Computers and Electrical Engineering 46 (2015) 328–337

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Integer-based accurate conversion between RGB and HSV color spaces ☆

Vladimir Chernov *, Jarmo Alander, Vladimir Bochko

Department of Electrical Engineering and Energy Technology, University of Vaasa, Finland

ARTICLE INFO

Article history:
Received 2 December 2014
Received in revised form 5 August 2015
Accepted 6 August 2015
Available online 19 August 2015

Keywords:
HSV
RGB
Integer algorithm
Color space conversion

ABSTRACT

This paper introduces a new fast integer-based algorithm to convert the RGB color representation to HSV and vice versa. The proposed algorithm is as accurate as the classical real-valued one. The use of only integer operations increases performance and portability. Performance measurement results show a speed gain of about two times when compared with the classical C++ language implementation on PC platforms. Lookup tables are not involved, thus the memory usage is minimal. The resulting HSV color can be packed into 48 bits. The proposed method can safely replace the commonly used floating-point implementation.

1. Introduction

The Hue Saturation Value (HSV) color space was created by Alvy Ray Smith in 1978 [1]. Initially, it was designed to improve a color picking interface in computer graphics software. The HSV model is a popular tool for color mixing since it is well consistent with human color perception. Furthermore, the HSV color space is often used in image processing and computer vision fields.

HSV space is well proven in image segmentation. Ganesan et al. used HSV color histogram for segmentation of region of interest in satellite images [2]. Liu et al. developed a method of fog level detection in highspeed road traffic based on image HSV color histogram [3]. Feng et al. proposed online method for rare colored capsule detection based on RGB and HSV color space [4].

Several works apply HSV for skin detection. Rahman et al. combined HSV and YCbCr color spaces in a new human skin detection method [5]. Bhowmick et al. applied HSV and YCbCr mixed color space for skin color segmentation in hand gesture recognition method for English alphabets [6]. Chaudhary and Gupta used HSV in dynamic color space switching system for skin pixel segmentation [7].

HSV components are often used as feature vectors for image retrieval and classification. Rashno et al. proposed a feature extraction schema that includes color features in RGB and HSV domain [8]. Raja et al. used colors from HSV model as low level features in robust indoor/outdoor scene classifier [9]. Archana et al. applied HSV in multiple face detection method with accuracy of 93% [10].

Fast RGB to HSV conversion algorithm is required for video processing. Wang and Zhao developed robust image chroma-keying method using HSV colors [11]. Chernov et al. proposed underwater video enhancement method based on HSV color space [12]. Guo et al. adjusted *S* and *V* channels in order to improve visibility and fidelity of underwater images [13].

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Area Editor Dr. E. Cabal-Yepez.
* Corresponding author. Tel.: +358 40 6647727.
E-mail addresses: vladimir.chernov@student.uva.fi (V. Chernov), jal@uva.fi (J. Alander), vbochko@gmail.com (V. Bochko).

http://dx.doi.org/10.1016/j.compeleceng.2015.08.005
0045-7906/

The use of HSV rather than RGB space is beneficial in texture analysis. Paschos [14] demonstrates that perceptually uniform spaces, such as HSV and L*a*b*, outperform nonuniform RGB. He concludes, according to experimental results, that HSV could be a superior color space, compared to RGB, for color texture analysis. In noisy conditions the HSV performs better than the L*a*b*.

The HSV model is designed as a transform of the RGB space. Practically, the only way to obtain the color in HSV is to convert it from RGB first. Therefore, characteristics of the H, S, and V components are dependent on the transformation algorithm. The classical conversion formulas, proposed by Smith [1], assume computations on real (not integer) numbers. Otherwise, the result would not have the expected precision. This causes the consideration for the H, S, V to be real numbers as well. The conversions have a classical implementation in programming languages with floating-point variables involved. As a reference of such implementation, the source code [15] from the GIMP editor (GNU Image Manipulation Program) is recommended by the authors.

