A Novel FPGA Based Reconfigurable Architecture for Image Color Space Conversion

M.C. Hanumantharaju[1], G.R. Vishalakshi[1], Srinivas Halvi[2], and S.B. Satish[3]

[1] Department of Information Science & Engineering
[2] Department of Medical Electronics
[3] Department of Electronic & Communication Engineering,
Dayananda Sagar College of Engineering, Bangalore, India
(mchanumantharaju,vishala.gr2005,satishsbhai)@gmail.com

Abstract. In this paper, a novel Field Programmable Gate Array (FPGA) based reconfigurable architecture for image color space conversion is presented. The color space conversion plays an important role in preprocessing stage of digital image processing. Although it is possible to process an image in RGB color space the resulting image may not be processed optimally. The hardware implementation of algorithms like image enhancement, image compression, and communication techniques use color spaces such as Hue-Saturation-Value (HSV), Luminance-Chrominance blue-Chrominance red (YCbCr) and YUV etc. Therefore, it is desired to have an efficient and high speed architectures for color space conversion. The proposed scheme implements two novel architectures for color space conversion suitable for FPGA's and Application Specific Integrated Circuits (ASICs). The architectures developed exploits high degrees of pipelining and parallel processing in order to achieve real time performance. The hardware realization of color space conversion is based on Register Transfer Level (RTL) compliant Verilog code and is implemented using Xilinx XC2V2000-6bf957 FPGA device. The experimental results show that proposed approaches exhibit better performances when compared with the other existing implementations.

Keywords: Field Programmable Gate Arrays, Color Space Conversion, Hue-Saturation-Value, YCbCr.

1 Introduction

The color space [1] or color models or color systems provides a standard method of defining and representing colors. In the literature, there exist a many color space converters and most of them represent each color as a point in a three dimension (3D) coordinate system [2]. Further, color space selection and its optimization depends on specific application [3]. The three most popular color models are RGB (used in computer graphics); YIQ, YCbCr, and YUV (used in video systems); CMYK (used in color printing). All color spaces can be derived from the RGB information supplied by devices such as cameras and scanners. Jung et al. [4] proposed a visual tracking system based on adaptive color histograms.

P.V. Krishna, M.R. Babu, and E. Ariwa (Eds.): ObCom 2011, Part II, CCIS 270, pp. 292–301, 2012.

The RGB to HSV conversion architecture developed in this work is computationally complex. In addition, the latency achieved in the conversion process is not satisfactory. Ming et al. [5] has developed high performance architecture for enhancement of video stream captured under non-uniform lightning conditions. The RGB to HSV conversion adapted in this scheme use log-domain in order to avoid complex division process. However, this technique also increases the latency of the conversion process. Further, the HSV to RGB conversion adopted in this scheme is not an efficient from computation point of view. The RGB to YCbCr color space conversion proposed by Stefano Marsi et al. [6] had consumed a logic cells of 901 (0.75%). Iakovidou et al. [7] has been used YCbCr color space for image enhancement provides a satisfactory results. However, the algorithm is capable of processing 2.5 Mpixels at frame rate of 25 frames per sec. Although it is possible to enhance or compress a digital real color image by applying existing gray-level image processing algorithms to each red (R), green (G) and blue (B) channel, the resulting image may not be enhanced or compressed optimally . In addition, the algorithm is applied to individual channels without considering the correlation between R, G, and B component of the image. RGB color space has weakness in representing shading effects or rapid illumination changing [8]. In order to solve this problem, we consider converting an image from RGB space to other spaces [9].

This paper is organized as follows: Section 2 describes the review of color space conversion and its implementation issues. Section 3 gives brief details of hardware implementation. Section 4 provides experimental results and comparative study. Finally conclusion is presented in Section 5.

2 Color Space Conversion: A Review

Numerous researchers have proposed different color models, each oriented toward supporting a specific task or solving a particular problem [10]. Two popular color spaces widely used by many researchers are described below.

