ISBN 978-93-5156-328-0
International Conference of Advance Research and Innovation (ICARI-2014)

Identification of Colors in Photographic Images Using Color Quantization

Shailendra Kumar Dewangan
Department of Electronics & Instrumentation Engineering, Chhatrapati Shivaji Institute of Technology, Durg, Chhattisgarh, India

Article Info

Article history:
Received 2 January 2014
Received in revised form
10 January 2014
Accepted 20 January 2014
Available online 1 February 2014

Keywords

Color Identification,
Color Quantization,
Photographic Image

Abstract

A simplification of the color histogram indexing algorithm is proposed and analyzed. Instead of taking a histogram consisting of hundreds of colors, each input image is first quantized to only a few colors and the feature vector is generated by taking a histogram of this smaller space. This increases the efficiency of the system by orders of magnitude. Quantization, involved in image processing, is a loss compression technique achieved by compressing a range of values to a single quantum value. When the number of discrete symbols in a given stream is reduced, the stream becomes more compressible. For example, reducing the number of colors required to represent a digital image makes it possible to reduce its file size. This is important for displaying images on devices that supports a limited number of colors and for efficiently compressing certain kinds of images.

1. Introduction

The color of an object is its most characteristic property and this suggests that it should be possible to find an object in a visual scene based on its color. In principle, this could be done by searching for all pixels in an image with a particular range of RGB values, but in practice, this does not work very well. There are a number of reasons for this, but the most important is that the RGB representation of color mixes intensity and color in all the three channels. For example, if the illumination increases, the values in all three channels will increase although perceptually, the color would not have changed. To allow for such changes, the range of RGB values has to be very large and this would include colors that are not necessarily of the target color. Another problem is that the color of the illumination may change over the day, which will have different effects on the different RGB channels. [1] This is in general a very hard problem to solve and requires color-constancy. However, it is possible to get by with simpler heuristics in many cases. Here, we describe a fast algorithm that can compensate for these variations and find pixels in an image with a particular color. There are a number of problems that need to be solved. First, it is useful to map the colors from RGB space to a more useful representation. Here we will map the colors onto the rag-chromaticity plane where the each color is coded as a hue and saturation, but the intensity is discarded.

Corresponding Author,
E-mail address: shailendra.dev@gmail.com

1.1 Color

In the RGB color space, each color is described as a combination of three main colors, namely Red, Green and Blue. This color space can be visualized as a 3d matrix with the main colors set out on the axis (Fig. 1). The values for the main colors vary from 0 to 1. Each color is coded with three values, a value for red, blue and green. In this color space, an imported image on a computer is thus transformed into 3 matrices with values per pixel for the representing main color.

1.1.3 Color Definitions

The fact is that for camera vision this is not desirable because of the loss of speed. So for camera vision in Mat lab, it is the best option to make use of the RGB color space. The size of the matrix that represents the RGB color space is dependent on the bit rate that is used. Mat lab uses a standard bitrates of 8 bits when an image is imported. This means that there are 256 tones of each main color, so the size of the color space 7 matrixes is 256x256x256. A 3D region inside this matrix must be defined to indicate a

ISBN 978-93-5156-328-0
International Conference of Advance Research and Innovation (ICARI-2014)

particular color. This can be done by intuition, but it is a lot easier when the colors are visualized. Because of the fact that colors in the RGB space depend on three variables, a 2D image is not sufficient to visualize all colors. This matrix must be transformed into the RGB-space with the 'hsv2rgb'-function. Each combination of red, green and blue defines a single point in the RGB-space matrix and these points can be labeled as 1 into this matrix. Now the color for the ball is a little area with ones inside a 256x256x256 matrix with for the rest zeros.

