Volume 5, Issue 12, December 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Image Storage Using Bit-Sequential Technique

Basava Raju. K
CSE Dept JNTUK Scholar
India

Dr. Y. Rama Devi
HOD of CSE Dept, CBIT, OU
India

Dr. P. V. Kumar
CSE Dept., O.U., Telangana,
India

Abstract*— Recent advances in technologies such as image digitization, storage and transmission caused the number of digital images to increase tremendously. Therefore, content-based image classification and retrieval system has been the subject of many multimedia data mining research works in recent years. Image mining deals with the extraction of implicit knowledge, image data relationship or other patterns not explicitly stored in the image databases. The main objective is to emphasize on efficient image transformation and retrieval for fast image recognition. Image mining deals with the extraction of image patterns from a large collection of images, whereas the focus of computer vision and image processing is in understanding and/or extracting specific features from a single image.

Keywords*— BSQ, Image mining, image Indexing and retrieval, bands.

I. INTRODUCTION

Image mining deals with the extraction of implicit knowledge, image data relationship, or other patterns not explicitly stored in the image database. Many issues of image mining [1] such as image processing feature extraction, image indexing and retrieval and, pattern and knowledge discovery can be optimized with different data mining techniques. These issues have not been considered for image retrieval. In this paper, emphasis is on the performance of transformation and retrieval processes using BSQ (bit Sequential) the new storage format of the images and transformation processes. The data is in BSQ format, with each line of the data followed immediately by the next line in the same spectral band. This format is optimal for spatial (X, Y) access of any part of a single spectral band. The main objective is to emphasize on efficient image transformation and retrieval for fast image recognition. For content-based analysis, images are first processed to extract a set of descriptors or features that represent the colors, textures, and shapes in the image. Many methods, such as Euclidean and Minkowski distance functions draw inferences about image similarity or semantic category-label etc using all of these features. Early systems, such as QBIC [2], VisualSEEK [3], Netra [4] and MARS [5], facilitate classification, indexing, and retrieval of images, mainly, based on low-level features of images, such as color, texture and shape. However such methods, which rely on the complete feature set, have been found to not model perceptual similarity adequately.

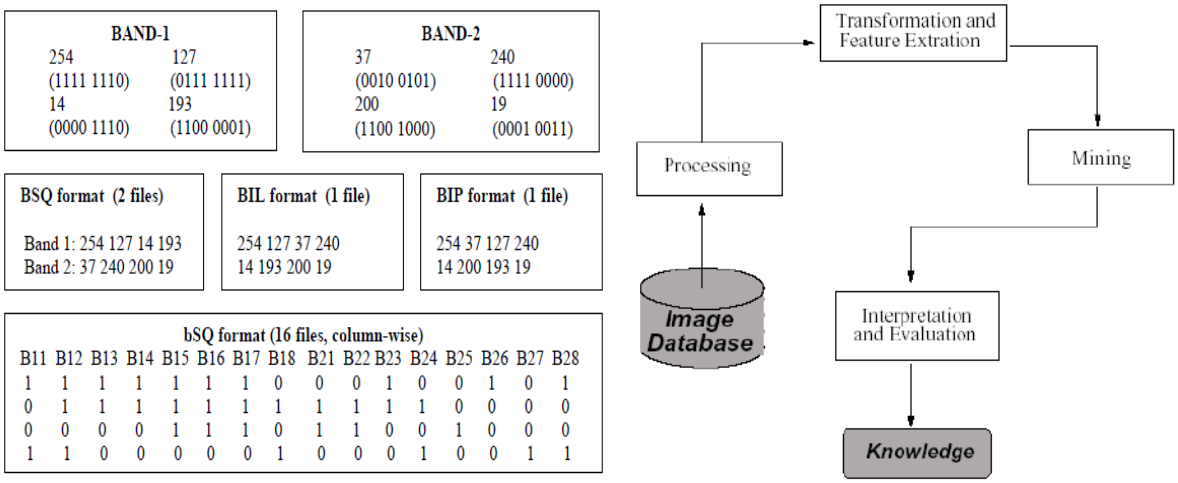

Figure 1(a) convert bands of pixels into bsq format for each region of image
Figure 1(b) General Image Mining System.

