Algorithms of Clustering and Classifying Batik Images Based on Color, Contrast and Motif

Veronica S. Moertini & Benhard Sitohang

Department of Informatics Engineering – Bandung Institute of Technology
Jl. Ganesha 10 Bandung 40132, Indonesia
Email: moertini@home.unpar.ac.id & benhard@informatika.org

Abstract. An interactive system could be provided for batik customers with the aim of helping them in selecting the right batiks. The system should manage a collection of batik images along with other information such as fashion color type, the contrast degree, and motif. This research aims to find methods of clustering and classifying batik images based on fashion color, contrast and motif. A color clustering algorithm using HSV color system is proposed. Two algorithms for contrast clustering, both utilize wavelet, are proposed. Six algorithms for clustering and classifying batik images based on group of motifs, employing shape- and texture-based techniques, are explored and proposed. Wavelet is used in image pre-processing, Canny detector is used to detect image edges. Experiments are conducted to evaluate the performance of the algorithms. The result of experiments shows that fashion color and contrast clustering algorithms perform quite well. Three of motif based clustering and classification algorithms perform fairly well, further work is needed to increase the accuracy and to refine the classification into detailed motif.

1   Introduction

Although it is well known that wearing proper outfit could increase self-confidence, lots of people do not learn seriously of how to select the most suitable apparels. Indonesians tend to select expensive, the most up to date model, fine fabrics and fashion color with less consideration of whether the apparels would really "look good" on them. This conclusion is drawn after some observation at public places. Despite this fact, to the writers' knowledge, traditional and online fashion stores do not give suggestion of how to select the proper apparels for customers. Fashion stores let the customers pick apparels by themselves, which could lead to miss-selection.

Supposedly, in order to give better service to customers, future fashion stores provide interactive systems to help customers picking the right clothes. The interactive systems would need databases to manage their apparel collections,

along with some information that could be matched with customers' profile. This early stage of research attempts to study what information regarding clothes is needed by the system. Batik is chosen as a case study, as it is not only a textile product but also a philosophical art work having rich meaning. (Batik designers claim that batiks are intended to give visual beauty as well as to radiate the "*inner beauty*" of the wearers. See Doellah (2002).) Also, although batik images are potential objects to be clustered and classified for fashion purpose, we found that no prior research attempted to analyze them. Among other variables, it is identified that color, contrast and motif of batik textile are important in determining what batik is suitable for a certain profile of wearers. Batik images are needed to be clustered or classified based on color, contrast and motif. Each class contains a collection of batiks suitable for certain skin color, a combination of skin-hair color and impression.

This paper presents the back-ground theory of fashion color classification, batik motif and impression. With the aim of processing batik images, this paper discusses algorithms and experiments result of: (1) Fashion color clustering using *HSV* color model. (2) Contrast clustering using wavelet. (3) Motif classifying and clustering using wavelet, Canny edge detector, shape-based and texture-based approaches. The evaluation of the algorithms, conclusion and further work needed are given on the last section.

2   Fashion Color Classification

Irma Hardisurya (2004), an expert of color and appearance, highlights the important of selecting the right fabric color for every combination of skin and hair color. She classifies fashion color into neutral and accent. For formal apparels, accent colors should be combined with neutral colors to avoid glamour or sensational look. Hardisurya suggests that someone who has high contrast between his skin and hair put "contrast touch" on his or her apparels for better appearance. Or, the more contrast of skin and hair color, the more contrast of accent and neutral colors that ones should pick. Based on their skin and hair color, Hardisurya divides people pallete into several classes, each with its own color collection suitable for them. Examples are given on Table 1 (detailed table is given on Moertini (2005)). Selecting the proper accent and neutral color by considering the skin and hair color would result in healthy or lively skin look and more beautiful appearance.

Table 1 Examples of skin-hair color and suggested fashion colors (Hardisurya (2004)).

People Palette	Skin Color	Hair Color	Contrast Effect	Suggested Accent Colors	Suggested Neutral Colors
Cool	White pink, pink yellow, beige, rose beige	Black	High	Cool colors: magenta, zamrud, purple, lemon, etc.	Black, gray, taupe, navy, white.
Dark warm	Brown, dark golden brown	Brown to Black	Low	Red, brown-red, dark yellow, green-blue.	Dark brown, black, dark green, dark navy, red-grape, deep purple, ivory.
Soft Cool	White pink, peach, olive, dark olive, golden bown	Gray-brown	Medium	Soft colors of blue, green, coral, yellow, peach, olive.	Soft black, dark gray, dark brown, red grape, cream, off-white.

3   Batik Motif, Meaning, and Impression

Over the last decades, many kinds of batik style have been produced, such as *batik kraton* (batik from the courts), *batik sudagaran*, *batik Belanda*, *batik Cina*, *batik Djawa Hokokai*, and *batik Indonesia* which also called *batik modern*. These types of batik differ in terms of their motifs, the way they are produced, color variety, coloring material, and cloth material. Although batik has been changing over the time, according to Iwan Tirta, a well known batik artist and designer, Indonesian batik has unique properties. They are their motifs, *pakem* (the way the motifs are organized), produced using wax, having philosophies, and containing *isen-isen* (small ornaments used to fill the empty space in or between the main motifs (Kompas (19 Agustus 2001)). The items formed motifs are called ornaments. *Pakem*, which could be geometrical and non-geometrical, plays important roles in defining the philosophies or meaning of batik (Doellah (2002), Kerloug (2004), Hamzuri (1981) and Susanto (1980)). Batik produced on the coastal area (Pekalongan, Cirebon, Madura, etc.), called *batik pesisir*, expresses more freedom and dynamical statement through the pakem, as opposed to the ones produced close to the court (Yogyakarta, Solo, etc.) Although nowadays batik has been modernized, many classical motifs are still preserved (but, the motifs are somewhat modified for varieties). Iwan Tirta states that as batik motifs or *pakems* have meaning, people could select certain motifs to match with their profile or to give certain impression when wearing batiks. Some examples of classical motifs, their philosophical meaning and impressions that the wearers give is shown on Table 2. Some motifs sometimes are also designed for special occasions, such as wedding parties, engagement, funeral, child-birth ceremonies, dating (in order to attract others), for daily use, etc. Some examples of batik motifs are given on Figure 1.

Table 2 Examples of batik motif, meanings and impression (Doellah (2002), Kerlouge (2004), Hamzuri (1981), Susanto (1980)).

