PERGAMON

Computers & Geosciences 29 (2003) 837–850

COMPUTERS & GEOSCIENCES

www.elsevier.com/locate/cageo

Grey level co-occurrence integrated algorithm (GLCIA): a superior computational method to rapidly determine co-occurrence probability texture features ☆

David A. Clausi*, Yongping Zhao

Department of Systems Design Engineering, University of Waterloo, Waterloo, Ont., Canada ON N2L 3G1

Received 4 June 2002; received in revised form 11 March 2003; accepted 15 March 2003

Abstract

A critical shortcoming of determining co-occurrence probability texture features using Haralick's popular grey level co-occurrence matrix (GLCM) is the excessive computational burden. In this paper, the design, implementation, and testing of a more efficient algorithm to perform this task are presented. This algorithm, known as the grey level co-occurrence integrated algorithm (GLCIA), is a dramatic improvement on earlier implementations. This algorithm is created by integrating the preferred aspects of two algorithms: the grey level co-occurrence hybrid structure (GLCHS) and the grey level co-occurrence hybrid histogram (GLCHH). The GLCHS utilizes a dedicated two-dimensional data structure to quickly generate the probabilities and apply statistics to generate the features. The GLCHH uses a more efficient one-dimensional data structure to perform the same tasks. Since the GLCHH is faster than the GLCHS yet the GLCHH is not able to calculate features using all available statistics, the integration of these two methods generates a superior algorithm (the GLCIA). The computational gains vary as a function of window size, quantization level, and statistics selected. Using a variety of test parameters, experiments indicate that the GLCIA requires a fraction (27–54%) of the computational time compared to using the GLCHS alone. The GLCIA computational time relative to that of the standard GLCM method ranges from 0.04% to 16%. The GLCIA is a highly recommended technique for anyone wishing to calculate co-occurrence probability texture features, especially from large digital images.

Keywords: Data structures; Grey level co-occurrence matrices (GLCM); Computational time reduction; Digital images; Texture analysis

1. Introduction

A popular method for texture feature extraction in remote-sensing image interpretation is the use of co-occurrence probabilities, first introduced by Haralick et al. using the grey level co-occurrence matrix (GLCM) (Haralick et al., 1973). Co-occurrence probability texture features have been used in many different applications such as Systeme Probatoire d'Observation de la Terra (SPOT) land cover classification (Marceau et al., 1990; Franklin and Peddle, 1990), cloud recognition (Gu et al., 1991), synthetic aperture radar (SAR) sea ice classification (Barber and LeDrew, 1991; Soh and Tsatsoulis, 1999), as well as Landsat MSS ice recognition (Welch et al., 1990). However, since the GLCMs are recognized as sparse matrices, excessive computation is required to generate the co-occurrence texture features.

☆ Code on server at http://www.iamg.org/CGEditor/index.htm

*Corresponding author. Tel.: +1-519-888-4567; fax: +1-519-746-4791.

E-mail address: dclausi@engmail.uwaterloo.ca (D.A. Clausi).

0098-3004/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
doi:10.1016/S0098-3004(03)00089-X

Other methods have been used to reduce the computational demands of generating co-occurrence texture features relative to the GLCM approach (Unser, 1986; Clausi and Jernigan, 1998; Svolos and Todd-Pokropek, 1998; Clausi and Zhao, 2002). Unser (1986) investigated sum and difference histograms to store the co-occurring data, which are efficient relative to using matrices. No computational comparisons with the GLCM were performed, but the theory clearly indicates a dramatic reduction in the computational time relative to the GLCM. Clausi and Jernigan (1998) reduce the computational time relative to the GLCM computational time by presenting a grey level co-occurrence linked list (GLCLL) structure that stores the non-zero co-occurring probabilities in a sorted linked list. An advancement on the GLCLL is illustrated by Svolos and Todd-Pokropek (1998) where a tree data structure is used to store the co-occurring probabilities. This represents a reduction in the computational demands since the tree structure search has computational order $O(\log m)$ compared to $O(m)$ of the GLCLL approach. Computational order is a standard technique to estimate the total number of calculations required to perform a particular algorithm. Another computational improvement on the GLCLL is the grey level co-occurrence hybrid structure (GLCHS) (Clausi and Zhao, 2002). The GLCHS is more efficient than the GLCLL since it avoids the need for maintaining sorted linked lists by using a combined hash table and linked list data structure.

In this paper, the sum and difference histograms are implemented using a GLCHS framework. The resulting implementation (the grey level co-occurrence hybrid histogram (GLCHH)) is demonstrated to reduce the computational demands relative to the GLCHS alone. However, the sum and difference histograms are not able to act as a basis to use all of Haralick's statistics exactly. For these statistics, the GLCHS is used. Consequently, an integrated method called the grey level co-occurrence integrated algorithm (GLCIA) is created, which offers a dramatically improved solution by combining the GLCHS and GLCHH methods. Fig. 1 illustrates the research history leading to the development of the GLCIA.

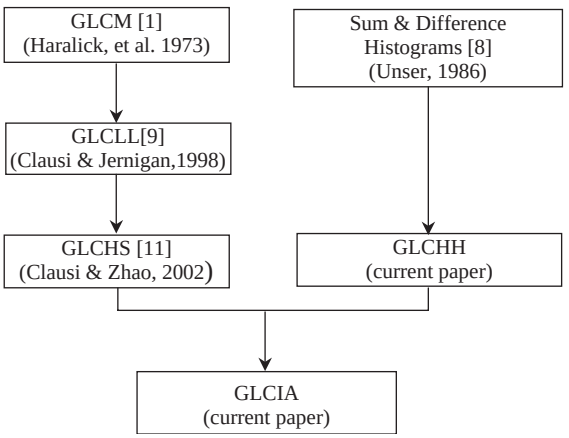

Fig. 1. Research history of GLCIA algorithm.

The quantization level (where G indicates the number of grey levels) is an important consideration when determining co-occurrence texture features. Quantization was originally used in the co-occurrence texture method to minimize memory needs and to increase the computational speed. A certain degree of quantization also reduces noise; however, significant quantization can destroy pertinent signal content (Clausi, 2002; Soh and Tsatsoulis, 1999). The GLCM method is highly sensitive to the number of grey levels, given that the computational complexity is proportional to $O(G^2)$ (Clausi and Jernigan, 1998). The GLCIA algorithm's computational performance will be shown to be not as sensitive to G as previous methods. As a result, there is interest in using the GLCIA algorithm in commercial remote-sensing software packages to reduce the number of computations for various G. For example, PCI's Geomatica© software constrains G to 16 grey levels when producing co-occurrence texture features, which can reduce their effectiveness since recent research recognizes the need for a greater number of quantization levels for discriminating SAR sea ice imagery, e.g. $G = 24$ (Clausi, 2002) and $G = 64$ (Soh and Tsatsoulis, 1999).

This paper will first briefly discuss the individual algorithms used as a basis for the GLCIA. This leads to the development of the GLCHH design. From this, the design and implementation of the novel GLCIA is presented. Computational speed testing is performed to compare different algorithms. Also, the GLCIA is applied to a 1000×1000 pixel image to demonstrate total computation time required for a full image. A summary concludes the paper.

2. Basic algorithms for determining co-occurrence texture features

After briefly defining co-occurrence probabilities, this section then describes each of the algorithms that act as a basis for the preferred algorithm, the GLCIA.

