Adaptive Gray Level Co-occurrence Probabilities for Texture Segmentation Based on Coarse-to-fine Strategy

Yao Hong Tsai*

Received 12 September 2014; Published online 11 October 2014

© The author(s) 2014. Published with open access at www.uscip.us

Abstract

In the applications of image segmentation, the tasks usually focused on extracting the exact boundaries among different texture classes. Traditionally, the popular texture feature extraction approach, gray level co-occurrence probability (GLCP) method, is widely used for extracting the boundaries. It scans the whole image by windows with fixed size to count the gray level co-occurrence and uses statistics to calculate the texture characteristics at each window. However, it has a tendency to misclassify and erode texture boundaries, especially for large window sizes and irregular texture boundaries in images. This paper presents the adaptive gray level co-occurrence probability (AGLCP) approach to improve the original GLCP method for the analysis of the image textures. Based on the coarse-to-fine strategy, the AGLCP method can adjust the scanning window and analyze texture features, adaptively. Therefore, it can generate a more suitable statistic characteristic matrix for further image segmentation. Experimental results show that this method can preserve the edge strength between textures and provide better segmentations when comparing with the previous methods.

Keywords: Texture analysis; Adaptive window; Gray level co-occurrence probability; Image segmentation

1. Introduction

In pattern recognition and computer vision areas, texture is considered as a significant property of visual content. In the literatures, applications using textures are found in various areas including aiding diagnoses (Wu et al., 1992), remote sensing (Yang, 1998), analysis of geo-logical structures (Heidelbach et al., 2000), meningioma classification (Al-Kadi, 2010) and so on. As a result, texture analysis plays an important role in many applications for classification or segmentation of images. A commonly used strategy for texture segmentation is first to extract texture features on a pixel-by-pixel process from a texture image and then perform a clustering technique on the extracted texture features. Many research efforts have been focused on texture analysis in recent years.

Corresponding e-mail: tyh@hcu.edu.tw
1* Department of Information Management, Hsuan Chuang University, Hsinchu 300, Taiwan
According to the results of Tuceryan and Jain (1999), the various texture analysis methods can be categorized into four main groups: statistical, geometrical, model-based and signal-based approach.

For texture-based image segmentation, pixels are grouped together to form regions of uniform texture based on the distribution of local features. For statistical methods, texture features are extracted from the local area of pixels and represented by a set of statistics. Among those various statistical techniques for texture analysis, the gray level co-occurrence probability (GLCP) method (Haralick et al., 1973) is considered for its ability to perform texture image segmentation. It is a second-order statistical method for texture analysis. Based on the spatial relationship of pixels, the GLCP method describes the conditional probability of two gray levels in a given image window. Afterwards, there are also some variations of the GLCP method. Solberg and Jain (1997) fused co-occurrence probability features, local statistics features, fractal features and random field features for image classification. Because of the great ability, the GLCP methods are still widely studied and applied to many recent results (Gupta et al., 2012; Guo and Ping, 2012; Wang et al., 2012).

When using the GLCP method, many parameters should be set to generate texture features in images. The window size is one of the most important parameters. Large window sizes are necessary to gather sufficient data to characterize local texture regions. However, it will tend to misclassify and erode texture boundaries when extracting features by large windows in images with irregularly shaped texture boundaries (Jobanputra and Clausi, 2006; Baraldi and Parmiggiani, 1995). On the contrary, it might be deficient to describe the texture features of the input image when using small windows.

Jobanputra and Clausi (2006) presented a modification of the GLCP technique which successfully improves the accuracy of the boundary extraction for image segmentation applications. They named it as the weighted gray level co-occurring probability (WGLCP) method. For the original GLCP method, all pixel pairs in the image window are given a uniform probability weighting, such that pixel pairs far from the center of the window will have the same impact to the feature measurement as those close to the center. A Gaussian weighting scheme was proposed in WGLCP for calculating the co-occurring probabilities in the given window. Pixel pairs closer to the center of the image window are given a higher probability than those on the outlying edges. They applied WGLCP method on solving the segmentation problem of sea ice image from synthetic aperture radar. They also provided experimental results on comparing the texture features generated by WGLCP method to the GLCP features with respect to their boundary preservation and segmentation ability.