Motif or Pakem	Philosophical Meaning	Impressions, Effects
Kawung (repetitive ellips, circles), *nitik*, *truntum*.	Giving hope, wisdom, guiding.	Neat, calm, mature.
Parang, lereng, wave. Modified versions of *meru* (mountain), fire, *garuda*, bird that come in sharp edges.	Showing changes, dynamic, strongness.	Dynamic, masculine.
Original *meru, garuda,* building (*candi*, gate, etc.)*.*	Preserver, showing wisdom, firm will.	Classical, protective, mature.
Fire, dragons.	Showing strongness, braveness*, sakti*, toughness.	Masculine, brave
Buketan (bucket of flowers), butterflies.	Expressing beauty.	Feminime

kawung   **nitik**

parang

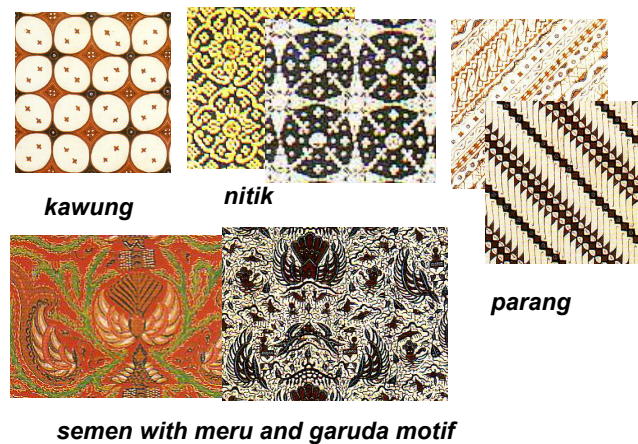**semen with meru and garuda motif**

Figure 1 Examples of batik motif.

4   Batik Images Clustering and Classification Algorithms

Based on the contents of Section 2 and 3, three groups of algorithms are proposed with the aim of clustering and classifying batik images based on their color, contrast and motif.

4.1   Clustering Batik Images Based on Fashion Color

After some study, we have not found any algorithm for clustering textile images based on fashion color as described in Hardisurya (2004). Therefore, in this section, we propose one. By studying the available color models (*RGB, HSV, CMY, CMYK*, etc), it is found that *HSV* is the most suitable color model for fashion color clustering (see Foley et.all. (1996) for color model discussion). Here, the values of *H, S* and *V* color component of batik image suitable for one's skin and hair color are approximated, some combination examples are given Table 3 (the table content is determined after some experiments). The classes of color *H, S*, and *V* (*clH, clS, clV*) are obtained as follows. *H* values are binned into 8 equally space containers, *S* values into 3 and *V* values into 4 (in experiments using batik images, these number of bins give the closest approximation to fashion colors). Each space represents a class. Let {*clH*}, {*clS*}, and {*clV*} be the set of *H, S, V* classes. Hence, $clH \in \{1,2,...,8\}, clS \in \{1,2,3\}, clV \in \{1,2,3,4\}$. *clH, clS*, and *clV* are computed from a batik image, and used to determine the batik image membership cluster in regard to a certain skin and hair color (see Algorithm 1). An image having a set of *clH, clS*, and *clV* can be a member of more than one skin and skin-hair color clusters.

Table 3   The class of H, S, V used to determine images cluster member.

Skin Cluster (Cluster Number)	H,S,V Class	Hair Cluster (Cluster Number)	H,S,V Class
White to pink or yellow (1)	clS = 1	Black (11)	clV == 1 and clS ==1, clV >= 3 and clS ==1
		Brown (12)	clV >=2 and clS ==1 and clH <=4
		Gray (13)	clV >= 2 and clS == 1 and clH > 4
Brown to black (4)	clV <=2 and clS >= 2 or clV ==3 and clS >= 2 and clH <3 clV==4 and clS >= 1 and clH >=1	Black (41)	clV ==1 and clS >= 2, clV ==2 and clS >= 2, clV ==3 and clS >= 2, clV ==4 and clS >= 1
		Gray (43)	clV ==1 and clS == 1 and clH >3, clV ==2 and clS == 1 and clH >3, clV ==3 and clS == 1 and clH >=2

Algorithm: Fashion color clustering of batik images
Input: (1) A set of image objects, *I* ={*Img*} (2) Width and height of image cropping, *Cr*
Output: arrays of list of integer, *SCl* and *SHCl*, containing images cluster members.
(*SCL*: skin cluster members. *SHCl*: skin-hair cluster members)
Method:
(1) create and initialize array *SCl*(*N*) and *SHCl*(N) with no member of list, where *N* is the number of images in *I*.
(2) for *i* = 1 to *N* do // process each image

(3)   Let $I(i) = \{p_w^{kl} \mid 1 \leq k \leq W,\ 1 \leq l \leq H,\ 1 \leq w \leq 3\}$ where p_w^{kl} is the pixel intensity of *I(i)* at *(k,l,)*, *W* = width, *H* = height of *I(i)* (in pixels), and *w* = 1 for *H*, *w* = 2 for *S* and *w* = 3 for *V* color component. Then

$$tempH(h) = \sum_{k=1}^{Cr} \sum_{l=1}^{Cr} binH_{kl}^{h}$$ where $1 \leq h \leq 8$, $binH_{kl}^{h}$ is the corresponding

bin number of *H* color component for p_1^{kl}. *tempS(s)* where $1 \leq s \leq 3$ and *tempH(h)* where $1 \leq h \leq 8$ can be computed in a similar way using p_2^{kl} and p_3^{kl}.

(4)   *clH = idH* where *tempH(idH)* = max(*tempH*), *clS = idS* where *tempS(idS)* = max (*tempS*), *clV = idV* where *tempV(idV)* = max (*tempV*)

(5)   Determine the skin cluster member, *clSk* (see Table 3). Insert *clSk* in list *SCl*(i).

(6)   Determine the skin-hair cluster member, *clSH* (see Table 3). Insert *clSH* in list *SHCl*(i).

Algorithm 1 Fashion color clustering of batik images.

4.2   Clustering Batik Images Based on Contrast

The applications of wavelet, a new tool for signal analysis, have been explored over the last few years. Two-dimensional discrete wavelet transform (2-D DWT) with different wavelets has been applied to images for compression, de-noising, retrieving them from databases, etc. In this paper, we use Haar wavelet because it is the fastest to compute and have been found to perform well in practice (Natsev et.all.(1999)). 2-D DWT would be utilized to cluster images based on their contrast. 2-D DWT produce average (*LL*), horizontal detail (*LH*), vertical detail (*HL*) and diagonal detail (*HH*) coefficients on each level of transformation, see Mallat (1999) for details.

