View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by CiteSeerX

DESIGN AND IMPLEMENTATION OF A REAL-TIME GLOBAL TONE MAPPING PROCESSOR FOR HIGH DYNAMIC RANGE VIDEO

Tsun-Hsien Wang[1] , Wei-Su Wong[2] , Fang-Chu Chen[3], and Ching-Te Chiu[2]

[1]SoC Technology Center, Industrial Technology Research Institute, Hsin-Chu, Taiwan, R.O.C..
[2]Department of Computer Science, National Tsing Hua University, Hsin-Chu, Taiwan, R.O.C.
[3]Information & Communications Research Laboratories, Industrial Technology Research Institute, Hsin-Chu, Taiwan, R.O.C.
Email: [1]thwang@itri.org.tw, [2]ctchiu@cs.nthu.edu.tw

ABSTRACT

*As the development in high dynamic range (HDR) video capture technologies, the bit-depth video encoding and decoding has become an interesting topic. In this paper, we show that the real-time HDR video display is possible. A tone mapping based HDR video architecture pipelined with a video CODEC is presented. The HDR video is compressed by the tone mapping processor. The compressed HDR video can be encoded and decoded by the video standards, such as MPEG2, MPEG4 or H.264 for transmission and display. We propose and implement a modified photographic tone mapping algorithm for the tone mapping processor. The required luminance wordlength in the processor is analyzed and the quantization error is estimated. We also develop the digit-by-digit exponent and logarithm hardware architecture for the tone mapping processor. The synthesized results show that our real-time tone mapping processor can process a NTSC video with 720*480 resolution at 30 frames per second.*

Keywords: High Dynamic Range (HDR), Low Dynamic Range (LDR), tone mapping, tone reproduction, global tone mapping.

1. INTRODUCTION

The dynamic range of natural world luminance can reach 10^8:1 but that of displayed luminance is about 100~1000:1 for most screen devices such as CRTs or LCDs. The discrepancy makes the accurate display of real world scenes difficult. As the advances of HDR capture technologies, HDR images or videos are available. Image processing schemes such as tone mapping resolve the issue of rendering HDR images on low dynamic range (LDR) displays while preserving the visual contents.

Tone mapping is commonly classified into the "global" and "local" tone reproduction [2]. The global tone reproduction adopts the same mapping scheme for all the pixels in the image. This makes it computationally efficient. Miller, et al., propose a global tone mapping scheme using nonlinear operators [5]. Tumblin and Rushmeier introduce the tone reproduction based on the human visual system [6]. This system is designed to preserve the apparent brightness of an image based on the actual luminance presented in the image and the target display characteristics. Since most nonlinear mappings use logarithmic and exponential operators, Drago et al. show how logarithmic response curves can be extended to handle a wider dynamic range than the simple operators [7]. Furthermore the logarithmic responses are close to the human visual system; Reinhard et al., present the photographic tone reproduction method for digital images. The method is simple and produced good results for a wide variety of images [1].

The global operators of tone reproduction are mostly limited in their capability to compress HDR images. To some extent, local operators that compress each pixel according its luminance value as well as the luminance values of neighboring pixels resolve this problem. Classic local tone reproduction operators are to determine how many neighboring pixels needed in the computation, how to weight each neighboring pixel's contribution, and how to use this adaptation level within a compressive function [8,9,10]. Consequently, the computation of local tone mapping is complicated.

The computation time of global tone mappings is shorter than that of local tone mappings. The time required to execute each tone-mapping operator is analyzed and some results are summarized in Table 1 [2]. To compress a 1600x1200 image with an Apple iBook G3 (800MHz) processor takes 0.96 to 3.7 seconds for different global tone mapping schemes. The run time is much longer for local tone mapping operators.