In this paper, a novel adaptive gray level co-occurrence probability (AGLCP) algorithm is proposed for analyzing the image textures. Since a window with fixed size is not suitable for the GLCP feature extraction on all texture images, the proposed algorithm adjusts the window size from large to small adaptively and analyzes the subimage in the window based on the GLCP method. The algorithm relies on a coarse-to-fine approach that involves first examining the texture by large windows, and then adaptively refining the analysis and approximation in regions of interest (ROIs) by smaller windows. For each adjustment, the ROI is also narrowed down to improve the accuracy and reduce the computation time, simultaneously. Experimentations show that the proposed method had better results on boundary preservation and segmentation ability when comparing the texture features generated by AGLCP method to the GLCP and WGLCP method.
This paper is arranged in the following manner. In Section 2, an introduction of the GLCP method is given. A complete formulation of the AGLCP texture features and implementation are described in Section 3. Section 4 provides experimental results for the comparison of the GLCP, WGLCP and AGLCP texture features. Section 5 summarizes and concludes the paper.

2. References

Gray Level Co-occurrence Probability (GLCP) Texture Features

The GLCP method proposed by Haralick et al. (1973) is a common method for texture feature extraction. It considers the orientation and distance between image pixels and provides a second-order approach for generating texture features from an input image. This method finds the conditional joint probabilities of all pair-wise combinations of gray levels in a given window within the image. Because of the real processing requirements, it is prohibitive to calculate the probabilities for the full dynamic range of an image. Suppose $G$ is the number of quantized gray levels. By default, we reduce the number of intensity values in a grayscale image from 256 to $G$. The set of the probabilities can be defined as:

$$C_{ij} = \frac{P_{ij}}{\sum_{i,j=0}^{G-1} P_{ij}}$$

(1)

where $P_{ij}$ represents how often a pixel with the gray level value $i$ occurs in a specific spatial relationship to a pixel with the value $j$ within the specified window. The probabilities are usually stored in a matrix with size of $G \times G$, namely gray level co-occurrence matrix (GLCM). Each element in location $(i, j)$ of the matrix represents the probability $C_{ij}$. After the GLCM is obtained, statistics are applied to it to generate texture features and assigned to the position of a GLCP feature matrix corresponding to the center of the image window. Finally, the GLCP feature matrix from the input image reveals the ability for texture segmentations. Several commonly used gray level shift invariant statistics are listed in Table 1 (Haralick et al., 1973; Jobanputra and Clausi, 2006; Baraldi and Parmiggiani, 1995; Barber and LeDrew, 1991). In addition, Baraldi and Parmiggiani (1995) recommend that entropy and contrast are the most significant texture statistics.

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum probability</td>
<td>$\max(C_{ij})$ for all $(i,j)$</td>
</tr>
<tr>
<td>Entropy</td>
<td>$- \sum_i C_{ij} \log C_{ij}$</td>
</tr>
<tr>
<td>Dissimilarity</td>
<td>$\sum_i C_{ij}</td>
</tr>
<tr>
<td>Contrast</td>
<td>$\sum_i C_{ij} (i-j)^2$</td>
</tr>
<tr>
<td>Correlation</td>
<td>$\sum_i \frac{(i-\mu_i)(j-\mu_j)C_{ij}}{\sigma_i \sigma_j}$</td>
</tr>
</tbody>
</table>
The extraction of GLCP texture features requires some parameters setting. Some suggestions for parameters setting were provided in previous results (Jobanputra and Clausi, 2006; Soh and Tsatsoulis, 1999; Zucker and Terzopoulos, 1980; Tsai and Leu, 2008). For the quantized gray levels $G$, 32 is sufficient. The spatial relationship, offset, is defined as the pixel of interest and the pixel to its adjacent pixels. Typically, four common orientations, $0^\circ$, $45^\circ$, $90^\circ$ and $135^\circ$, are used as they are easy to calculate. These offsets can be specified as a two-element vector, $(row\_offset, col\_offset)$, such that they are denoted by $(1, 0), (1, 1), (0, 1)$, and $(-1, 1)$.