Figure 1(a) Shows how to convert bands of pixels into bsq format for each region of image.
Pixel level features describe spectral and textural Information about each individual pixel. Polygon level features describe connected groups of pixels. In the segmentation process, each polygon is described by its boundary and by a number of attributes that present information about the content of the region in terms of shape, size, etc. The spectral and texture properties are based on pixel features of points within the polygon. Tile level features present spectrum and texture information about whole image tiles.

II. IMAGE MINING SYSTEM

A general structure model for image mining System is shown in figure 1(B). The system considers a specified sample of images as an input, whose image features are extracted to represent concisely the image content. After representing the image content, the model description of a given image - the correct semantic image interpretation - is obtained. Mining results are obtained after matching the model description with its complementary symbolic description. The symbolic description might be just a feature or a set of features, a verbal description or phrase in order to identify a particular semantic.

Image mining system has two main themes. The first is mining large collections of images and the second is the combined data mining [6] of large collections of images and associated alphanumeric data. A vast collection of raw image data should be mined to discover new and valuable knowledge. Raw images consist of a 2-D array of pixels, usually named *iconic* format. An image database containing raw information cannot be used for mining purposes. All low level information must be preprocessed and transformed into a new suitable format. Numerous image mining techniques are based - explicitly or implicitly – in image models redefined at different representation levels.

These new representations try to reflect invisible image features established by association, resemblance, or convention; what is called *symbolic* representation. Given the variety of known image formats and the significant quantity of information every image represents - explicitly o implicitly -, it is necessary to select the only format representing the required information for knowledge processing.

2.1 Preprocessing

During preprocessing stage, an image set is selected from image database to be the minable view. The most difficult task of this process is to know the image domain. Sometimes, it is necessary a human expert who makes the required image adjustment and selection or just a query is executed on the database to retrieve the training images. Most of the time, once images are selected, segmentation and clustering techniques are applied to divide images into significant blocks for mining.

2.2 Recovery and Association

All existing mining paradigms depend on input and output data types. Input data types consider two kinds of image representation: iconic o symbolic. Any adopted paradigm considers as input image set a certain image class representing specific semantic entities: objects, concepts, among others and the possible relationships defined among the entities and its attributes.

2.3 Content Based Recovery

It finds in the databases those images similar or semantically equivalent to an input image. The aim is to resolve the recovery by means of keyword-based methods that consider human perception features, so users must express their queries in a semantic way. This kind of task have two disadvantages: low level of retrieval accuracy and high response time for huge dimensional data spaces, resulting in poor performance systems.

2.4 Pattern Recognition

It involves the use of a database with reference images representing diverse semantic entities and its different visions. As a result of a query the user obtains the entity or requested object or a ―no‖ answer if the database does not contain it. Patterns[7] also can be used to represent complex knowledge combining diverse levels of abstraction. This is a process that allows new pattern[8] discovery or to validate the existing ones.

2.5 Image model description

Consist of transforming the iconic description of an image into the symbolic one using a particular model structure and a set of parameters.

Besides input and output data types, two other basic tasks must be kept in account: knowledge search and extraction. To develop the first one, the majority of content base image recoveries are applied. For the second one, an input sample image is specified and a symbolic description of the sample image is get as the output.

The above project work revealed on data mining processes for image retrieval and transformations. The construction of the image data formats has not been concentrated. Hence, the main disadvantage is less performance transformation and retrieval from large image databases.

III. BIT SEQUENTIAL FORMAT(BSQ)

The following is an example of the band sequential (BSQ) file format as it would be written for the following figure. An actual .bsq file will be binary; however, for the purpose of this example it will be shown using ASCII characters.

RGBcomposite.bsq

0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
64 64 64 64 64 64 64 64
64 64 64 64 64 64 64 64
128 128 128 128 128 128 128 128

128 128 128 128 128 128 128 128
255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255
0 0 64 64 128 128 255 255
0 0 64 64 128 128 255 255
0 0 64 64 128 128 255 255
0 0 64 64 128 128 255 255
0 0 64 64 128 128 255 255
0 0 64 64 128 128 255 255
0 0 64 64 128 128 255 255
255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255
128 128 128 128 128 128 128 128
128 128 128 128 128 128 128 128
64 64 64 64 64 64 64 64
64 64 64 64 64 64 64 64
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0

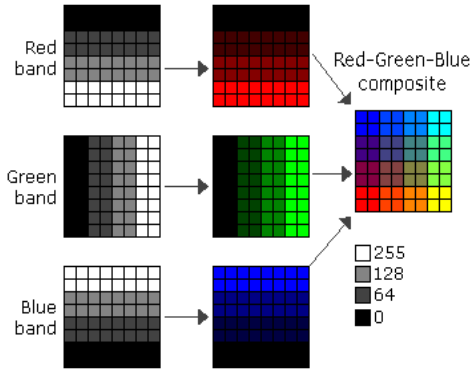

Figure2: BSQ Image composition

The red, green, blue colors separation from the .bsq image and also giving the relevant integer values to relevant colors with respect to the color contrast is shown in Fig.2.