2.1. Co-occurrence probabilities defined

The co-occurrence probability between two grey levels i and j given a spatial offset (δ_x, δ_y) can be computed for all possible co-occurring grey level pairs in an image window (Haralick et al., 1973). Co-occurrence probabilities are mathematically defined in the following manner. Consider elements $r_{a,b}$ in a fixed image window

R with dynamic range G, where a and b represent indexed positions in the window ($a = 1, \ldots, A$; $b = 1, \ldots, B$). Given a certain spatial offset (δ_x, δ_y), the total number of co-occurring grey level pairs (i,j) (where $0 \leqslant i < G$ and $0 \leqslant j < G$) is determined by

$$C(i,j|\delta_x,\delta_y) = C(i,j) = \mathrm{Card}\{(a,b) \in R, r_{a,b} = i, r_{a+\delta_x,b+\delta_y} = j, 1 \leqslant a+\delta_x \leqslant A, 1 \leqslant b+\delta_y \leqslant B\}, \quad (1)$$

where 'Card' (the cardinality) indicates the total number of elements for the given set. Effectively, a two-dimensional histogram is created (dimensioned to $G \times G$) that represents the total number of occurrences for each (i,j) grey level pair that occurs in a fixed sized window given a certain spatial offset. The constraints at the end of the equation are necessary to ensure that all grey level pairs are selected from within the given window. The co-occurring probabilities are determined by dividing $C(i,j)$ by the total number of counts across all i and j, i.e.

$$P(i,j|\delta_x,\delta_y) = P_{ij} = C(i,j)/N, \quad (2)$$

where

$$N = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C(i,j). \quad (3)$$

Comparably, the co-occurring probabilities can be implemented using a relative orientation (θ) and distance (δ). However, the (δ_x, δ_y) notation is more convenient to the end user since orientations other than the basic 0°, 45°, 90°, and 135° are awkward in the (θ, δ) notation.

2.2. Grey level co-occurrence matrix (GLCM)

Early implementations stored the co-occurrence probabilities P_{ij} in a GLCM dimensioned to the number of grey levels, G. To determine texture features, selected statistics are applied to each GLCM by iterating through the entire matrix (see the GLCM column in Table 1).

Table 1
Common statistics applied to co-occurrence probabilities

Statistics name	GLCM	GLCLL/GLCHS	GLCHH
Uniformity (UNI)	$\sum_{i=1}^{G}\sum_{j=1}^{G} P_{ij}^2$	$\sum_{k=1}^{L} P_k^2$	N/A
Entropy (ENT)	$-\sum_{i=1}^{G}\sum_{j=1}^{G} P_{ij}\log P_{ij}$	$-\sum_{k=1}^{L} P_k \log P_k$	N/A
Maximum probability (MAX)	$\max\{P_{ij}\}\ \forall(i,j)$	$\max\{P_k\}\ \forall(k)$	N/A
Dissimilarity (DIS)	$\sum_{i=1}^{G}\sum_{j=1}^{G} P_{ij}\lvert i-j\rvert$	$\sum_{k=1}^{L} P_k\lvert i_k - j_k\rvert$	$\sum_{k=1}^{L_d} d_k D_k$
Contrast (CON)	$\sum_{i=1}^{G}\sum_{j=1}^{G} P_{ij}(i-j)^2$	$\sum_{k=1}^{L} P_k(i_k - j_k)^2$	$\sum_{k=1}^{L_d} d_k^2 D_k$
Inverse difference moment (IDM)	$\sum_{i=1}^{G}\sum_{j=1}^{G} \frac{P_{ij}}{1+(i-j)^2}$	$\sum_{k=1}^{L} \frac{P_k}{1+(i_k-j_k)^2}$	$\sum_{k=1}^{L_d} \frac{D_k}{1+d_k^2}$
Inverse difference (INV)	$\sum_{i=1}^{G}\sum_{j=1}^{G} \frac{P_{ij}}{1+(i-j)}$	$\sum_{k=1}^{L} \frac{P_k}{1+(i_k-j_k)}$	$\sum_{k=1}^{L_d} \frac{D_k}{1+d_k}$
Correlation (COR)	$\sum_{i=1}^{G}\sum_{j=1}^{G} \frac{(i-\mu)(j-\mu)P_{ij}}{\sigma^2}$	$\sum_{k=1}^{L} \frac{(i_k-\mu)(j_k-\mu)P_{ij}}{\sigma^2}$	$\frac{\sum_{k=1}^{L_s} S_k(s_k-2\mu)^2 - \sum_{k=1}^{L_d} D_k d_k^2}{4\sigma^2}$
	$\mu = \sum_{i=1}^{G} i \sum_{j=1}^{G} P_{ij}$	$\mu = \sum_{k=1}^{L} i_k P_k$	$\mu = \frac{1}{2}\sum_{k=1}^{L_s} s_k S_k$
	$\sigma^2 = \sum_{i=1}^{G}(i-\mu)^2 \sum_{j=1}^{G} P_{ij}$	$\sigma^2 = \sum_{k=1}^{L}(i_k-\mu)^2 P_k$	$\sigma^2 = \frac{\sum_{k=1}^{L_s} S_k(s_k-2\mu)^2 + \sum_{k=1}^{L_d} D_k d_k^2}{4}$

P_{ij} and P_k represent co-occurring probability in GLCM and GLCHS. $C_s(k)$ and $C_d(k)$ represent sum and difference histograms required for GLCHH. (i,j) represents co-occurring pair for GLCM and (i_k, j_k) represents co-occurring grey levels pair within GLCHS linked list. s_k and d_k represent sum $(i_k + j_k)$ and difference $(|i_k - j_k|)$ of co-occurring pairs. L, L_s, and L_d are linked list lengths of these three algorithms. μ and σ^2 represent mean and variance necessary for calculating only the COR statistic. These are the mean and

3	3	3	1	1
1	3	3	1	1
1	3	2	3	1
1	2	2	1	1
2	2	2	2	1

(A)

i \ j	0	1	2	3
0	0	–	–	–
1	0	0.15	–	–
2	0	0.15	0.2	–
3	0	0.25	0.1	0.15

(B)

Fig. 2. (A) 5×5 image window with $G = 4$ (range of 0–3). (B) Corresponding lower triangle GLCM given $(\delta_x, \delta_y) = \{(1,0), (-1,0)\}$.

Given that the GLCM is quite sparse (relatively few non-zero entries), this leads to many unnecessary calculations and considerable computing overhead. When applying this process to large remote-sensing images, exceptional computation is required which makes the algorithm not feasible in operational environments.

An example of a GLCM is presented in Fig. 2. Fig. 2A depicts a square image window with a window size $A = B = n = 5$ and $G = 4$. Fig. 2B represents the corresponding GLCM given $(\delta_x = 1, \delta_y = 0)$. Note that this GLCM is implemented in a symmetric manner, i.e. using $(\delta_x = 1, \delta_y = 0)$ and $(\delta_x = -1, \delta_y = 0)$ at the same time, so that only the lower triangle needs to be stored. This symmetry method has no loss of generality with respect to generating discriminating features (Clausi, 1996) and is advantageous due to reduced calculations when applying the statistics. Minor adjustments to the application of the statistics to a lower triangular matrix are required so that identical features are determined using a full matrix. With larger G, the matrix becomes sparser with a corresponding increase in the computational burden to apply the statistics.

