Histogram Modification Of Colour Images

Rajesh Bhimrao Khotre, Asst.Prof,PVPPCOE,Mumbai.
E-mail- rajesh_khotre@rediffmail.com

ABSTRACT

Digital images are enhanced for extracting the additional information that is not by itself perceivable prior to enhancement. Histogram modeling is considered one of the well known methods for enhancing the images. Histogram being a global description of an image in terms of probabilities of various intensity levels Histogram modeling is normally achieved in two steps namely histogram equalization and histogram specification.

Histogram equalization and histogram specification have been widely used to enhance information in a gray scale image, with histogram specification having the advantages of allowing the output histogram to beo specified as compared to the histogram equalization which attempts to produce an output histogram that is uniform. Unfortunately, expanding histogram techniques to colour images is not straightforward. Since humans are sensitive to chromatic changes, care must be taken to ensure that incorrect colours are not produced.

Additionally, expanding the 1-D histogram used in gray level histogram techniques to a joint histogram (usually of three variables representing the primary colours of red, green and blue) can yield specified histograms which have no physical meaning hence making it difficult to determine the set of histograms required for the desired enhancement. Method of gray level histogram specification is extended to colour images by performing histogram specification on intensity, saturation and hue components. These methods take into account the relationship between the intensity and saturation components while yield specified histograms that produce natural looking results.

I. INTRODUCTION

How does one can judge a scanned picture? More basically, what are the criteria by which one judges the quality of a scan? Conceptually simple factors such as focus and frame alignment are immediately obvious from viewing the monitor and are easily adjustable through simple controls or manual adjustment. Other factors such as colour and tonal balance are much more difficult to evaluate by looking at scanned image on the monitor. A full-frame scan on a scanner with a default resolution of 1350 dots per inch will generate approximately 6 million bytes. At a resolution of 2700 dpi, the file will contain over 24 million bytes. Obviously no one is intellectually capable of following such a mass of points individually. At the other extreme, the person who can consistently obtain quality scans just previewing images on the monitor is exceptionally talented indeed.

An Image Histogram is the way of making quantitative sense of this mass of data.

II. INTENSITY PROCESSING FOR COLOUR IMAGES

The RGB (red-green-blue) coordinate system is commonly used for representing digital colour images. However, coordinate systems related to the human visual system's perceptual attributes (intensity, hue, and saturation) are often useful for processing colour images. Much research has been done toward the development of colour measurement techniques, but there is not anyone colour coordinate system that is universally accepted as corresponding to human perception. There are several colour coordinate systems, each having its particular merits, that are being used in colour image processing. For examples, HSI and YIQ coordinate systems are the most popular.

- Quantization error may exist during coordinate transformations. The effect of quantization error on colour image processing is analyzed in order to minimize the artifacts that created by the error.
- The transformations between colour coordinate systems during the processing is very inefficient. Efficient techniques of intensity and saturation processing are developed to bypass the colour transformations.

A common approach for colour image processing is to first transform the RGB image to a new colour space (such as HSI and YIQ), do the processing, and then transform back to RGB space. In many applications, only the intensity component is processed, and the hue and saturation are preserved. The efficient luminance processing are done for HSI and YIQ colour coordinate systems. However, when the processed values in the new colour spaces are transformed back to RGB, the transformed values may lie outside the RGB cube. Some researchers clip the processed intensity values before transformation from the processing colour space to RGB space. This can lead to a loss of intensity contrast because many pixels may be clipped to approximately the same intensity value. Instead of clipping the intensity ,clipping the saturation component is proposed.

III. HISTOGRAM

The histogram of digital image with gray levels in the range [0 , L-1] is a discrete function p(rk) = nk / n, where rk is the kth gray level, nk is the number of pixels in the image with that gray level, n is the total number of pixels in the image, and k = 0,1,2,……..,L-1.

