Journal of Applied Sciences

ISSN 1812-5654

Journal of Applied Sciences 11 (19): 3351-3361, 2011
ISSN 1812-5654 / DOI: 10.3923/jas.2011.3351.3361

Nested Circles Boundary Algorithm for Rotated Texture Classification

[1]Mutaz M. Al-Debei, [2]Suzan Tayseer Awienat and [1]Rana Rasem Abu-Laila
[1]Department of Management Information System, University of Jordan, Jordan
[2]Madrid, Spain

Abstract: A new simple yet efficient classification algorithm named as Nested Circle Boundary (NCB) is proposed in this study. This algorithm provides features from measuring the average of the sum of boundary pixels for a number of nested circles inside the texture image. It was tested on different 91 rotated texture images for 13 texture classes using Brodatz texture database. The proposed algorithm achieves 100% accuracy when it comes to rotated texture classification. The methodology of NCB algorithm is based on two phases. Phase one mainly measures the features of the 13 texture classes and the original texture images. Then, the derived textures details are stored to be compared later with the rotated texture images. In phase two, the NCB algorithm repeats the same measures conducted in phase one but on the rotated texture images. The matching process is conducted by measuring the center values of each rotated and original image and comparing the summation of the 8 neighbors of each center with those of original images so as to find the best center value. The best center values are then assumed to be the base points for the circles, where the algorithm calculates the average of the summation of its boundary pixel values. Finally, the features of rotated textures are compared with the features of original texture images so as to find the classification solution. In conclusion, the Nested Circles Boundary Algorithm (NCB) proposed in this paper achieves the highest accuracy level using a simple technique.

Key words: Nested circles boundary algorithm, texture analysis, rotated texture classification, digital image, image processing, statistical methods

INTRODUCTION

The classification of rotated texture is one of the main challenges for image processing and computer graphics research domains. The digital image is a numerical array that is generated after digitizing and converting the original form of the image into digital form. So, the digital image can be defined as a 2-D function f (x, y), where x and y denotes co-ordinates and the value of f called the gray level of image (Gonzalez and Woods, 2002). Digital Image Processing incorporates actions and operations that are usually performed in order to achieve a set of specific aims. Indeed, research in the domain of image processing is very important as results are deemed to be useful for many applications in several important fields such as medicine, science and commerce. Given its perceived usefulness, several studies have developed different algorithms to classify rotated textures and the highest accuracy that has been obtained is 99%.

There is no formal definition of texture (Pratt *et al*., 1978) and it is very difficult to be described. Notwithstanding, most of previous research define texture as the properties held and sensations caused by the external surface of objects received through the sense of touch (Haralick *et al*., 1973; Bovik *et al*., 1990; Jain and Karu, 1996). Four methods to analyze textures can be found on literature (Zhang and Tan, 2002a; Jiahua, 2003; Chen *et al*., 1998) statistical (Poonguzhali and Ravindran, 2008) geometrical, model-based and signal processing methods.

The major property of the statistical method is using statistical measures to analyze the spatial distribution of gray values. Some of the statistical methods that can be used for texture analysis are Co-occurrence method (Al-Bashish *et al*., 2011), gray level differences method and autocorrelation function (Jiahua, 2003; Materka and Strzelecki, 1998; Zhang and Tan, 2002a). The geometrical model depends on the pattern of the texture. Therefore and in order to analyze textures, this method normally uses statistical features or displacement rules (Amirani *et al*., 2008; Khan *et al*., 2007; Zhang *et al*., 2008). Moreover, extracting and identifying the primitives of the texture can be done by any of these three methods: Edge detection with laplacian-of-gaussian filter or difference-of-gaussian filter, adaptive region extraction and/or mathematical morphology.

Corresponding Author: Mutaz M. Al-Debei, Department of Management Information System, University of Jordan, Jordan

However, the model-based methods are based on determining the process which generates the texture. Three model-based methods are defined in this regard: Markov Random Fields (MRF), Fractals and Multi-Resolution Autoregressive (Jiahua, 2003; Materka and Strzelecki, 1998; Zhang and Tan, 2002a). On the other hand, signal processing methods are based on filtering process in order to perform frequency analysis. Spatial filters in the frequency domain, Low's filter, Gabor filters and pseudo-Wigner distribution can be applied for this method of processing.

Since the touch sense of the external surface of objects is represented by texture, the importance of classifying texture objects appeared. Texture image is spatially homogeneous and typically contains repeated structures, often with some random variation (Portilla and Simoncelli, 2006). In natural, capturing textures can not be in same deflection. So, texture analysis should be invariant to rotations which make texture recognition operation more difficult for rotated textures.

Fountain *et al*. (1998) compared four different methods: (1) Gabor filtering method, (2) Edge attribute processing methods, (3) The CSAR model method and (4) Wavelet decomposition method. Based on their results, they observed that the best recognition method is the Gabor method with 98.90% accuracy. Based on this result, Zhang and Tan (2002b) reported their new rotation invariant texture classification method with higher level of accuracy, i.e., 98.95% accuracy. However, at that point, the best accuracy rate was attributed to Haley and Manjunath (1995) modified Gabor filter algorithm as it achieved 99% accuracy rate.

This study propose a new simple rotated texture classification algorithm; namely Nested Circles Boundary (NCB) algorithm. NCB shows the highest accuracy performance. It was tested on 91 different rotated textures belong to 13 texture classes from Brodatz database (Brodatz, 1966) which is considered the standard evaluating texture algorithm (Picard *et al*., 1993). The 13 texture classes that are used in this study. The main aim of this study was to develop a new algorithm for classifying rotated texture images with a high accuracy level and without using any filtering methods to analyze the texture so as to make it simpler.