GLOBAL Operators	Time (sec.)	**LOCAL Operators and Others**	Time (sec.)
Tumbin–Rushmeier's operator	3.2	Chiu's spatially variant operator	10
Ward's scale factor	0.96	Ashikhmin's operator	120.0
Drago's logarithmic operator	2.8	Reinhard's local photographic operator	80.0
Reinhard's global photographic operator	3.7	Fattel's gradient domain compression	45.0

Table 1. Run time for different tone mapping operators

Due to rapid progress in the HDR video capture technology, HDR video display on conventional LCD devices becomes crucial. We apply tone mapping operators on each video frame before display. The tone mapping operations need to compress each HDR video frame within 1/30 sec. As indicated by Table 1, it appears that the computational speed for most of the tone mapping schemes can not meet the video display requirement. Therefore, we

focus on the development of a real-time tone mapping processor for HDR video applications.

The tone mapping based HDR video architecture pipelined with a video CODEC is presented in section II. The modified tone mapping scheme, the color reproduction and gamma correction are also described. The luminance wordlength selection and precision analysis are discussed in section III. The design and implementation of a real-time tone mapping processor is presented in section IV. Finally, section V gives the conclusion.

2. GLOBAL TONE MAPPING AND COLOR REPODUCTION

The tone mapping based HDR video architecture pipelined with a video CODEC is shown in Figure 1. The original HDR video is in the 32-bit per pixel RGBE format with 76 orders of dynamic range. The HDR video is compressed by the tone mapping processor frame by frame. The compressed HDR video contains the 8-bit/channel RGB data that are ready for LDR display or video CODEC processing. The video CODEC standards, such as MPEG-2, MPEG-4 or H.264, can be used for video transmission and storage purpose in the architecture.

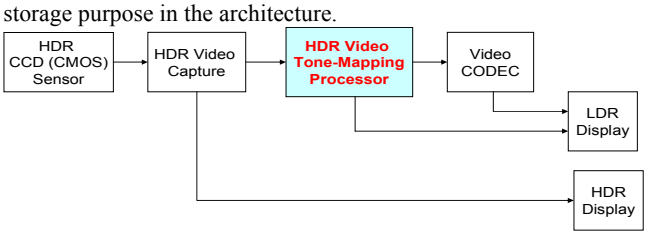


Figure 1. The tone mapping based HDR video architecture pipelined with a video CODEC.

Only global tone mapping schemes are considered in the tone mapping processor due to the speed requirement of video display. In ref [2], it shows that the photographic tone reproduction proposed by E. Reinhard works the best. The photographic tone reproduction has lower complexity and better quality for HDR images. We develop a real-time tone mapping processor for HDR video based on this photographic tone reproduction scheme. Reinhard's method [1] is delveloped for HDR images and the modified photographic tone reproduction for HDR video is described below.

The luminance of each HDR video frame is calculated first as Eq. (1).

$$L_w = 0.2654\mathrm{R}_{\mathrm{w}} + 0.6704\mathrm{G}_{\mathrm{w}} + 0.0642\mathrm{B}_{\mathrm{w}} \qquad (1)$$

where R_w, G_w and B_w are scene-referred color and L_w is the scene luminance. After the luminance of each video frame are obtained, the log-average luminance L_{avg} is calculated as shown Eq. (2a). The log-average luminance is a useful information for calcualting the key of the scene.

$$L_{avg} = \exp\left(\frac{1}{N}\sum \log(\delta + L_w)\right) \qquad (2a)$$

In Eq. (2a), the N indicates the number of pixels in the video frame, and δ is a very small value. In our implementation, we choose δ as 0.0001.

For a video sequence, the temporal frames are highly correlated with successive video frames hence we modify Eq (2a) for HDR video frames as shown in Eq. (2b).

$$L_{avg}(t_k) = \exp\left(\frac{1}{N}\sum \log(\delta + L_w(i,j,t_{k-1}))\right), \quad for\ k \neq 1 \qquad (2b)$$

where the $L_{avg}(t_k)$ indicates the average scene luminance of the k-th HDR video frame and $L_w(i,j,t_{k-1})$ indicates the scene luminance of pixel *(i,j)* of the *(k-1)*-th HDR video frame.

Next, the average or normal luminance can be mapped to a desired key controlled by the parameter α with a linear scaling. The scaling luminance $L_s(i,j,t_k)$ is shown in Eq. (3).

$$L_s(i,j,t_k) = \frac{\alpha}{L_{avg}(t_{k-1})} L_w(i,j,t_k) \qquad (3)$$

The parameter α affects the brightness of the video and α=0.18 is chosen to map the normal luminance to middle-grey key.