A $5\times5$ image window as shown in Fig. 1(a) is used to illustrate the extraction of the GLCP texture features. For convenience, let $G=4$ and then the quantized result is shown in Fig. 1(b). All of the elements in the matrix belong to the set $\{0, 1, 2, 3\}$. Fig. 1(c) represents the frequency of occurrence $P_{ij}$ in the horizontal direction. For example, the element at the location $(3, 1)$ in Fig. 1(c) is 4 since there are four pairs of adjacent pixels with the gray values 3 and 1 in the orientation $0^\circ$, which are denoted by red rectangles in Fig. 1(b). Finally, the GLCM in Fig. 1(d) is calculated by Eq. (1) and statistics listed in Table 1 are applied to it to generate texture features and assigned to the center of the image window.

![Fig. 1](image_url)

Fig. 1. (a) A $5\times5$ image window; (b) the corresponding quantized result; (c) the frequency of occurrence $P_{ij}$; (d) the GLCM.

However, the size of the given window is very important for GLCP texture features generation. Small window size will cause poorly sampled co-occurring probabilities and lead to an inconsistent estimate of the individual texture features (Jobanputra and Clausi, 2006). Larger windows can be
used to capture lower frequency information, but these same window sizes will capture all frequencies at the same time not just the lower frequencies. As a result, if the window size is large, it will gather more data to characterize the texture features but likely erode the class boundaries. Both of them will result in poorer segmentations. Thus, an adaptive window scheme for improving the original method of GLCP is necessary for real applications.

3. Adaptive Gray Level Co-occurrence Probability Method

In the previous work of Jobanputra and Clausi (2006), they defined the boundary transect as the profile view of a texture feature when the image window moves across a texture boundary. Boundary transect is also used to describe the concept of the AGLCP algorithm. The bipartite image and the corresponding boundary transition of row 50 are shown in Fig. 2(a) and Fig. 2(b), respectively. For each column in the given row of Fig. 2(a), calculate the GLCP statistics and plot the feature value versus image column position in Fig. 2(b). The value of feature response will ideally have a step-like transition as it crosses the texture boundary if those chosen parameters for GLCP are appropriate. Consequently, optimal image segmentation would be possible by the extracted texture features. The vertical line at column 100 in Fig. 2(b) represents the true boundary corresponding to Fig. 2(a).

![Fig. 2. (a) The bipartite image with different textures: paper and ridge; (b) the corresponding boundary transect of row 50 in (a).](image)

Fig. 3 illustrates the different boundary transects under different window sizes. Large window of size $27 \times 27$ capture lower frequency information in the image such that it will result in a smooth boundary transition and also a gradual step from column 85 to column 117 as it cross the texture boundary. The result is shown in Fig. 3(a). Based on these features, it will probably erode the class boundaries in image segmentations. On the contrary, small window of size $9 \times 9$ produce many sharper transect responses in Fig. 3(b) such that it might be difficult to describe the difference between the exact boundary and the others.
Based on the above considerations, the proposed algorithm first applies larger windows to capture lower frequency information. It helps to enlarge the variation among feature responses near the true boundary but depress that for the other area. Then, it adaptively reduces the window size and performs the same operations only in the corresponding ROI, which contains the true boundary. The region between two vertical lines in Fig. 4 shows an example of ROI for further processing to have an ideal step-like transition. For real applications, there might be some ROIs when it scans the whole image by windows on a specific row or column. For locating those ROIs, the original boundary transition is first approximated by line segments in the following subsection. The goal of the line approximation is to eliminate those small variances of feature response in the boundary transect.