The histogram p(rk) of an image gives an estimate of the probability of occurance of gray level rk. A plot of this function for all values of k provides a global description of an image. For example, fig. 1 shows the four basic types of Histograms of an image. The histogram shown in fig. 1 (a) shows that the gray levels are concentrated towards the dark end of the gray scale range. Thus this histogram corresponds to an image with overall dark characteristics. Just the opposite is true in fig 1 (b). the histogram shown in the fig 1 (c) has a narrow shape, thus it corresponds to an image having low contrast. Finally, 1 (d) shows a histogram with significant spread, corresponding to an image with high contrast.

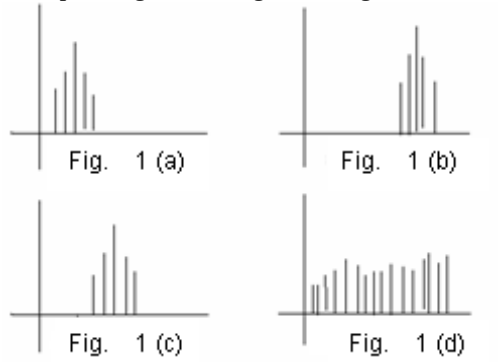


Fig. 1 Four basic types of Histograms of an image

Histogram Equalization

Histogram equalization is the important step of histogram processing. The goal of the histogram equalization is to obtain a uniform histogram. This technique can be used on a whole image or just on a part of an image.

Histogram equalization will not "flatten" a histogram. It redistributes intensity distributions. If the histogram of any image has many peaks and valleys, it will still have peaks and valleys after equalization, but the peaks and valleys will be shifted Because of this, "spreading" is a better term than "flattening" to describe histogram equalization.

Because histogram equalization is a point process, new intensities will not be introduces into the image. Existing values will be mapped to new values but the actual number of intensities in the resulting image will be equal or less than the original number of intensities.

Operation

1. Compute histogram.
2. Calculate normalized sum of histogram.
3. Transform input image to output image.

The first step is accomplished by counting each distinct pixel value in the image. You can start with an array of zeros. For 8 bit pixel of the array is 256 (0-255). Parse the image and increment each array element corresponding to each pixel processed.

The second step requires another array to store the sum of all the histogram values. In this array, element l would contain the sum of histogram elements I and 0. Element 255 would contain the sum of histogram elements 255, 254, 253 ,……,1 ,0. This array is then normalized by multiplying each element by (maximum pixel value / number of pixels). For an 8 bit 512 X 512 image that constant would be 255/252144.

Histogram Specification

Although the histogram equalization is quite useful method, it does not lead to interactive image enhancement applications. The reason is that it is capable producing only one result that is an approximation to a uniform histogram. Sometimes a uniform histogram is not what is desired, but the ability to specify particular histogram shapes capable of enhancing some particular feature of an image is what is required. This can be accomplished using histogram specification technique. Histogram specification is a simple process that requires both a desired histogram and the original image as input.

It is performed using following two steps:

1. The first is to histogram equalize the original image.
2. The second is to perform an inverse histogram equalization of desired image on the equalized original image

IV. IMPLEMENTATION

Image Enhancement Using RGB Model:

The first step in implementing colour image enhancement is to digitize an image and obtain its RGB components. Image in RGB colour model consist of three independent image planes one for each primary colour therefore while enhancing an image in RGB model, the obvious approach is to subject each image plane to histogram modeling independently. The intensities in all the three image planes will be altered differently, resulting in a change of the relative intensities between them due to which important colour properties in an image will not appear natural when viewed on an RGB monitor.

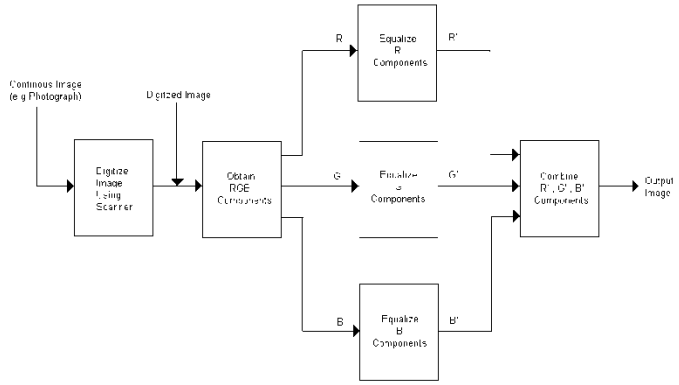


Fig. 2 Image Enhancement using RGB Model

Image Enhancement Using HSI Model:

The best approach for colour image enhancement is to convert RGB to HSI as the intensity components from the colour information in an image. The next step is to process intensity component and converting the result to RGB for display. The colour content of the image is not affected.