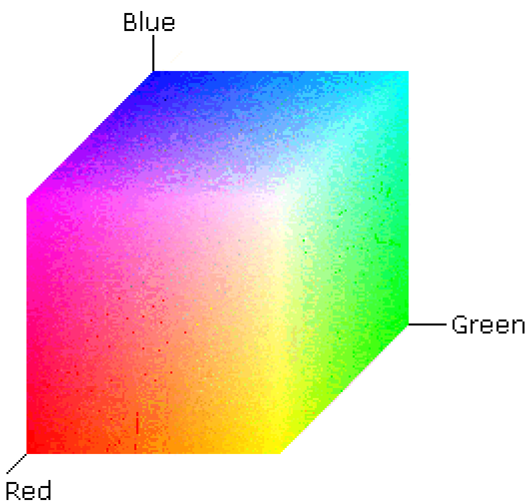

Fig: 1. RGB Color Space

1.2 Pixel Labeling

The color matrix is now a matrix of size 256x256x256 with all zeros, except for every color that is associated with the ball, the value on these coordinates is one. This matrix is used for pixel labeling. An image consists of a lot of pixels, depending on the resolution of the camera. Each pixel has its own color and by means of the color matrix this color can be recognized. The process, in which each pixel is given a value depending on its color, is called pixel labeling.

2. Color Quantization

The goal of this algorithm is to display an image as close as possible to the original image. Ideally, a human viewer should find the output image indistinguishable from the original. In the second case, quantization is used to store or transmit the image as efficiently as possible. Again, the desired output is assumed to be as close as possible to an observer's perception of the original. Time-dependent effects of the human visual system, such as chromatic adaptation and degradation of the image due to memory are explicitly ignored because the purpose of this quantization is to allow the human observer to view the image independent of these effects. Because of this goal, a second assumption inherent in all these quantization methods is that comparison to the original image is the best measure of the method's impact. [2] [3] This means that generally, quantization itself is seen as introducing noise and degrading the image. In our case, the best measure of the effect of the algorithm is comparison to other images of the same scene. The original image contains noise in the pixel values. A second image of the same view will generally contain different (although similar) values. If the lighting is unchanged, a quantization algorithm that is effective for these purposes should map the images to the same result, rather than to different results. Comparison to the original image will give a sense of the degradation to the image caused by the algorithm. Comparison to different images, rather than comparison of storage or display constraints, should provide the metric for the effectiveness of the Quantization algorithm. Instead, we will measure an algorithm by its effectiveness when used in the context of the object recognition algorithm.

2.1 Uniform Quantization

This is the simplest of the quantization algorithms. The color space is divided up into n blocks of equal volume. The centroid of each block is the color used in the color palette. The color palette is therefore fixed, data-independent, and takes no notice of the combinations that humans may find more pleasing. Generally this algorithm is considered to produce poor results from a human standpoint.

In uniform quantization each axis of the color space is treated independently. Each axis is then divided into equal sized segments. The planes perpendicular to the axis' that pass through the division points then define regions in the color space. The number of these regions is dependent on the scheme used for dividing the color space. One possible scheme is to divide the red and green axis into 8 segments each and the blue axis into 4 resulting in 256 regions. Another possibility is dividing the red and blue into 6 and the green into 7 segments resulting in 252 regions. Each one of these regions will produce a color for the color map.

Once the color space has been divided each of the original colors is then mapped to the region which it falls in. The representative colors for each region is then the average of all the colors mapped to that region. Because each of the regions represents an entry in the color map, the same process for mapping the original colors to a region can be repeated for mapping the original colors to colors in the color map. While this algorithm is quick and easy to implement it does not yield very good results. Often region in the color space will not have any colors mapped to them resulting in color map entries to be wasted.

This algorithm can also be applied in a non-uniform manner if the axis is broken on a logarithmic scale instead of linear. This will produce slightly better results because the human eye cannot distinguish dark colors as well as bright ones.

2.2 Modified Uniform Quantization

In this algorithm, instead of dividing the space evenly into n blocks of equal volume in three dimensions, the n blocks are either cubic or rectangular, with an integer number of them to a side. In this way, the largest difference between the maximum and minimum number of blocks in any single dimension is one, and the number of blocks in all other dimensions have neither a maximum nor a minimum. This is a data-independent method for simple quantization to few colors. The order in which the axis along which the variable block size occurs can be chosen in advance to fit the data best, making this a data dependent method, in advance to fit the data best, making this a data-dependent method, or it can be determined in advance and fixed, making this a data-independent method.