Fig. 4. The example of the ROI from Fig. 3(a).

3.1 Line Segments Approximation

Algorithms for line segments approximation have been developed over the years for the purpose of deleting redundant or unnecessary coordinate information from line features, while retaining the perceptual characteristics of the line. They generally work by some geometric criterion to the line's
coordinate pairs, such as the perpendicular distance from a line segment. A well-known algorithm for line simplification is the classical Douglas-Peucker (D-P) algorithm (Douglas and Peucker, 1973). Given $N$ initial points, it takes $O(N \log N)$ time at best and $O(N^2)$ at worst and gives the closest line approximation to the points a human being would choose to simplify a line. Afterwards, improvements on the original algorithm have been written by Hershberger and Snoeyink (1992). More recently, it was extended to three dimensional problems (Fei et al., 2006). D-P algorithm can be used to simplify the generalization of contour lines during the compilation of maps at smaller scales.

Given a set, $P = \{p_0, p_1, ..., p_{N-1}\}$, of points from the original boundary transition and a tolerance $\varepsilon$ for distance. This algorithm starts with an initial guess that is the single line segment joining the first and last points. Then the remaining points are tested for closeness to the line segment. It is determined by the perpendicular distances from all points to the line segment. If all these distances are less than $\varepsilon$, the approximation is accepted and all the other points are eliminated except the endpoints. However, if any of these distances exceeds $\varepsilon$, we choose the point that is furthest away as a new endpoint subdividing the original set $P$ into two subsets. This procedure is repeated recursively on these two subsets. The line approximation of the boundary transition from Fig. 3(a) is shown in Fig. 5. It is observed that the simplified boundary transect after the line approximation only reflects lower frequency of texture information. It would be helpful for locating each ROI.

![Fig. 5. Line segments approximation of the boundary transition.](image)

### 3.2 Locating The ROIs

Line approximation transforms the original problem for locating ROIs into finding the line segments that satisfy the following criteria. Since end points of a selected line segment describe the range of the corresponding ROI, the slope and projected length on the $y$ coordinate of a line segment will point out the likelihood of it being an ROI or not. Then, the simple measure $\eta$ combining the above two features is used to evaluate each line segment. Given $S=\{S_i \mid 0\leq i \leq n-1\}$, where $n$ is the total number of line segments.
\[ \eta(S_v) = |\Delta y \times \alpha| + \tan^{-1}(\frac{\Delta y}{\Delta x}) \]  

(2)

where \( \Delta x (\Delta y) \) represents the difference of \( x \) coordinates (\( y \) coordinates) for the end points of \( S_v \) respectively. \( \alpha \) is a constant to rescale the \( y \) coordinates. Using the above formula, the algorithm computes \( \eta \) of each line segment and finds those which are large enough with a user-defined threshold.

However, the choice for the threshold of the measurement \( \eta \) is very difficult under the consideration of real applications. Fig. 6 is an example to illustrate this kind of situations. The input image is of size 1024\( \times \)768. A large window with size of 27\( \times \)27 is used to capture lower frequency information and find possible ROIs. The corresponding boundary transect of row 384 in Fig. 6(a) is shown in Fig. 6(b). The foreground object in the input image is the head of the giraffe. For the boundary transect in Fig. 6(b), the giraffe is approximately between column 350 and 800, which are denoted by red lines. It is noted that we should focus on the ROI before column 350 and another one after column 800 to get the true boundary of the giraffe’s head. Ideally, all the local maximums of the boundary transect between column 350 and 800 should be ignored. How to set a suitable threshold for the measurement \( \eta \) turned out to be a thorny problem. Therefore, an optimization technique should be applied here. Simulated annealing (Kirkpatrick et al., 1983) was used to explain the optimization process to avoid setting threshold case by case. In fact, other optimization methods (Fletcher, 2000) can also be applied to solve this problem.