2.1 HSV Color Space

The HSV color space was defined by Smith (1978) and is based on cylindrical coordinates. The HSV [11] model defines a color space in terms of three constituent components. Hue represents type of color such as red, blue, or yellow that ranges from 0 to 360 degrees. Saturation is the vibrancy of color that ranges from 0 to 100 % . Value (intensity) is the brightness of the color that ranges from 0 to 100 % [4]. HSV color space, also known as hex cone color model; however, the human color space is a horseshoe-shaped cone [12]. In this work, hardware realization of RGB to HSV conversion[13] and vice-versa has been chosen since it offers satisfactory results for color image enhancement algorithms.

The digital hardware realization of RGB to HSV color space conversion is based on the Eqns. (1) to (3). The hue (H) component of HSV color space can be obtained from R, G and B channels using Eqn. (1)

$$H = \begin{cases} 0 + \frac{43\times|G-B|}{Max(R,G,B)-Min(R,G,B)}, & \text{if Max(R, G, B) = R} \\ 85 + \frac{43\times|B-R|}{Max(R,G,B)-Min(R,G,B)}, & \text{if Max(R, G, B) = G} \\ 171 + \frac{43\times|R-G|}{Max(R,G,B)-Min(R,G,B)}, & \text{if Max(R, G, B) = B} \end{cases} \tag{1}$$

where Max(R, G, B) is the maximum of three red, green, and blue pixel.

The saturation (S) component of HSV color space can be derived from R, G and B channels using Eqn. (2)

$$S = \left[\frac{Max(R,G,B) - Min(R,G,B)}{Max(R,G,B)}\right] \tag{2}$$

The value (V) component of HSV color space is given by Eqn. (3)

$$V = Max(R,G,B) \tag{3}$$

The flowchart for RGB to HSV conversion and HSV to RGB conversion is shown in Fig. 1. The RGB to HSV conversion process starts from computation of maximum and minimum element among R, G and B pixel values. The maximum value of RGB indicates the amount of white in the color image. The minimum value of RGB indicates the amount of black in the color image. The maximum of R, G and B represents the value or luminance or brightness of HSV. The difference component, which is computed by subtracting the minimum component from the maximum component of a gray image gives an indication of how much gray the color contains. The hue is computed by considering three cases. The hue represents type of color varying between 0 to 360 degrees and has fitted in the range 0 to 255. The digital value 85 is equivalent to 120 degree. The digital value 171 is equivalent to 240 degrees and finally the digital value 0 is equivalent to 360 or 0 degrees. The green color channel has been moved to 85 and blue is shifted to 171. The factor 60 degree has changed to a digital value of 43. Hue is no longer less than zero.

The HSV to RGB conversion is defined in Eqn. (4) and (5)

$$\begin{aligned} h_1 &= \frac{h}{43} = \frac{3h}{128} \\ h_2 &= floor(h_1) \\ f &= \frac{255 \times h - 8192 \times h_2}{43} \\ p &= v - vs \\ q &= v - vfs \\ t &= v - vs + vfs \end{aligned} \tag{4}$$

$$R'G'B' = \begin{cases} v, t, p: & \text{if } f = 0 \\ q, v, p: & \text{if } f = 1 \\ p, v, t: & \text{if } f = 2 \\ p, q, v: & \text{if } f = 3 \\ t, p, v: & \text{if } f = 4 \\ v, p, q: & \text{if } f = 5 \end{cases} \tag{5}$$