For full image segmentation, each pixel is represented by its own feature vector (excluding the border pixels). Given a particular window in the image, the derived co-occurrence texture features (as a function of δ_x, δ_y, G, n, and the selected statistics) represent a feature vector for the window's centre pixel. Once the co-occurrence probabilities and statistics are determined for a given window, the next centre pixel's features are determined by moving the window over one column. Since most of the probabilities will remain the same, the probabilities only have to be updated by including co-occurring probabilities introduced by the new column and subtracting co-occurring probabilities introduced by the column left behind. Note that the larger the window, the smaller the percentage of pixel pairs that will change. By updating these probabilities when moving the window column by column (or row by row), the entire image can be covered using a "zigzag" path (Clausi and Jernigan, 1998). This approach is computationally advantageous and has been used for each algorithm implemented in this paper.

2.3. Grey level co-occurrence linked lists (GLCLL)

In the GLCLL method, a sorted linked list stores only the non-zero co-occurring probabilities. A linked list is a data structure that allows rapid access from node to node using pointers. The kth node in the list contains a probability value (P_k) and the grey level pair (i_k, j_k). Therefore, double summations over the entire GLCM are avoided and only single summations over the length of the linked lists (L) are required (Table 1). The linked list length L is much smaller than the matrix size $G \times G$, so tremendous gains are achieved. GLCLLs are a reduction in the computational demands relative to the GLCM (Clausi and Jernigan, 1998). Given a variety of window sizes and quantizations, the GLCLL method required 0.20–18% of the computing time relative to the GLCM, for a given test image across various window sizes and grey level quantizations.

Maintaining the sorted list as the window moves requires additional computation. First, whether or not the co-occurring pair is represented on the linked list must be determined. This check is easily performed if the lists are sorted. If the co-occurring pair is represented on the linked list, its probability value is updated. If the pair is not represented, then a node is inserted and initialized at the proper location in the list. Without a sorted list, one would always have to search the entire list for a particular grey level pair, which would be more time consuming.

2.4. Grey level co-occurrence hybrid structure (GLCHS)

The GLCHS (Clausi and Zhao, 2002) uses a hybrid implementation—a hash table to access a linked list (Fig. 3). The hash table is dimensioned to the lower triangle size that would be required by an equivalent GLCM as in Fig. 2, i.e. $G(G+1)/2$. In this sense, the constraint $i \geqslant j$ is imposed. Access to the hash table is provided by using (i, j) as a unique key, i.e. $iG + j$. Each entry in the hash table contains a pointer. If this pointer is null, then the particular co-occurring pair (i, j) does not have a representative node on the linked list. In this case, a new node would be created, inserted at the end of the linked list, and its grey level values set. If the pointer is not null, then it points to an existing corresponding node on the linked list. In this case, the only necessary step is to increment the probability. This framework allows for ease of modifying a probability associated with a node, as well as adding and deleting nodes without the need for a sorted list. The GLCHS will be

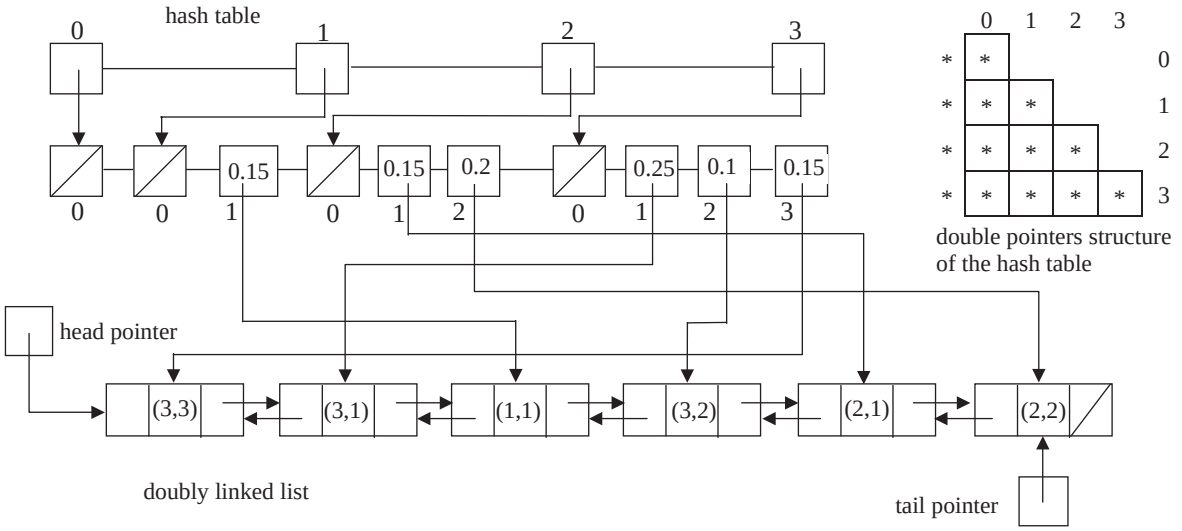

Fig. 3. GLCHS method for determining image texture features. Nodes created are based on sample image in Fig. 2A. Double pointers are created to access lower triangular matrix to represent hash table. Each pointer in hash table may then point to node on linked list.

built in the order in which the co-occurring pairs are encountered.

Fig. 3 illustrates the implementation of a GLCHS using the image window from Fig. 2A. The pair (3,3) is the first co-occurring pair in the image window, so it is inserted at the start of the linked list. Similarly, the pair (2,2) is inserted as the last node. This design significantly reduces the completion times for determining co-occurrence probability texture features. The hash table allows rapid access to an (i,j) pair and the linked list provides a fast means to apply the statistics. The GLCHS is an improvement on the GLCLL method requiring 28–38% of the GLCLL computation time and 0.10–32% of the GLCM computation time for the given test image (depending on quantization and window size) (Clausi and Zhao, 2002).

2.5. Sum and difference histograms

There exists a means of using vectors in the form of sum and difference histograms for the purpose of generating co-occurrence texture features (Unser, 1986). A sum and a difference associated with the relative displacement (δ_x, δ_y) are defined as

$$s_{a,b} = r_{a,b} + r_{a+\delta_x,b+\delta_y}, \tag{4a}$$

$$d_{a,b} = r_{a,b} - r_{a+\delta_x,b+\delta_y}. \tag{4b}$$

A sum histogram is defined as

$$\begin{aligned} C(i;\delta_x,\delta_y) &= C_s(i) \\ &= \mathrm{Card}\{(a,b)\in R, s_{a,b} = i, \\ &\quad 1\leqslant a+\delta_x\leqslant A, 1\leqslant b+\delta_y\leqslant B\}, \end{aligned} \tag{5}$$

where $i = 0, 1, \ldots, 2(G—1)$. The corresponding difference histogram is defined as

$$\begin{aligned} C_d(j;\delta_x,\delta_y) &= C_d(j) \\ &= \mathrm{Card}\{(a,b)\in R, d_{a,b} = j, \\ &\quad 1\leqslant a+\delta_x\leqslant A, 1\leqslant b+\delta_y\leqslant B\}, \end{aligned} \tag{6}$$

where $j = -(G-1), \ldots, G-1$. The normalized sum and difference histograms are defined by

$$S(i) = C_s(i)/N_H \quad \text{for all } i, \tag{7a}$$

$$D(j) = C_d(j)/N_H \quad \text{for all } j, \tag{7b}$$

where

$$N_H = \sum_{i=1}^{H_s} C_s(i) = \sum_{j=1}^{H_d} C_d(j), \tag{8}$$

and H_s and H_d represent the lengths of the sum and different histograms, respectively.