In this section, we discuss Algorithm 2 and 3. In Algorithm 2, we adopt the method found in Bartolini (2001) with wavelet coefficient computation taken from Natsev et.all. (1999). As in experiments we found that Algorithm 2 did not perform well, we propose our own algorithm, which is Algorithm 3.

Contrast clustering of batik images using covariance matrix trace

Windsurf, a wavelet-based image retrieval algorithm handling HSV color images, uses covariance matrix trace, *TLL*, to determine the heterogeneity or homogeneity of an image (Bartolini (2001)). First, it segments the images into windows, then it computes the 3-level average (*LL*) wavelet coefficients of color component *H*, *S* and *V* from the windows. (Bartolini finds that 3-level wavelet decomposition performs well.) Second, a covariance matrix of the color component, C^{3LL}, which is a 3 × 3 matrix, is computed from the wavelet coefficients. Its elements are:

$$cov_{c,d}^{3LL} = \frac{1}{N}\left(\sum_{j=1}^{N} w_{c_j}^{3LL} w_{d_j}^{3LL} - \frac{1}{N}\sum_{j=1}^{N} w_{c_j}^{3LL} \sum_{j=1}^{N} w_{d_j}^{3LL}\right) \tag{4.1}$$

where w_c^{3LL} is 3-level average wavelet coefficient of color component c, N is the number of wavelet coefficient. c and d ($c,d \in \{0,1,2\}$) denote the color component (H=0, S=1, V=2). The covariance matrix trace, *TLL*, is defined as the sum of the diagonal elements, or,

$$TLL = c_{11} + c_{22} + c_{33} \tag{4.2}$$

The more homogeneous of an image, the smaller the variance of the bit map of the image, the *LL* sub-band coefficient of each color component, the *eigen* values of the covariance matrix, and *TLL* (see Bartolini (2001) for detail discussion.) Here, *TLL* is used to cluster batik images into three clusters, which are low-contrast (*cl*=1), medium-contrast (*cl*=2) and high-contrast (*cl*=3) cluster. Hence

$$cl = \begin{cases} 1, \text{if } 0 \leq TLL \leq tLow \\ 2, \text{if } tLow < TLL \leq tMed \\ 3 \text{ otherwise} \end{cases} \tag{4.3}$$

where *tLow* and *tMed* are threshold values for low and medium contrast.

Algorithm: Contrast clustering of batik images using covariance matrix trace.
<u>Input</u>: (1) Image objects, $I = \{Img\}$ (2) Size of cropping, *Cr* (3) *tLow, tMed*
<u>Output</u>: array SConT containing image contrast cluster membership.
<u>Method</u>:
(1) create and initialize array SCon(*N*) with zero values, where M is the number of images in *I*.
(2) for i = 1 to M do // process each image
(3)   Get cropped of *I(i)*, $I_c = \{p_w^{kl} \mid 1 \leq k \leq Cr, 1 \leq l \leq Cr, k,l \in Z\}$
(4)   compute the trace value of I_c, *TLL*, of the image using (4.2)
(5)   determine *cl* using (4.3)
(6)   SConT(i) = *cl*;

Algorithm 2 Contrast clustering of batik images using covariance matrix trace.

Contrast clustering of batik images using Haar wavelet

The idea is to use *LH, HL, HH* wavelet coefficients, which could be viewed as the intensity difference between neighboring pixels, in determining the contrast cluster of each image. The steps of the algorithm proposed are as follows. First, each color image is cropped, then, it is converted to gray-scale image. Second,

3-level DWT using Haar wavelet is applied to the image to produce *LH, HL, HH* coefficients. The larger the magnitudes of these coefficients, the more contrast the neighboring pixels (see Figure 2). Let *aLH, aHL* and *aHH* be the absolute values of *LH, HL, HH*. Third, the mean of *Pct* percent of the highest values of *aLH, aHL* and *aHH* are computed and stored as *mLH, mHL* and *mHH*. Let *tLH1, tLH2, tHL1, tHL2, tHH1, tHH2* be the mean thresholds. Fourth, cluster member of the image is determined by comparing *mLH, mHL* and *mHH* to the thresholds (see Algorithm 3).
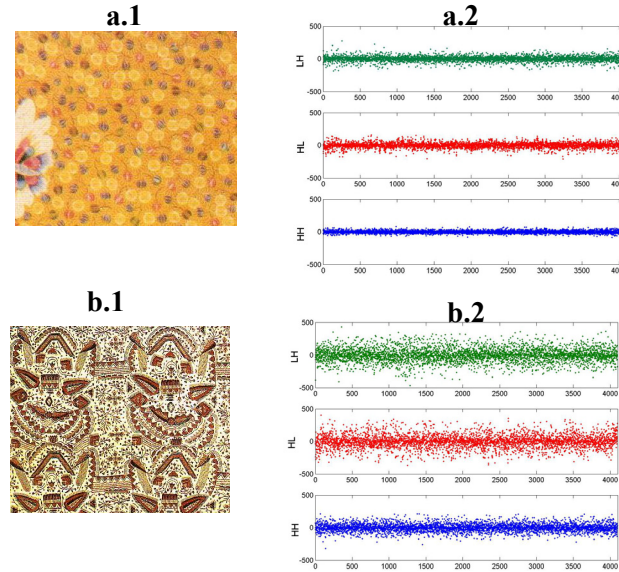

Figure 2 A low (top, a.1) and high-contrast (bottom, b.1) batik images with their distribution of *LH, HL, HH* wavelet coefficients (a.2 and b.2).

Algorithm: Contrast clustering of batik images using Haar wavelet.

Input: (1) Image objects, $I = \{Img\}$ (2) Size of cropping, *Cr* (3) Percentile threshold of wavelet coefficient taken, *Pct* (4) Mean thresholds, *tLH1, tLH2, tHL1, tHL2, tHH1, tHH2*

Output: array SConW containing image contrast cluster membership.

Method:

(1) create and initialize array SCon(*N*) with zero values, where *N* is the number of images in *I*.