For display consideration, we use Eq. (4) to map the high luminance in a controllable way.

$$L_d(i,j,t_k) = \frac{L_s(i,j,t_k)\left(1.0 + \frac{L_s(i,j,t_k)}{L^2_{white}(t_{k-1})}\right)}{1.0 + L_s(i,j,t_k)} \qquad (4)$$

The L_{white} is the luminance value of the white point. The maximum luminance value of the previous video frame is used as the white point for the current frame.

The ratio of the color channels before and after compression is kept constant to maintain the color shift to be minimum [2]. This can be achieved if the compressed image R_d, G_d, B_d is computed as follows:

$$C_d = L_d \frac{C_w}{L_w} \qquad (5)$$

where C_w indicates R_w, G_w, B_w. The C_w and L_w denote the color and luminance before the HDR compression. The C_d indicates R_d, G_d, B_d. The C_d and L_d denote the color and luminance after the HDR compression.

In order to control the amount of saturation in the image, we apply an exponent γ at the ratio (C_w/L_w) in Eq. (5). It represents a gamma correction per channel as follows.

$$C_d = L_d\left(\frac{C_w}{L_w}\right)^{\gamma} \qquad (6)$$

The exponent γ is given as a user parameter that takes values between 0 and 1.

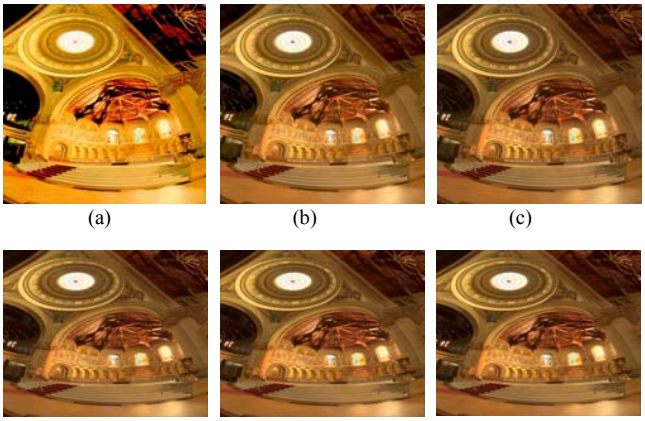
3. WORDLENGTH AND PRECISION

The number of bit used for luminance in the hardware implementation affects the image quality and hardware cost. Considerations of both cost and quality are important issues before the hardware realization. The wordlength of the luminance L_w is analyzed below.

The value of the luminance includes the integer and fractional parts. In our design, the targeted dynamic range of luminance is from 10^{-3} to 10^{5} so we choose 16 bits for the integer part. The quantization error ε satisfies Eq. (7), where y and y_k indicate the full precision and quantized value with k-bit fractional part.

$$0 \le y - y_k = \varepsilon < 2^{-k+1} \qquad (7)$$

If the number of bits in the fractional part is 16, the quantized error can be reduced to $3*10^{-5}$. The images with different number of the fractional bits are shown in Figure 2 and Figure 3. The PSNR for different fractional precision per color channel are shown in Table 2 and Table 3. According to the above analysis, over 40dB average PSNR and satisfied image quality can be reached with the 16 bits realization for the fractional part. As the result, a 32-bit wordlength (16 bits for the integer and 16 bits for the fraction) is used to implement a tone mapping processor.



(a) (b) (c)

(d) (e) (f)

Figure 2. The (a)~(f) indicate the fractional precision of 8 bits, 10bits, 12 bits, 16 bits, 20 bits and floating point realization respectively.

(a) (b)

(c) (d)

Figure 3. The (a)~(d) indicate the fractional precision of 8 bits, 12 bits, 16 bits and floating point realization respectively.

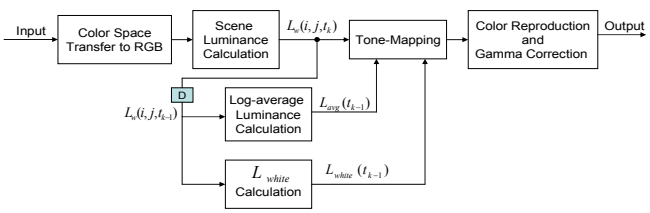


Figure 4. The overall architecture of the photographic tone mapping processor.

