Advances in Artificial Intelligence, Big Data and Algorithms
G. Grigoras and P. Lorenz (Eds.)

doi:10.3233/FAIA230915

Fast Algorithm of Brightness and Contrast Enhancement Based on HSV

Yuxiang ZOU[a,1] and Xiangqun ZOU[b]
[a]*Department of electronic Commerce, Guangdong University of Technology, Guangzhou, 510520, China*
[b]*Guangzhou Intelligence Oriented Technology Co., Ltd., No. 604, Tian'an Technology Development Building, Tian'an Hi-Tech Ecological Park, No. 555, North Panyu Avenue, Panyu District, Guangzhou 511493, China*

Abstract. By analyzing RGB-to-HSV conversion algorithms, this paper proposes a fast algorithm for enhancing brightness and contrast based on HSV. Unlike traditional algorithm that convert RGB to HSV, enhance the brightness V, and then convert back to RGB, our approach allows for enhancing brightness and contrast directly in HSV without any conversion to RGB. Avoiding using the floating point multiplication and division, it used integer operations, shift operator and look-up table methods in this algorithm, so that improved the procession suggest admit the look-up table methods. Finally, the experiments show that the fast algorithm can save 70% of the computing time than the traditional algorithm. It can process the 720p video images real time.

Keywords. RGB, HSV, brightness, contrast

1. Introduction

With the development and popularity of computer technology and image processing technology, digital image processing is becoming more and more widely used. Colour digital images can be represented by a variety of colour space models, such as RGB, YUV and HSV. In practical applications, in order to improve the visual effect or to facilitate the analysis and understanding of the image, it is often necessary to do different degrees or aspects of enhancement of the original image, and certain image enhancement needs to be processed in a different colour space. For example, Sri Dianing Asri[1] et al. detected fish in seagrass ecosystems using Masked-Otsu algorithm in the HSV color space. Lv Chen[2] et al. proposed a stereo matching algorithm based on the HSV color space and improved census transform. Nugroho Hanung Adi[3] et al. segmented Plasmodium parasite by extracting the saturation channel of the HSV color space. Omori Yuki[4] proposed a brake lamp detection method based on the HSV color space. Fidel Indalecio Mamani Maquera[5] et al. removed wakes-ship in high-resolution optical images based on histograms in the HSV color space. Hdioud Boutaina[6] et al. detected shadows in the HSV color space using dynamic thresholds. In addition, there are a series of image enhancement algorithms based on HSV space[7-13] which are frequently

[1] Corresponding Author: Yuxiang ZOU, Senior Algorithm Engineer, Guangdong University of Technology; E-mail: nkwavelet@163.com

mentioned. However, due to the need to process large batches of data and the high demands on the speed of the algorithms with limited computational resources, especially for real-time, the interconversion between different colour spaces can no longer be limited to traditional algorithms alone. Therefore, numerous scholars have proposed some fast algorithms. For example, Qiansheng Zhou[14] optimized the conversion of YCbCr to RGB by proposing a table look-up method in media playback software that reduced computation and improved speed; Yongchao Feng et al.[15] proposed a table look-up method for converting YUV to RGB, which improves efficiency and eliminates the need for multiplication; Yunlin Liu et al.[16] proposed a table look-up method by analysing the conversion algorithm between YUV and RGB format, proposed the use of integer calculations instead of floating-point operations, using integer division by 256 corresponding to a right shift of eight bits operation, thus providing operational speed; Yifang Liu et al.[17] proposed a fast conversion algorithm for YCrCb to RGB on DSP using fixed-point shift and addition operations instead of floating-point multiplication.

A common color image enhancement algorithm involves transforming the color image through a color space. This converts the three closely related components to a largely unrelated chromaticity space, most commonly the HSV space. Luminance contrast enhancement based on the HSV space is the most common form of image enhancement.

Brightness refers to how bright an image is. Contrast refers to the measurement of the different brightness levels between the brightest white and darkest black in a light and dark area of an image, with a greater range of difference representing greater contrast and a smaller range representing less contrast. Contrast has a critical impact on visual effects. Generally speaking, higher contrast results in clearer and more striking images with vivid colors, while low contrast can make the whole image appear grey and dull. Simply put, contrast reflects the level of light and dark areas on the image. In image editing, adjusting contrast means widening or narrowing the difference between light and dark points while keeping the average brightness constant. As the average brightness is to be kept constant, the adjustment scale for each point must be applied to the difference between that value and the average brightness, so that the calculated average brightness remains the same.