NCB ALGORITHM

The idea of the developed algorithm came from the property of the circle as its shape does not change with rotations. Therefore, taking a set of pixels values for a circle boundary in a texture image would give the same set

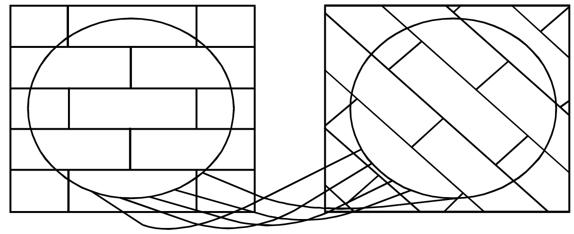

Fig. 1: Circle boundary pixels are equal for the texture image and it's rotated one

(Xc-1, Yc-1)	(Xc-1, Yc)	(Xc-1, Yc+1)
(Xc, Yc-1)	(Xc, Yc)	(Xc, Yc+1)
(Xc+1, Yc-1)	(Xc+1, Yc)	(Xc+1, Yc+1)

Fig. 2: The 8 neighbors for the center point

of pixel values when doing so for a rotated image of the same texture. Figure 1 can better represent the idea. It shows two rotated texture images belonging to the same class. Taking a set of pixel values for a circle boundary for the both images, will give the same set but with a shift of value positions in both sets.

However, this is only true if the rotation base point is the center of the texture image which can not always be. In this case the rotation base point should be found so as to make it the circle center in the rotated image. This approach assumed that the rotation base point is within a 70×70 block in the center of the rotated image. This block can be found by searching for the original image center value and its 8 neighbors, as shown in Fig. 2. Also note that the 8 neighbors can be shifted.

NCB algorithm incorporates two phases, as illustrated in Fig. 3. The first stage measures the features f (N) for each texture class, so as to be used later for comparison reasons with the rotated texture images. These measures include the center values, summation of 8 neighbors of the center and the average of the sum of the circle boundary pixel values for each texture image. This phase is called the initializing phase. On the other hand, the second phase is mainly concerned with analyzing and classifying the sample textures. In this phase the rotation based point is measured for the rotated texture sample. This base point is then assigned to be the center of the circle for each texture. Thereafter, the pixel values of the 8-neighbors of the center are used to calculate the sample image features f (s) in order to be compared later to the classes features. Accordingly, the class with the closest feature value will be returned as the class of this sample.

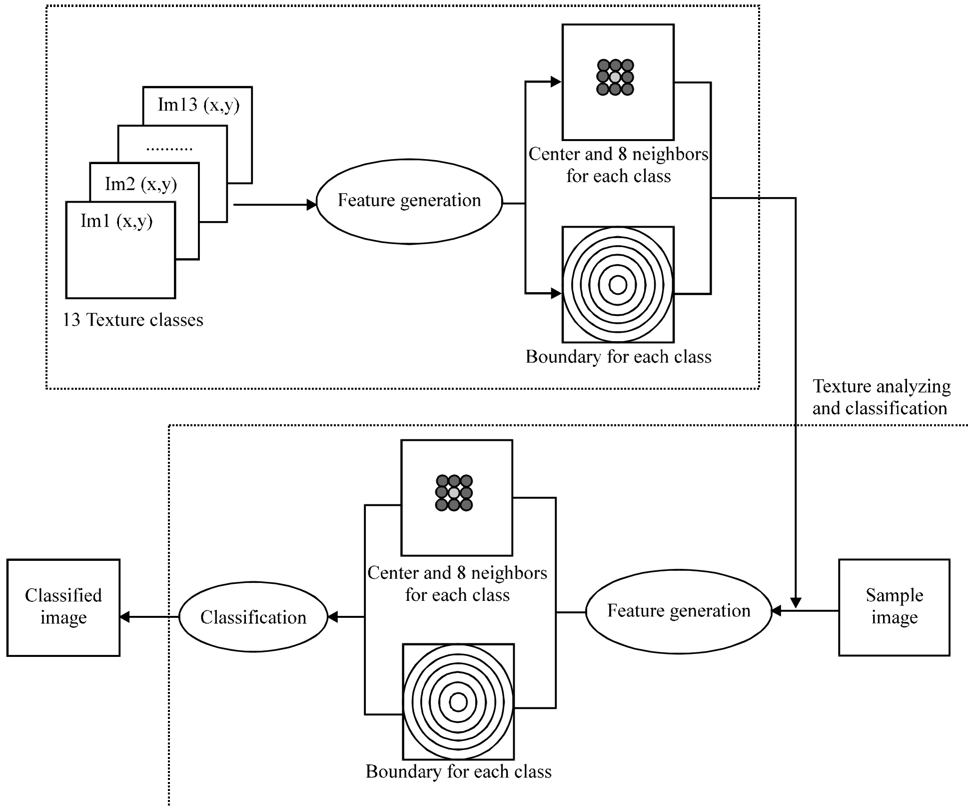

Fig. 3: NCB algorithm

Phase 1: Initializing.