(2) for $i = 1$ to *N* do // process each image

(3)   Get cropped of *I(I)*, $I_c = \{p_w^{kl} \mid 1 \le k \le Cr, 1 \le l \le Cr\}$

(4)   *IcBW* = gray scale of *Ic*

(5) $\{LH, HL, HH\}$ = compute_3-level_DWT(*IcBW*). $LH = \{lh_{mn}\}$, $HL = \{hl_{mn}\}$, $HH = \{hh_{mn}\}$, $1 \le m \le (Cr/8)$, $1 \le n \le (Cr/8)$, $m, n \in Z$.

(6) $aLH = \{\mid lh_{mn} \mid\}$, $aHL = \{\mid hl_{mn} \mid\}$, $aHH = \{\mid hh_{mn} \mid\}$

(7)   *pLH = percentile(aLH,Pct), pHL = percentile(aHL,Pct) pHH =* percentile*(aHH, Pct)*

(8)   *mLH* = mean*(pLH), mHL* = mean*(pHL), mHH* = mean*(pHH) (see (4.3))*

(9)   if(*mLH <=tLH1* and *mHL<=tHL1* or *mLH <=tLH1* and *mHH<=tHH1* or

(10)   *mHL<=tHL1* and *mHH<=tHH1*),

(11)   then *cl* = 1;

(12)   elseif(*mLH>tLH2* and *mHL>tHL2* or *mLH>tLH2* and *mHH>tHH2* or

(13)   *mHL>tHL2* and *mHH>tHH2*)

(14)   then *cl* = 3;

(15)   else *cl* = 2;

(16)   SConW(i) = *cl*;

Algorithm 3 Contrast clustering of batik images using Haar wavelet.

The mean in Algorithm 3 is defined as

$$mean(plH) = \sum_{i=1}^{Sz} plh_i / Sz \qquad\qquad (4.4)$$

where *Sz* = size of array *plH*.

4.3   Classifying and Clustering Batik Images Based on Motif

After studying batik images, it is concluded that batik image motifs could not be detected using the shapes or the textures alone, see Moertini (June 2005). It is also found that detecting motifs in batik images is a challenging task. The difficulties arise because many motifs are crowded with *isen-isen*. Also, the ornaments may come in various sizes, angles and styles as ornaments are art works. Given these facts, in this early stage of research, several novel classification and clustering algorithms for detecting motif group or class (instead of single motif) are proposed. The classes of motif used in the experiments are *banji-ceplokan-nitik*, *parang-lereng*, *semen*, *lung-lungan*, *buketan-other*, *truntum* or small shape ornaments. The reason of this grouping is that each class or group, more or less, gives a certain impression to the wearers.

The classification system, which adopts image retrieval algorithm technique, is given on Figure 3. During off-line stage, first, the training images are first preprocessed. Secondly, texture or shape features are extracted from the images and stored. On on-line mode, the new image being classified is preprocessed, and the features are generated. The features are then compared to each set of features stored in database, which generated a set of similarity or dissimilarity values between the new image and the training images. Using the similarity or dissimilarity values, *k*-nearest neighbor determines the class or group of motif

of the new batik image. Here, four classification algorithms are explored and proposed.

Unlike the classification techniques, clustering techniques do not require training images. The images are preprocessed, then, fed to the algorithms, and the output is the cluster member, which represents a group of motif, of each image. One clustering algorithm and a hybrid algorithm employing clustering and classification techniques are proposed.

On Fig. 3, the image preprocessing steps includes: cropping the images to find the part of the images that are meaningful, converting them to gray-scale image and normalize the intensities. In Algorithm 4 to 6, the techniques of texture and shape feature generation for training and new images are adopted from Lu (1999) and Hirata et.all. (1992). After some study and experimenting with batik images, in these algorithms, we choose 2-level of DWT computation to "filter out" or to reduce *isen-isen*, and Canny edge detector to detect the edge of the main ornaments (see Canny (1986) for Canny algorithm discussion).
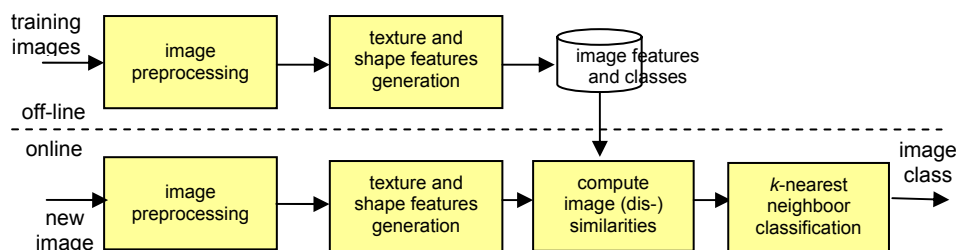

Figure 3 A schematic of batik images classification system.

Algorithm: Classifying batik images using wavelet and moments (shape-based).
Input: (1) Training image features and their classes, $TI =\{Ft, class\}$ (2) Preprocessed new image, I. (3) Segment size, Sg. (4) Segment shifting distance in horizontal and vertical direction, dx and dy. (5) Nearest neighbors, k
Output: Cl, the class of the new image.
Methods:

(1) LL = average coefficient of 2-level DWT of I

(2) Ed = Canny_edge_detector(LL), $Ed= \{e_{wh} \mid 1 \le w \le W , 1 \le h \le H \}$

(3) ct_max = round($(SI- Sg)/dx$)-1 // SI is the vertical or horizontal image dimension.

(4) initialize $DistSeg$, an array used to store the dissimilarity values, and $Dist$, an array of dissimilarity values between the new image and all of the training images.

(5) for i = 1 to N do // N is the number training images

(6) $x = 1$

(7)   for ctx = 1 to ct_max,   //Shift the segment in horizontal direction

(8) $y = 1$

(9)   for cty = 1 to ct_max //Shift the segment in vertical direction

(10) $Ed_c = \{e_{(x+m)(y+n)} \mid 1 \le m \le Sg, 1 \le n \le Sg\}$, $F_c =$ moment(Ed_c), $d_i =$

Euclidean_distance(F_c, TI(i).Ft), store d_i into *DistSeg* array.

(11) $y = y+dy$

(12) $x = x +dx$

(13) $Dist$(i) = min(*DistSeg)*

(14) sort *Dist* into ascending order and store it in *SortedDist*.

(15) compute Cl using equation (4.5)

Algorithm 4 Classifying a batik image using wavelet and moments (shape-based).