In the early days of PCs, it was common for video adapters to support only 2, 4, 16, or (eventually) 256 colors due to video memory limitations; they preferred to dedicate the video memory to having more pixels (higher resolution) rather than more colors. Color quantization helped to justify this tradeoff by making it possible to display many high color images in 16- and 256-color modes with limited visual degradation. The Windows operating system and many other operating systems automatically perform quantization and dithering when viewing high color images in a 256 color video mode, which was important when video devices limited to 256 color modes were dominant. Modern computers can now display millions of colors at once, far more than can be distinguished by the human eye, limiting this application primarily to mobile devices and legacy hardware.

Nowadays, color quantization is mainly used in GIF and PNG images. GIF, for a long time the most popular lossless and animated bitmap format on the World Wide Web, only supports up to 256 colors, necessitating quantization for many images. Some early web browsers constrained images to use a specific palette known as the web colors, leading to severe degradation in quality compared to optimized palettes. PNG images support 24-bit color, but can often be made much smaller in file size without much visual degradation by application of color quantization, since PNG files use fewer bits per pixel for palettized images.

The infinite number of colors available through the lens of a camera is impossible to display on a computer screen; thus converting any photograph to a digital representation necessarily involves some quantization. Practically speaking, 24-bit color is sufficiently rich to represent almost all colors perceivable by humans with sufficiently small error as to be visually identical (if presented faithfully), within the available color space. However, the digitization of color, either in a camera detector or on a screen, necessarily limits the available color space. Consequently there are many colors that may be impossible to reproduce, regardless of how many bits are used to represent the color. For example, it is impossible in typical RGB color spaces (common on computer monitors) to reproduce the full range of green colors that the human eye is capable of perceiving.

With the few colors available on early computers, different quantization algorithms produced very different looking output images. As a result, a lot of time was spent on writing sophisticated algorithms to be more lifelike.

3. Advantages of the Process

Color quantization or color image quantization is a process that reduces the number of distinct colors used in an image, usually with the intention that the new image should be as visually similar as possible to the original image. This methodology can reduce the effects of lighting change on the algorithm and that this would be a better model for the human object recognition mechanism than the algorithm combined with color constancy alone. [4] This method of color quantization can occasionally compensate for small lighting changes. Any three-dimensional clustering algorithm can be applied to color quantization, and vice versa. After the clusters are located, typically the points in each cluster are averaged to obtain the representative color that all colors in that cluster are mapped to. [5] The three color channels are usually red, green, and blue, but another popular choice is the Lab color space, in which Euclidean distance is more consistent with perceptual difference. Color quantization methodology reduces the number of colors used in an image.

4. Experimental Set-Up Development

In this work uniform quantization method is used for recognition of colors. We have used different images for testing of this methodology.

4.1 MATLAB Implementation

ISBN 978-93-5156-328-0
International Conference of Advance Research and Innovation (ICARI-2014)

The Name MATLAB Stands For Matrix Laboratory. MATLAB Is A High-Performance Language For Technical Computing. It Is Developed By Math works. MATLAB Allows Matrix Manipulations, Plotting Of Functions And Data, Implementation Of Algorithms, Creation Of User Interfaces, And Interfacing With Programs Written In Other Languages, Including C, C++, Java, And FORTRAN. It Integrates Computation, Visualization, And Programming In An Easy-To-Use Environment Where Problems And Solutions Are Expressed In Familiar Mathematical Notation.