- **Input:** N texture classes
- **Output:** f (n): n features; one for each class to be compared with, Acenter (n): Array of center point values for each texture class and A8nei (n): array of the center 8 neighbors vectors
- **Methodology:** Feature is the average of the summation of the circle boundary pixel values, as defined in Eq. 1

Assuming N is the number of texture classes and n is the number of circles:

$$f(N)=\frac{1}{n}\sum_{c=1}^{n}\sum_{th=1}^{360}N(xb,yb) \quad (1)$$

Where,

$$xb=xc+r(c)*\cos(\theta) \quad (2)$$

$$yb=yc+r(c)*\sin(\theta) \quad (3)$$

- **(xc, yc):** The center of the circles which is for the original texture image, the center of the image
- **r(c):** The radius for circle c which is increased by 1 for each circle. So, the distance between every two circles is 1 pixel
- **θ:** is the angle used to calculate the boundary coordination pixels, to get all pixels on the boundary; all thetas should be taken 1-360

Figure 4 shows the previous parameters. Since the circle is a set of points that are located on a plane with fixed distances from the center, measurements of the circle boundary require the value of the center, the values of all points on the plane and the distance between the center and the plane. As shown in Fig. 4, the parameter (Xc, Yc) presents the center of the circles. The parameter r (c) presents the distance between the center and the plane which called radius where it increases by 1 to find n circles. On the other hand, the parameter (Xb, Yb) presents one point in a plane. Indeed, using theta values is necessary to find all points on the plane with values ranging from 1 to 360 to complete the boundary for each circle.

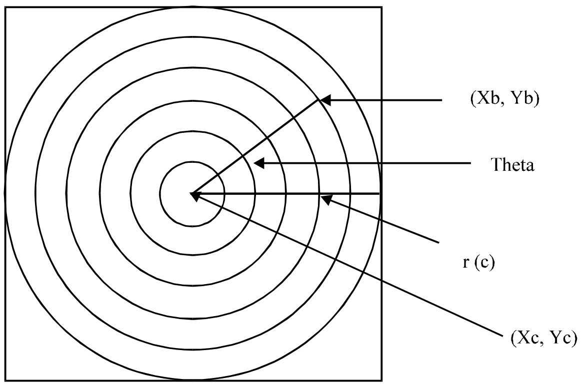

Fig. 4: Texture image feature equation parameters

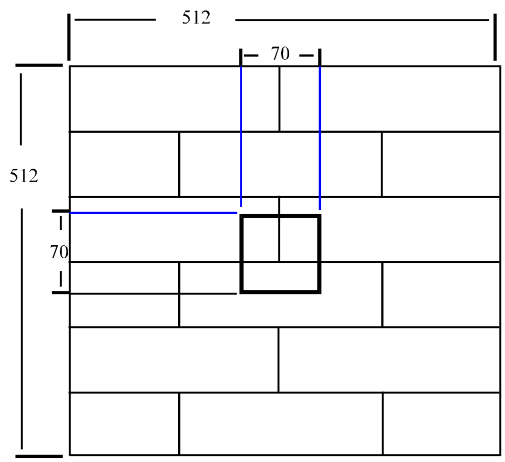

Fig. 5: The block, where the rotation base point can be found in the sample image

Phase 2: Texture analysis and classification.

- **Input:** The outputs of phase (1) and a sample image S
- **Output:** The class to which S belongs
- **Methodology:** Find pixels with values equal to each texture class center point in the 70×70 sample image block, that shown in Fig. 5. Find 8 neighbors for each pixel. Then compare between these values to get the correct pixel coordinate which represents the rotation base point of S. This point will be taken as a center of circles to calculate S feature, using the same Eq. 1

If more than one point are found, then all of them will be taken as a center of circles This will provide more than one feature for S which will be compared with texture classes features and get the texture class that S belongs to.

TESTING AND ANALYSIS

The algorithm described in this paper was developed in MATLAB using image processing toolbox (Gonzalez *et al*., 2006). The implementation of these algorithm are shown Fig. 8a, b. Ninety one texture images were used to test the classification algorithm. Fig. 6 shows 13 texture classes. Each class has 7 rotated texture images. For example, Fig. 7 shows the 7 rotated textures that belong to the class (brick). The size of images is 512×512 pixels.

Each class in brodatz database has 7 rotated texture images. Figure 7 shows the 7 rotated textures that belong