The class of the new image, *Cl,* is computed as $Cl = c$ in $max(\{vote(c)\})$ (4.5)
where $c \in \{1,2,..., MaxClass\}$, *MaxClass* is the maximum class number,

$vote(c) = \sum_{i=1}^{k} IClass_i$, and $IClass_i = \begin{cases} 1, \text{if the class of } TI \text{ having } SortedDist(i) \text{ is } c \\ 0, \text{otherwise} \end{cases}$

The computation of seven moments used as image feature in Algorithm 4 could be found in Lu (1999). Euclidean distance is used to compute the distance between the new and training image features.

Algorithm: Classifying batik images using wavelet and Fourier descriptor (shape-based).

Input: The same with the ones of Algorithm 4.   Output: *Cl*, the class of the new image.

Methods:The methods are the same with of Algorithm 4 except for feature computation.

Algorithm 5 Classifying a batik image using wavelet and Fourier descriptor (shape-based).

The steps of image feature computation using Fourier descriptor in Algorithm 5, which are modified steps of Lu (1999), are: (1) Compute *LL* coefficient component of 2-level DWT of the image. (2) Present the result of step 1 to Canny edge detector. (3) Compute the centroid of the edge points. (4) Compute the distance of each edge point to the centroid (radii). (5) Apply Discrete Fourier Transform (DFT) to the result of step 4. (6) Normalize the coefficients of step 4 by dividing each coefficient (F_n) with the first coefficient (F_0) and store them as the image features. The DFT (of step 5) is

$F_n = \dfrac{1}{M} \sum_{i=0}^{M-1} f(i)e^{\frac{-j2\pi ni}{M}}$, where $n = 0$ to M-1,$M = Sg^2$,which is the number of

edge points, *f(i)* is the radii of edge point *i*.

Algorithm: Classifying batik images using wavelet and correlation (shape-based).

Input: The same with the ones of Algorithm 4.

Output: *Cl*, the class of the new image.

Methods: Step (1) to (9) are the same with Algorithm 4.
(10) obtain Ed_c (see Alg. 5), cr = correlation(E_{dc}, TI(i).Ft), store cr into $CorSeg$.
(11) $y = y + dy$
(12) $x = x + dx$
(13) Cor(i) = max($CorSeg$)
(14) sort Cor into descending order and store it in $SortedCor$
(15) compute Cl using equation (4.5)

Algorithm 6 Classifying a batik image using wavelet and correlation (shape-based).

The method of correlation computation between 2 segments (in Step 10 of Algorithm 6) of image is adopted from Hirata et.all. (1992), which is reported to perform well in Sajjanhar (1998). The steps are as follows: <u>Step 10.1</u>: Segment the edges Sg to obtain 8 blocks of edges, $Q = \{q_{ij}\}$, and the training image edges to obtain 8 blocks of edges, $T = \{t_{ij}\}$. q_{ij} and t_{ij} are blocks of size of m x n (here, $m = 8$, $n = 8$, Sg is 64 x 64 points). <u>Step 10.2</u>: For each q_{ij} and t_{ij}, calculate the blocks correlation between q_{ij} and t_{ij} with sifting by d and e (this means that this algorithm accepts at most 12.5% shifted shape error):

$$C_{de}^{ab} = \sum_{r=ma}^{m(a+1)-1} \sum_{s=nb}^{n(b+1)-1} (\alpha q_{rs}.t_{r+ds+e} + \beta \bar{q}_{rs}.\bar{t}_{r+ds+e} + \gamma q_{rs} \oplus t_{r+ds+e})$$. Here, α, β and

γ are control parameters used to estimate matching and mismatching patterns (in the experiments, $\alpha = 10$, $\beta = 1$, $\gamma = -3$), and $\oplus$ is binary OR operator. <u>Step 10.3</u>: Take the maximum value of Step 10.2 results: $C^{ab} = \max(C_{de}^{ab})$ for

$-\dfrac{m}{2} \le d \le \dfrac{m}{2}, -\dfrac{n}{2} \le d \le \dfrac{n}{2}$ <u>Step 10.4</u>: The correlation is the sum of the result

of Step 10.3, which is: $CorSeg = \sum\limits_{a=0}^{7}\sum\limits_{b=0}^{7} C^{ab}$.

During the experiments of observing the output of 2-level DWT and Canny edge detector with *parang* and *ceplok* motif as the inputs, it is found that diagonal lines are shown (see Figure 4.a.3 and b.3). So, in Algorithm 7, we propose a new technique, which is used to detect and count segments of diagonal lines, both left and right direction, using masks. Four masks, which are *m11*, *m12*, *m21, m22*, are designed (see Figure 5). *m11* and *m21* are for right, and *m12* and *m22* are for left diagonal lines. If the counts of diagonal lines are above the threshold, either *parang* or *ceplok* motif may be found (see Algorithm 7).
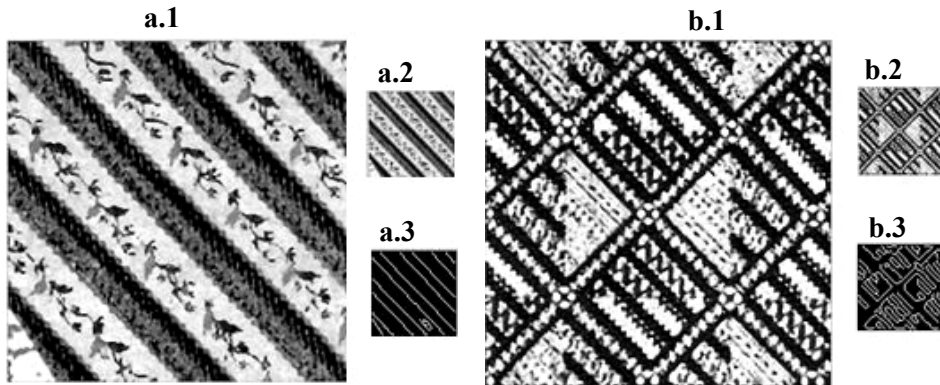
Figure 4 The original images of *parang* and *ceplok* motif (a.1 and b.1), *LL* coefficient of 2 level-DWT (a.2 and b.2) and Canny edge detector output (a.3 and b.3), where element array of 1 is displayed in white and 0 is in black color.

m11

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

m21

0	1	0	0	0
0	1	0	0	0
0	1	1	0	0
0	0	1	1	0
0	0	0	1	0

m12

0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	1	0	0	0
1	0	0	0	0

m22

0	0	0	1	0
0	0	0	1	0
0	0	1	1	0
0	1	1	0	0
0	1	0	0	0

Figure 5 The masks used in Algorithm 7.