The normalized sum and difference histograms can be used to determine co-occurrence texture features. Due to similarity, and for succinctness, the statistics are effectively represented in Table 1 under the GLCHH heading. Here, the GLCHH uses L_s and L_d as upper summation limits. The normalized sum and difference histogram statistics would simply substitute H_s for L_s and H_d for L_d for the GLCHH equations identified in Table 1. Additional Haralick statistics (Haralick et al., 1973) implemented using sum and difference histograms are provided. Only a subset is used here for demonstration of commonly used statistics. The computational advantage is obvious: summations over vectors of length $H_s = H_d = 2G-1$ are used as opposed to summations

0	1	2	3	4	5	6
0	0	0.15	0.15	0.45	0.1	0.15

(A)

-3	-2	-1	0	1	2	3
0	0.15	0.15	0.5	0.1	0.1	0

(B)

Fig. 4. (A) Normalized sum histogram based on sample image in Fig. 2A. Indices represent possible sums of co-occurring pairs in sample image given relative displacement of ($\delta_x = 1, \delta_y = 0$). (B) Normalized difference histogram based on sample image in Fig. 2A. Indices represent possible differences of co-occurring pairs in sample image given relative displacement of ($\delta_x = 1, \delta_y = 0$).

over matrices of size $G \times G$ in the GLCM approach. Figs. 4A and B show the normalized sum and difference histograms created based on the window in Fig. 2A. These represent the probability of a given sum or difference occurring within the window of interest. For example, half of the co-occurring pairs have the same grey level value; therefore, '0' has a value of 0.5 in the normalized difference histogram.

The drawback of the sum and difference histograms is that they cannot be used to determine all of Haralick's statistics exactly (Unser, 1986). For example, the statistics UNI, ENT, and MAX (see Table 1) cannot be determined exactly using sum and difference histograms (although estimates of two of these terms, namely UNI and ENT, can be made).

3. Grey level co-occurrence integrated algorithm (GLCIA)

This section will first discuss the design of the more efficient method to implement the sum and difference histograms using the GLCHS's data structure. Then, the complete final algorithm known as the GLCIA will be presented by integrating the GLCHH and GLCHS algorithms. The description of the GLCIA implementation concludes Section 3.

3.1. Grey level co-occurrence hybrid histogram (GLCHH)

The sum and difference histograms are generally sparse vectors, especially with smaller window sizes and/or larger G. As a result, they may be implemented using the hybrid data structure used by the GLCHS. These pointer-based data structures will run faster than standard vectors. This implementation will be referred to as the GLCHH. The normalized sum and difference histograms must each be represented using a hybrid data structure. These will be referred to as the grey level co-occurrence hybrid sum histogram (GLCHSH) and the grey level co-occurrence hybrid difference histogram (GLCHDH).

This concept is illustrated in Fig. 5. Fig. 5A represents the sum histogram using a hash table with pointers pointing to nodes on a linked list and Fig. 5B represents the comparable implementation for a difference histogram. Since a histogram is represented, only a one-dimensional hash table is created which is accessed by a single key (either the sum or the difference). As with the GLCHS, the linked list is built based on the order in which the co-occurring pairs are encountered in the window. For example, the first node in Fig. 5A holds the sum '6' which represents the sum of the first co-occurring pair encountered in the window of Fig. 2A given $\delta_x = 1$ and $\delta_y = 0$. The dimension of the hash table for the GLCHSH is $2G - 1$. To accommodate the lower triangle GLCM (i.e. $i \geqslant j$) for the GLCHDH (Fig. 5B) and maintain consistency across the earlier algorithms, the hash table only needs to contain G elements. That is, since $i \geqslant j$, no negative values would appear in the difference histogram.

In the sum and difference histograms, each histogram was a fixed length ($H_s = H_d = 2G - 1$). In the above implementation, only the sum or differences with non-zero values are stored so that these fixed lengths only represent the maximum lengths of the GLCHH implementation. L_d and L_s represent the variable lengths for the GLCHH implementation. The eligible statistics used to calculate texture features based on the GLCHH are indicated in Table 1.

3.2. Grey level co-occurrence integrated algorithm (GLCIA)

Since only sums and differences need to be determined using the GLCHH implementation, it represents, in theory, a faster algorithm to calculate co-occurrence texture features compared to the GLCHS, which requires a two-dimensional hash table with longer linked lists. However, as mentioned earlier, not all the statistics commonly used can be calculated using normalized sum and difference histograms. As a result, the GLCHH method and the GLCHS method can be integrated to produce a preferred method for determining co-occurrence probability texture features. This method is called the GLCIA. Note that all of the methods compared here calculate identical texture feature values.

Table 2 indicates the order comparison for each technique. Each order is divided into two components. The first term refers to determining the probabilities; the second refers to the application of the statistics. With regard to the first term, most methods are of O(n) due to the zigzag updating scheme. The only method that

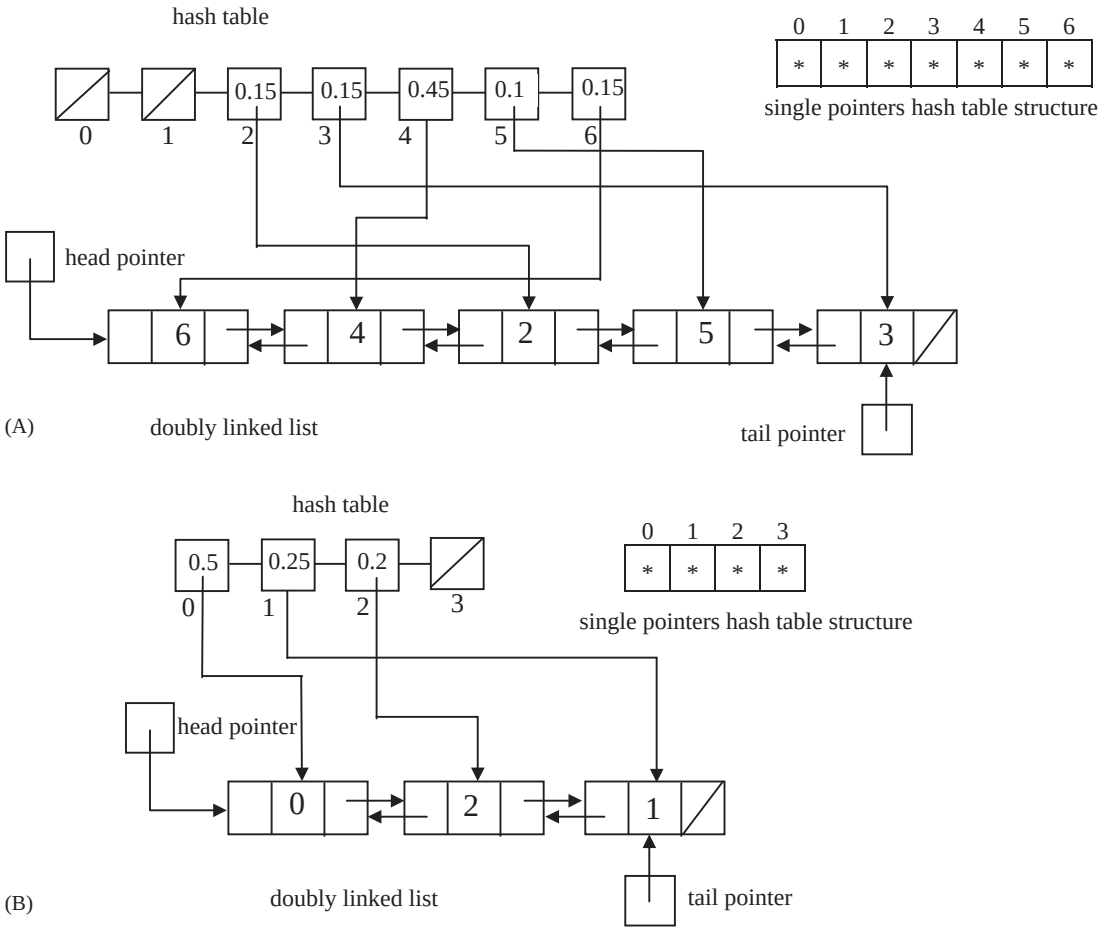

Fig. 5. Sum and difference components of GLCHH data structure based on sample image in Fig. 2A. In each case, hash table is dimensioned to number of indices indicated in Fig. 4 and relative displacement of ($\delta_x = 1, \delta_y = 0$) is used. (A) GLCHSH data structure to store normalized sum histogram. (B) GLCHDH data structure to store normalized difference histogram.