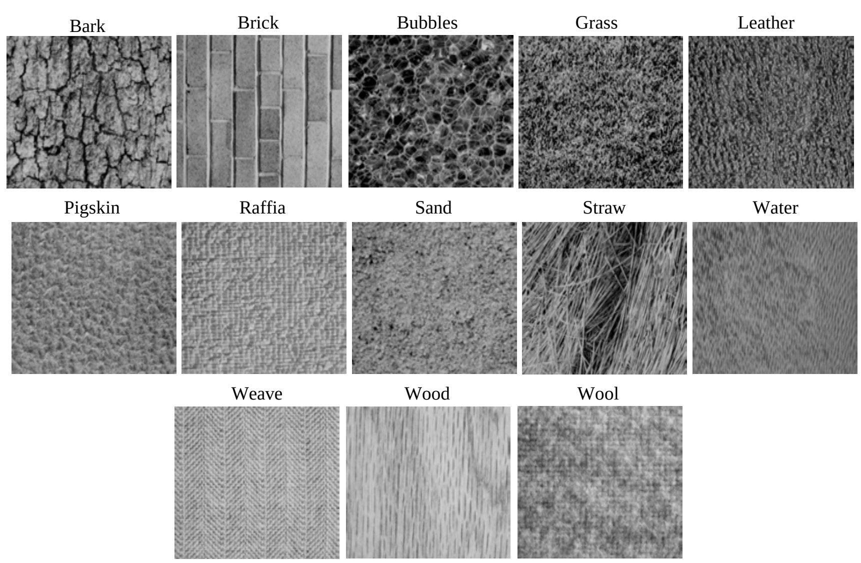

Fig. 6: Thirteen texture classes from brodatz album

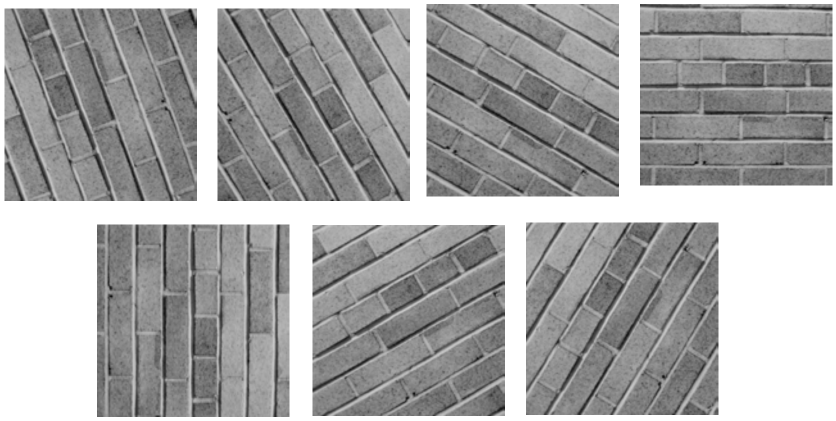

Fig. 7: Seven rotated textures for brick texture (0, 30, 60 and 90, 120, 150, 200)

```
Phase 1:

for i=1:13
   s=0;
   imagess=images{1,i};

   %find the center for each image
   xc=256;
   yc=256;
   A0CenValue(i,1)=imagess(xc,yc);

 %Find the sum of the center 8 neighbors
   A0nie8(i,1)=imagess(xc-1,yc-1);
   s=s+A0nie8(i,1);
   A0nie8(i,2)=imagess(xc-1,yc);
   s=s+A0nie8(i,2);
   A0nie8(i,3)=imagess(xc-1,yc+1);
   s=s+A0nie8(i,3);
   A0nie8(i,4)=imagess(xc,yc-1);
   s=s+A0nie8(i,4);
   A0nie8(i,5)=imagess(xc,yc+1);
   s=s+A0nie8(i,5);
   A0nie8(i,6)=imagess(xc+1,yc-1);
   s=s+A0nie8(i,6);
   A0nie8(i,7)=imagess(xc+1,yc);
   s=s+A0nie8(i,7);
   A0nie8(i,8)=imagess(xc+1,yc+1);
   s=s+A0nie8(i,8);
   A0nie8(i,9)=s;

Phase(1) cont.:

%Find f(n), The Boundary of nested circles
 k=zeros(1,360);
  n=0;
  AllCs=0;
     for r=1:220
        n=n+1;
        for theta=1:360

x0=round(xc+r*cos((theta*pi)/180));
y0=round(yc+r*sin((theta*pi)/180));
 k(theta)=imagess(x0,y0);
        end
        EachC=sum(k);
        AllCs=AllCs+EachC;
     end
  avgs=AllCs/n;
  feature0(i)=avgs;
end
```

Fig. 8a: The implementation of phase (1) for NCB algorithm

to the class called (Brick) which is rotated by the following angles (0, 30, 60 and 90, 120, 150, 200).

Figure 8a shows the implementation and coding of phase (1) as it was developed in MATLAB by using image processing toolbox. Phase (1) reads the 13 texture image and finds the center for each image. Thereafter, the sum of the center 8 neighbors is calculated. Finally, phase 1 finds the f(n) which refers to the boundary of nested circles and stores all these information.

Figure 8b shows the implementation and coding of phase (2) in MATLAB. Phase 2 reads the sample image and finds the pixels where the values of centers for all classes in block 70×70. Thereafter, it finds the 8 neighbors for each pixel and stores the sum of their values. Finally,

```
Phase 2:
for c=1:13

  MaxCount(c,1)=1;
%Read the 70*70 Square
   Sq70=Img(221:291,221:291);
% Find the pixels where the values of center in all
classes
 [x,y]=find(Sq70 == A0CenValue(c));
   s1=length(x);

%find  the 8 neighbors for each pixel, and stor the
sum of its values
  Anewnie8=zeros(s1,9);
  for v=1:s1
     xx(v)=x(v)+220;
     yy(v)=y(v)+220;
  end

   min=100000;
   minp=0;
   count=1;

    for ii=1:s1
centerImg(c,ii)=Img(xx(ii),yy(ii));
Anewnie8(ii,1)=Img(xx(ii)-1,yy(ii)-1);
Anewnie8(ii,2)=Img(xx(ii)-1,yy(ii));
Anewnie8(ii,3)=Img(xx(ii)-1,yy(ii)+1);
Anewnie8(ii,4)=Img(xx(ii),yy(ii)-1);
Anewnie8(ii,5)=Img(xx(ii),yy(ii)+1);
Anewnie8(ii,6)=Img(xx(ii)+1,yy(ii)-1);
Anewnie8(ii,7)=Img(xx(ii)+1,yy(ii));
Anewnie8(ii,8)=Img(xx(ii)+1,yy(ii)+1);
Anewnie8(ii,9)=sum(Anewnie8(ii,1:8));
```

```
Phase 2 cont.:

if(abs(Anewnie8(ii,9)-A0nie8(c,9))<min);
min=abs(Anewnie8(ii,9)-A0nie8(c,9)); minp=ii;
 end
 end
minCen(c,1)=min;
minpCen(c,count)=minp;
    for cc=(minp+1):s1
if(abs(Anewnie8(cc,9)-A0nie8(c,9))==   min);
      count=count+1;
      minpCen(c,count)=cc;
      MaxCount(c)=count;
 end
    end
    for h=1:MaxCount(c)
   Allnewnie8{c,h}=Anewnie8;
    End
for jj=1:MaxCount(c)
    if(minpCen(c,jj)>0)
   Xnewc=xx(minpCen(c,jj));
   Ynewc=yy(minpCen(c,jj));

   nR=0;
   AllCsR=0;
   kR=zeros(1,360);
   for rR=1:220
      nR=nR+1;
       for thetaR=1:360
x0R=round(Xnewc+rR*cos((thetaR*pi)/180));
y0R=round(Ynewc+rR*sin((thetaR*pi)/180));
    kR(thetaR)=Img(x0R,y0R);
      end
      EachCR=sum(kR);
      AllCsR=AllCsR+EachCR;
   end
   avgsR=AllCsR/nR;
   featrenew1(c,jj)=avgsR;
    else
   featrenew1(c,jj)=0;
   end
end

end
```