Algorithm: Clustering a batik image into *parang*, *ceplok* and *other* motif using wavelet and masks (shape-based).

Input: (1) Preprocessed images, *I* (2) Mask arrays: *m11*, *m12*, *m21*, *m22*. (3) Threshold of counts of 1 in dot product of edge segments and masks, *Th1*. (4) Threshold of mask count, *Th2*. (5) Block size, *SB*; threshold block count of 1s, *Th3*.

Output: *Cl*, image cluster member.

Methods:

(1) LL = average coefficient of 2-level DWT of *I*

(2) Ed = Canny_edge_detector(LL), $Ed = \{e_{wh} \mid 1 \le w \le W , 1 \le h \le H \}$

(3) $Blk = \{b_{ij} \mid b_{ij} = 1 \; \forall i \forall j, 1 \le i \le SB, 1 \le j \le SB\}$

(4) *ctB*=0, *ct1*=0, *ct12*=0, *ct21*=0, *ct22* = 0.

(5) Overlay and move masks *Blk, m11, m12, m21, m22* on *Ed* (by non-overlapping method). Increase the values of *ctB, ct11, ct12, ct21* and *ct22* on each overlay if the count of 1 found is above *Th1*.

(6) If *ctB* < *Th3*, // segments are not block of 1s

(7) If pair of *ct11* and *ct12*, *ct11* and *ct22*, *ct12* and *ct21* or *ct11* and *ct22* > *Th2*, *Cl* =1 //ceplok

(8) else if *ct11, ct12, ct21* or *ct22* > *Th2*, *Cl* = 2 //parang

(9) else *Cl* = 8 //other

Algorithm 7 Clustering a batik image using wavelet and masks (shape-based).

In Algorithm 8, the method of generating image features and comparing between features are basically adopted from Wang et.all. (1999). 4-level of DWT using Haar wavelet is employed to obtain features of 1/64 of the original images.(In experiments, generating feature using 5-level DWT increased classification errors, so we choose 4-level to obtain small size of feature.) The compare 2 images, the Euclidean distance between the features are computed.

Algorithm: Classifying batik images using wavelet (texture-based).

<u>Input</u>: (1) Training image features and the classes, $TI =\{Ft, class\}$ (2) Preprocessed new image, I. (3) Segment size, Sg. (4) Segment shifting distance in horizontal and vertical direction, dx and dy. (5) Nearest neighbors, k

<u>Output</u>: Cl, the class of the new image.

<u>Methods</u>:

(1) $\{LL, LH, HL, HH\} =$ 4-level DWT coefficient of I.

    $LL = \{ll_{ij} \mid 1 \leq i \leq (S/64), 1 \leq j \leq (S/64), \ i, j \in Z\}$, LH, HL, HH are also array of

    $(S/64)(S/64)$, where S is the size of the original image

(2) $ct\_max = \text{round}((SI\text{-} Sg)/dx)\text{-}1$ // SI is the vertical or horizontal image dimension.

(3) initialize $DistSeg$, an array used to store the dissimilarity values, and $Dist$, an array of dissimilarity values between the new image and all of the training images.

(4) for $i = 1$ to N do // N is the number training images

(5) $x = 1$

(6)   for $ctx = 1$ to ct_max, //Shift the segment in horizontal direction

(7) $y = 1$

(8)   for $cty = 1$ to ct_max //Shift the segment in vertical direction

(9) $LL_c = \{ll_{(x+m)(y+n)} \mid 1 \leq m \leq Sg, 1 \leq n \leq Sg\}$. LH_c, HL_c, HH_c are crops of LH, HL, HH, which are analogous to LL_c. $d_i = $ distance(LL_c, TI(i).$Ft.LL$) + distance(LH_c, TI(i).$Ft.LH$) + distance(HL_c, TI(i).$Ft.HL$) + distance(HH_c, TI(i).$Ft.HH$). Store d_i into array $DistSeg$.

(10) $y = y+dy$

(11) $x = x +dx$

(12) $Dist$(i) = min($DistSeg)$

(13) sort $Dist$ into ascending order and store it in $SortedDist$.

(14) compute Cl using equation (4.5)

Algorithm 8 Classifying a batik image using wavelet (texture-based).

Since experiments show that Algorithm 7 and 8 performs quite well (see section Experiments), we propose Algorithm 9 to take the advantages of both.

Algorithm: Classifying batik images using Algorithm (Alg) 7 and Alg 8 (shape- and texture-based).

<u>Input</u>: Both input of Alg 7 and Alg 8 input. <u>Output</u>: Cl, the image class.

<u>Methods</u>:

(1) *Cl* = the cluster of *I* computed using Alg 7 (2) if *Cl* = 8, *Cl* = the class of *I* computed using Alg 8

Algorithm 9 Classifying a batik image using Algorithm 7 and 8 (shape- and texture-based).

5   Experiments and Result Discussion

A total of 489 batik images (consisting classical, modern, Pekalongan, Lasem, Madura, Cirebon, Belanda, Djawa-Hokokai batiks) have been collected. They are obtained from Doellah (2002), Kerlouge (2004), van Roojen (2001), Susanto (1981) and The Institute for Research and Development of Handicraft and Batik Industries, Yogyakarta. The batiks, which are in bitmap format, are used in a series of experiment with the aims of measuring the accuracy and performance of the algorithms. As we found no prior research report of batik images clustering and classification, here, we could only present quantitative measures with no comparison to other performance's methods.

5.1   Fashion Color Clustering

In the experiments, Algorithm 1 is run to cluster the batik images of three cropping sizes (64×4, 128×128 and 200×200), then, the clustering result is observed by viewing each image member of each skin and skin-hair clusters. The wrong members and total members are recorded to determine the percentages of error. Experiments result shows that Algorithm 1, in regard of clustering batik images suitable for certain skin color, works well as its error average is less than 5% (see Moertini (2005) for detail). The best clustering result is achieved for white skin color, and the worst is for black skin. There are many pastel color batik images found to be the member of black skin cluster. Where as black skin fits with bright colors, it is difficult to define the *H*, *S* and *V* classes for bright color without including the pastel color.

Table 4 shows that the error rate of the skin-hair color clustering result varies. The best error rate is achieved for yellow skin - brown hair and white skin - black hair, and the worst is for gray hair of white, yellow and brown skin. While gray hair would match with soft colors of batik, many bright color batiks have been mistakenly included in the clustering result. It is difficult to define the *H, S* and *V* classes for gray hair color of white, yellow and brown skin.