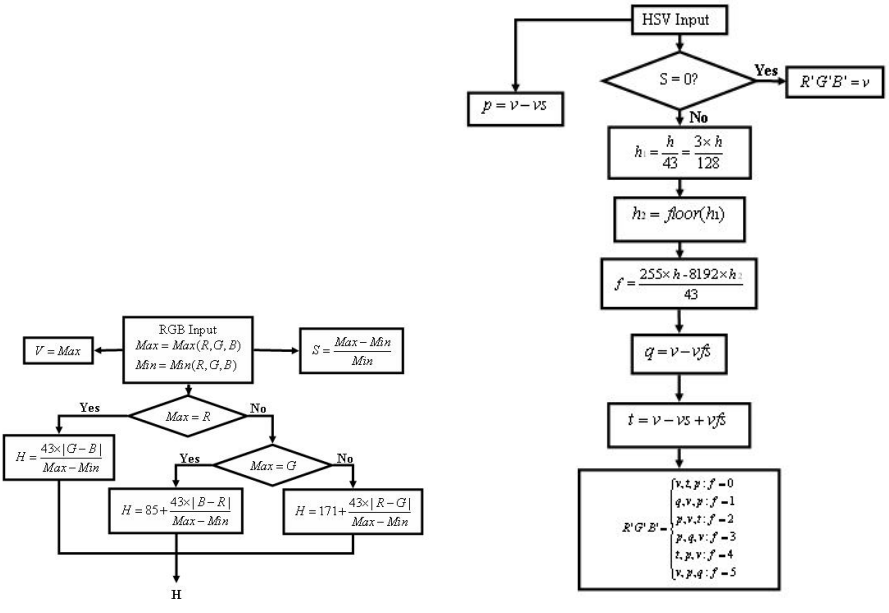

Fig. 1. Flow Chart for RGB to HSV and HSV to RGB conversion

2.2 YCbCr Color Space

In color image or video processing applications, a popular format is the YCbCr [14]. The RGB to YCbCr conversion and vice-versa is governed by the following expressions:

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{6}$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1.406 \\ 1 & 0.352 & 0.711 \\ 1 & 1.781 & 0 \end{pmatrix} \begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} \tag{7}$$

where R, G, and B are the picture element (pixel) values of three fundamental colors: red, green, and blue. Each of these color components is of size, 8 bits. An apt algorithm is developed without using multipliers for the computation of Y, Cb, Cr so that it may be efficiently mapped on to an FPGA/ASIC. The coefficients in the Eqn. (6) and (7) are scaled up by 128, and integers are retained, replacing multiplication operations by addition of relevant decimal weights of an multiplier or multiplicand. Finally, the results are scaled down to 128 in order to get back the correct results.

3 Hardware Implementation

In this section, hardware architectures designed for RGB-HSV and RGB-YCbCr color space conversion is described. The architecture has been designed in accordance with the color space conversion algorithm.

3.1 RGB-HSV Conversion Architectures

Hardware implementation of color space conversion starts with the Matlab verification followed by hardware architecture development, Verilog coding, logic synthesis, place and route and FPGA implementation [15]. First, we begin with realization RGB to HSV converter on FPGA. The RGB to HSV converter module design is based on Eqns. (1) to (3). The Hardware components used in RGB to HSV conversion are comparator, subtractor, divider, adder and hue selection. The Verilog codes developed for these modules are pipelined and processed parallel in order to speed up the conversion process [16]. The most complex module in this conversion process is the divider block. However, the divider employed in the conversion process is highly efficient. Finally, the hue selection block selects the individual hue values computed based on the maximum pixel value among R, G, and B.

The top level signal diagram for RGB-HSV color space conversion module is shown in Fig. 2. The detailed hardware architecture for RGB to HSV and HSV to RGB color space conversion is shown in Fig. 3.