Table 2
Comparison of computational orders across different algorithms

Method	Order
GLCM	$O(n) + O(\alpha G^2)$
GLCLL	$O(nL) + O(\alpha L)$
GLCHS	$O(n) + O(\alpha L)$
Sum and difference histograms	$O(n) + O(\alpha H_s)$
	$O(n) + O(\alpha H_d)$
GLCHH	$O(n) + O(\alpha L_s)$
	$O(n) + O(\alpha L_d)$

First term represents computational order for creating probabilities. Second term represents computational order for determining statistics. Computational order improves as one moves further down this list.

differs is the GLCLL which requires updating and searching for a particular (i, j) pair which leads to $O(nL)$, where L represents the linked list length. With regard to the second term, α indicates a generic order for each method to account for applying each statistic. The GLCM requires looping through the entire matrix which requires $O(G^2)$. The other methods depend on the length of their linked lists, which is different for each of GLCHS, sum and difference histograms, and GLCHH. GLCHH is the preferred method since $(L_s, L_d) < (H_s, H_d) < L$.

3.3. GLCIA implementation

The software is implemented in the C programming language in a Unix environment using an in-house library as a basis. Fig. 6 represents the flowchart of the routine. First, the image file and the necessary parameters (δ_x, δ_y, G, n, statistics) are read. Window selection begins at the top left corner of the image. Since the window will move in a zigzag pattern to cover the entire image, the window will move left to right for even rows

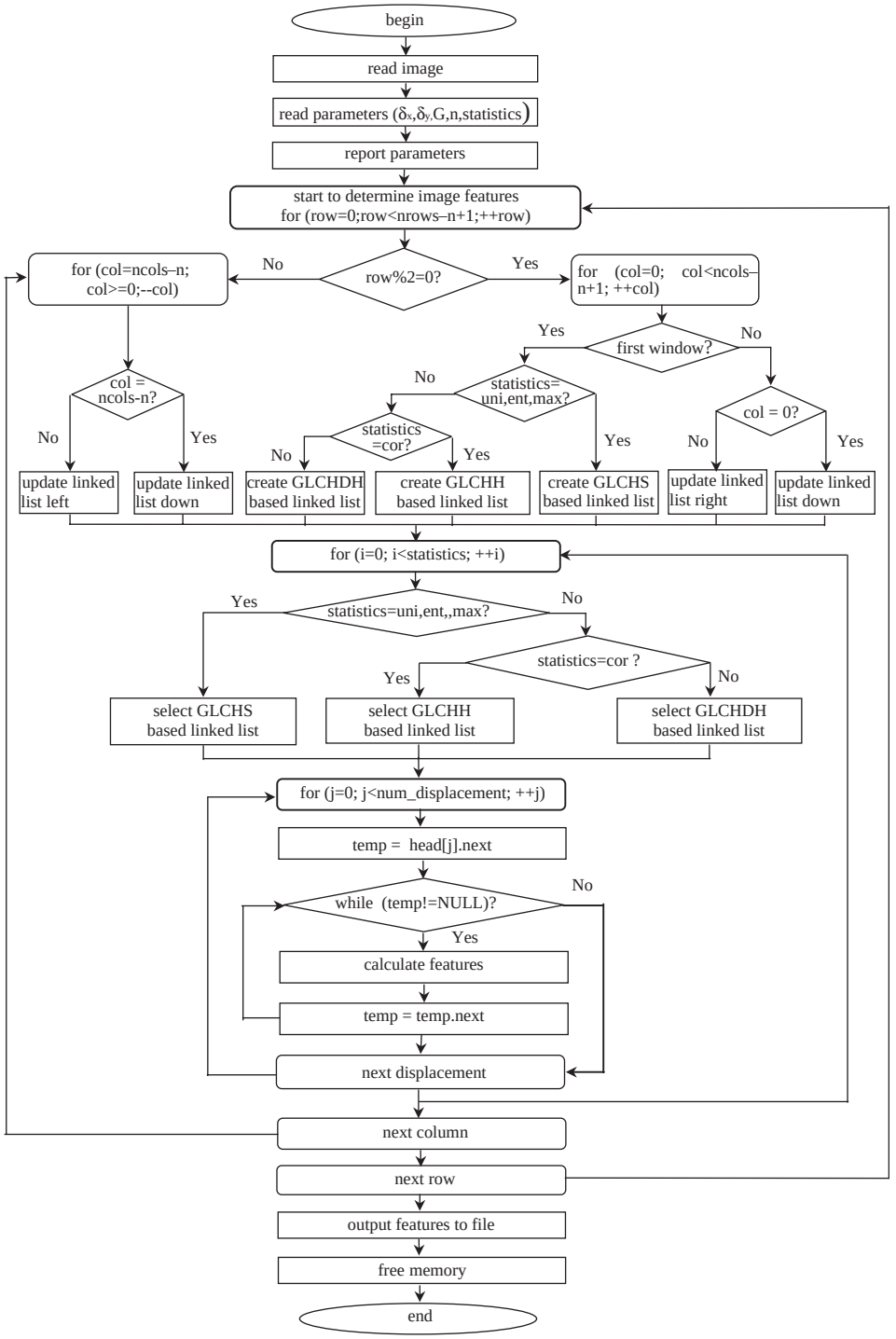

Fig. 6. Flowchart of GLCIA routine for determining co-occurrence probability texture features.

and right to left for odd rows. Each movement of the window leads to an updating step (left, right, or down) and, as mentioned earlier, instead of re-calculating all probabilities following a window shift, the data structures are simply updated with the new information.

Decisions concerning which data structures to create are made based on which statistics have been selected. Algorithm efficiencies (both speed and memory) are introduced by limiting, when necessary, the co-occurring information collected from the window. Not all GLCHH statistics require both sum and difference histograms (see Table 1). For any of DIS, CON, IDM, and INV, the GLCHDH data structure is required. For COR, both the GLCHDH and GLCHSH are required. The GLCHS is used for any of the statistics UNI, ENT or MAX. For each window, looping over all selected displacement pairs is required to calculate the necessary feature using the proper statistic.

4. Testing and results

This section is divided into four areas. First, the methodology is outlined (Section 4.1). Following this, completion times for the GLCIA method are provided (Section 4.2). Then, these results are compared to the GLCHS in terms of completion times and in terms of linked list lengths (Section 4.3). Finally, the GLCIA is applied to a large image to gain improved understanding of its computational capabilities (Section 4.4).