The simulated annealing (Kirkpatrick et al., 1983; Jeng and Woods, 1990) is a well know stochastic method for optimization problems. Since it does not require any derivative information and specific conditions on the objective function, it is one of the most used algorithms in solving optimization problems. For image processing applications, it had been used to solve the problem like image restoration for degraded images by Geman and Geman (1984). Applying the algorithm of simulated

![Fig. 6. (a) The input image with size of 1024\( \times \)768; (b) the corresponding boundary transect of row 384 in (a).](image-url)
annealing, the proposed method can efficiently find the large enough $\eta$ in those approximated line segments instead of using a human-selected threshold.

The concept of simulated annealing is based on the manner in which liquids freeze in the process of annealing. The annealing process, initially at high temperature and disordered, is slowly cooled down so that the system at any time is approximately in thermodynamic equilibrium. The initial state of a thermodynamic system is chosen at the energy $E$. In this paper, $E$ is set to be the value $\eta$. Then the change in energy, $\Delta E$, is obtained from the difference of $\eta$ between $S_i$ and $S_{i+1}$. For that matter, the process accepts a new candidate whenever an increase of the energy is verified. Otherwise, it is accepted with a probability $p$. The acceptance criterion has the following mathematical form:

$$C_{j+1} = \begin{cases} 
\eta(S_{j+1}) & \text{if } u \leq p, \\
C_j & \text{otherwise}, 
\end{cases} \quad (3)$$

where $C_j$ is the current approximation to the local maximum, $\eta(S_{j+1})$ is the new candidate, and $u$ is a random number drawn from uniform distribution $U(0,1)$. The acceptance criterion allows the algorithm to avoid getting stuck in some local solutions. The probability $p$ of accepting $\eta(S_{j+1})$ depends on the positive control parameter $t$ and the difference of the energy $\Delta E$. It can be computed by the following function:

$$p = \min \{1, \exp(-\Delta E / t)\}, \quad t > 0. \quad (4)$$

The control parameter $t$, is also known as the temperature in the cooling schedule, must be updated in order to define a positive decreasing sequence. The temperature $t$ after each cooling process is obtained by the following equation with a parameter $\gamma$, where $t'$ is the previous temperature and $\gamma$ is usually set to 0.9 in the experimentations.

$$t = \gamma t', \quad 0 \leq \gamma \leq 1 \quad (5)$$

When $t$ is high, the process searches in the whole feasible set. Actually, the slower the cooling schedule, or rate of decrease, the more likely the algorithm is to find an optimal or near-optimal solution. Toward the end of the process, the temperature or probability of accepting a worse solution is nearly zero. The stopping criteria for the simulated annealing are based on the idea that the loop should terminate when no further changes occur. The algorithm for locating ROIs is listed in the following.

\textbf{Algorithm 1} \\
\hspace{1em} Given a set of line segments, $S$ \\
\hspace{1em} Set $j=0$ \\
\hspace{1em} \textbf{while} the end of $S$ is not reached \textbf{do} \\
\hspace{2em} Given an initial temperature $t$ \\
\hspace{2em} $C_j = \eta(S_j)$
while stopping criterion is not reached do
    Generate a new feasible candidate, $\eta(S_{j+1})$
    Let $\Delta E = \eta(S_{j+1}) - C_j$
    If $\Delta E \geq 0$, set $C_{j+1} = \eta(S_{j+1})$
    If $\Delta E < 0$, set $C_{j+1} = \eta(S_{j+1})$ with probability $p$ or $C_{j+1} = C_j$
    Reduce temperature $t$
    Set $j=j+1$
end while

Output the line segment corresponding to $C_j$
end while

The algorithm consists of two while loops. The outer loop controls the total set of line segments and initializes each process of maximum finding, i.e., the inner loop. The inner loop works based on the concept of simulated annealing described above. It outputs a line segment with a considerable $\eta$ and locates the current ROI each time when the inner loop stops. After algorithm 1 stops, it outputs ROIs for the coarse window.