There are two aspects to adjusting luminance in HSV space, one is to increase or decrease overall luminance and the other is to enhance contrast, both of which can be expressed in one formula:

$$\mathrm{Out} = \mathrm{Inc} + \mathrm{Mean} + (\mathrm{In} - \mathrm{Mean}) * (1 + P) \tag{1}$$

Where In denotes the original pixel point brightness, Inc denotes the brightness of the whole picture increased or decreased, Mean denotes the average brightness of the whole picture, Out denotes the adjusted brightness, and P is the adjustment range [-1, 1]. From this formula, it is clear that when Inc is 0 it means only contrast enhancement. When P is greater than 0 it means contrast enhancement, P is less than 0 it means contrast reduction, and when P is 0, there is no change in contrast before and after.

Example of formula (1):

Because Inc represents the increase or decrease in brightness of the entire image, it does not affect contrast. Therefore, it can be assumed that Inc=0 here. We give an example of how formula (1) enhances contrast.

Example 1: Assuming Mean=128, P=0.3, and the brightness values of pixels x_1 and x_2 are 100 and 200, respectively, substituting them into formula (1) can obtain

$$\mathrm{Out}_1 = 128 + (100 - 128) \times 1.3 = 91.6$$

$$\mathrm{Out}_2 = 128 + (200 - 128) \times 1.3 = 221.6$$

The brightness difference before applying formula (1) is 200-100=100, and the brightness difference after enhancement is 221.6-91.6=130. Therefore, it can be seen that the contrast has improved.

Example 2: Assuming Mean=128, P= -0.3, and the brightness values of pixels x_1 and x_2 are 100 and 200, respectively, substituting them into formula (1) can obtain

$$\mathrm{Out}_1 = 128 + (100 - 128) \times 0.7 = 108.4$$

$$\mathrm{Out}_2 = 128 + (200 - 128) \times 0.7 = 178.4$$

The brightness difference before applying formula (1) is 200-100=100. After applying formula (1), the brightness difference is 178.4-108.4=70. Therefore, it can be seen that the contrast has decreased.

2. Background Knowledge

2.1. Introduction to HSV Space

HSV (Hue, Saturation, Value) is a colour space based on the intuitive properties of colour, also known as the hexagonal cone model. The colour parameters in this model are: hue (H), saturation (S) and luminance (V). The HSV colour model is a colour model for the user's perception, focusing on colour representation, what colour, shade, lightness and darkness:

a) The model of the HSV colour space corresponds to a conical subset of the cylindrical coordinate system (the top face of the cone corresponds to V=1). It contains the three faces R=1, G=1 and B=1 of the RGB model, represented by the brighter colours. The colour H is measured by the angle around the V-axis at a fixed position. We specify that red corresponds to an angle of 0°, green to an angle of 120° and blue to an angle of 240°. In the HSV colour model, each colour differs from its complementary colour by 180°, i.e. yellow corresponds to 60°, cyan to 180° and magenta to 300°.

b) The saturation S is used to represent the distance from the position coordinate point to the centre of the horizontal plane, the radius of the top surface of the cone is defined as 1, so S takes values from 0 to 1. A colour with a saturation of 100 per cent in the HSV model is generally less than 100 per cent pure.

c) At the vertex (i.e. origin) of the cone, V = 0, H and S are undefined and represent black. At the top centre of the cone, S = 0, V = 1, H is undefined and represents white. From this point to the origin represents grey with diminishing brightness, i.e. grey with different shades of grey. For these points, the values of S = 0,H are undefined. It can be said that the V-axis in the HSV model corresponds to the main diagonal in the RGB colour space. A colour on the circumference of the top surface of a cone with V = 1 and S = 1, this colour is a pure colour.