Precision (bits)	PSNR R (dB)	PSNR G (dB)	PSNR B (dB)	PSNR AVG (dB)
8	11.93	15.48	21.43	16.28
10	26.18	28.54	30.98	28.57
12	38.87	40.45	39.82	39.71
16	43.31	44.05	42.43	43.26
20	55.06	55.71	53.71	54.83

Table 2. PSNR comparisons for different fractional precision per color channel of Figure 2

Precision (bits)	PSNR R (dB)	PSNR G (dB)	PSNR B (dB)	PSNR AVG (dB)
8	9.72	8.05	15.36	11.04
12	25.93	29.86	33.11	29.63
16	45.05	46.49	43.50	45.01

Table 3. PSNR comparisons for different fractional precision per color channel of Figure 3

4. REAL-TIME PHOTOGRAPHIC TONE MAPPING PROCESSOR

The overall architecture of the photographic tone mapping processor is shown in Figure 4. The log-average luminance and white point value of the previous video frame are calculated and used in the current frame tone mapping. It is to insure the real-time operation for video display. In the modified photographic tone mapping algorithm, operations such as division, logarithm, and exponent are needed.

Fully pipelined hardware architectures for the division, logarithm, and exponent are implemented for real-time operations. Because the power function can be replaced by the logarithm and exponent functions, we only discuss the exponent and logarithm operators here. We adapt a digit-by-digit algorithm that uses only shift registers and adders to reduce the hardware complexity. The digit-by-digit exponent operation is described below [4].

We calculate the exponent function by iteration and consider the case that x is limited to the range [0, ln2] first. The power x of natural number e is set to y, i.e. $y = e^x$. We approximate the y value by building a set of data pair (x_i , y_i) and set the initial value as ($x_0 = x$, $y_0 = 1$). Each x_i and y_i always satisfy Eq. (8).

$$y_i \cdot e^{x_i} = e^x \quad \text{or} \quad y_i = e^{-x_i} \cdot e^x \qquad (8)$$

The x_i is updated by subtracting a constant k_i where $k_i = \ln(b_i)$ as shown in Eq. (9).

$$x_{i+1} = x_i - k_i = x_i - \ln(b_i) \qquad (9)$$

We take the $b_i = 1 + s_i \cdot 2^{-i}$, where the $s_i \in \{0,1\}$. We store the values of $\ln(1+2^{-i})$ with 0<=i<=15 to a ROM for table lookup. If we take 16-bit precision, the ROM size is 16*16 bits. At each iteration, we compare x_i and $\ln(1+2^{-i})$. If $x_i >= \ln(1 + 2^{-i})$, we choose $s_i = 1$ and

$x_{i+1} = x_i - \ln(b_i)$, otherwise we choose the s_i as zero and $x_{i+1} = x_i$. When x_i equals to zero, then $y_i = e^{-x_i} \cdot e^x = e^x$. The final exponent is computed iteratively from Eq. (10).

$$y_{i+1} = e^{-x_{i+1}} \cdot e^x = e^{-(x_i - k_i)} \cdot e^x = e^{k_i} \cdot y_i = b_i \cdot y_i \qquad (10)$$

Since all b_i are either 1 or $1+2^{-i}$, we can use a shift-and-add method to get the $b_i \cdot y_i$. Notice that we only calculate e^x in this way when x is limited to the range [0, ln2]. For any arbitrary number *x*, we can take $x \cdot \log_2 e = I + f$ or $x = (I+f)\ln 2$, where *I* and *f* are the integer and fractional part of $x \cdot \log_2 e$, respectively. Then

$$y = e^x = e^{(I+f)\ln 2} = e^{I\ln 2 + f\ln 2} = 2^I e^{f\ln 2} \qquad (11)$$

where $0 \le f \ln 2 < \ln 2$. Take $x_0 = f \ln 2$ and we can use the iteration method to calculate exponential function. The item 2^I can be implemented by the shift operation to reduce hardware complexity for any arbitrary number. The architecture of the exponential operation is shown in Figure 5.