Fig. 8b: The implementation of phase (2) for NCB algorithm

phase 2 finds f (n) as well as the Boundary of nested circles for the rotated images in order to classify and find the original texture images.

In the initializing phase (phase 1), the input is N which refers to 13 texture classes in this case. In this phase, three steps will be applied:

Initializing:

- **Step 1:** Find the center of each texture class images. Fig. 9 shows that value
- **Step 2:** Calculate the sum of 8 neighbors for each texture class image and store it. Fig. 10 shows the summations of the 8 neighbors for 13 texture class images
- **Step 3:** calculate the f feature using Eq. 1 which presents the average of the sum of the circle boundary pixel values for each texture image and store it. The results of Eq. 1 for original images are shown in Fig. 11

In phase 2, that is sample texture analysis and classification phase, the NCB algorithm is used to find the pixels according to the values of centers in all classes. Afterwards, it is used to calculate the sum of the values of 8 neighbors for each pixel. Table 1 illustrates that result for the sample image (bark. 030) which includes the first 10 result, just to give one example.

In each Sample (S) image, phase 2 search for the original image center value, this illustrated in Fig. 9. Then,

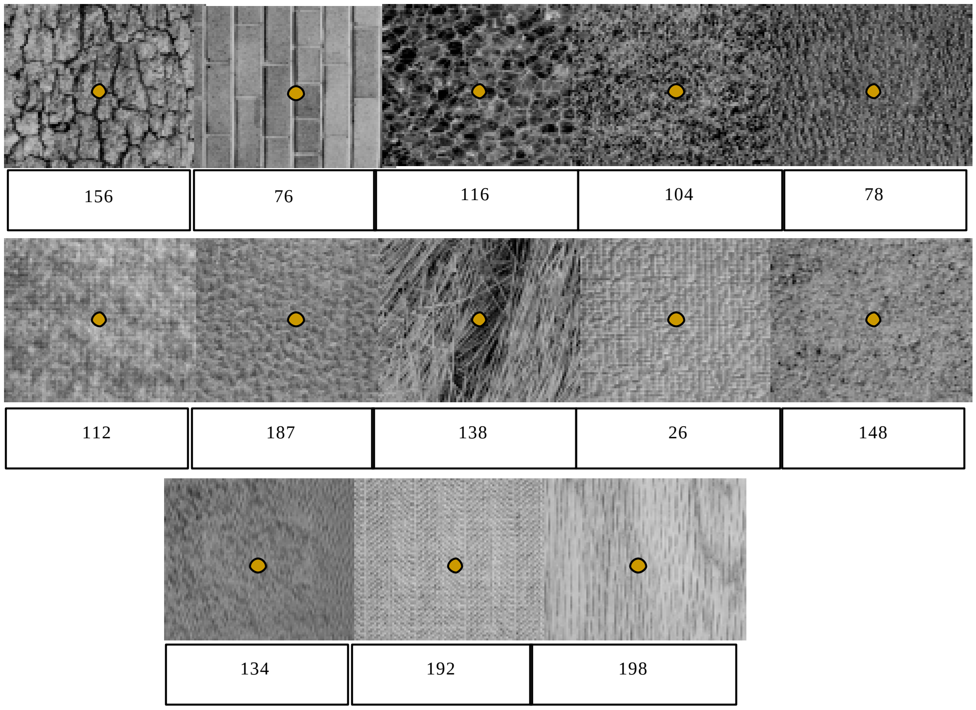

Fig. 9: A center values for each texture class image

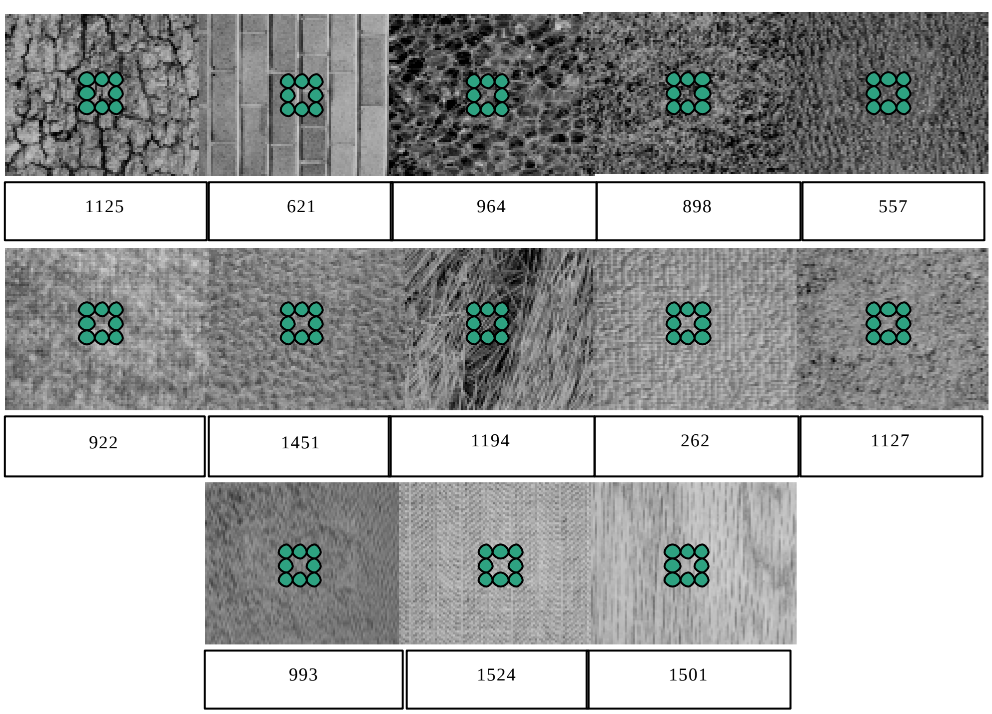

Fig. 10: The sum of the center 8 neighbors' vector values for each texture class image

Table 1: Coordinates of the center values and summation of 8 neighbors

Class No.	Center value	X	Y	Sum (8)
1	156	281	224	1191
		271	228	1119
		283	228	1213
		230	229	1218
		283	230	1276
		272	231	1252
		286	231	1237
		285	232	1241
		289	232	1193
		291	233	1207
2	76	278	221	715
		225	223	630
		237	223	676
		222	226	764
		246	233	678
		288	241	707
		262	252	626
		232	258	754
		290	262	630
		276	266	602
3	116	246	221	983
		251	224	1014
		258	228	979
		258	229	966
		227	233	986
		231	233	989
		251	233	797
		238	234	1005
		236	236	954
		227	237	861
4	104	257	222	943
		290	222	890
		250	223	934
		223	224	681
		247	227	866
		221	228	921