3.1 Image Description Files

There are three description files that can be provided with BSQ file: a header file (hdr) that describes the layout of the image pixel data and must be provided, a color file (clr) that describes the image color map, and a statistics file (stx) that describes image statistics for each band of the image. These ASCII text files can be generated in a text editor using the information you know about the image.

Band row bytes—The number of bytes per band per row. This must be an integer. Bandrowbytes can be thought of as an index to the starting point of the next band of data. Starting at the beginning of any band in a row, moving bandrowbytes along that row leads to the beginning of the next band.

The figure 4 illustrates one row of data for a three-band image.

Band row bytes = image data + extra bits

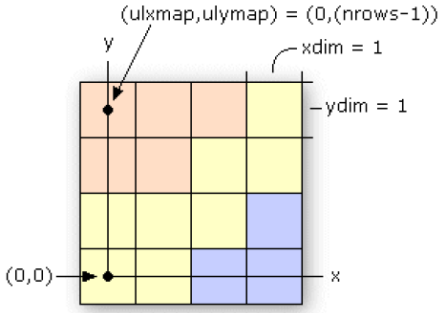

Figure 3: An example for different mapping values.

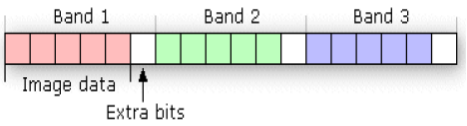

Figure 4: An example for bandrowbytes

To set bandrowbytes, you must know the layout of the image data or, more specifically, how many bytes are used to store pixel values for each band in a row. If bandrowbytes is not specified, a default value is calculated with the following equation:

bandrowbytes = the smallest integer(ncols x nbits) / 8

The default value handles cases when there are no extra trailing bits at the end of each band in a row and when the number of bytes per band per row is the smallest integer number of bytes that will adequately store the data for the band; for example, if the data requires 2.5 bytes, 3 bytes is the smallest integer number of bytes that could store the data. In these two cases, bandrowbytes does not need to be set. If, however, the number of bytes per band per row is greater than the default, set bandrowbytes accordingly.

The following two examples show the default behavior of bandrowbytes. The first example describes the case for which there are no trailing bits at the end of a band in a row, and the second describes the case for which there are. If there are no extra trailing bits at the end of a band, bandrowbytes equals the number of bytes used to store the image data.

bandrowbytes = image data

Figure 5: An example of bandrowbytes2

For example, given a 6-by-6 image with three bands and 8 bits (1 byte) per pixel, the image data will require 6 bytes per band per row.

bytes per band per row = ncols x nbits =6x8=48bits Or 6bytes

By default, bandrowbytes is set to 6 bytes, as shown by the following equation:

bandrowbytes = (ncols x nbits) / 8 = (6 x 8) / 8 = 48 / 8 bandrowbytes = 6 bytes

Because the number of bytes per band per row equals bandrowbytes, the default value is the appropriate setting. Thus, bandrowbytes does not need to be explicitly specified.

If there are trailing bits at the end of a band in a row, then bandrowbytes does not equal the number of bytes of data per band per row.

Suppose you have a three-band image of 5 rows and 5 columns with 4 bits per pixel. By default, bandrowbytes is set to the smallest integer number of bytes that will adequately hold the data. In this case, the default value is 3. This is calculated as follows:

bandrowbytes = (ncols x nbits) / 8 = (5 x 4) / 8 = 20 / 8 = 2.5 =3 (when rounded up to the nearest integer)

The image data, however, only requires 2.5 bytes, which is calculated by multiplying ncols by nbits. Thus, the number of bytes that will be skipped is .5 bytes (4 bits), or the difference between 3 bytes (bandrowbytes) and 2.5 bytes (image data bytes). The figure below shows one band of data for one row of the image.