The HSV colour space can be described by a cone space model, as shown in figure 1:

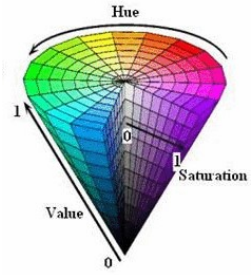

Figure 1. Visual representation of colour space

2.2. RGB space to HSV space

Let (r, g, b) represent the red, green and blue values of a colour that take on fractional values in the interval [0, 1], respectively. Let max be the largest of r, g and b, let min be the smallest of these values, H ∈ [0, 360) be the angular hue angle, and s, v ∈ [0, 1] be the saturation S and luminance V. To find the value of (h, s, v) in HSV space, the calculation is

$$\begin{cases} h = \begin{cases} 0, & \text{if } \max = min \\ 60 * \frac{g-b}{\max - min}, & \text{if } \max = r \\ 60 * \frac{b-r}{\max - min} + 120, & \text{if } \max = g \\ 60 * \frac{r-g}{\max - min} + 240, & \text{if } \max = b \end{cases} \\ s = \begin{cases} 0, & \text{if } \max = 0 \\ \frac{\max - min}{max}, & otherwise \end{cases} \\ v = max \end{cases} \tag{2}$$

2.3. HSV space to RGB space

Given a colour defined by the (h, s, v) values in HSV, h in the value range [0, 360) indicative of the hue angle, and s and v varying between 0 and 1 for saturation and luminance, the corresponding (r, g, b) triplet in RGB space can be calculated as

$$\begin{aligned} \mathrm{h}_i &= \left\lfloor \frac{h}{60} \right\rfloor \pmod 6 \\ f &= \frac{h}{60} - h_i \\ p &= v * (1 - s) \\ q &= v * (1 - f * s) \\ t &= v * (1 - (1 - f) * s) \end{aligned} \tag{3}$$

For each colour component (r, g, b), one has

$$(\mathrm{r},\mathrm{g},\mathrm{b}) = \begin{cases} (v,t,p) & if\ h_i = 0 \\ (q,v,p) & if\ h_i = 1 \\ (p,v,t) & if\ h_i = 2 \\ (p,q,v) & if\ h_i = 3 \\ (t,p,v) & if\ h_i = 4 \\ (v,p,q) & if\ h_i = 5 \end{cases} \tag{4}$$

3. Traditional Algorithms

Commonly used image data is in RGB format and each component is represented by 8 bits, so each component is an integer in the range 0 to 255. The traditional luminance contrast enhancement algorithm is to first convert the RGB values to HSV, which is a process that involves multiplying and dividing floating point numbers, as shown by the RGB to HSV formula, and a number of branching judgements in the calculation of the H component. This conversion also involves multiplying and dividing floating point numbers and multiple branching judgements. Thus the whole process involves multiple multiplications and divisions of floating point numbers, and although this is the most intuitive method, it is also the most inefficient. As a rough estimate, one conversion is equivalent to 10 floating-point multiplications, which is extremely inefficient when you consider that you have to do this for every pixel in the image. For example, an image with a resolution of 1000 x 1000 requires 100 million floating point multiplications, which is a very inefficient and inefficient way to enhance the brightness and contrast of a video image, far from meeting the demands of real time. This enhancement process can be represented by a simple model:

$$\mathrm{RGB} \rightarrow \mathrm{HSV} \rightarrow \mathrm{HSV}' \rightarrow \mathrm{R'G'B'}.$$

We give the pseudo-code for this process below:

```
while (! lastPixel)
{
    GetPixelRGB(R, G, B).
    RGB2HSV(R, G, B, H, S, V).
    Sum += V.
    PixelNumber++.
}

Mean = Sum / PixelNumber.

while (! lastPixel)
{
    GetPixelRGB(R, G, B).
    RGB2HSV(R, G, B, H, S, V).

    V = Inc + Mean + (V - Mean) * ContrastLevel.
    V = CLAMP(V, 0, 1).

    HSV2RGB(H, S, V, R, G, B).
    PutRGBToDst(R, G, B).
}
```

4. Fast Algorithms

The luminance contrast enhancement only involves the adjustment of the luminance component V. Careful analysis of the interconversion equations (2), (3) and (4) between RGB and HSV reveals that before and after the adjustment of the V component, the variables h, s, f, in the conversion equationh_i are unchanged. Assuming that the value of the luminance component before the transformation isv and the value of the luminance component after the luminance contrast enhancement isv' and that the ratio between them is$F = v'/v$ The ratio between them is . Since both the f and s variables remain unchanged and the values of p, q and t are only influenced by the variable v, after the transformation we obtain equation (5).