4.1. Methodology

Testing is performed on a Sun Sparc Ultra 1 200E (200 MHz, 128 Mbyte RAM, 322 SPECint, 462 SPECfp) workstation using a four-class 128×128 pixel Brodatz (Brodatz, 1966) test image (Fig. 7). Six window sizes (5×5, 10×10, 15×15, 20×20, 25×25, and 30×30) and five quantization levels (128, 64, 32, 16, and 8) are used as parameters. The interpixel displacements (δ_x, δ_y) are selected as $\{(1, 0), (1, 1), (0, 1), (-1, 1)\}$. The program for each algorithm is based on the same framework, i.e. the same routines for reading in the image, writing out the features, etc. All that differs between programs is the algorithm itself.

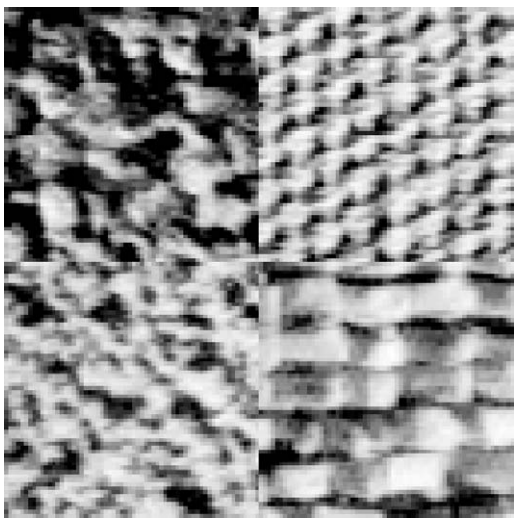

Fig. 7. Brodatz (1966) 128×128 test image.

4.2. GLCIA completion times

Timed testing is performed for the GLCIA using three scenarios.

(i) The first scenario uses all *eight* statistics indicated in Table 1. This requires both the GLCHS (to determine MAX, ENT, and UNI) and the GLCHH (for the other five statistics CON, IDM, INV, DIS, and COR);
(ii) The second scenario uses *five* statistics in Table 1 (CON, IDM, INV, DIS, and COR) that require the GLCHH data structure, i.e. both the GLCHDH and GLCHSH; and
(iii) The third scenario uses *four* statistics (CON, IDM, INV, DIS) that require only the GLCHDH data structure.

Table 3 indicates the test results using the GLCIA algorithm for each of these three scenarios. The indicated computation times are in units of μs/window. Comparing across the three different scenarios is not appropriate since each method applies a different number of statistics. Determining speeds on a per statistic basis (i.e. μs/window/statistic) is not appropriate since statistics have different computational requirements. This table does indicate that larger n as well as larger G increase computational times for the GLCIA method, supporting the orders indicated in Table 2. This increase is due to a higher number of co-occurring pairs which leads to longer linked lists. Longer linked lists require additional computation to add more nodes to the linked list and, more significantly, to apply the statistics.

4.3. Comparing GLCIA to GLCHS completion times

Fig. 8 illustrates the GLCHS compared to the GLCIA computational times as a function of G and n for each of the three scenarios provided in Section 4.2. The GLCIA completion times indicated in Table 3 as well as comparable completion times for the GLCHS are plotted. Table 4 indicates the percentage ratio of the GLCHS vs. the GLCIA completion times for each test indicated in Fig. 8. For all cases, GLCIA shows a demonstrated improvement over GLCHS. In the case of calculating all eight statistics, the percentage of the computational time required ranges from 27.0% to

Table 3
Average GLCIA computational time (μs/sample) given G and n for each of three scenarios: eight statistics {DIS, CON, IDM, INV, COR, UNI, ENT, MAX} (requiring GLCHH and GLCHS), five statistics {DIS, CON, IDM, INV, COR} (requiring only GLCHH), and four statistics {DIS, CON, IDM, INV} (requiring only GLCHDH)

	Grey levels (G)	Window size ($n \times n$)					
		5×5	10×10	15×15	20×20	25×25	30×30
GLCIA method (8 statistics)	128	12.4	38.4	75.3	119.9	169.3	217.5
	64	11.2	33.1	62.0	92.3	122.8	149.5
	32	9.8	24.9	41.6	54.2	63.9	71.7
	16	7.7	16.1	21.4	25.5	28.6	31.6
	8	5.9	9.5	12.0	13.9	15.9	17.9
GLCIA (GLCHH) method (5 statistics)	128	10.1	19.8	27.3	34.9	36.5	40.3
	64	8.1	13.8	18.1	21	23.1	25.0
	32	6.8	10.2	12.7	15.2	16.6	19.4
	16	5.5	7.8	10.0	11.8	13.9	15.8
	8	4.6	6.4	8.5	10.1	12	14.3
GLCIA (GLCHDH) method (4 statistics)	128	6.9	11.9	15.4	19.5	20.2	21.7
	64	6.1	8.8	11.1	13.2	14.4	16.6
	32	4.9	6.6	8.7	10.0	11.6	13.4
	16	4.1	5.7	7.2	8.9	10.4	11.5
	8	3.3	4.9	6.7	7.9	9.2	10.8

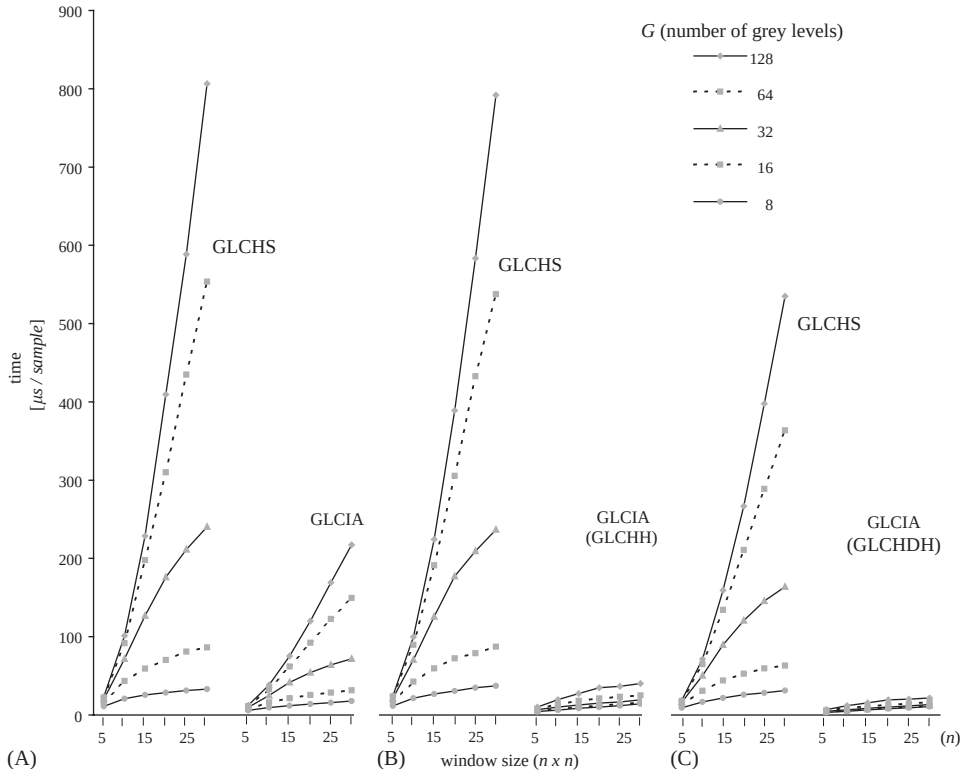

Fig. 8. Comparison of GLCIA and GLCHS computational times (μs/sample). (A) Average computational time comparison between GLCHS and GLCIA data structures. Eight statistics {DIS, CON, IDM, INV, COR, UNI, ENT, MAX} are used. (B) Average computational time comparison between GLCHS and GLCIA (GLCHV) data structures. Five statistics {DIS, CON, IDM, INV, COR} are used. (C) Average computational time comparison between GLCHS and GLCIA (GLCHDV) data structures. Four statistics {DIS, CON, IDM, INV} are used.