3.2 RGB-YCbCr Conversion Architectures

The hardware realization of RGB to YCbCr conversion uses five pipeline stages. The format conversion scale-up the coefficients by 128, retaining integer and rearranging, we get the following expressions:

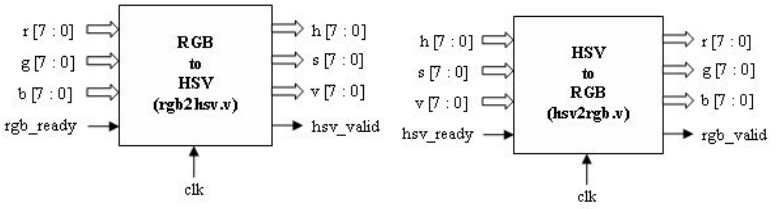

Fig. 2. Signal Diagram of RGB to HSV and HSV to RGB Converter Module

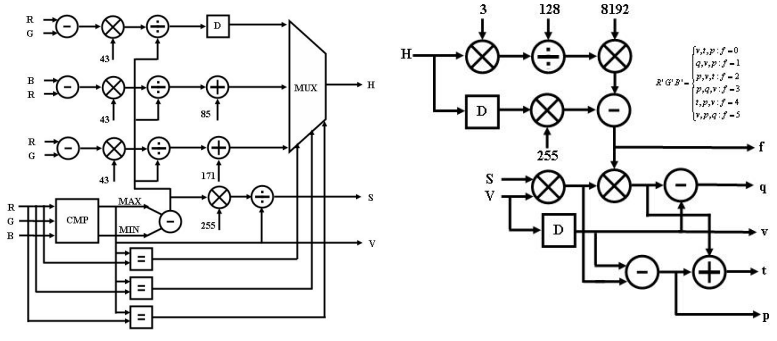

Fig. 3. Hardware Architecture for RGB to HSV and HSV to RGB Conversion

$$
\begin{aligned}
Y &= 38R + 75G + 15B \\
Y &= (32 + 4 + 2)R + (64 + 8 + 2 + 1)G + (8 + 4 + 2 + 1)B \\
Cb &= -22R - 42G + 64B \\
Cb &= -(16 + 4 + 2)R - (32 + 8 + 2)G + (64)B \\
Cr &= 64R - 54G - 10B \\
Cr &= (64)R + (32 + 16 + 4 + 2)G + (8 + 2)B
\end{aligned}
\tag{8}
$$

The above expressions may be easily realized by Verilog HDL since the multiplications involved are trivial, namely, multiplications by 64, 32, 16, 8, 4, 2 and 1. These multiplications are accomplished by left shifting by 6, 5, 4, 3, 2, 1 and nil bits respectively. The additions such as 32R + 4R + 2R ; 64 G + 8G + 2G + 1G, etc. are pipelined in such a way that only two terms are added at one time. The scaled coefficients are registered on the arrival of the rising edge of the clock. Finally, we scale down by 128 in the last clock cycle and thus accomplishing YCbCr conversion in about five pipeline stages.

$$
\begin{aligned}
R &= 128Y + 180Cr = (128)Y + (128 + 32 + 16 + 4)Cr \\
G &= 128Y + 45Cb + 91Cr \\
G &= (128)Y + (32 + 8 + 4 + 1)Cb + (64 + 16 + 8 + 2 + 1)Cr \\
B &= 128Y + 228Cb = (128)Y + (128 + 64 + 32 + 4)G
\end{aligned} \tag{9}
$$

From the above expressions, multiplication can be easily replaced by left shift operation. The scaled coefficients are registered on the arrival of the rising edge of clock pulse. Finally, we scale down weights by 128 in the last clock cycle and thus accomplishing the RGB conversion from YCbCr conversion in about five pipeline stages. The top level signal diagram for RGB-YCbCr color space conversion module is shown in Fig. 4. The detailed hardware architecture for RGB to YCbCr and YCbCr to RGB color space conversion is shown in Fig. 5.