		253	231	813
		224	237	888
		278	239	776
		275	241	832
5	78	239	224	573
		279	239	734
		278	241	775
		275	242	722
		291	252	638
		260	262	632
		276	265	492
		278	268	597
		246	273	722
		255	274	585
6	112	236	222	841
		289	222	1004
		247	224	858
		248	224	894
		258	225	813
		229	226	964
		268	227	981
		224	229	998
		276	229	1008
		254	230	879
7	187	253	227	1428
8	138	240	221	1055
		256	223	1044
		290	223	945
		279	224	1068
		283	225	1129
		290	225	1035
		227	226	1055
		281	226	1102
		232	228	1046
		259	228	1092
9	26	247	245	300
		248	245	349
		252	246	217
		254	248	208
		254	249	241
		258	249	177
		255	250	217
		258	255	280
		254	257	345
		270	258	207
10	148	265	223	1097
		289	225	1183
		279	226	1108
		254	229	1100
		283	229	1260
		259	235	1103
		266	235	1189
		286	236	1183
		285	237	1095
		268	244	1154
11	134	262	224	1036
		223	229	1022
		263	233	1157
		273	236	1040
		284	252	1021
		284	254	1057
		280	255	1116
		237	258	1065
		233	263	1120
		241	275	1056
12	192	----	----	----
13	198	----	----	----

we need to find the X and Y coordinates where value is equal to the center value for the original texture image as shown in Table 1. Then calculate the sum of the values of 8 neighbors for each pixel.

It is also important to note that in some cases, more than one point with values equal to the center value can be found as it shows in class 1 in Table 1. In other cases, the value of the center for a class cannot be found as in class 12 and 13 in Table 1. If the value of the center for an image is missing, then this sample image does not belong to these classes and thus should be excluded.

To determine the center of sample image, the algorithm compares between sum (8) in these tables with the sum of the center of the 8 neighbors for each class as shown in Fig. 10 and choose the closest value.

Table 2 shows the closest value that represents the rotation base point of sample image S. This point will be used as the center of circles to calculate the average of the sum circle boundary pixel values by the Eq. 1. The results are shown in Table 3, by comparing it with the result shown in Fig. 11. Accordingly, the classification of each texture can be carried out.