$$p' = F * p,\ q' = F * q,\ t' = F * t. \tag{5}$$

And since theh_i the values of are unchanged before and after the transformation, the branches in the calculation of (r', g', b') are the same as before the transformation, and in either branch, since the values of p', q', t' , and v' all differ from the original values by only a constant factor F, after the transformation we obtain equation (6).

$$(r', g', b') = F * (r, g, b) \tag{6}$$

From the above analysis, it is clear that the key to obtaining the r, g and b values after the luminance contrast enhancement is to obtain the ratio of the luminance before and after the transformation, i.e.$F = v'/v$. Again, the luminance contrast enhancement formula is listed as follows:

$$\begin{aligned} v' &= Inc + Mean + (v - Mean) * (1 + P) \\ &= Inc + Mean + (v - Mean) + (v - Mean) * P \\ &= Inc + v + (v - Mean) * P \end{aligned} \tag{7}$$

All the variables in equation (7) are decimals between [0, 1]. Considering that RGB values for images are usually integers between 0 and 255, it is necessary to derive an integer version of the luminance contrast enhancement formula in order to increase the speed of the operation. Multiplying the left and right sides of the above formula by the same 255 gives

$$V' = I + V + (V - A) * P \tag{8}$$

In equation (8), $V' = v' * 255$,$I = Inc * 255$,$V = v * 255$, and$A = Mean * 255$. Therefore there are

$$F = V'/V = v'/v \tag{9}$$

Taking the R component as an example, assuming that the enhanced value is R', we combine equation (9) to obtain

$$\begin{aligned} R' &= R * F = R * V'/V \\ &= R * [I + V + (V - A) * P]/V \end{aligned} \tag{10}$$

In equation (10), there is floating point multiplication and division, and within the margin of error we use table look-up and integer multiplication instead of division, and integer shift operations instead of floating point multiplication, thus substantially increasing the speed of the operation.

First consider $(V - A) * P$ this floating point multiplication, the actual application of floating point numbers P Some discrete values can be chosen, e.g. 0.125, 0.25, 0.375, 0.5, etc. By expressing these discrete values as binary, performing various shifts on$(V - A)$ various shift operations and then summing them, you get $(V - A) * P$.

For any integer X , we want to convertX/ V convert to an integer operation with a very small error control. Since V = Max(R, G, B) takes values in the range [0, 255], when V is 0 nothing can be done, other than the reciprocal of the divisor 1/V can be put into an array TAB[255]. To ensure that the values in the array are integers, and to keep the multiplication error as small as possible as well as the fact that each value in the array is different, we adjust the value to $2^{16}/V$ and finally the value is shifted to the right by 16 bits.

Thus for any component X in RGB, we have:

$$X' = (X * TAB[V] * V') \gg 16 , \tag{11}$$

In equation (11), $V' = I + V + (V - A) * P$, the specific expression for the TAB array is

$$TAB[V] = (int)(2^{16}/V + 0.5), \tag{12}$$

In equation (12), $V = 1, 2, \ldots, 254, 255$, int means rounded and +0.5 means rounded.

5. Illustrative Examples

In the following we give an example of the calculation process and give an error analysis of the traditional and fast algorithms. Assuming$(R, G, B) = (175,126,153)$, the average luminance$A = 138$, the intensity of contrast enhancement $P = 0.375$, the overall brightness is increased by 3.

5.1. Traditional algorithm steps

a) Calculated from the RGB to HSV formula

$$h = 326.9388, \quad s = 0.2800, \quad v = 0.6863.$$

b) Calculate the brightness after enhancement according to equation (1):

$$v' = 3/255 + 138/255 + (0.6863 - 138/255) * (1 + 0.375) \ = 0.7525.$$

c) Calculated from the HSV to RGB formula gives

$$R = 192, \quad G = 138, \quad B = 168 .$$

5.2. Fast algorithm steps

a) First calculate the luminance:

V=maxR,G,B $V = \max(R, G, B) = 175$.

b) according to the formula $V' = I + V + (V - A) * P$ and

$P = 0.375 = 1/2^2 + 1/2^3$, the calculation gives

$$V' = 3 + 175 + (175 - 138) \gg 2 + (175 - 138) \gg 3 = 191.$$

c) Based on V = 175, look up the table TAB[175] = 374.

d) Calculated separately:

$$R' = (R * TAB[V] * V') \gg 16 = (175 * 374 * 191) \gg 16 = 190.$$

$$G' = (G * TAB[V] * V') \gg 16 = (126 * 374 * 191) >> 16 = 137.$$

$$B' = (B * TAB[V] * V') \gg 16 = (153 * 374 * 191) >> 16 = 166.$$

In this example, we obtain that the error of the fast algorithm relative to the conventional algorithm is

$$|192 - 190| + |138 - 137| + |168 - 166| = 5$$

The error is defined here as the sum of the absolute values of the differences of each component.