3.3 The AGLCP Algorithm Based on The Coarse-to-fine Strategy

The proposed algorithm applies the coarse-to-fine strategy to generate an ideally step-like transition close to each true boundary by adjusting the window size and ROI, adaptively. First, a scanning window of large size is used to examine the textures and produce a coarse GLCP feature matrix. Initially, the ROI is the whole image. Line approximation and simulated annealing help to find those new ROIs in the coarse GLCP feature matrix on each row and column. For the next refinement, the window size is reduced by $K$ to generate new GLCP features only in each new ROI. Then, the smaller window is applied to the corresponding subimage of the ROI to recalculate the GLCP texture features. For each time, the new ROI is also narrowed down in the coarse-to-fine process. This procedure is repeated until there is no further change of the ROI or the smallest window size is achieved. The range of window size varies from $27 \times 27$ to $9 \times 9$ and $K$ is set to be 2 in the experimentations.

![Fig. 7. Two examples of the ROI.](image-url)
Fig. 7(a) is the temporary result of the algorithm with the window of size 15×15 in the ROI from Fig. 3(a). Similarly, Fig. 7(b) is the result with the window of size 9×9 from the ROI in Fig. 7(a) after three iterations. It is quite obvious that the algorithm successfully narrows the ROI toward the true boundary and produce a shaper transect response. The algorithm for the coarse-to-fine strategy is listed in the following. After that, a new GLCM from adaptive gray level co-occurrence probabilities is obtained.

**Algorithm 2**

- Initialization: parameters selection
- Gray level quantization
- Calculate the GLCP
- Calculate statistical features to form the GLCM
- Locate ROIs by **Algorithm 1**
- For each ROI
  - Reduce the window size
  - Calculate GLCP
  - Calculate statistical features to update the GLCM
  - Locate new ROI by line segment approximation and maximum η finding

While the result is satisfied or minimum window size is reached

4. Experimentations

Our experimental platform was a personal computer with an Intel(R) Core(TM) i7 CPU 870 @ 2.93GHz and 2GB RAM running Windows 7 and Matlab of version R2009b. In the experiments, parameters setting for the feature extraction are described in Table 2. Under the same parameters, the GLCP, WGLCP and AGLCP texture features were calculated and scaled using linear normalization for all test sets. In addition, the GLCP and WGLCP method adopted a fixed window size of 19×19. The features were scaled to improve segmentation by providing a consistent resolution along all dimensions of the features space. From the resulting feature matrix, K-means clustering (Rafael & Richard, 2004) was applied for further segmentations.

**Table 2** Common parameters used in feature extraction for GLCP, WGLCP, and AGLCP.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>32</td>
</tr>
<tr>
<td>(row_offset, col_offset)</td>
<td>(1, 0), (1, 1), (0, 1), (-1, 1)</td>
</tr>
<tr>
<td>Statistic</td>
<td>entropy</td>
</tr>
</tbody>
</table>

4.1 Experimentations on Bipartite Texture Images

The first image set consists of 111 images from the Brodatz texture images database which can be accessed at [http://www.uxuis.no/~tranden/brodatz.html](http://www.uxuis.no/~tranden/brodatz.html) and 160 images from the Imageafter image database at [http://www.imageafter.com/](http://www.imageafter.com/). The website of Brodatz texture images contains image D1 to D112 but without D14. Since there are too many texture images in the Imageafter image database, only the first 8 images in each category were collected to be the testing images. The size of each texture image in the first image set was normalized to 256×256. Some small samples
cropped from the center of original images are shown in Fig. 8. All pair-wise combinations of images were chosen from the dataset to form a bipartite image with a vertical boundary. All bipartite images generated from the dataset were used to test the accuracy of boundary detection by GLCP, WGLCP, and the proposed AGLCP. Ideally, the boundary line between two texture images generated by the testing algorithms should be as close to the true boundary as possible.