The logarithm operation $y = \log(x)$ is similar to the exponential operation mentioned above. The digit-by-digit algorithm is suitable for hardware implementation and the architecture of logarithmic operation is similar to Figure 5.

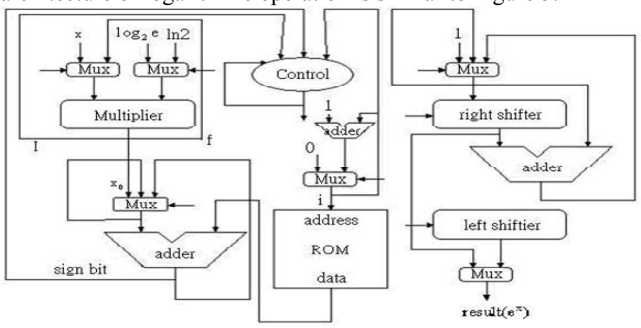


Figure 5. The architecture of the exponential operation

To improve the data throughput rate, we adapt the pipeline architecture. The images of the hardware implementation and software simulations with the same number of fractional bits are shown in Figure 6(a) and Figure 6(b) respectively. The synthesis results show that photographic tone mapping process can run at 30 MHz for a video stream with frame size 720*480. The total area is 4.18 mm^2 based on TSMC 0.18 um technology.

5. CONCLUSION

In this paper, we present a tone mapping based HDR video architecture pipelined with a video CODEC. The original HDR video is compressed by the tone mapping processor. The compressed HDR video can be encoded and decoded by the video standards for transmission and display. We propose a modified photographic tone mapping algorithm for the tone mapping processor. We analyze the required wordlength for the luminance and estimate the quantization error. We also present and implement the digit-by-digit exponent and logarithm algorithm hardware architecture. The synthesis results show that our real-time tone mapping processor can process a NTSC video with 720 *480 resolution at 30 frames per second.

(a) (b)

Figure 6. (a),(b) are hardware implementation result and software simulation result respectively.

ACKNOWLEDGEMENTS

CTC acknowledges support from National Science Council project No. NSF-95-2220-E-007-033. We also acknowledge support from Information and Communications Research Laboratories, ITRI. THW acknowledges support from SoC Technology Center, ITRI.

REFERENCES

[1] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic Tone Reproduction for Digital Images", *SIGGRAPH*, pp. 267 – 276, 2002

[2] E. Reinhard, G. Ward, S. Pattanaik, and P. Debevec "High Dynamic Range Imaging", *Morgan Kanfmann.*

[3] I. Koren, "Computer Arithmetic and Algorithms". Prentice Hall,

[4] V. Kantabutra, "On Hardware for Computing Exponential and Trigonometric Functions", *IEEE Trans., Computers,* vol. 45, NO. 3, March 1996

[5] G.S. Miller and C. R. Hotffman, "Illumination and Reflection Maps : Simulated Object in Simulated and Real Environments", *SIGGRAPH 84 Course Notes for Advanced Computer Graphics Animation,* July,1984.

[6] J. Tumblin and H. Rushmeier, " Tone Reproduction for Computer Generated Image", *IEEE Computer Graphics and Application,* pp42-48, November ,1993.

[7] F. Drago, K. Myszkowski, T. Annen, and N. Chiba. " Adaptive Logarithmic Mapping for Displaying High Contrast Scene", *Computer Graphics Forum,* 2003.

[8] K. Chiu, M.Herf, P.Shirley,. S. Swamy, C. Wang, and K. Zimmerman, " Spatially Nonumuniform Scaling Function for High Contrast Images", in *Proceeding of Graphics Interface '93,* pp.245-253, May 1993.

[9] M. Ashikhmin, "A Tone Mapping Algorithm for High Contrast Images", *Proceeding of 13 th Eurographics Work shop on Rendering ,* pp.145-155, 2002.

[10] R. Fattal , D. Lischinski, and M. Werman., " Gradient Domain High Dynamic Range Compression", *ACM Trans. on Graphics*, pp.249-256, 2002.