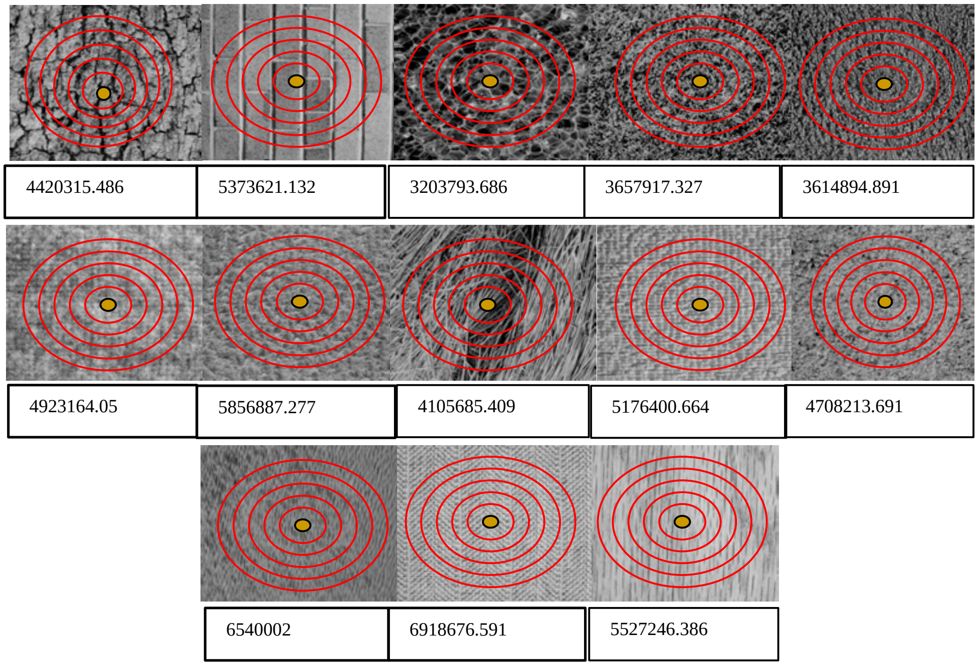

Fig. 11: The average (sum (circle boundary pixel values)) for each texture image

Table 2: Nearest rotation base point of sample image S

Class	1	2	3	4	5	6	7	8	9	10	11	12	13
X	286	262	258	290	236	277	253	264	282	244	284	-	-
Y	245	252	229	222	278	285	227	244	266	276	252	-	-
Sum(8)	1121	626	966	890	545	920	1428	1185	266	1126	1021	-	-

Table 3: The average of the boundary circle for the sample image S

Class	AVG (Boundary)
1	4466235.655
2	4463931.018
3	4488150.164
4	4474635.914
5	4466146.677
6	4466842.7
7	4487226.245
8	4475395.636
9	4448701.259
10	4461069.673
11	4460140.714
12	0
13	0

EXPERIMENTAL RESULTS

Based on the experiment conducted in this research, It is proven that this algorithm is quite successful in rotated texture classification. Table 4 shows the results. The value 1 in the table indicates that the texture sample has been classified correctly to its class. Calculating the sum of each row gives the correct classification results for each rotation textures for every class. As shown in Table 4, the sum of every row is 7 which is the number of rotated textures for every class. The sum of each column gives the number of correct classification results for every rotation angle for all classes. Finally, the sum of all columns or rows gives the total number of correct results for all 91 texture samples. This paper shows the sum is 91 which gives 100% accuracy rate for our proposed algorithm.

Table 4: 91 different rotated texture images testing results

Angle image	0	30	60	90	120	150	200	Result (/91)
Bark	1	1	1	1	1	1	1	7
Brick	1	1	1	1	1	1	1	7
bubbles	1	1	1	1	1	1	1	7
Grass	1	1	1	1	1	1	1	7
Leather	1	1	1	1	1	1	1	7
Pigskin	1	1	1	1	1	1	1	7
Raffia	1	1	1	1	1	1	1	7
Sand	1	1	1	1	1	1	1	7
Straw	1	1	1	1	1	1	1	7
Water	1	1	1	1	1	1	1	7
Weave	1	1	1	1	1	1	1	7
Wood	1	1	1	1	1	1	1	7
wool	1	1	1	1	1	1	1	7
Result (/91)	13	13	13	13	13	13	13	91/91=100%

Based on these results, the proposed algorithm in this paper achieves the accuracy rate compared to previous algorithms. In addition NCB is simple and easy to use as it does not need any extraordinary experiments in image processing.

DISCUSSION OF THE RESULTS

Several research (Fountain *et al.*, 1998; Fountain and Tan, 1998; Fan and Xia, 2003; Materka and Strzelecki, 1998; Singh and Singh, 2002) studied the classification approach of rotated textures and a number of algorithms were developed. The most four popular algorithms in this area are: Gabor filters (Haley and Manjunath, 1995), Edge attribute processing method (Fountain and Tan, 1998), CSAR and Wavelet transformation (Fan and Xia, 2003; Patel and Jameison, 1996). The highest rotated texture classification accuracy achieved by Gabor algorithm is 98.90%.

Haley and Manjunath (1995) applied a modification on Gabor algorithm by using multi resolution filters with 2D Gabor wavelet. The algorithm was tested on the same set of texture images used in this study. This modification has increased the accuracy rate to 99%.

When comparing between the NCB algorithm and the algorithms developed in prior research, differences appear as follow:

The database used in NCB algorithm contains 13 texture images extracted from the Brodatz set, as it used in modified gabor filters. However, the New Signatures algorithm (Zhang and Tan, 2002b) used 20 Brodatz texture classes. Other algorithms such as CSAR, Wavelet decomposition method, Edge attribute processing methods and gabor filtering method used either 10 or 44 Brodatz Textures.

On other hand, all previous algorithms used Model-based methods and employed different methods in texture analyses as in (Jin and Peng, 2006; Loum *et al.*, 2007; Raju *et al.*, 2008). But, NCB algorithm used statistical method that does not need to employ any other methods as filtering methods (Fountain *et al.*, 1998; Stambouli *et al.*, 2010). This is a significant improvement in terms of simplicity.

Finally, the accuracy of CSAR was 60.1%, Wavelet decomposition method achieves 70% accuracy, Edge attribute processing methods enjoys 84% accuracy level, gabor filtering method with a 98.90% accuracy level, the new texture signatures achieves 98.95% and modified gabor filters reaches 99.0% accuracy level. The NCB Algorithm raised the accuracy level of such algorithms to each 100% which the highest one.

It is also important to note that all previous algorithms make assumptions that the texture images are acquired from the same view point, where the NCB algorithm is more dynamic as it finds the base point as appropriate regardless of the position of the texture images.

