ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (127-132), Month: April 2015 – September 2015, Available at: www.paperpublications.org

Study of Color Image Processing For Capturing and Comparing Images

Richa Gupta

Department of Information Technology Banasthali Vidyapeeth University Jaipur-302001, Rajasthan-India

Abstract: **Study of processing techniques applicable to full color images. Although the process followed by the human brain in perceiving and interpreting color is a phenomenon that is fully understood, the physical nature of color can be expressed on the formal basis supported by experimental and theoretical results. By examining some principal areas it can be applied for detecting, Capturing, Comparing the images. The objective of this article is to educate newcomer to basic and fundamental techniques of detecting & Comparing images and to find out the compared result. All fundamental algorithms of color image processing will be discussed and quality of processed image output comparison will be shown to find out comparison.**

Keywords: **color models, color detection, pixel by pixel image comparison, open CV for image capturing and comparing.**

I. INTRODUCTION

Color Image Processing is motivated by two basic principals factors. First, color is powerful descriptor that often simplifies object identification and extraction from a scene. Second, humans can discern thousands of color shades and intensities, compared to about only two dozens shades of gray. Although the process followed by the human brain in perceiving and interpreting color is a phenomenon that is fully understood, the physical nature of color can be expressed on the formal basis supported by experimental and theoretical results. However, in the past decade, color sensors and hardware processing color images have become available at reasonable prices. The result that with the use some different patterns full color image processing techniques are now used in broad range of applications including detection, visualization and the internet.

II. COLOR MODELS

The purpose of a color model (*also called color space or color system)* is to facilitate the signification of colors in some standard, generally accepted way. In essence, color model is a specification of coordinate system and a subspace within that system where each color is represented by a single point.There are numerous color models in use today due to fact that color science is a broad field that encompasses many areas of applications. In terms of Digital Image Processing, the hardware-oriented models most commonly in used in practice are the:

2.1 RGB (red, green, blue) Color Model

This model is for monitors and broad class of color video cameras. In RGB model, each color appears in its primary components of red, green, blue. This model is based on Cartesian coordinate system.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (127-132), Month: April 2015 – September 2015, Available at: www.paperpublications.org

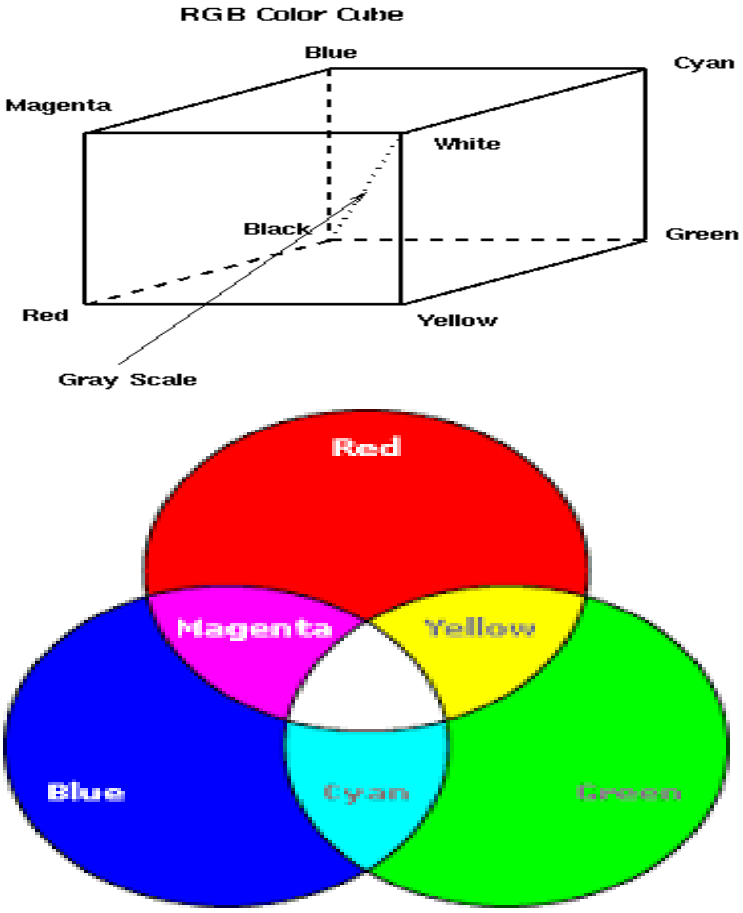


Fig. 2.1 RGB color Model

RGB model consist of three component images, one of each primary color. The number of bits used to represent each pixel in RGB space is called the pixel depth. The term full-color image is used often to denote a 24-bit RGB color image.

