute perceptual Luminance L
rL(:, :) = 100*sqrt(l)
% compute local contrast for each pixel
lc = 0.0
for x=1:H
for y=1:W
if (x == 1) && (x == H)
if (y == 1) && (y == W)
lc = lc + 0
else if (y == 1)
lc = lc+abs(rL(x, y)-rL(x, y+1))
else if (y == W)
lc = lc+abs(rL(x, y)-rL(x, y-1))
else
lc = lc+( abs(rL(x, y)-rL(x, y-1))+...
abs(rL(x, y)-rL(x, y+1)) )/2
end
else if (x == 1)
if (y == 1) && (y == W)
lc = lc+abs(rL(x, y)-rL(x+1, y))
else if (y == 1)
lc = lc+( abs(rL(x, y)-rL(x, y+1))+...
abs(rL(x, y)-rL(x+1, y)))/2
else if (y == W)
lc = lc+( abs(rL(x, y)-rL(x,y-1))+...
abs(rL(x, y)-rL(x+1, y)))/2
else
lc = lc+( abs(rL(x, y)-rL(x, y-1))+...
abs(rL(x, y)-rL(x, y+1))+...
abs(rL(x, y)-rL(x+1, y)))/3
end
else if (x == H)
if (y == 1) && (y == W)
lc = lc+abs(rL(x, y)-rL(x-1, y))
else if (y == 1)
 lc = lc+( abs(rL(x, y)-rL(x, y+1))+...
abs(rL(x, y)-rL(x-1, y)))/2
else if (y == W)
lc = lc+( abs(rL(x, y)-rL(x, y-1))+...
abs(rL(x, y)-rL(x-1, y)))/2
else
lc = lc+( abs(rL(x, y)-rL(x, y-1))+...
abs(rL(x, y)-rL(x, y+1))+...
abs(rL(x, y)-rL(x-1, y)))/3
end
else%x>1 && x<H
 if (y == 1) && (y == W)
 lc = lc+( abs(rL(x, y)-rL(x+1, y))+...
abs(rL(x, y)-rL(x-1, y)))/2
else if (y == 1)
lc = lc+( abs(rL(x, y)-rL(x, y+1))+...
abs(rL(x, y)-rL(x+1, y))+...
abs(rL(x, y)-rL(x-1, y)))/3
else if (y == W)
lc = lc+( abs(rL(x, y)-rL(x, y-1))+...
abs(rL(x, y)-rL(x+1, y))+...
abs(rL(x, y)-rL(x-1, y)))/3
else
lc = lc+( abs(rL(x, y)-rL(x, y-1))+...
abs(rL(x, y)-rL(x, y+1))+...
abs(rL(x, y)-rL(x-1, y))+...
abs(rL(x, y)-rL(x+1, y)))/4
end        end        end        end
%compute average local contrast c
c(i) = lc/(W*H)
w(i) = (-0.406385*(i/9)+0.334573)*(i/9)+ 0.0877526
%compute local contrast victor
LC(i) = c(i)*w(i);        end
toc
```

Different color images with different sizes and types were selected, each time the features victor was generated and Table 1 shows the results of implementation. The same experiment was implemented by PLBP method by executing the following MATLAB code:

Algorithm 2; PLBP method:

```
clc; clear all
a = imread('a1.bmp')
[rr1 rr2 rr3] = size(a)
rr1*rr2*rr3
tic
im = reshape(a, rr1*rr3, rr2)
fea = zeros(4, 1)
for I = 2:rr1*rr3-1
   for j=2:rr2-1
      a00 = [(im(i-1,j)+im(i+1,j))>=2*im(i, j)]
      a01=[(im(i, j-1)+im(i, j+1))>=2*im(i, j)]
      kk = a00+2*a01
      fea(kk+1) = fea(kk+1)+1
   end  end
toc
fea'
```

The results of implementation are shown in Table 2. From Table 1 and 2, we can see that each features victor is a unique for each image, thus, we can use these victors as a signatures or keys to identify or retrieve a certain image. The features victor obtained by each method is a unique and is very sensitive to any changes in the image (even for a very small changes as illustrated in Table 3.