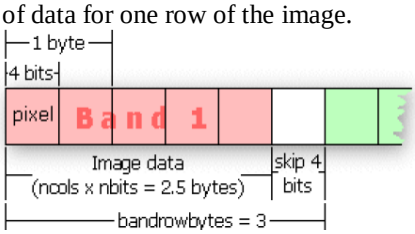

Figure6: An example for image data

totalrowbytes—The total number of bytes of data per row. Use totalrowbytes when there are extra trailing bits at the end of each row.

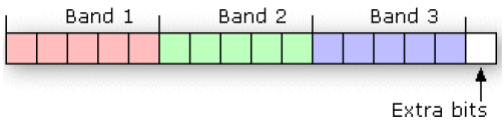

Figure 7. An example for itotal row bytes

For a BIL file, the default value for totalrowbytes is calculated with the following equation:

totalrowbytes = nbands x bandrowbytes

The default value assumes that there are no extra trailing bits at the end of each row. If there are, set totalrowbytes accordingly. For example, given a three-band image with bandrowbytes equal to 3, totalrowbytes will equal 9 by default. If there is an extra trailing byte of data at the end of the row, set totalrowbytes to 10.

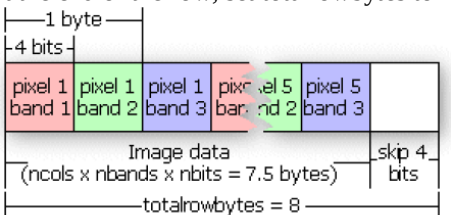

Figure8 : An example for itotal row bytes pixel and band wise.

This figure illustrates the default setting of totalrowbytes for a BIP image. If the default value for total row bytes does not accurately represent the layout of the data, totalrowbytes must be set to the appropriate number of bytes in each row.

bandgapbytes—The number of bytes between bands in a BSQ format image. The default is 0. The following is a typical header file that might be generated for a band interleaved by line satellite image where the image data is preceded by a 128-byte header. Sample BIL header file Lines that don't begin with a keyword are treated as comments. nrows 1024 Comments can be placed here as well. ncols 1024 nbands 3 nbits 8 layout bilskip bytes 128

Table: Gives a summary of the keywords that can be used in the .hdr file:

The color file

The color file (.clr) is an optional file that describes the image color map for single-band pseudo color images. If this file doesn't exist, the image displays as a gray scale[9] image. The color file records the colors to be associated with pixel values in the image. Colors are defined using the RGB color model that describes colors in the amount of red, green, and blue they contain. The file consists of a set of entries, each on a separate line that describes the color corresponding to a pixel value in the image.

Table 1. Each entry has the format:

The methodology of over all work includes as Follows but in this paper we are presenting only primary work like how to convert the give image into bit-sequential format.

1. Select an Input Image.
2. Detect skin area in Input Image
3. Detect Features.
4. Save Face into Database with its record, class label, attributes, and age.
5. Design a learning data set for more images.
7. The values are converted into bSQ file format.
8. Spatial data organization using Peano count tree.
9. Decision Tree is constructed using ID3 method to classify the data.
10. Using the Classified data test image age is predicted.

<value> <red> <green> <blue>

in which <value> is a given pixel value and <red>, <green>, and <blue> are the color components for the pixel. Sort all entries in ascending order by pixel value. If the first nonblank character on the line is not a number, the line is considered a comment and is ignored. Any nonblank characters in a line beyond the fourth parameter (blue) are ignored and may be used as comments as well.

The red, green, and blue components are described using a scale with values ranging from 0 to 255. As the color value increases, so does the intensity of the particular color component.

The default color for a pixel value with no entry is black. A sample color file for a raster soils map with pixel values of 11, 16, 18, 19, 21, 98, and 99 is shown below:

<value> <red> <green> <blue>

in which <value> is a given pixel value and <red>, <green>, and <blue> are the color components for the pixel. Sort all entries in ascending order by pixel value. If the first nonblank character on the line is not a number, the line is considered a comment and is ignored. Any nonblank characters in a line beyond the fourth parameter (blue) are ignored and may be used as comments as well.