2.2 HSI Color Model

When humans view a color object, we describe it by hue, saturation, and brightness. Hue is a color attraction that describes a pure color (pure yellow, orange or red) whereas saturation gives a measure of the degree to which a pure color is diluted by white light. Brightness is a subjective descriptor that is practically impossible to measure. It embodies the achromatic notion of intensity and is the one key factors in descring .HSI model decouples the intensity component from color carrying information (hue and saturation) in a color image.

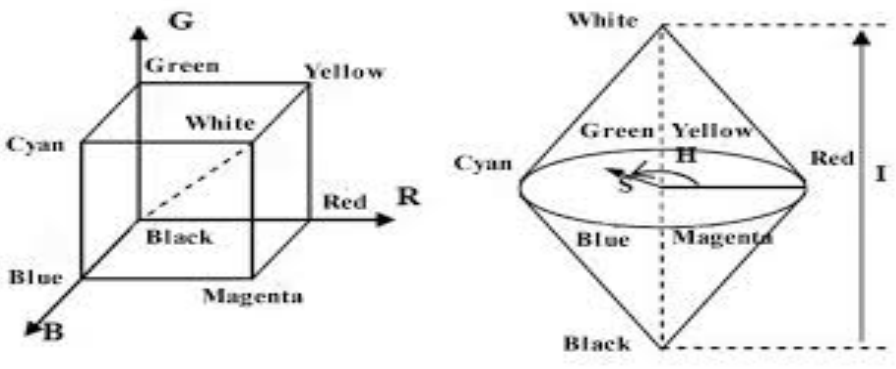
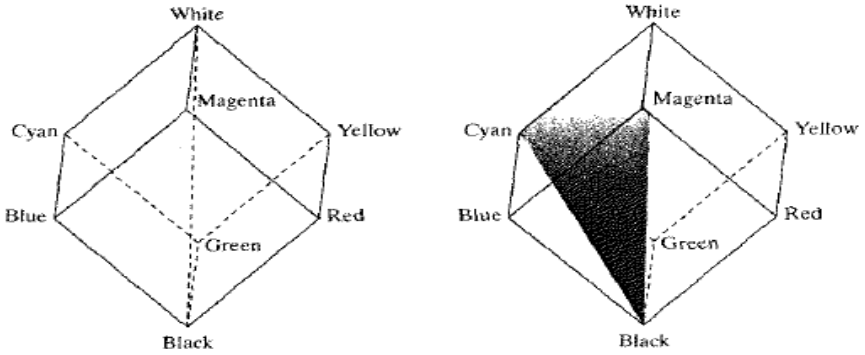


Fig. 2.2 a. Hue and saturation in HSI color model. The intensity of all colors in any of these planes is given by the position of the plane on the vertical intensity axis.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (127-132), Month: April 2015 – September 2015, Available at: www.paperpublications.org

Hue , saturation, and intensity values required to form the HSI space can be obtained from the RGB color cube. That is, when we convert any RGB point to corresponding point in the HSI color model by working out the geometrical formulas.



a b

Fig. 2.2 b. Conceptual Relationship between the RGB and HSI color model

The Hue of the point is determined by the angle from the some reference point. the Saturation is the length of the vector from the origin to the point. The important components of the HSI color space are the vertical intensity axis The length of the vector to the color point ,the angle this makes the red axis.

2.3 Conversion colors from RGB to HSI

Suppose R, G, and B are the red, green, and blue values of a color. The HSI intensity is given by the equation

$$I = (R + G + B)/3.$$

Now let m be the minimum value among R, G, and B. The HSI saturation value of a color Image is given by the equation

$$S = 1 - m/I \quad \text{if } I > 0, \text{ or}$$
$$S = 0 \quad \text{if } I = 0.$$

To convert a color's overall hue, H, to an angle measure, use the following equations:

$$H = \cos^{-1}[\,(R - \tfrac{1}{2}G - \tfrac{1}{2}B)/\sqrt{R^2 + G^2 + B^2 - RG - RB - GB}\,] \quad \text{if } G \geq B, \text{ or}$$
$$H = 360 - \cos^{-1}[\,(R - \tfrac{1}{2}G - \tfrac{1}{2}B)/\sqrt{R^2 + G^2 + B^2 - RG - RB - GB}\,] \quad \text{if } B > G,$$

where the inverse cosine output is in degrees.