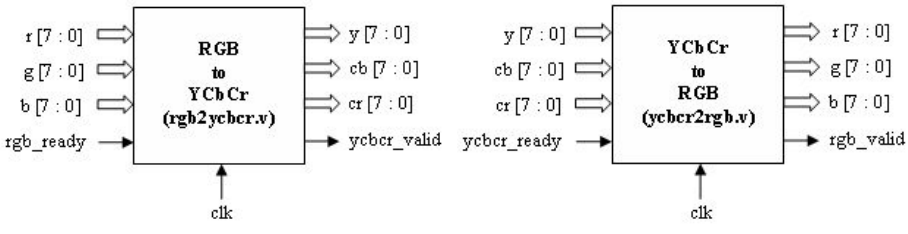

Fig. 4. Signal Diagram of RGB to YCbCr and YCbCr to RGB Converter Module

4 Experimental Results and Comparative Study

The proposed FPGA implementation of image color space conversion has been coded in Matlab (Ver. 8) and tested first in order to ensure the correct working of the color space conversion developed. The tests have provided satisfactory results. Subsequently, the entire design has been coded in Verilog HDL and has been simulated using Modelsim (Ver. SE 6.4). The Synthesis, Place & Route, and bit file generation is done using Xilinx 9.2i.

The ModelSim waveform results for RGB-HSV and RGB-YCbCr conversion is shown in Fig. 6. The Xilinx FPGA implementation of proposed RGB-HSV color space conversion is shown in Fig. 7. The proposed FPGA implementation of RGB to HSV conversion is compared with another implementation proposed by Jung et al. The data latency reported by Jung et al. method is 23 clock cycles. However, our method is capable of achieving conversion process within 16 clock cycles. The RGB to HSV and HSV to RGB color space conversion results using standard images with different environmental conditions are shown in Fig. 8.

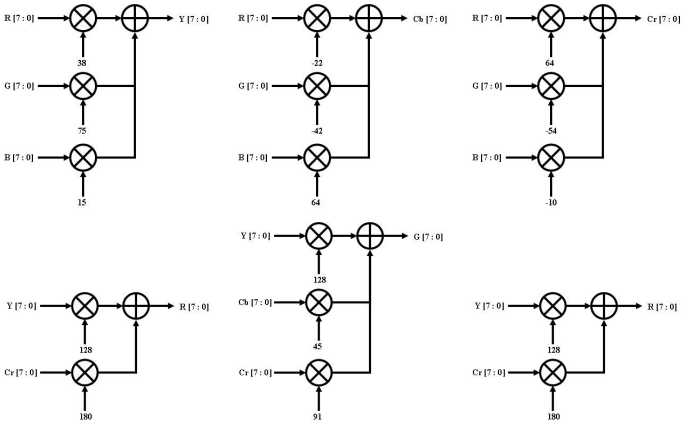

Fig. 5. First Row : RGB to Y, RGB to Cb, RGB to Cr, **Second Row** : YCbCr to R, YCbCr to G, YCbCr to B

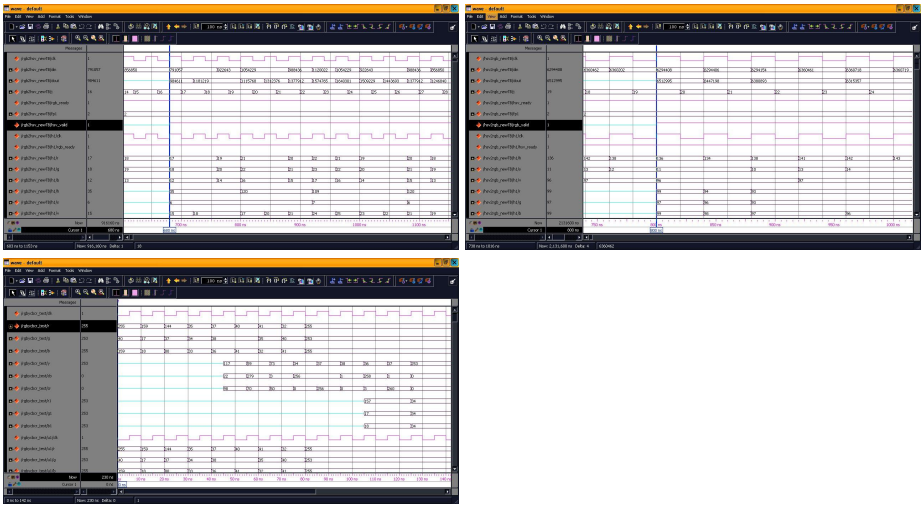

Fig. 6. First Row: Simulation Waveforms for RGB to HSV, HSV to RGB Conversion, **Second Row:** RGB to YCbCr and YCbCr to RGB Conversion