The following figure shows the entire process of colour image enhancement using HSI model.

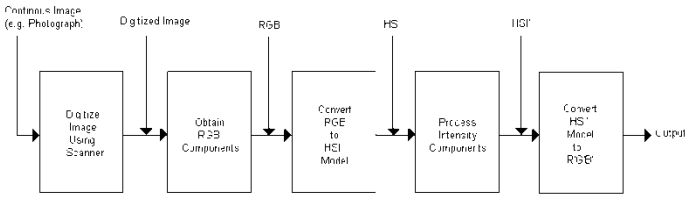


Fig. 3 Image Enhancement Using HSI Model.

It is also possible to equalize saturation and intensity individually or both together as required by the application. That is we can equalize both intensity and saturation converting HSI to HSI' and then converting back to the R'B'G' for display.

Basic Steps of Implementation:

1. Create a new project with the application wizard
2. copy the interface file (dibkernel.h) and implementation file (dibkernel.cpp) of C Dib class to the new project directory.
3. Insert these files in the project.
4. Include the interface file (dinkernel.h) of the Cdib class in the interface file of the application document class (C enhancement Doc.h)
5. Add a C Dib object, called m_dib as a member variable to the C enhancement Doc class.
6. Implement the various functions using applications view class (C enhancement view).

For ex = 1) CEenhancement view : : RGB to HSI ( )

This function extracts the RGB components of every image and converts them to HSI components in order for intensity component (I) to be processed separately from chromaticity components (i.e. Hue and saturation).

V. SOFTWARE TOOL USED: VISUAL C++

Using Bitmaps In VC++:

Windows bitmaps are arrays of bits mapped to display pixels. There are basically two types of windows bitmaps:

1. Device dependent GDI bitmap
2. Device Independent Bitmap (DIB)

Device Independent Bitmap Class (CDib) provides an easy access to the values of the pixels in the image as well as ways for manipulating pixels of an image, thus, enabling us to process individual pixels of an image. This class incorporates two files:

1. dibkernel.h
2. dibkernel.cpp

dibkernel.h

This is the interface file of Cdib class and it enumerates all the function implemented by Cdib class to process device independent bitmap image. The purpose of this file is to hide all implementation details of the CDib class. It specifies the declaration of all the functions along with arguments and return values.

Dibkernel.cpp

This is the implementation file of CDib class and it implements all the function listed in the interface file to process device independent bitmap images. These two files together form CDib class and provide an efficient way of manipulating the pixels of bitmap image.

Some of the most important member functions of CDib class are listed below:

DWORD CDib :: GetPixel (Cpoint pt) const :

This function takes co-ordinates of a particular point (pt) of an image as argument and returns the 32 bit DWORD value of pixel at that point.

DWORD CDib :: Setpixel (Cpoint pt, DWORD colour) :

This function takes two arguments, co-ordinates of a particular point (pt) of an image and 32 bit value of colour to be assigned to the pixel represented by the first argument.

Document / View architectur in VC++

In this paper CEnhancementDoc and CenhancementView forms document view architecture. Each of these two classes incorporates two files interface file with .h extension and implementation file .cpp extension.

Class CEnhancementDoc :

This is the document class and it declares the variable m_dib which is the object of class Cdib. The variable m_dib holds the bitmap image currently in the RAM. The object m_dib and its member functions can be accessed from the view class by using document pointer pDoc.

For example, to access GetPixel member function of m_dib object we use the following statement in view class :

PDoc → m_dib.GetPixel (Cpoint pt)

Class CEnhancementView :

This is the view class and it uses document pointer pDoc to access m_dib object declared in document class. It also implements all the function required to perform histogram equalization and histogram specification of colour images.