Although there is some error in the results obtained by the fast algorithm compared to the traditional algorithm, the error is very small. We have programmed the fast algorithm to verify that the error is very small, and the results of the program show that the average error is within 2 and the maximum error is no more than 5, which is too small to be distinguished by the naked eye.

6. SIMD Introduction

SIMD (abbreviation for Single Instruction Multiple Data), is a type of data stream that uses a single controller to control multiple processing units , while simultaneously processing a set of data (also known as " data vector It is a technique that uses a single controller to control multiple processing units to simultaneously perform the same operation on each of a set of data (also known as a "data vector"), thus achieving spatial parallelism. SIMD technologies currently supported by Intel processors include MMX, SSE, SSE2, SSE3, SSE4, AVX and AVX2.

SIMD technology first appeared on the Pentium MMX processor, corresponding to the MMX instruction set, which at the time was SIMD support for integer operations, using 64-bit floating-point registers to operate, allowing two 32-bit integers or four 16-bit short integers to be stored at the same time for operations, exponentially improving the processor's media computing performance.

SSE2 was launched in 2001 with the release of Intel's first generation Pentium 4 processor, the instruction set was also released. It extends the earlier SSE instruction set and can completely replace the MMX instruction set. Compared to SSE, SSE2 extends the MMX technology and SSE technology with 144 new instructions that improve the performance of a wide range of applications. The SIMD integer instructions introduced with the MMX technology have been extended from 64 to 128 bits, resulting in an exponential increase in the effective execution rate of SIMD integer type operations. sse2 has twice the register capacity of the MMX registers and twice the register storage data. SSE2 is a significant performance improvement for the processor, as programs and software can run twice as fast with SSE2 optimisation, while the instruction processing speed remains the same.

AVX is based on the SSE instruction set by extending the 128-bit XMM register into a 256-bit-length YMM register, making it support 256-bit vector calculations, and AVX is fully compatible with SSE/SSE2/SSE3/SSE4, which means that the lower 128 bits of the YMM register are the XMM register. As many of the current ARM embedded processors do not support the AVX instruction set. In many scenarios, such as live video streaming, video surveillance and online video editing, images need to be enhanced in real time in an embedded environment. Therefore on embedded platforms we still use the SSE family of instruction sets to handle parallel computing.

7. Experimental Results

Two platforms were chosen for testing, one was a PC platform with an Intel Core i5-3570 CPU @ 3.40 GHz and 8 GB of RAM, and the other was a laptop with an Intel Core i5-4200 CPU @ 2.50 GHz and 4 GB of RAM. We also used SSE2 instructions to optimise the fast algorithm. The speed increase is very significant and the test data is as follows:

Table 1. Comparison of consumption times for different resolutions on PC platforms

Algorithms	1920 × 1080	1280×720	640×480	400×300
Tradition	125ms	64 ms	25 ms	12 ms
Fast	35 ms	18 ms	6 ms	2 ms
Fast (SSE2)	25 ms	11 ms	4 ms	1 ms

Table 2. Comparison of consumption times for different resolutions on Laptop platforms

Algorithms	1920 × 1080	1280×720	640×480	400×300
Tradition	140 ms	78 ms	47 ms	16 ms
Fast	48 ms	30 ms	16 ms	2 ms
Fast (SSE2)	32 ms	16 ms	10 ms	1 ms

From Table 1 and Table 2 test data, it can be seen that the speed of the fast algorithm is significantly improved compared to the traditional algorithm, and the combination of the SSE2 multimedia instruction set increases the speed even more, which can meet the real-time enhancement of 720p video images.