RGBHSV Project Status			
Project File:	RGBHSV.ise	Current State:	Programming File Generated
Module Name:	rgb2hsv_new	• Errors:	No Errors
Target Device:	xc2v2000-6bf957	• Warnings:	351 Warnings (340 new, 0 filtered)
Product Version:	ISE 9.2i	• Updated:	Tue Jun 7 09:21:27 2011

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	595	21,504	2%	
Number of 4 input LUTs	2,190	21,504	10%	
Logic Distribution				
Number of occupied Slices	1,366	10,752	12%	
Number of Slices containing only related logic	1,366	1,366	100%	
Number of Slices containing unrelated logic	0	1,366	0%	
Total Number of 4 input LUTs	2,501	21,504	11%	
Number used as logic	2,190			
Number used as a route-thru	95			
Number used as Shift registers	216			
Number of bonded IOBs	51	624	8%	
IOB Flip Flops	32			
Number of GCLKs	1	16	6%	
Total equivalent gate count for design	34,386			
Additional JTAG gate count for IOBs	2,448			

RGBHSV Project Status			
Project File:	RGBHSV.ise	Current State:	Programming File Generated
Module Name:	hsv2rgb_new	• Errors:	No Errors
Target Device:	xc2v2000-6bf957	• Warnings:	467 Warnings (453 new, 0 filtered)
Product Version:	ISE 9.2i	• Updated:	Tue Jun 7 09:17:07 2011

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	430	21,504	1%	
Number of 4 input LUTs	382	21,504	1%	
Logic Distribution				
Number of occupied Slices	367	10,752	3%	
Number of Slices containing only related logic	367	367	100%	
Number of Slices containing unrelated logic	0	367	0%	
Total Number of 4 input LUTs	562	21,504	2%	
Number used as logic	382			
Number used as a route-thru	14			
Number used as Shift registers	166			
Number of bonded IOBs	51	624	8%	
IOB Flip Flops	24			
Number of MULT18X18s	1	56	1%	
Number of GCLKs	1	16	6%	
Total equivalent gate count for design	21,427			
Additional JTAG gate count for IOBs	2,448			

Fig. 7. RGB to HSV and HSV to RGB Color Space Conversion Implementation using XC2V2000-6bf957 Xilinx FPGA Device

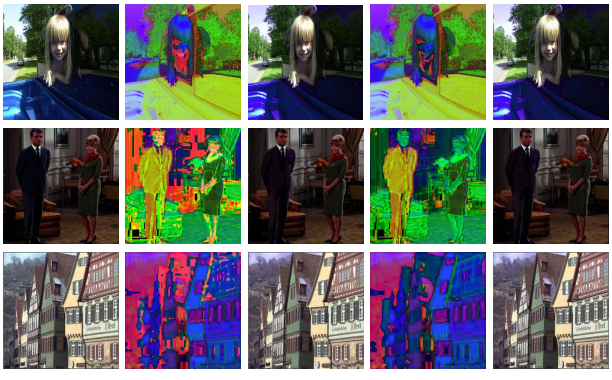

Fig. 8. RGB to HSV and HSV to RGB Conversion Results: **First Column :** Original 'Girl', 'Couple' and 'House' Images, **Second Column :** RGB to HSV Conversion using Matlab Software, **Third Column :** Restored RGB Images from HSV Color Space, **Fourth Column :**RGB to HSV Conversion using Proposed Hardware Technique, **Fifth Column :**Restored RGB Images from HSV Color Space using Proposed Hardware Technique