Functions implemented by this class include the following:

CEnhancementView :: OnEnhanceHistogramProcessingHistogram()

This function displays the histogram of an image with intensity levels on x axis and probabilities of the intensity on y axis.

CEnhancementView :: RGBtoHSI( )

This function extracts the RGB components of every image and converts them to the HSI components in order for intensity component (I) to be processed separately from chromaticity components (i.e. Hue and saturation).

CEhancementView :: HSItoRGB( )

This function converts the HSI components of every pixel of an image back to the RGB components of display.

CEnhancementView :: OnEnhanceHistogramProcessingEqualization( )

This function first calls the RGBtoHSI( )function depending upon whether the input image is colour or grayscale image, then it performs histogram equalization of either intensity component (I) or saturation component (S) as required. Then it calls HSItoRGB( ) function to convert processed HSI components back to RGB components for display.

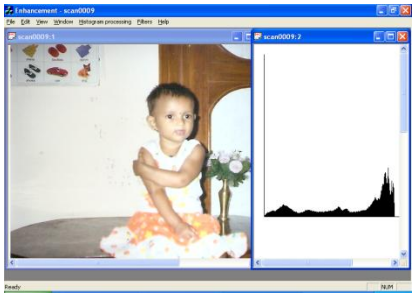
If RGB model is to be used, then this function equalizes R, G & B components separately.

CEnhancementView :: OnEnhanceHistogramProcessingSpecification( )

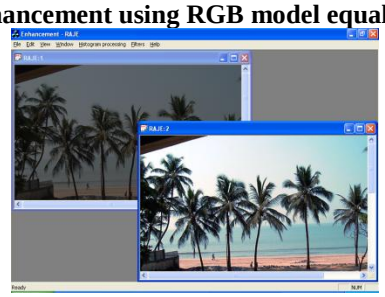
This function operates on the input image in the same way as histogram equalization, the only difference is that histogram specification allows user to specify the desired histogram. On the basis of the selected histogram, the histogram specification is performed on intensity component (I) to produce enhanced intensity component (I') whose histogram is approximately same as the specified histogram.

VI. RESULTS

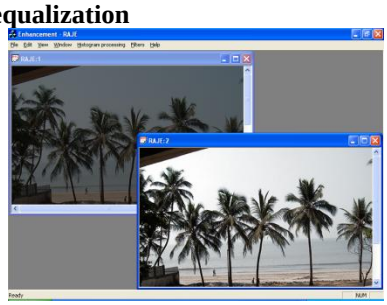
1. Histogram of an image



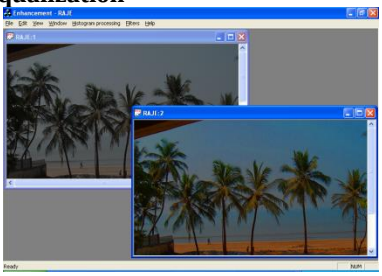
2. Image enhancement using RGB model equalization



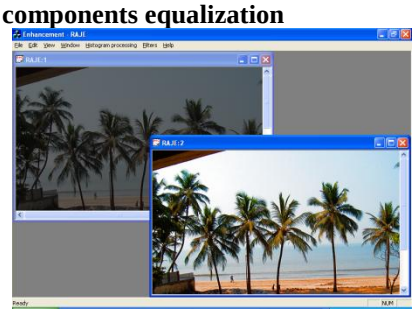
3. Image enhancement using HSI model with Intensity component equalization



4. Image enhancement using HSI model with Saturation component equalization



5. Image enhancement using HSI model with Intensity and Saturation components equalization



REFERENCES

[1] Christopher C. Yang, and Jeffery J. Rodriguez, "Graphics models and image processing".
[2] htpp://www.imada.ou.dk/~edr
[3] htpp://www.mip.ou.dk/~hansm
[4] Digital image processing- by Gonzalez and Woods
[5] Programming with VC++ by Sanghavi
[6] Microsoft visual C++ Reference book
[7] htpp://msdn.microsoft.com/visualc