2.4 Equations to Convert HSI Values to RGB Values

To convert hue, saturation, and intensity to a set of red, green, and blue values, you must first note the value of H. If H = 0, then R, G, and B are given by

$$R = I + IS \qquad G = 3I - (R+B) \qquad B = I - IS.$$

If $0 < H < 120$, then

$$R = I + IS*\cos(H)/\cos(60-H) \qquad G = I + IS*[1 - \cos(H)/\cos(60-H)]$$
$$B = I - IS.$$

If H = 120, then the red, green, and blue values are

$$R = I - IS$$
$$G = I + 2IS \qquad B = I - IS$$

If $120 < H < 240$, then

$$R = I - IS \qquad G = I + IS*\cos(H-120)/\cos(180-H)$$
$$B = I + IS*[1 - \cos(H-120)/\cos(180-H)].$$

If H = 240 and if $240 < H < 360$, we have

$$G = I - IS \qquad R = I + IS*[1 - \cos(H-240)/\cos(300-H)] \qquad B = I + IS*\cos(H-240)/\cos(300-H)$$

III. PIXEL BY PIXEL IMAGE COMPARISON

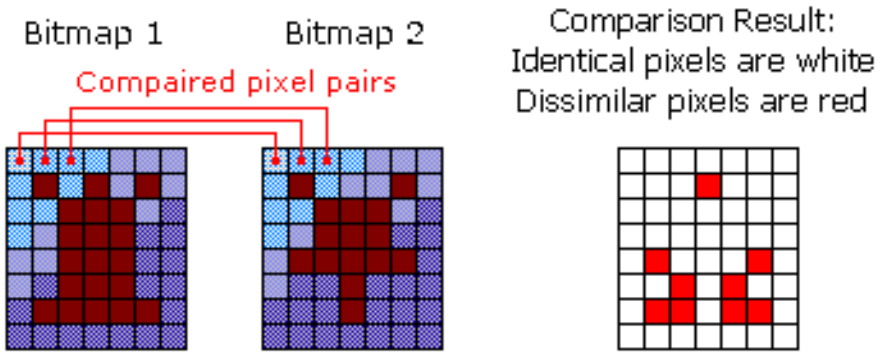
The comparison engine gets the color of pixels that have the same coordinates within the image and compares this color. If the color of each pixel of both images coincides, Test Complete considers the two images to be identical.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (127-132), Month: April 2015 – September 2015, Available at: www.paperpublications.org

3.1 Comparison Engine Parameters

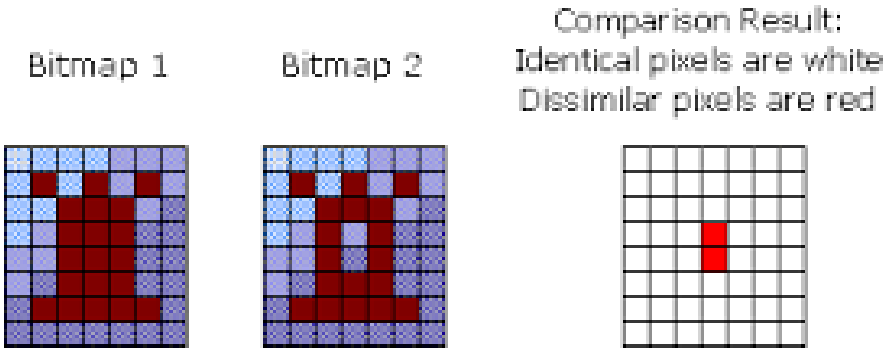
The comparison engine has a number of parameters. Each of the parameters enables additional processing of bitmaps, which makes comparison more flexible.



Below is a list of the comparison engine's parameters and their descriptions. You specify these parameters when creating a region checkpoint or when calling scripting methods that perform image comparison. The parameters are not used by the test log when it compares images that were captured during recording and playback and posted to the log by Test Visualize.

3.2 Pixel Tolerance

(Pixel Tolerance in scripting methods) - Defines the allowed number of dissimilar pixels. If the number of different pixels is less than or equal to Pixel Tolerance, then TestComplete considers the images to be identical.



For instance, in the image above, one bitmap differs from another by two pixels. If the PixelTolerance value were 2, TestComplete would consider these bitmaps to be identical.

3.3 Color Tolerance

(ColorTolerance in scripting methods) - Specifies an acceptable color difference at which two pixels should be treated as identical. The color difference is represented as an integer value within the range 0…255 that specifies an acceptable difference for each color component (red, green and blue) of the compared pixels. Two pixels are considered identical if the difference between intensities of each of their color components does not exceed the specified value.

When ColorTolerance is 0, which is the default value, the compared pixels are considered identical only if they have exactly the same color. When ColorTolerance is 255, pixels of any color are considered identical.

3.4 Include Mouse Pointer

(Mouse in scripting methods) - Defines whether TestComplete should capture the mouse cursor when it gets the image to be compared. By default, screenshots are taken without the image of the cursor. This makes image comparison independent of the mouse cursor position.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (127-132), Month: April 2015 – September 2015, Available at: www.paperpublications.org

However, if the cursor position is significant for you, you can enable the Mouse comparison parameter. In this case, TestComplete will include the mouse pointer in the captured image.