The result given on Table 4 suggest that the batiks used in the experiments are of nearly homogenous color, as the error rates of 64×64 pixels are just slightly higher than those of 200×200 pixels. By considering the execution times given on Figure 6 (where the cropping sizes do not contribute significantly to the execution times) and the error rates, here, the best cropping size is 128×128 pixels.

Table 4   Error rate of Algorithm 1.

Skin-	Hair Cluster	Image Size (pixels)		
		64×64	128x128	200×200
White-	Black (11)	0.7%	0.7%	0.7%
	Brown (12)	8.2%	8.2%	8.2%
	Gray (13)	28.6%	28.6%	28.6%
Yellow-	Black (21)	4.5%	4.5%	4.5%
	Brown (22)	0%	0%	0%
	Gray (23)	24.2%	18.8%	16%
Brown-	Black (31)	4.5%	3.9%	3.9%
	Brown (32)	4.5%	3.9%	3.9%
	Gray (33)	20.5%	20.3%	20.3%
Black-	Black (31)	17.6%	17.5%	17.5%
	Brown (32)	17.6%	17.5%	17.5%
	Gray (33)	8.3%	8.3%	8.3%
Average		**11.6%**	**11%**	**10.8%**
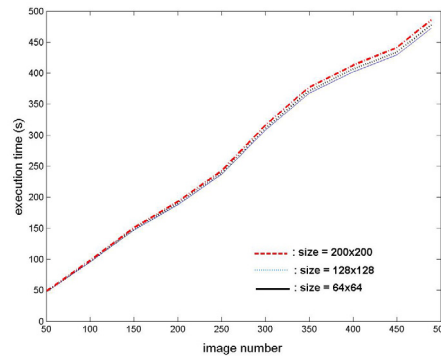
Figure 6   Execution time of Alg. 1.

5.2   Contrast Clustering

For Algorithm 2 (Alg 2), $tLow = 500$ and $tMed = 1900$ are used. (These values are selected after several runs using different values of $tLow$ and $tMed$. The ones that give the least error are picked). For Algorithm 3 (Alg 3), clustering using Haar wavelet, these values are used in experiments: $bLH1 = 30$, $bLH2 = 90$, $bHL1 = 40$, $bHL2 = 80$, $bHH1 = 40$, and $bHH2 = 100$.

Each of the 489 images being clustered is observed and marked as having low, medium or high contrast. The result is: 28 images are marked as low, 267 images are medium and 194 are high contrast. All of the images are then presented into Alg 2 and Alg 3 three times, by cropping them into 64×64, 128×128 and 256×256 pixels. To obtain the clustering error rate, the output of the run, which is the cluster member of each image, is compared with the clusters being set manually. The result is given on Table 5.

Table 5   The error of Algorithm 2 and 3.

Error	Algorithm 2			Algorithm 3		
	Image Size (pixels)			Image Size (pixels)		
	64×64	128×128	256×256	64×64	128×128	256×256
L as M	14	15	16	23	26	25
L as H	0	0	0	3	3	3
M as L	31	18	11	0	0	0
M as H	39	40	47	68	52	20
H as L	19	7	3	2	0	0
H as M	94	84	85	40	26	3
Total	197 (**40%**)	164 (**34%**)	162 (**33%**)	136 (**28%**)	107 (**22%**)	51 (**10%**)

L = low, M = medium, H = high contrast

Alg 3 performs better than Alg 2 as it gives smaller error rate for all size of images, and the larger size of the image cropping, the better the accuracy. Alg 3 fails in detecting low contrast images (error is 23/28, 26/28 and 25/28) but Alg 2 does better in detecting low contrast (error is 14/28, 15/28 and 16/28). But, where as Alg 3 never clusters images of medium and high contrast into low contrast, Alg 2 mistakenly clusters a number of medium and high contrast images into low contrast. The error of Alg 3 in clustering medium contrast images into high contrast is rather high (68/267, 52/267 and 20/267). This is due to the fact that human tend to view an image as a whole, and the contrast level is determined by comparing "segment" to "segment" of image. On the other hand, wavelet determines the contrast by comparing the intensity level of a pixel to a pixel. This gives result that a very fine image texture, with high different of intensity among a number of its pixels, is clustered into high contrast one. By comparing Fig. 7(a) and (b), it could be seen that Alg 3 executes three times as fast as Alg 2. Given the fact that the accuracy of Alg 3 is also better than Alg 2, it could be concluded that Alg 3 is a better algorithm.
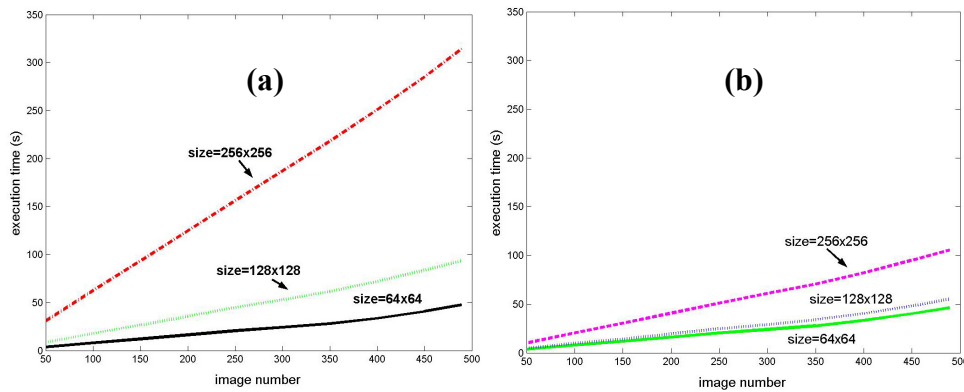

Figure 7   The execution time of image clustering using (a) CMT (b) Haar wavelet for images of size 64×64, 128×128 and 256×256.

5.3   Motif Classification and Clustering

The algorithms, except Algorithm (Alg) 7, are run several times with different value of nearest neighbors, k. The classification error for each run is observed and recorded. The best results, the ones that give the smallest percentages of error, are given on Table 8 and the execution times are given on Figure 8. It could be concluded that Alg 4,5 and 6 are inappropriate algorithms for classifying batik images, as their errors are greater than 50%. Alg 7 and 8 give promising results, and when both are combined (as Alg 9), the error rate is 19.3%, where as the execution time falls between the two.

Table 6   The error of Alg. 4 to 9.