**Fig. 8.** Some images of the first image set.

Six segmentation results from the first image set are illustrated in Fig. 9. The original testing images are shown in the leftmost column. They correspond to the bipartite images from the Brodatz texture images D19-D16, D21-D22 and D92-D93 in the first three rows and texture images paper-ridge, wool-leather and ice-carpet in the next three rows, respectively.

**Fig. 9.** Segmentation results of six bipartite images. From the leftmost column, they correspond to the (a) input images, (b) result images by GLCP, (c) WGLCP, and the (d) proposed AGLCP texture features.
Segmentation results of the GLCP, WGLCP and the proposed AGLCP are shown in column (b), (c) and (d), respectively. The white lines on the resulting images represent the detected texture boundary of two kinds of textures. In general, it is demonstrated that the AGLCP method improves the segmentations when comparing to the GLCP and WGLCP method from these results. In fact, there is serious boundary misclassification in the results for the testing image D92-D93, since the texture in D93 is calf fur and it has large variance itself. In this case, AGLCP got better result even though the testing image has large variance.

For further accuracy evaluations, the misclassification error (ME) method (Yasnoff et al., 1977) is adopted to be a numerical measurement. ME is a useful index for quantifying the percentage of background pixels wrongly assigned to objects, and conversely, the pixels of objects wrongly assigned to background. For the two-class segmentation problem, ME can be expressed as:

\[
ME = 1 - \frac{|R_G \cap R_S| + |L_G \cap L_S|}{|L_G| + |R_G|} 
\]

where \(R_G\) and \(L_G\) denote the right and left texture image of the ground-truth image, \(R_S\) and \(L_S\) denote the right and left texture image of the segmentation results, respectively. The symbols of \(\cap\) is the intersection between two sets and \(|.|\) is the number of pixels of the resulting set. The ME is 0 for a perfectly classified image. The corresponding ground-truth image of each testing image was generated by setting the center line as the boundary. Table 3 shows the evaluation of segmentations on the first image set according to the performance criteria, ME. It is observed that the average ME of AGLCP results is smaller than that of the original GLCP and WGLCP results. The proposed method got the smallest ME than other existing methods since it can adaptively approach the true boundary. Comparing with the original GLCP method, the proposed method improves 33.8% in ME.

### Table 3 Evaluation of the performance criteria, ME, on the first image set.

<table>
<thead>
<tr>
<th>Method</th>
<th>GLCP</th>
<th>WGLCP</th>
<th>AGLCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average ME</td>
<td>0.382</td>
<td>0.335</td>
<td>0.253</td>
</tr>
<tr>
<td>ME decreasing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rate from GLCP</td>
<td>12.3%</td>
<td>33.8%</td>
<td></td>
</tr>
</tbody>
</table>

4.2 Experimentations on Natural Images

The second image set containing 250 images was randomly collected from the web and used to test the performance of image segmentation on natural images. Some images of the second set are shown in Fig. 10. Each image of size 320×240 was captured from the natural scene to present the foreground object and the background with different textures. The corresponding ground-truth image of each testing image was generated by manually segmentation. For the simplicity of the comparison between different methods, the experimentation only illustrates the results for two-class segmentation.
Six segmentation results of the proposed method and the previous methods are illustrated in Fig. 11. It only shows a small portion of all results for the limitation of the paper. Fig. 11(a) corresponds to trees, clouds, tiger, smoke, fishes, and coral image from the top row to the bottom respectively. Each image was segmented to be the foreground and the background according to their textures. In column (b), the red lines overlaid in the images represent the true boundaries by manually segmentation. The white lines on the resulting images in column (c), (d) and (e) represent the detected texture boundary of two kinds of textures by using the GLCP, WGLCP and AGLCP method.