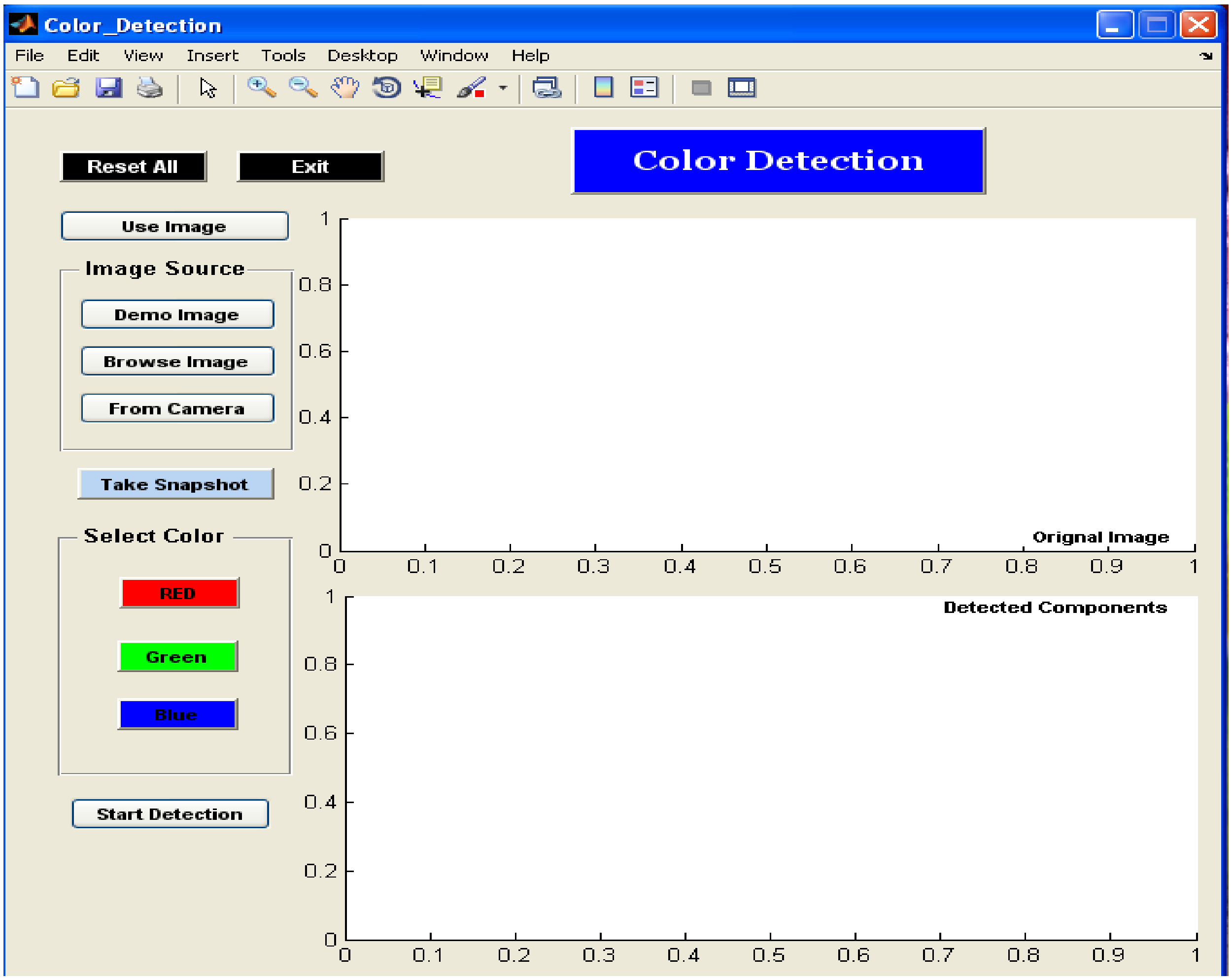

Fig: 2. Layout of GUI

4.2 Graphical User Interface (GUI)

MATLAB's Graphical User Interface development environment provides a set of tools for laying out the GUI. The Layout Editor is the control panel for Guide. Guide provides four templates that make it easier to construct GUIs. The templates are simple examples of GUIs that can be modified for the required purposes. The templates can be accessed from the new Guide Quick Start dialog that appears when open a new Guide, or when the option '**New**' is selected from the file menu. It is often easier to build a GUI from an existing template rather than starting with a blank GUI. Figure 2 shows the layout of GUI for this proposed methodology.

4.3 Experimental Results

This method of color quantization was tested for various photographic images using uniform quantization. The testing was done on image database containing 110 different color images. This method was able to recognize the RGB components of the images efficiently. One of the result set is shown in Figure 3.