For both methods features victor generation (features) time was obtained by applying the previous

Table 1: Feature victors obtained by LC method for different message

Image #	Size (pixel)	-----------------------Features			-----------------------
1	270948	0.7029	0.9420	1.3637	2.1304
2	151875	0.7521	0.9180	1.1601	1.4904
3	49152	1.0355	1.0376	1.3494	1.6101
4	1125600	0.9834	0.9469	1.0118	1.3161
5	540000	1.4120	1.2246	1.3452	1.9101
6	3396069	0.8420	0.7940	0.8817	1.2390
7	2359296	0.6520	0.7046	0.9217	1.4551
8	928800	1.2015	1.0986	1.1262	1.3937
9	432000	1.2783	0.9842	1.0443	1.4202
10	151353	1.7832	1.5315	1.8349	2.5978
11	151248	0.9219	1.1376	1.4450	2.0427
12	12402570	1.7710	1.1044	0.4100	0.6467
13	2193750	0.6066	0.6382	0.7843	1.1013
14	432000	1.2781	0.9841	1.0442	1.4203
15	151353	1.7872	1.5327	1.8357	2.5989
16	151512	5.3404	2.7915	2.0102	2.3124
17	150660	1.3354	1.0002	1.1267	1.6463
18	151032	1.2503	1.1044	1.3492	2.0258
19	151200	1.1384	1.1212	1.3146	1.7836
20	270000	0.8202	0.8445	1.0927	1.5716

Table 2: Feature victors obtained by PLBP method for different message

Image #	Features			
1	55061	45104	37108	131397
2	18640	23584	15027	92828
3	7121	6270	5699	29042
4	165902	145839	131100	678349
5	75906	67073	60331	333694
6	395295	301166	259191	2432369
7	201452	153677	129269	1868246
8	247859	175997	162563	338089
9	64466	50569	45900	268249
10	31273	22450	20959	74967
11	26933	23125	22579	76963
12	613632	2670430	1650336	7453250
13	153561	177628	163150	1692965
14	64454	50567	45908	268255
15	31344	22454	20982	74869
16	19964	27463	16365	85968
17	33121	20783	20029	75093
18	25658	16759	16525	90328
19	28715	20770	19494	80585
20	31121	25592	21890	189001

Table 3: Sensitivity of features victor

Variables	Feature extraction method			
Image 1 (LC)				
Original image features	0.7029	0.9420	1.3637	2.1304
Features after changing 1 pixel	**0.7030**	**0.9421**	**1.3638**	2.1304
Features after changing 2 pixel	0.7030	0.9420	1.3637	2.1304
Image 1 (proposed)				
Original image features	55061	45104	37108	131397
Features after changing 1 pixel	**55063**	**45102**	**37107**	131398
Features after changing 2 pixel	55063	45104	37109	131394

Bold values are sensitive features

Table 4: Extraction time comparisons

		Extraction time (sec)		
Image #	Size (pixel)	LC method (1)	Proposed method (2)	Speed up: (1)/(2)
1	270948	0.825000	0.103000	8.0097
2	151875	0.507000	0.172000	2.9477
3	49152	0.169000	0.057000	2.9649
4	1125600	3.429000	1.243000	2.7586
5	540000	1.676000	0.637000	2.6311
6	3396069	10.122000	4.666000	2.1693
7	2359296	7.105000	3.764000	1.8876
8	928800	2.857000	0.324000	8.8179
9	432000	1.580000	0.507000	3.1164
10	151353	0.743000	0.113000	6.5752
11	151248	0.752000	0.108000	6.9630
12	12402570	37.529000	17.670000	2.1239
13	2193750	6.871000	3.236000	2.1233
14	432000	1.600000	0.509000	3.1434
15	151353	0.752000	0.121000	6.2149
16	151512	0.741000	0.165000	4.4909
17	150660	0.752000	0.115000	6.5391
18	151032	0.741000	0.108000	6.8611
19	151200	0.753000	0.145000	5.1931
20	270000	1.101000	0.389000	2.8303
Average	1280500	4.0302	1.7076	2.3602

MATLAB codes, the results of implementation are shown in Table 4. From Table 4, we can see that the proposed PLBP method of features extraction is more efficient than CSLBP method and it has a speedup up average equal 2.3 times which means that the proposed method is 2.3 times faster than CSLBP method.

CONCLUSION

An efficient method of color features extraction was proposed, the obtained experimental results showed that this method is 2.3 times faster than CSLBP. The obtained features victor for each color image is a unique victor, so, this victor can be easily used as an image key or signature to identify or retrieve any color image with any type or size.

REFERNCES

Al Zudool, M.A., S. Khawatreh and Z.A. Alqadi, 2017. Efficient methods used to extract color image features. Intl. J. Comput. Sci. Mob. Comput., 6: 7-14.

Gehler, P. and S. Nowozin, 2009. On feature combination for multiclass object classification. Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Septmber 29-October 2, 2009, IEEE, Kyoto, Japan, ISBN:978-1-4244-4420-5, pp: 221-228.

Heikkila, M., M. Pietikainen and C. Schmid, 2006. Description of interest regions with center-symmetric local binary patterns. Proceedings of the 5th Indian Conference on Computer Vision, Graphics and Image Processing ICVGIP Vol. 6, December 13-16, 2006, Springer, Madurai, India, pp: 58-69.

Hunt, R.W.G., 1992. Measuring Color (Ellis Horwood Series in Applied Science and Industrial Technology). 2nd Edn., Ellis Horwood Limited, Quezon, Philippines, ISBN:9780135676868, Pages: 313.

Khawatreh, S., B. Ayyoub, A. Abu-Ein and Z. Alqadi, 2018. A novel methodology to extract voice signal features. Intl. J. Comput. Appl., 179: 40-43.

Matkovic, K., L. Neumann, A. Neumann, T. Psik and W. Purgathofer, 2005. Global contrast factor-a new approach to image contrast. Proceedings of the 1st Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging, May 18-20, 2005, Eurographics Association, Girona, Spain, ISBN:3-905673-27-4, pp: 159-168.

Moustafa, A.A. and Z.A. Alqadi, 2009. Color image reconstruction using a new R'G'I model. J. Comput. Sci., 5: 250-254.

Petruk, V., 2011. Application of local binary patterns to face recognition problem solving. Master Thesis, Bauman Moscow State Technical University, Moscow, Russia.

Qaryouti, G.M., D.Z. Alqadi and D.M A. Zalata, 2016. A novel method for color image recognition. Intl. J. Sci. Mob. Comput., 5: 57-64.

Qaryouti, G.M., S. Khawatreh, Z.A. Alqadi and M.A. Zalata, 2017. Optimal color image recognition system OCIRS. Intl. J. Adv. Comput. Sci. Technol., 7: 91-99.

Uma, S.K. and B.J. Srujana, 2015. Feature extraction for human detection using HOG and CS-LBP methods. Intl. J. Comput. Appl., 2: 11-14.

Zneit, D.R.S.A., D.Z. AlQadi and D.M.A. Zalata, 2017. A methodology to create a fingerprint for RGB color image. Intl. J. Comput. Sci. Mob. Comput., 6: 205-212.