<value> <red> <green> <blue>

in which <value> is a given pixel value and <red>, <green>, and <blue> are the color components for the pixel. Sort all entries in ascending order by pixel value. If the first nonblank character on the line is not a number, the line is considered a comment and is ignored. Any nonblank characters in a line beyond the fourth parameter (blue) are ignored and may be used as comments as well.

The red, green, and blue components are described using a scale with values ranging from 0 to 255. As the color value increases, so does the intensity of the particular color component. The default color for a pixel value with no entry is black. A sample color file for a raster soils map with pixel values of 11, 16, 18, 19, 21, 98, and 99 is shown below:

Color file for Soils map

Entries are sorted in ascending order by pixel value.

11 255 0 0 (red)
16 255 165 0 (orange)
18 255 255 0 (yellow)
19 0 255 0 (green)
21 0 0 255 (blue)
98 0 255 255 (cyan)
99 160 32 240 (purple)

Color files are only used with single-band images. Any single-band image with a color file will be interpreted as a pseudo color image. Color files that accompany multiband images are ignored.

The statistics file

The statistics file (.stx) is an optional file that describes image statistics for each spectral band in a grayscale or multiband image. The file consists of a series of entries, one for each band, that records the minimum pixel value, the maximum pixel value, the mean, the standard deviation, and two linear contrast stretch parameters.
Each entry has the format (all values appear on the same line in the file for each band):

<band> <minimum> <maximum> {mean} {std_deviation} {linear_stretch_min} {linear_stretch_max}

in which <band> is the band number, <minimum> is the minimum pixel value in the band, <maximum> is the maximum pixel value in the band, {mean} is the mean pixel value, {std_deviation} is the standard deviation, {linear_stretch_min} is the minimum pixel value for a linear contrast stretch, and {linear_stretch_max} is the maximum pixel value for a linear contrast stretch.

Keyword	Acceptable Value	Default
Nrows	any integer > 0	None
Ncols	any integer > 0	None
Nbands	any integer > 0	1
Nbits	1, 4, 8, 16, 32	8
Byteorder	I = Intel; M = Motorola	same as host machine
Layout	bil, bip, bsq	Bil
Skipbytes	any integer ≥ 0	0
Ulxmap	any real number	0
Ulymap	any real number	nrows – 1
Xdim	any real number	1
Ydim	any real number	1
bandrowbytes	any integer > 0	smallest integer ≥ (ncols x nbits) / 8
totalrowbytes	any integer > 0	for bil: nbands x bandrowbytes; for bip: smallest integer ≥ (ncols x nbands x nbits) / 8
Bandgapbytes	any integer ≥ 0	0

The values for each parameter are entered on one line. Any entry in which the first nonblank character is not a number is treated as a comment and ignored. The band number and the minimum and maximum pixel values are required parameters; the mean, the standard deviation, linear stretch minimum value, and linear stretch maximum value are optional parameters. Use a "#" to skip the optional parameters.
Band numbers can range from 1 to nbands. The linear_stretch_min and linear_stretch_max parameters are used to expand the contrast of the displayed image. Pixel values less than linear_stretch_min are displayed in black, and pixel values greater than linear_stretch_max are displayed in white. The pixel values that fall between the minimum and maximum linear stretch parameters are displayed using shades of gray, with lower pixel values being displayed in darkershad of gray.

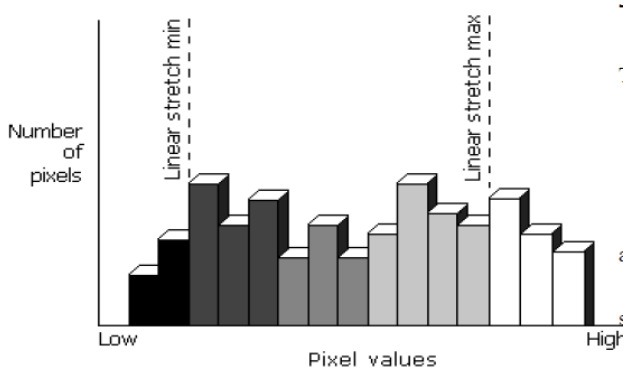

Figure 9: An example showing the ratio of pixel values and number of pixels

The pixel values that fall between the linear stretch parameters are displayed using the maximum number of gray shades available on the display device.

If linear_stretch_min and linear_stretch_max are not specified, they default to the mean minus two standard deviations for linear_stretch_min and the mean plus two standard deviations for linear_stretch_max. If the standard deviation is not given, the minimum and maximum pixel values are used as the contrast stretching parameters.