ISBN 978-93-5156-328-0

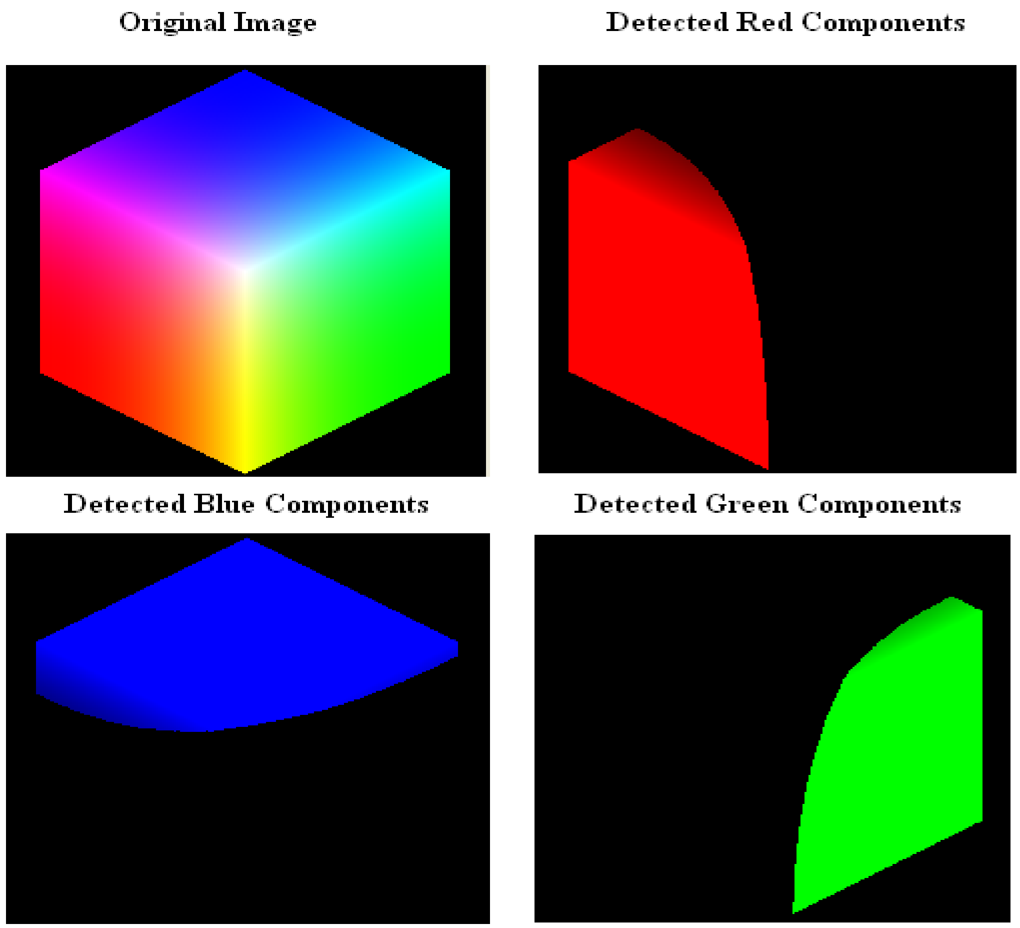

Fig: 3. Experimental Results

5. Conclusion

With our experiments and the corresponding results obtained we come to a conclusion that the color composition of the objects in an image can be significantly different from the color composition of the image as whole. A higher level of effectiveness in image retrieval can be achieved by closely approximating the color composition of the desired object.

References

[1] T. Batard, M. Berthier & C. Saint Jean, “Fourier Transform for Color Image Processing in Geometric Algebra Computing”, Engineering and Computer Science Springer Verlag, Chap. 8, pp. 135-161, 2010

[2] Adlington, K. Laws & T. Gale, “Visual Processing in Alzheimer's Disease: Surface Detail & Color Fail to Aid Object Identification”, Neuropsychologia, 47, pp. 2574-2583, 2009

[3] Damien Muselet & Ludovic Macaire, “Combining Colour and Spatial Information for Object Recognition across Illumination Changes”, Pattern Recognition Letters, 28, pp.1176-1185, 2007

[4] Christoph Palm, “Color Texture Classification by Integrative Co-Occurrence Matrices”, Elsevier, Pattern Recognition, 37, pp. 965-976, 2004

[5] C. Christian Balkenius, A.J. Johansson & A. Balkenius, “Color Constancy in Visual Scene Perception”, Lund University Cognitive Studies, 2007