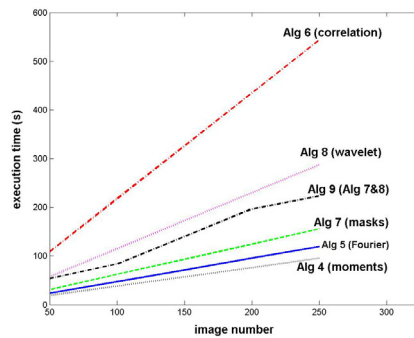| Algorithm	k	Error (%)
4 (moments)	9	66.7
5 (Fourier descriptor)	5	65.9
6 (correlation)	1	51.5
7 (masks)	-	23.1
8 (wavelet)	1	20.4
9 (Alg 7 & 8)	1	19.3

Figure 8 Execution time of Alg. 4 to 9.

The high errors of Alg 4,5 and 6 are due to the existing of *isen-isen*. The outputs of Canny edge detector show that the main ornament shapes defining batik motifs are mixed with small *isen-isen* shapes. Alg 4,5 and 6 fail in detecting the main shapes, or they could not distinguish which edges are main shapes and which ones are *isen-isen*. The method of Alg 7, which is detecting diagonal lines for clustering batik images into *parang*, *ceplok* and *other*, is quite effective. When observed, the images that are miss-clustered are the ones having unclear lines or lines disrupted by *isen-isen*. Alg 8, a texture-based classifier, performs quite well. The issue of this algorithm is selecting the smallest number of representative training images, which could reduce the error

and execution time. Alg 9 combines the advantages of Alg 7 and 8, which works well in detecting *parang-ceplok* motif and textures. Therefore, to reduce the error, the problems of Alg 7 and 8 should be resolved. The non-straight line of the Alg 9 execution time on Fig. 8 is due to the variety of the images, if lots of images being clustered are of *parang-ceplok* motif, the execution time would be less (as only Alg 7 is executed) compare to when lots of images are of non *parang-ceplok* (Alg 7 and 8 are executed).

6   Conclusion and Further Research

In this research, we have explored many existing methods of clustering and classifying images, used them in our algorithms, and proposed our own algorithms. We have also conducted experiments to compare the performance of all algorithms. The experiment results show that: (1) Color-based clustering algorithm works quite well, but further work is needed especially to search the right combination of *H,S,V* color component value for the gray hair owners. (2) Contrast-based algorithm using Haar wavelet also performs quite well. The issue remains to be solved is how to determine the contrast as how human do (human tend to compare segment to segment of images). (3) The best of clustering and classification algorithms based on batik motif is the algorithm that uses mask (shape-based) and wavelet (texture-based), but the accuracy is not very high. The main issue in shape-based classifying is how to "filter-out" *isen-isen* from the main shapes, and in texture-based is preparing the most representative training images, as well as finding a better method that could lead to better accuracy. Both groups of algorithms also need to be improved in dealing with different sizes and angles of ornaments. For better classifying batik motif, future work should classify batik images based on detailed motif (*kawung, nitik, buketan, semen*, etc.) that has specific meaning, as well as determining the origin of batiks (Pekalongan, Solo, Lasem, etc.), which also important in finding the meaning of the motif.

The algorithms discussed in this paper might also be used to analyze other textile images but with some modifications. Further interesting work would include experimenting the algorithms using other type textile images, finding the problems and modifying the algorithms to solve the problems.

References

1.   Doellah, Santosa, *Batik: Pengaruh Jaman dan Lingkungan*, Danar Hadi Solo, Indonesia, (2002).
2.   Hardisurya, Irma, *Warna bagi Citra dan Penampilan*, Gaya Favorit Press, Jakarta, (2004).

3.   Moertini, V. S., *Technical Report: Image Data Pre-Processing as a Step of Data Generalization for Object-Relational DBMS, Case Study: Batik Images*, Dept. of Informatics Eng, ITB, Bandung, (2005).

4.   Kompas Online, *Batik Tidak Mengenal Batas Pengembangan,* http://www.kompas.com, (19 Agustus 2001).

5.   Kerlogue, F., *The Book of Batik*, Archipelago Press, Singapore, (2004).

6.   Hamzuri, *Classical Batik*, Penerbit Djambatan, Jakarta, (1981).

7.   van Roojen, P., *Batik Design*, The Pepin Press, Singapore, (2001).

8.   The Institute for Research and Development of Handicraft and Batik Industries, *Handbook of Indonesian Batik*, Lembaga Penelitian dan Pendidikan Industri, Dept. Perindustrian, Indonesia, Yogyakarta, (1997).

9.   Susanto, S. S. K., *Seni Kerajinan Batik Indonesia*, Balai Penelitian Batik dan Kerajinan, Lembaga Penelitian dan Pendidikan Industri, Dept. Perindustrian R.I., (1980).

10.   Foley, van Dam, Feiner & Hughes, *Computer Graphics Principles and Practice*, Addison-Wesley Co., (1996).

11.   Mallat, S., *A Wavelet Tour of Signal Processing*, Academic Press, USA, (1999)

12.   Bartolini, I., *Efficient and Effective Similarity Search in Image Databases*, Dissertation, Dip. Elet. Informatika Sis., Univ. Degli Studi, Bologna, Italy, (2001).

13.   Natsev, A., Rastogi, R. & Shim, K., *WALRUS: A Similarity Retrieval Algorithm for Image Databases*, Duke University and Bell Laboratories, USA, (1999).

14.   Moertini, V. S., *Towards Classifying Classical Batik Images*, Proc. of Intl. Conference on Information and Communication Technology, UMB, Jakarta (June 9-10, 2005).

15.   Canny, J., *A Computational Approach to Edge Detection*, IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol. PAMI-8, No. 6, pp. 679-698, (1986).

16.   Lu, G., *Multimedia Database Management Systems*, Artech House, London, (1999).

17.   Hirata, K. & Kato, T., *Query by Visual Example Content Based Image Retrieval*, Lecture Notes on Computer Science, Advances in Database Technology- EDBT 92, Vol. 580, (1992).

18.   Sajjanhar, A. & Lu, G., *A Comparison of Techniques for Shape Retrieval*, Gippsland School of Computing and IT, Monash Univ., Australia, (1998).

19.   Wang, J. E., Wiederhold, G., Firschein, O. & Wei, S. X., *Content-Based Image Indexing and Searching Using Daubechies' Wavelets*, *International Journal of Digital Libraries*, Springer-Verlag, Vol.1, pp.311-328, (1997).