For multiband images, each band is stretched before the composite image displays. The presence of a color file (.clr) will override the linear contrast stretching of a single-band grayscale image and, instead, display the image as a pseudo color image.

The following is a sample statistics file for a four-band satellite image with 8 bits per pixel per band:

Image statistics file
1 2 118 67 10
Band 2 has linear contrast stretch parameters:
2 23 251 112 23 80 90
3 68 91 73 4
Band 4 does not contain values for mean and standard deviation:
4 126 198 # # 135 16

IV. IMPLEMENTATION

The proposed system is tested with several images. All images of the formats like jpeg, gif, bmp and png are converted into bsq format. The bsq images are stored into the database and retrieved on request, indexing also done to these bsq images.

4.1 Image conversion into .bsq format

The image conversion into .bsq format is given as follows:

Input: Image in the formats JPEG, BMP and PNG.
Output: Image in .BSQ format
Step 1: Image reading
Step 2: storing each pixel value of the image in an integer array
Step 3: for every pixel red, green, blue values are separated.
Step 4: Integer array wise red, green, blue values are converted into 8-bit stream.
Step 5: Row and pixel wise every red bits are appended.
Step 6: Row and pixel wise every green bits are appended.
Step 7: Row and pixel wise every blue bits are appended.
Step 8: Red, Green and blue bit stream values are stored into a text document with .bsq extension

4.2 Storing .bsq image into the database

The procedure for storing .bsq image into the database as follows:

Input: .bsq image.
Output: image is stored into the database.
Step 1: reading image.
Step 2: Converting image into .bsq format by using above procedure.
Step 3: Sending the image into the database.
Step 4: Storing the image into the database.

4.3 Image retrieval from the database

The procedure for retrieving .bsq image from the database as follows

Input: request for image.
Output: .bsq image.
Step 1: index is taken.

Step 2: Searching for the given index in the database.
Step 3: If the index is found the image is displayed else ―the image ia not found in the database‖ message will be displayed.

4.4 Comparison between two images

The procedure for comparing two .bsq images as follows

Input: Two .bsq images.
Output: Result.
Step 1: Two images are taken.
Step 2: These two images are converted into .bsq format.
Step 3: Each bit value of the first image is compared with the respect bit value of the second image.
Step 3: If the comparison gives TRUE value then the message ―Two images are same‖ is displayed else the message ―Two images are different‖ is displayed.

V. RESULT

The system is developed in MATLAB. The data sets are collected the books of Digital Library of India (http://dli.iiit.ac.in). The experimental results show that the proposed method is reliable to convert any image format into .bsq format even though there is a small color variation as in figure 10 and fig11. The comparison technique is more accurate as each bit values are compared.

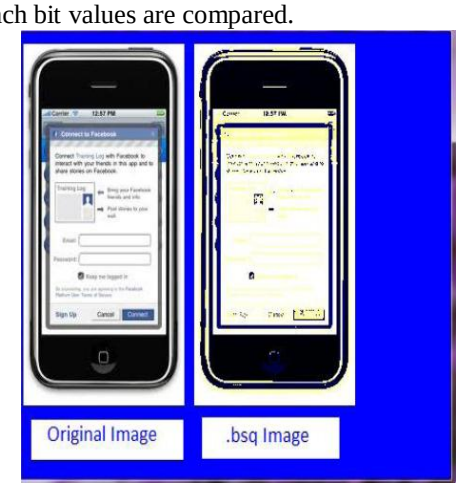

Figure 10: Image conversion to .bsq format.

Figure 11: Retrieving an image from the database.