Table 4
Percentage ratios (%) for GLCIA vs. GLCHS, GLCIA (GLCHH) vs. GLCHS, and GLCIA (GLCHDH) vs. GLCHS in each sample window based on data in Fig. 5

	Grey levels (G)	Window size ($n \times n$)					
		5×5	10×10	15×15	20×20	25×25	30×30
Percentage of computational time for full GLCIA compared to GLCHS (8 statistics)	128	53.4	37.9	32.9	29.3	28.8	27.0
	64	50.5	36.1	31.4	29.8	28.2	27.0
	32	48.8	34.7	32.8	30.8	30.2	29.8
	16	47.2	37.0	36.0	36.3	35.4	36.6
	8	53.6	45.9	47.1	48.4	51.0	53.9
Percentage of computational time for GLCIA (GLCHH) compared to GLCHS (5 statistics)	128	41.4	19.8	12.2	9.0	6.3	5.1
	64	34.2	15.4	9.5	6.9	5.3	4.6
	32	32.2	14.4	10.1	8.6	7.9	8.2
	16	32.5	18.3	16.8	16.3	17.6	18.1
	8	39.3	29.8	31.6	32.9	34.6	38.4
Percentage of computational time for GLCIA (GLCHDH) compared to GLCHS (4 statistics)	128	36.9	16.6	9.7	7.3	5.1	4.1
	64	34.1	13.6	8.3	6.3	5.0	4.6
	32	30.1	13.1	9.7	8.3	8.0	8.2
	16	30.6	18.4	16.3	16.9	17.5	18.2
	8	35.5	29.2	31.0	30.6	32.5	34.6

53.9%. For the cases of four and five statistics, these ranges are even lower (4.1–36.9% and 5.1–41.4%), as a result of not having to use the hybrid structure. If one wanted to determine texture features using only the statistic CON, $n = 25$, and $G = 64$, then approximately 5% of the GLCHS computing time would be required. If a GLCM is used for the same purpose, this would amount to 9.1×5–0.46% of the time required (using Table 4 of Clausi and Zhao (2002)).

Reasons for the overall improvement of GLCIA compared to GLCHS can be motivated by considering the average linked list lengths across each sample window in the image using GLCHS, GLCHSH, and GLCHDH (Table 5). The terms in parentheses represent the percentage of the indicated linked list length with respect to the maximum possible linked list length for that particular implementation (i.e. if a lower triangle GLCM implementation was used). The linked list lengths follow predictable patterns: increasing G or increasing n increases the average linked list lengths. As mentioned earlier and demonstrated here, longer linked lists require additional computational time to maintain the list (adding and deleting of nodes) and to apply the statistics.

With increasing window size, the relative linked list length between GLCHDH and GLCHS as well as GLCHSH and GLCHS decreases. For example, given $G = 128$ grey levels and $n = 5$ pixels, the GLCHDH average linked list length is 12.9 and that of the GLCHS is 17.4 (a ratio of 0.74). At $n = 30$, this ratio becomes 0.077 (50.6/653.9). That is, with increasing window size, the GLCHDH should become relatively more efficient than the GLCHS when calculating co-occurrence texture features. This result agrees with results in Fig. 8.

The percentage of the maximum linked list length (indicated within brackets in Table 5) indicates that, as expected, larger G and smaller n lead to sparser vectors and matrices, GLCMs or sum/difference histograms should be used. This demonstrates the enhanced need for a storage mechanism that does not store zero probability terms.

4.4. Application of GLCIA to full textured image

To give the reader an appreciation of the total time required to determine co-occurrence texture features using the GLCIA, a larger image will be considered. Table 6 indicates computation times (in seconds) required for determining co-occurrence texture features given different statistics using a previously published 1000×1000 pixel Brodatz image mosaic (Jain and Farrokhnia, 1991, Fig. 13). Parameters include: $G = 32$ and 64, $n = 32$, and $(\delta_x, \delta_y) = \{(1,0), (1,1), (0,1), (-1,1)\}$. The statistics include combinations of CON (requiring only the GLCHDH), COR (requiring both GLCHDH and GLCHSH, i.e. the complete GLCHH), and ENT (requiring the GLCHS). When CON alone is determined, a total of 47 s is required for $G = 64$ to determine texture features for the whole image. Determining features using both CON and COR (requiring

Table 5
Average linked list length and percentage with respect to maximum linked list length (in parentheses) for GLCHS, GLCIA (GLCHSH), and GLCIA (GLCHDH) as a function of n and G

	Grey levels (G)	Window size ($n \times n$)					
		5×5	10×10	15×15	20×20	25×25	30×30
Average linked list length per sample and (percentage of maximum) for GLCHS algorithm	128	17.4 (0.2)	80.2 (1.0)	183.4 (2.2)	318.9 (3.9)	478.4 (5.8)	653.9 (7.9)
	64	16.7 (0.8)	73.4 (3.5)	157.4 (7.6)	253.8 (12.2)	351.2 (16.9)	442.5 (21.3)
	32	15.0 (2.8)	57.6 (10.9)	105.4 (20.0)	145.3 (27.5)	175.1 (33.2)	197.1 (37.3)
	16	11.8 (8.7)	34.3 (25.2)	49.3 (36.3)	57.9 (42.6)	63.6 (46.8)	68.1 (50.1)
	8	7.5 (20.8)	15.2 (42.2)	18.4 (51.1)	20.2 (56.1)	21.4 (59.4)	22.3 (61.9)
Average linked list length per sample and (percentage of maximum) for GLCIA (GLCHSH) algorithm	128	16.2 (6.4)	62.9 (24.7)	112.5 (44.1)	148.0 (58.0)	169.0 (66.3)	180.8 (70.9)
	64	14.8 (11.7)	49.1 (38.7)	74.2 (58.4)	86.6 (68.2)	92.6 (72.9)	96.2 (75.7)
	32	12.6 (20.0)	33.2 (52.7)	42.6 (67.6)	46.3 (73.5)	48.4 (76.8)	49.9 (79.2)
	16	9.7 (31.3)	19.5 (62.9)	22.7 (73.2)	24.2 (78.1)	25.1 (81.0)	25.8 (83.2)
	8	6.5 (43.3)	10.6 (70.7)	11.8 (78.7)	12.3 (82.0)	12.7 (84.7)	12.9 (86.0)
Average linked list length per sample and (percentage of maximum) for GLCIA (GLCHDH) algorithm	128	12.9 (10.1)	30.5 (23.8)	39.1 (30.5)	44.1 (34.5)	47.7 (37.3)	50.6 (39.5)
	64	10.2 (15.9)	18.8 (29.4)	22.4 (35.0)	24.7 (38.6)	26.4 (41.3)	27.9 (43.6)
	32	7.2 (22.5)	11.0 (34.4)	12.6 (39.4)	13.7 (42.8)	14.5 (45.3)	15.3 (47.8)
	16	4.7 (29.4)	6.4 (40.0)	7.1 (44.4)	7.7 (48.1)	8.1 (50.6)	8.6 (53.8)
	8	3.1 (38.8)	3.8 (47.5)	4.2 (52.5)	4.5 (56.3)	4.7 (58.8)	4.9 (61.3)