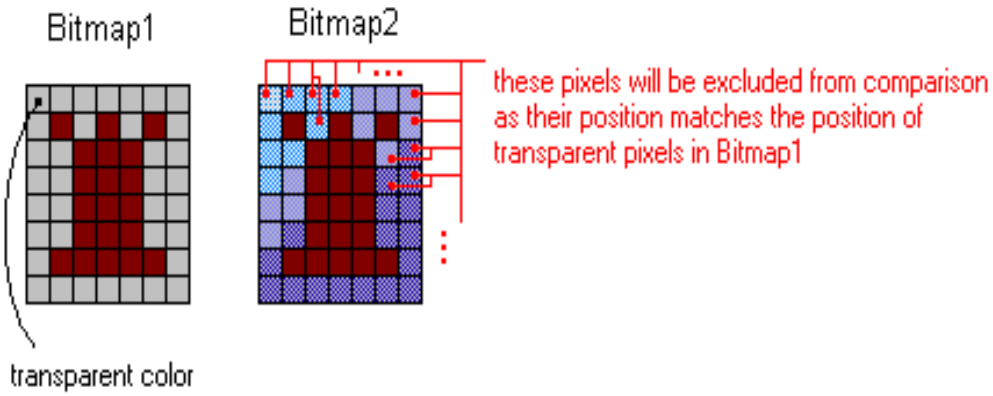
3.5 "Transparent" Color

(Transparent in scripting methods) If this parameter is True, TestComplete treats the color of the upper-left pixel of the baseline image as a transparent color and does not compare all the pixels that have the same coordinates in the compared image as the transparent pixels have in the baseline image. This parameter is similar to the way transparency is implemented in the jpeg and gif formats. For instance, if the top-left pixel is gray, then all gray pixels of the first image will match pixels of any color that have the same coordinates in the second image.

3.6 Comparison Mask

(Mask in scripting methods) Specifies which areas of bitmaps should be compared. A mask is another image (the Picture object) whose pixels can be either black or white. If a mask is specified, the comparison engine works in the following way: white pixels on the mask are taken into account during comparison, whereas black pixels are ignored. In contrast to using the Transparent parameter, you do not need to modify the baseline image, you just create a separate image instead.

For example, the following two different bitmaps can be considered identical if a mask is used.



3.7 Factors Affecting Comparison

Pixel-by-pixel comparison is considered successful only if the compared image and the baseline image are absolutely identical. To improve the quality of image comparison, you should capture the image you want to compare in the same conditions the baseline image was captured.

IV. ANALYZE OPENCV FOR IMAGE CAPTURING & COLOR DETECTION

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision, developed by Intel Russia research center in Nizhny Novgorod, and now supported by Willow Garage and Itseez. It is free for use under the open-source BSD license. It's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android.

OpenCV has lots of basic inbuilt image processing functions so that those who want to learn computer vision can develop their applications through proper understanding about what they do.

In other words, captured images can be considered as 3 matrices; BLUE, GREEN and RED (hence the name BGR) with integer values ranges from 0 to 255.

Color detection and segmentation is the most important and challenging fundamental task of computer vision. It is a critical part in many applications such as image search, scene understanding, etc. However it is still an open problem due to the variety and complexity of object classes and backgrounds. The easiest way to detect and segment an object from an image is the color based methods. The object and the background should have a significant color difference in order to successfully segment objects using color based detection.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (127-132), Month: April 2015 – September 2015, Available at: www.paperpublications.org

V. CONCLUSION AND FUTURE WORK

The proposed method is tested on different images. It produced stable and fairly good results. Consistent acceptable outputs over different kinds of real life images have proved robustness of the presented scheme. Thus, the proposed method may be handy for any computer vision task where extraction of edge maps is required for a large set of images for feature extraction or for any other work. Our next venture will be comparing those algorithms with the proposed one and analyze the performance on the basis of histograms like comparing histogram of one image to the another image, execution complexity and accuracy of the system output in presence of comparison.

REFERENCES

[1] J. F. Canny, .A computational approach to Image detection,. IEEE Trans.Pattern Anal. Machine Intell., vol. 8, no. 6, pp. 679.698, Nov. 1986.

[2] C. L. Novak and S. A. Shafer, “Color image detection,” in Proc. DARPA Image Understanding Workshop, 1987, pp. 35–37.

[3] R. Nevatia, “A color Image Processing and its use in scene segmentation,” IEEE Trans. Syst., Man, Cybern., vol. SMC-7,no. 11, pp. 820–826, Nov. 1977.

[4] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, Third Edition.

[5] A. Shiozaki, “Edge extraction using entropy operator,” Computer Vis., Graph., Image Process., vol. 36, no. 1, pp. 1–9, Oct. 1986.

[6] S. K. Naik and C. A. Murthy, “Standardization of Edge Magnitude in Color Images,” IEEE Trans. Image processing, vol. 15, no. 9, Sept. 2006.