In most cases, the classical floating-point realization is sufficient. However, there is a branch of algorithms that aims to improve the performance and portability of the color space conversions. A reasonable way in such improvement is to use integer (fixed-point) arithmetic and data types. Note, that the terms fixed-point and integer are practically identical in this context and used as synonyms in this paper.

Kabi et al. switched from floating-point to fixed-point arithmetic in order to reduce power consumption, time and market price of handheld devices for the application of speech-based emotion recognition [16]. Dobashi et al. proposed a fixed-point tone mapping operation (TMO) as a replacement for a floating-point TMO in order to reduce a computational cost for generating a low dynamic range (LDR) image from a high dynamic range (HDR) image [17].

The main practical reasons motivating the usage of the integer instead of floating-point arithmetic are performance and portability. For most processors, integer arithmetic is faster and well-suited to optimize hardware implementations like FPGA and allow for nice embedded system realizations. In addition, the integer-based algorithm is able to be run on hardware without an FPU (Floating-Point Unit or math coprocessor).

The integer data types for both RGB and HSV color schemes are preferable in many applications. The integer-based representation is generally more robust. Following good programming practices, the float and double types should not be used where exact results are required. This rule is applicable for many pixel manipulation applications, where floating-point representation is redundant. However, the straightforward change to integer types in the classical implementation is impossible without precision loss.

Liu et al. [18] proposed a fast algorithm for color space conversion based on fixed-point calculations on digital signal processors (DSPs). In the publication, the following conversion methods are included: YCbCr (luminance, chrominance-blue, and chrominance-red) to RGB, RGB to HSV, and YCbCr to HSV. The authors replaced floating-point multiplications with fixed-point shifts, 16-bit fixed-point multiplications, and additions. They used supporting small-sized (225 bytes and 256 bytes) lookup tables (LUT) in order to compensate for rounding error and to realize the 16-bit fixed-point multiplication. According to the experimental results, the authors achieved the error in acceptable range, maximum equal to 3. At the same time, they significantly improved computational speed, about 10 times faster on a fixed-point DSP and about 1.41 times faster on a personal computer (PC).

This paper introduces an integer-based algorithm to convert color representation from RGB to HSV and vice versa. The proposed algorithm is as accurate as the classical one. Each color from the RGB space has a unique equivalent in the HSV space. All HSV and RGB components in the proposed conversions are integer numbers. The use of floating-point variables is excluded entirely. The implementation of the proposed algorithm is fast, which is important for real-time image processing applications. Performance measurement results show a speed gain of about two times when compared with the classical implementation [15] on PC platform. Lookup tables are not involved. Therefore, the memory usage is minimal. The proposed algorithm can safely replace the classical one. If the floating-point data types are still needed for complex math operations, the HSV channels can be type-casted afterward.

This paper is organized as follows. In Sections 2 and 3 the theoretical background about RGB and HSV models, respectively, is given. In Section 4 the classical real-valued conversion algorithm is explained. The proposed integer-based algorithm is presented in Section 5. The performance analysis is done in Section 6. The experimental results are shown in Section 7. Finally, the paper is concluded in Section 8.

2. The RGB model

The RGB model encodes a color using three components: red (R), green (G), and blue (B). It is an additive color scheme that employs the principle of human eye functionality, i.e., sensitivity of three types of cones in retina to specific light spectra [19]. Thus visible colors can be reproduced by adding various intensities of red, green, and blue lights. The concept of RGB mixing is widely utilized in display and camera devices that makes this color model essential for most computer graphics applications.

RGB space can be visualized in the form of a cube in a three-dimensional Cartesian coordinate system (Fig. 1). The black color is in the axis origin while the white color is located in the diagonally opposite corner. The rest of the vertices represent primary colors (red, green, and blue) and secondary colors (cyan, yellow, and magenta).

The standard 24-bit RGB format is required for the proposed integer algorithm. In the 24-bit RGB pixel, all components have 8-bit depth. The total number of possible colors in this case is $2^8 \cdot 2^8 \cdot 2^8 = 16,777,216$. The components are interpreted as unsigned integers in range $[0, 255]$, which is exactly the value range of a single byte.