5 Conclusion

In this paper, efficient architectures suitable for FPGA/ASIC implementation of RGB-HSV and RGB-YCbCr color space conversion was presented. The potential applications of the proposed techniques includes image compression, image enhancement, and segmentation etc. Pipelining and parallel processing techniques are adopted in order to speed up the conversion process. The Verilog code developed for the complete system is RTL compliant and works for ASIC design. The implementation presented in this paper has been realized on an Xilinx XC2V2000-6bf957 FPGA device. The experimental results show that proposed color space conversion approaches exhibit better performances when compared with the other existing implementations.

References

1. Jain, A.K.: Fundamentals of Digital Image Processing. Prentice-Hall, Inc, Upper Saddle River (1989)
2. Bensaali, F., Amira, A.: Design and Implementation of Efficient Architectures for Color Space Conversion. ICGST International Journal on Graphics, Vision and Image Processing 5, 37–47 (2004)
3. Asmare, M.H., Asirvadam., V.S., Iznita, L.: Color Space Selection for Color Image Enhancement Applications. In: IEEE International Conference on Signal Acquisition and Processing (ICSAP), pp. 208–212 (2009)
4. Cho, J.U., Jin, S.H., Dai Pham, X., Kim, D., Jeon, J.W.: FPGA based Real Time Visual Tracking System Using Adaptive Color Histograms. In: IEEE International Conference on Robotics and Biomimetics (ROBIO 2007), pp, pp. 172–177 (2007)
5. Zhang, M., Seow, M., Tao, L., Asari, V.K.: Tunable High-Performance Architecture for Enhancement of Stream Video Captured Under Non-Uniform Lighting Conditions. Microprocessors and Microsystems 32(7), 386–393 (2008)
6. Marsi, S., Ramponi, G.: A Flexible FPGA Implementation for Illuminance-Reflectance Video Enhancement. Journal of Real-Time Image Processing, 1–13
7. Iakovidou, C., Vonikakis, V., Andreadis, I.: FPGA Implementation of a Real Time Biologically Inspired Image Enhancement Algorithm. Journal of Real-Time Image Processing 23(4), 269–287 (2008)
8. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color Based Probabilistic Tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 661–675. Springer, Heidelberg (2002)
9. Hanumantharaju, M.C., Ravishankar, M., Rameshbabu, D.R., Ramachandran, S.: Adaptive Color Image Enhancement Based Geometric Mean Filter. In: Proceedings of International Conference on Communication, Computing & Security, pp. 403–408 (2011)
10. Tkalcic, M., Tasic, J.: Color Spaces: Perceptual. Historical and Applicational Background 1 (2003)
11. Zeng, D.: Future Intelligent Information Systems, vol. 1. Springer, Heidelberg (2011); ISBN: 3642197051
12. Siddiqui, A., Nazzal, S.: Measurement of Surface Color as an Expedient QC Method for the Detection of Deviations in Tablet Hardness. International Journal of Pharmaceutics 341(1-2), 173–180 (2007)
13. Bailey, D.: Design for Embedded Image Processing on FPGAs. Wiley (2011)
14. Bensaali, F., Amira, A.: Design and Efficient FPGA Implementation of an RGB to YCbCr Color Space Converter Using Distributed Arithmetic. In: Becker, J., Platzner, M., Vernalde, S. (eds.) FPL 2004. LNCS, vol. 3203, pp. 991–995. Springer, Heidelberg (2004)
15. Ramachandran, S.: Digital VLSI Systems design: A Design Manual for Implementation of Projects on FPGAs and ASICs using Verilog. Springer, Heidelberg (2007)
16. Bensaali, F., Amira, A.: Accelerating Color Space Conversion on Reconfigurable Hardware. Image and Vision Computing 23, 935–942 (2005)