Figure 12 : Band1 Image Pixels

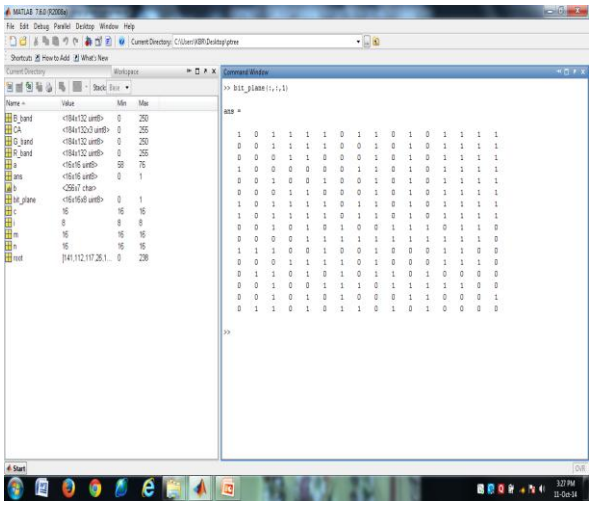

Figure 13: Band1_bSQ plane1 File

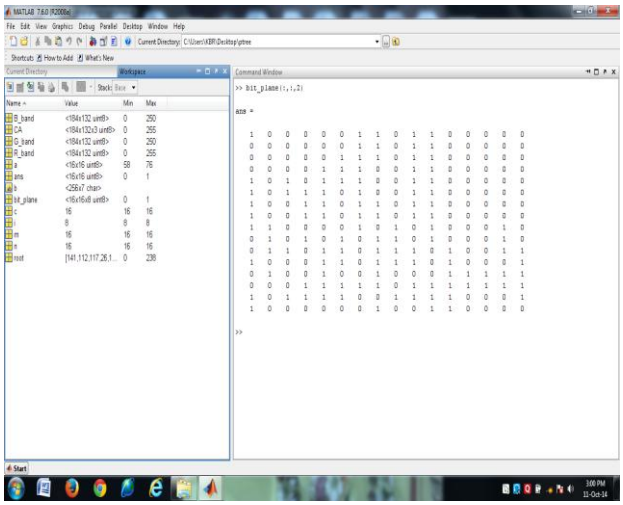

Figure 14: Band1_bSQ plane2 File

For a given face image 3 bands are generated which is shown in figure 12 and appropriate equivalent bsq equivalent planes are generated by using bsq files which are show in figure 13 and figure 14.

VI. CONCLUSION

The proposed system is tested intially with several images. All images of the formats like jpeg, gif, bmp and png are converted into bsq format. The bsq images are stored into the database and retrieved on request, indexing also done to these bsq images. After that we applied this on Yale and CAPSEAL databases and obtained effective results.

The results show that the proportionality of accuracy depends on the size of image and clarity of image. The red, green and blue colors presented in the .bsq image are separated and stored independently in bits hence maximum accuracy is possible. Finally by storing images in .bsq format we can reduce the amount of memory which is required for the image. It is advisable to store the image in bsq format by higher order bits of each pixel of given images. so that we can save around half of the storage space when comare with other few popular methods.

REFERENCES

[1] J.Zhang, W.Hsu, M-L.Lee. ―Image Mining: Issues, Frameworks and Techniques‖. Int'l Workshop on Multimedia Data Mining (MDM/KDD) 2001: 13-20.

[2] M.Flickner, H.Sawhney, J.Ashley, Q.Huang, B.Dom, M.Gorkani, J.Hafner, D.Lee, D.Petkovic, D.Steele, P.Yanker ―Query By Image and Video Content: The QBIC System‖. IEEE Computer Magazine, Sep. 1995.

[3] J.R.Smith, S-F.Chang. ―VisualSEEK: A fully automated content-based image query system‖. ACM Multimedia Conf., Nov. 1996.

[4] W.Y.Ma, B.S.Manjunath. ―Netra: A Toolbox for Navigating Large Image Databases‖. IEEE Int'l Conf. On Image Processing (ICIP), Oct. 1997.

[5] S.Mehrotra, Y.Rui, M.Ortega, T.S.Huang. ―Supporting Content-Based Queries over Images in MARS‖. IEEE Int'l Conf. On Multimedia Computing and Systems, 1997

[6] Ming-Syan Chen; Jiawei Han; Yu, P.S., "Data mining: an overview from a Data-base perspective", Knowledge and Data Engineering,IEEE Tran on,Vol 8 Issue 6,Dec,1996 ,P866-883

[7] R.duda and P.Hart. " pattern classification and seene analysis". Bayes Decision Theory.John Wiley & Sons.pp-10-13.1973

[8] G. Seni and E. Cohen, "External Word Segmentation of Off- Line Handwritten Text Lines," Pattern Recognition,vol. 27, no. 1, pp. 41-52, Jan. 1994

[9] N. Otsu. (1979): A threshold selection method from gray-level histograms, IEEE transactions on systems, man,and cybernetics Vol.SMC-9,No.1.