8. Conclusion

This paper presents a fast algorithm for HSV luminance contrast enhancement that can meet the real-time requirements of video image enhancement. Table look-up, shift and integer multiplication are used instead of floating-point multiplication and division, reducing the complexity of the operation and greatly increasing the speed of the operation. The final experimental results show that the algorithm proposed in this paper runs much faster than traditional algorithms on PCs and laptops, and therefore has good applicability in video image enhancement.

References

[1] Sri Dianing Asri,Indra Jaya,Agus Buono,Sony Hartono Wijaya. Fish Detection in Seagrass Ecosystem using Masked-Otsu in HSV Color Space[J]. International Journal of Advanced Computer Science and Applications (IJACSA),2022,13(12).

[2] Lv Chen,Li Jiahan,Kou Qiqi,Zhuang Huandong,Tang Shoufeng. Stereo Matching Algorithm Based on HSV Color Space and Improved Census Transform[J]. Mathematical Problems in Engineering,2021,2021.

[3] Nugroho Hanung Adi,Goratama Rezqy Dwikara,Frannita Eka Legya. Saturation channel extraction of HSV color space for segmenting Plasmodium parasite[J]. IOP Conference Series: Materials Science and Engineering,2021,1088(1).

[4] Omori Yuki. HSV Color Space Based Lighting Detection for Brake Lamps of Daytime Vehicle Images[J]. Journal of Computers,2019,14(1).

[5] Fidel Indalecio Mamani Maquera,Eveling Gloria Castro Gutierrez. Wakes-Ship Removal on High-Resolution Optical Images based on Histograms in HSV Color Space[J]. International Journal of Advanced Computer Science and Applications (IJACSA),2018,9.

[6] Hdioud Boutaina,Haj Tirari Mohammed El,Haj Thami Rachid Oulad,Faizi Rdouan. Detecting and Shadows in the HSV Color Space using Dynamic Thresholds[J]. International Journal of Electrical and Computer Engineering (IJECE),2018,8(3).

[7] S. Biswas and S. Barma, "Foldscope Image Enhancement in HSV Space by PSO Optimization Technique," 2019 12th Biomedical Engineering International Conference (BMEiCON), Ubon Ratchathani, Thailand, 2019, pp. 1-5, doi: 10.1109/BMEiCON47515.2019.8990276.

[8] Dazhang You, Jatao Tao, Yepeng Zhang, Min Zhang. Low-light image enhancement based on grayscale transform and improved Retinex[J]. Infrared Technology,2023,45(02):161-170.

[9] Kui Wang, Fuzhen Huang. Illumination compensation-based multi-scale Retinex image enhancement in HSV space[J]. Advances in Laser and Optoelectronics,2022,59(10):102-113.

[10] Qilong Sun, Xin Guo, Zhenhui Si, Ping Yu. An algorithm for fog image and dust image enhancement[J]. Journal of Jilin University (Science Edition),2022,60(02):381-391.DOI:10.13413/j.cnki.jdxblxb.2021124.

[11] Yixin Cao, Yaonan Ding. Extraction method of track surface area based on S-component of HSV colour space[J]. Journal of Nanjing University of Technology,2021,45(04):464-471.DOI:10.14177/j.cnki.32-1397n.2021.45.04.010.

[12] Hongying Zhang, Jindong Zhao. RetinexNet low-illumination image enhancement algorithm in HSV space[J]. Advances in Laser and Optoelectronics,2020,57(20):294-301.

[13] Qiang Wang, Pei Tao, Bo Yuan, Liqiang Wang. A multi-color space approach for endoscopic image vascular enhancement[J]. Optoelectronic Engineering,2020,47(01):48-53.

[14] Qiansheng Zhou, Lin Dai. Fast look-up table method to optimize YCrCb to RGB conversion in video decoding[J]. Modern Electronics Technology, 2007(15):167-169.

[15] Yongchao Feng, Min Luo, Guiming He. A fast YUV-RGB colour space transformation method[J]. Microcomputers and Applications, 2002(7):59-60.

[16] Yunlin Liu, Shudong Wang. YUV and RGB colour space conversion based on SSE2[J]. Chinese Journal of Image Graphics, 2010, 15(1):45-49.

[17] Yifang Liu, Zhaoguang Liu, et al. A fixed-point DSP-based YCbCr to HSV The fast conversion algorithm[J]. Computer Applications Research, 2012, 29(2):741-743.