Computers and Electrical Engineering 40 (2014) 1405–1414

Contents lists available at SciVerse ScienceDirect

Computers and Electrical Engineering

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

A fast algorithm for color space conversion and rounding error analysis based on fixed-point digital signal processors ☆

Zhao-Guang Liu [a,*], Sheng-Yong Du [b], Yang Yang [c], Xiu-Hua Ji [a]

[a] Shandong Provincial Key Laboratory of Digital Media Technology, School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan, China
[b] School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan, China
[c] School of Information Science and Engineering, Shandong University, Jinan, China

ARTICLE INFO

Article history:
Available online 14 February 2013

ABSTRACT

The YCbCr (luminance, chrominance-blue, and chrominance-red) color space is adopted in video codecs or transmission, while the HSV (hue, saturation and value) color space is used in some video analysis algorithms. In this paper, a fast algorithm based on fixed-point digital signal processors (DSPs) is proposed for YCbCr to HSV conversion. Floating-point multiplications are replaced with fixed-point shifts, 16-bit fixed-point multiplications, and additions. To compensate for rounding error and convert floating-point multiplication into 16-bit fixed-point multiplication in the calculation, tables occupying 225 bytes and 256 bytes are established, respectively. According to the experimental results, the error caused by the proposal is within the acceptable range. At the same time, the proposed algorithm is about 10 times faster than traditional YCbCr to HSV conversion algorithm on a fixed-point digital signal processor (DSP) platform and about 1.41 times faster on a personal computer (PC) platform.

1. Introduction

A color image can be represented by multiple color space models, such as RGB (red, green, and blue), YCbCr (luminance, chrominance-blue, and chrominance-red) [1], and HSV (hue, saturation and value) [2,3]. RGB describes a color based on the three primary colors and light system theory. YCbCr describes a color in accordance with the principles of brightness and chromatic aberration. In YCbCr space, Y is the luminance component, Cr reflects the difference between the red component and the illumination value of the input RGB signal, and Cb reflects the difference between the blue component and the illumination value of the input RGB signal. In the HSV model, H, S, and V represent hue, saturation, and brightness, respectively. The V-axis of the HSV model corresponds to the main diagonal of the RGB color space.

Since RGB representations are highly correlated, they are not well suited to independent coding. Therefore, compression standards such as joint photographic experts group (JPEG), moving pictures experts group (MPEG), and H.264/AVC (AVC: advanced video codec) employ the YCbCr color space. At the same time, HSV is always used in video analysis; e.g., in fire or flame detection [4,5], skin color detection [6], and image segmentation [7]. In some applications, video is decoded from a bitstream directly, and the HSV color space is utilized for image or video analysis, so conversion from YCbCr to HSV is certainly necessary. Therefore, a fast conversion algorithm is helpful to reduce the computational complexity in such applications.

☆ Reviews processed and recommended for publication to Editor-in-Chief by Guest Editor Dr. Jing Tian.
* Corresponding author.
E-mail address: liuzhg@sdufe.edu.cn (Z.-G. Liu).

0045-7906/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compeleceng.2013.01.011

Table 1
Color spaces of YCbCr and RGB.

	RGB			YCbCr		
	R	*G*	*B*	*Y*	Cb	Cr
Max	255	255	255	235	240	240
Min	0	0	0	16	16	16

There exists no direct conversion from YCbCr to HSV, and RGB variables are needed as intermediates. There are several different conversion standards in use. Without loss of generality, we focus on ITU-R Recommendation BT.601-5 [8] as well as Refs. [2,3]. According to the standard [8], the relationship between RGB and YCbCr is

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.402 \\ 1 & -0.344 & -0.714 \\ 1 & 1.772 & 0 \end{bmatrix} \times \begin{bmatrix} Y \\ \mathrm{Cb} - 128 \\ \mathrm{Cr} - 128 \end{bmatrix}. \tag{1}$$

The only difference between YCbCr and YUV (luminance and chrominance) is the direct component (DC) component of 128, and the conversion matrix is the same, so we focus on YCbCr below. According to the standard [8], the range of each component of YCbCr or RGB is that listed in Table 1.

According to Refs. [2,3], the formula for RGB to HSV conversion is

$$\begin{cases} H = 60 * \begin{cases} (G-B)/\mathit{delta}, & \text{if } R = \max(R,G,B) \\ 2 + (B-R)/\mathit{delta}, & \text{if } G = \max(R,G,B) \\ 4 + (R-B)/\mathit{delta}, & \text{if } B = \max(R,G,B) \end{cases} \\ S = \mathit{delta}/\max(R,G,B) \\ V = \max(R,G,B) \\ \text{where delta} = \max(R,G,B) - \min(R,G,B). \end{cases} \tag{2}$$

The rest of this paper is organized as follows. Section 2 gives a detailed introduction to related work. Section 3 proposes a fast algorithm for YCbCr to HSV conversion. Section 4 presents experimental results based on a fixed-point DSP and a general-purpose PC. Finally, Section 5 states the conclusions.

2. Related work

As far as we know, there is currently no fast algorithm for conversion from RGB to HSV. There are several conventional methods for conversion from YCbCr to RGB, including hardware and software methods. Based on comparisons of those methods, this study prescribes a way to find the most economical expression at given quantification error, which is more appropriate for fixed-point DSPs.

2.1. Lookup table method

The lookup table (LUT) method is a highly efficient method, especially for embedded systems [9]. There are two kinds of LUT methods for YCbCr to RGB conversion; i.e., one-check LUTs and two-check LUTs. For the 24-bit case, the one-check LUT needs no calculation and simply checks the index table of $6 \times 256 \times 256 \times 256 = 100663296$ bytes long, which becomes impractical for embedded systems. For the two-check LUT, there are two series corresponding to the upper and lower 4 bits of YCbCr and RGB data, respectively, and there are six tables in each series, so the index table has only $2 \times 6 \times 16 \times 16 \times 16 = 49152$ bytes. The 8-bit data are first broken into two sets of 4-bit data, the two index tables are then checked, and the results are added to obtain the final result. Compared with a one-check LUT, a two-check LUT needs extra blocking, shift, and addition operations, so a two-check LUT is relatively slow but occupies much less memory. The drawback of any LUT is the great amount of memory occupied.

2.2. Shifting method

In our previous work [10], a fast algorithm for YCbCr to RGB conversion on the fixed-point DSP was presented. Within the allowable range of error, the floating-point operations can be converted into fixed-point shift and addition operations, thus the computation speed was improved dramatically. In this paper, we will further analyze the shifting method and its rounding error.