Table 6
Total time (s) required using 200 MHz computer to determine co-occurrence texture features given combinations of three frequently used statistics

	Total time (s)		Probability time (s)		Percentage probability time	
	$G = 64$	$G = 32$	$G = 64$	$G = 32$	$G = 64$	$G = 32$
CON/ENT/COR	653	346	102	87	15.6	25.0
CON/ENT	637	303	80	66	12.6	21.7
CON/COR	108	83	59	58	54.6	70.7
ENT/COR	637	340	102	87	16.0	25.5
CON	47	42	37	37	79.7	87.7
ENT	538	274	56	41	10.3	25.1
COR	109	78	59	58	59.5	74.1

CON: contrast; ENT: entropy; COR: correlation. "Probability time" refers to the time used just to calculate probabilities. The right hand side column contains percentages of the "probability time" with respect to "total time".

the full GLCHH) requires 108 s, again for $G = 64$. However, once a statistic that requires the GLCHS is used, the computational time required increases appreciably. Determining ENT alone requires 538 s and determining all three indicated statistics requires 653 s. Keep in mind that these routines were run on a 200 MHz machine, and in theory, current 2.0 GHz processors would require 10% of the indicated computing time. That is, in theory, CON would require less than 5 s and CON/COR would require less than 11 s to perform the processing for the indicated examples.

Using CON, ENT, and COR is a reasonable choice of statistics for image texture segmentation (Clausi, 2002) and it is practical to require approximately 1 min (using a 2.0 GHz processor) to determine whether texture features for a 1000×1000 pixel image is practical. Granted, remote-sensing images are often larger and one may choose additional relative displacements to increase

the feature space dimensionality; however, these computational times allow the co-occurrence texture method to be used in a wider context given the dramatic decrease in times. Although not tested, it is reasonable to assume that larger images would have a linear increase in the computational time. For example, a 2000×2000 image would require approximately 4 min to process. By considering the results for the GLCHS algorithm (Clausi and Zhao, 2002, Table 4), the GLCIA algorithm will require (given all eight statistics and depending on quantization and window size) 0.04–16% of the computation time required by the GLCM. For the preferred quantization levels of $G \geqslant 32$ (Soh and Tsatsoulis, 1999; Clausi, 2002), this is reduced to the range 0.05–5.0%. If statistics that do not require the GLCHS are used (i.e. MAX, ENT, and UNI are not used) and $G \geqslant 32$, then the GLCIA algorithm requires 0.04–1.3% of the GLCM computation time.

The GLCIA code fundamentally completes two tasks given a particular window: determine the probabilities and then apply the statistics to the probabilities to generate the texture features. The second and third column pairs in Table 6 indicate an approximation of the amount of time and its associated percentage of the total time required to determine the probabilities. For ENT, using the GLCHS, which generates longer linked lists than the GLCHH, close to 10% (or 56 s) of the computational time is spent determining the probabilities. The rest of the time (482 s) is spent applying the statistics. CON, on the other hand, uses 80% (37 s) of its computational time to determine its GLCHDHs, with 10 s spent on applying the statistic. Anytime the GLCHS is required, significant computing is required to apply the statistics relative to the GLCHH. Using GLCHHs requires approximately the same amount of time to generate the probabilities as the GLCHSs (note the 59 s for COR and 56 s for ENT); however, the considerably shorter linked lists of the GLCHHs allows the application of the statistics to proceed much faster.

These results indicate that the GLCIA is a substantial improvement on earlier implementations for rapid determination of co-occurrence probability texture features. This method would also be an improvement on the binary tree implementation by Svolos and Todd-Pokropek (1998) since the binary tree requires $O(\log m)$ (where m is the cardinality of the set of elements stored in the tree) to store the probabilities while the GLCHS (using a hash table) requires only O(1).

5. Conclusion

This paper describes a rapid means of calculating co-occurrence probability texture features. One reason that the co-occurrence approach has not been suitable for operational use is due to the exceptional computation required. By applying the GLCIA, completion times are dramatically reduced compared to traditional and other recent methods to perform the same task. The GLCIA is a highly recommended technique for anyone wishing to calculate co-occurrence probability texture features, especially from large-scale remote-sensing images.

Acknowledgements

Support for this project was provided by Geomatics for Informed Decisions (GEOIDE) (http://www.geoide.ulaval.ca) and Cryospheric System in Canada (CRYSYS) (http://www.crysys.uwaterloo.ca/).

References

Barber, D.G., LeDrew, E.F., 1991. SAR sea ice discrimination using texture statistics: a multivariate approach. Photogrammetric Engineering and Remote Sensing 57 (4), 385–395.

Brodatz, P., 1966. Textures: A Photographic Album for Artists and Designers. Dover, New York, NY, 128pp.

Clausi, D.A., 1996. Texture segmentation of SAR sea ice imagery. Ph.D. Dissertation, Department of Systems Design Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1, 176pp, http://rousseau.uwaterloo.ca/~dclausi/Theses/daclausi_PhD.ps.

Clausi, D.A., 2002. An analysis of co-occurrence texture statistics as a function of grey level quantization. Canadian Journal of Remote Sensing 28 (1), 45–62.

Clausi, D.A., Jernigan, M.E., 1998. A fast method to determine co-occurrence texture features. IEEE Transactions on Geoscience and Remote Sensing 36 (1), 298–300.

Clausi, D.A., Zhao, Y., 2002. Rapid co-occurrence texture feature extraction using a hybrid data structure. Computers & Geosciences 28 (6), 763–774.

Franklin, S.E., Peddle, D.R., 1990. Classification of SPOT HRV imagery and texture features. International Journal of Remote Sensing 11 (3), 551–556.

Gu, Z.Q., Duncan, C.N., Grant, P.M., Cowan, C.F.N., Renshaw, E., Mugglestone, M.A., 1991. Textural and spectral features as an aid to cloud classification. International Journal of Remote Sensing 12 (5), 953–968.

Haralick, R.M., Shanmugam, K., Dinstein, I., 1973. Texture features for image classification. IEEE Transactions on Systems, Man and Cybernetics 3 (6), 610–621.

Jain, A.K., Farrokhnia, F., 1991. Unsupervised texture segmentation using Gabor filters. Pattern Recognition 24 (12), 1167–1186.

Marceau, D.J., Howarth, P.J., Dubois, J.-M.M., Gratton, D.J., 1990. Evaluation of grey level co-occurrence matrix method for land-cover classification using SPOT imagery. IEEE Transactions on Geoscience and Remote Sensing 28 (4), 513–519.

Soh, L.-K., Tsatsoulis, C., 1999. Texture analysis of SAR sea ice imagery using gray level co-occurrence matrices. IEEE

Transactions on Geoscience and Remote Sensing 37 (2), 780–795.

Svolos, A.E., Todd-Pokropek, A., 1998. Time and space results of dynamic texture feature extraction in MR and CT image analysis. IEEE Transactions on Information Technology in Biomedicine 2 (2), 48–54.

Unser, M., 1986. Sum and difference histograms for texture classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (1), 118–125.

Welch, R.M., Kuo, K.-S., Sengupta, S.K., 1990. Cloud and surface texture features in polar regions. IEEE Transactions on Geoscience and Remote Sensing 28 (4), 520–528.