CONCLUSION AND FUTURE AVENUES

This study proposed a new simple rotated texture algorithm; called Nested Circles Boundary algorithm (NCB). It was tested on 91 rotated texture images for 13 texture classes. NCB achieved higher effectiveness and the best accuracy rate over all previous algorithms. In addition, it has advantages of ease and high computational speed compared with others. Note that, NCB can be used not only for rotated texture but also for general texture classifications. On the future NCB could be apply on different texture image scales and it could be modified to solve skew texture images by using nested ovals instead of nested circles.

REFERENCES

Al-Bashish, D., M. Braik and S. Bani-Ahmad, 2011. Detection and classification of leaf diseases using K-means-based segmentation and neural-networks-based classification. Inform. Technol. J., 10: 267-275.

Amirani, M.C., Z.S. Gol and A.A.B. Shirazi, 2008. Efficient feature extraction for shape-based image retrieval. J. Applied Sci., 8: 2378-2386.

Bovik, A.C., M. Clark and W. S. Geisler, 1990. Multichannel texture analysis using localized spatial filters. IEEE Trans. Pattern Anal. Machine Intel., 12: 55-73.

Brodatz, P., 1966. Textures: A Photographic Album for Artists and Designers. 1st Edn., Dover, New York.

Chen, C.H., L.F. Pau and P.S.P. Wang, 1998. Pattern Recognition and Computer Vision. World Scientific Publishing Co., Singapore, pp: 207-248.

Fan, G. and X. Xia, 2003. Wavelet-based texture analysis and synthesis using hidden markov models. IEEE Trans. Circuits Syst., 50: 106-120.

Fountain, S.R. and T.N. Tan and K.D. Baker, 1998. A comparative study of rotation invariant classification and retrieval of texture images. BMVC, 1: 266-275.

Fountain, S.R. and T.N. Tan, 1998. Efficient rotation invariant texture features for content based image retrieval. Pattern Recognit., 31: 1725-1732.

Gonzalez, R.C. and R.E. Woods, 2002. Digital Image Processing. Prentice Hall, New Jersey.

Gonzalez, R.C., R.E. Woods and S.L. Eddins, 2006. Digital image processing using. ATLAB. http://www.mathworks.com/products/image.

Haley, G.M. and B.S. Manjunath, 1995. Rotation invariant texture classification using modified gabor filters. Proc. Int. Conf. IEEE, 1: 262-266.

Haralick, R.M., K. Shanmugam and I.H. Dinstein, 1973. Texture features for image classification. IEEE Trans. Syst. Man Cybernetics, 3: 610-621.

Jain, A.K. and K. Karu, 1996. Learning texture discrimination masks. IEEE Trans. Pattern Anal. Mach. Intel., 18: 195-205.

Jiahua, W., 2003. Rotation invariant classification of 3D surface texture using photometric stereo. Ph.D. Thesis, Department of Computer Science, School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK.

Jin, C. and J. Peng, 2006. A robust wavelet-based blind digital watermarking algorithm. Inform. Technol. J., 5: 358-363.

Khan, M.A.U., M.K. Khan, M.A. Khan and M.T. Ibrahim, 2007. Endothelial cell image enhancement using non-subsampled image pyramid. Inform. Technol. J., 6: 1057-1062.

Loum, G., C.T. Haba, J. Lemoine and P. Provent, 2007. Texture characterisation and classification using full wavelet decomposition. J. Applied Sci., 7: 1566-1573.

Materka, A. and M. Strzelecki, 1998. Texture analysis methods - a review. Technical Report, Technical University of Lodz, Institute of Electronics.

Patel, J.N. and L.H. Jameison, 1996. Scalability of 2-d wavelet transform algorithms: Analytical and experimental results on coarse grained parallel computers. Proceedings of the 9th IEEE Workshop on VLSI Signal Processing, 30 Oct- Nov. 1, San Franciso, USA., pp: 378-387.

Picard, R.W., T. Kabir and F. Liu, 1993. Real-time recognition with the entire Brodatz texture database. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Jun. 15-17, New York, USA., pp: 638-639.

Poonguzhali, S. and G. Ravindran, 2008. Automatic classification of facal lesions in ultrasound liver images. Int. J. Computer. Sci., 7: 205-209.

Portilla, J. and E. Simoncelli, 2006. Representation and synthesis of visual texture. http://www.cns.nyu.edu/~lcv/texture/.

Pratt, W.K., O.D. Faugeras and A. Gagalowicz, 1978. Visual discrimination of stochastic texture fields. IEEE Trans. Syst. Man Cybernetics, 8: 796-804.

Raju, U.S.N., B.E. Reddy, V.V. Kumar and B. Sujatha, 2008. Texture classification based on extraction of skeleton primitives using wavelets. Inform. Technol. J., 7: 883-889.

Singh, M. and S. Singh, 2002. Spatial texture analysis: A comparative study. Proc. Int. Conf. Pattern Recognition, 1: 676-679.

Stambouli, T.B., M. Keche and A. Ouamri, 2010. Iterative feature selection for classification. J. Applied Sci., 10: 1015-1018.

Zhang, J. and T. Tan, 2002a. New textures Signatures and their use in rotation invariant texture classification. Chinese Academy of Sciences, Beijing, China. http://lear.inrialpes.fr/people/zhang/ECCV_paper.pdf.

Zhang, J. and T. Tan, 2002b. Brief review of invariant texture analysis methods. Pattern Recog., 35: 735-747.

Zhang, M., J. Shen and Z. Shang, 2008. A novel unsupervised color image fuzzy classification algorithm with salient regions. Inform. Technol. J., 7: 560-569.