**Fig. 10.** Some images of the second image set.

**Fig. 11.** Segmentation results of six sample images. From the leftmost column, they correspond to the (a) input images, (b) result images by manually segmentation, (c) GLCP, (d) WGLCP, and the (e) proposed AGLCP texture features.
From these results, it is demonstrated that the AGLCP method improves the segmentations when comparing to the GLCP and WGLCP method, although there are some misclassifications for the dark clouds on the left side of the clouds image. Since the texture of the dark clouds is more similar to that of the sky than the white clouds in the image, GLCP based methods will tend to classify them into the background for two-class segmentation. According to the texture analysis, AGLCP obtained the boundary line which is closer to the boundary between white and dark clouds. In general, AGLCP seems to have some improvements even though the input image has complex background. Table 4 shows the evaluation of segmentation according to the performance criteria, ME. It is observed that the average ME of AGLCP result is smaller than the original GLCP and WGLCP results. The proposed method got the smallest ME than other existing methods since it can automatically adjust the window size for texture analysis and refine the ROI. Comparing with the original GLCP method, the proposed method improves 36.4% in ME. From these result, it is showed that adaptive window selection really improves segmentation results by GLCP based method.

Table 4 Evaluation of the performance criteria, ME.

<table>
<thead>
<tr>
<th>Method</th>
<th>GLCP</th>
<th>WGLCP</th>
<th>AGLCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average ME</td>
<td>0.140</td>
<td>0.125</td>
<td>0.089</td>
</tr>
<tr>
<td>ME decreasing rate from GLCP</td>
<td>10.7%</td>
<td>36.4%</td>
<td></td>
</tr>
</tbody>
</table>

4.3 Computation Time

Since the coarse-to-fine strategy is applied for adaptively locating the texture boundary, the AGLCP method is slower than the original GLCP and WGLCP method in computation time. Since the ROI in each fine process is very small compared with the original one in coarse process, it takes relative little computation time for each fine process. For an image of size 320×240, the AGLCP (WGLCP, GLCP) method takes approximate 36 (28, 20) minutes in average to calculate the texture features according to the parameter setting in the experimentation, respectively. In fact, the exact computation time heavily depends on the image content. If there is only simple object in the image and the background has extremely different textures, it will need low extra computation time for locating ROIs. Furthermore, users can also reduce the initial window size for extracting texture features in coarse-to-fine process if the computation time is a very important consideration. Recently, the parallel implementations (Shahbahrami et al., 2012) using 16 Synergistic Processor Elements significantly reduce the computational times of the GLCMs and texture features extraction algorithms by a factor of 10 over non-parallel optimized implementations for different image sizes from 128×128 to 1024×1024. Nevertheless, with the advancements of computer hardware, this computational burden will continue to be reduced in the future.

5. Conclusions

The AGLCP algorithm is proposed to calculate the texture characteristics and segment different objects from the input images. Based on the adaptive window scheme, the proposed method might exactly analyze the image textures by coarse-to-fine strategy. That could preserve the edge strength between textures of different regions and provide better image segmentations than the previous
methods, GLCP and WGLCP, under the same parameter setting. In fact, the proposed algorithm could also be applied to other GLCP based texture analysis methods as well.

Acknowledgements

This work was supported in part by a grant from National Science Council MOST 103-2221-E-364-003 and Hsuan Chuang University.

References


http://dx.doi.org/10.1126/science.220.4598.671

http://dx.doi.org/10.1007/s11227-011-0556-x


http://dx.doi.org/10.1007/s11227-011-0556-x

http://dx.doi.org/10.1007/s11227-011-0556-x

http://dx.doi.org/10.1007/s11227-011-0556-x


http://dx.doi.org/10.1016/j.csi.2011.05.001

http://dx.doi.org/10.1109/42.141636

http://dx.doi.org/10.1080/01431169821426

http://dx.doi.org/10.1016/0031-3816(77)90006-1

http://dx.doi.org/10.1016/0146-664X(80)90016-7