ARCHITECTURE OF MPEG-7 COLOR STRUCTURE DESCRIPTION GENERATOR FOR REALTIME VIDEO APPLICATIONS

Jing-Ying Chang, Hung-Chi Fang, Yen-Wei Huang, and Liang-Gee Chen

DSP/IC Design Lab.,
Graduate Institute of Electronics Engineering,
National Taiwan University, Taipei, Taiwan
Email: {jychang, honchi, yenwei, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

Color structure descriptor (CSD) provides satisfactory image indexing and retrieval results among all color-based descriptors in MPEG-7. The superiority comes from the consideration of space distribution of colors. Hardware accelerator is a must because its good performance is at the expense of high computational complexity. In this paper, a design approach of specific hardware accelerators for descriptors is explored. The characteristics of CSD algorithm are also analyzed and an efficient architecture is proposed. The proposed architecture can generate CSD description of 256×256 image at 30 frames per second (fps). The architecture provides about 4.5 giga instructions per second (GIPS) to achieve real-time applications like assisting rate control in video coding system and circumstance change detection in surveillance system.

1. INTRODUCTION

With mature digital video technology, inexpensive camcorders gradually enter our life. More and more multimedia are produced and shared among the world. Original intention of MPEG-7 is to provide a powerful search engine which helps people easily find what they are looking for. Several MPEG-7 toolkits further integrate useful functionalities for categorizing and organizing their personal collection. However, some related research showed most people only categorize their albums at semantic level. The recognition technique nowadays is still not able to meet this kind of demand [1]. MPEG-7 descriptors are good tools for indexing and retrieval but should not be limited to them. Those descriptors can be creatively extended and linked to applications such as rate control in real-time video coding and movement detection in surveillance systems. In these applications, computational loads of the real-time implementation for these descriptors will not be a trivial issue.

With statistics derived from MPEG-7 descriptors, good indication of image and video properties can provide referable adjustment parameters for video pre-processing like auto white balance, RGB gains tuning, saturation control, auto contrast, and edge enhancement. In video coding, they can assist fast algorithm of motion estimation, rate control policy, probability distribution model of entropy coding, and so on [2]. A recent research showed that edge histogram descriptor and scalable color descriptor are applied to segmentation for content-based video coding [3]. When we use them in surveillance system, the system can notice police to keep an eye on unusual behavior by analyzing object trajectory. Face descriptor can also provide auto identification of uncertified people in certain degree.

MPEG-7 visual descriptors record statistics of images and video sequences in color, texture, shape of objects, and motion. Because the variety of possible applications, we first take implementation of color descriptors as our start point. Color is one of important visual attributes for human vision and image processing. It is also an expressive visual feature in image and video retrieval. Color descriptions usually are irrelevant to viewing angle, translation and rotation. This advantage possesses good resistance to undesired shacking of camera. In MPEG-7, six descriptors are selected to record color statistics of images and video. Among them, Color structure descriptor (CSD) provides best image indexing and retrieval results [4]. The superiority comes from that CSD considers space distribution of pixel colors by recording appearance of each color in every structuring window (SW) in its histogram [5]. In this paper, we focus on the architecture and analysis of CSD.

The challenge to realize CSD hardware accelerator for real-time video system is that each pixel in one frame needs to be scanned 64 times on average. The vast data bandwidth and then excessive operating frequency make CSD unsuitable for real multimedia systems. Analysis of the trade-off between input bandwidth and local buffer size is the first issue needed to be evaluated. Then, the index algorithm of the color appearance in one SW has to be considered carefully to lower operating frequency. Moreover, along with exploring suitable solutions, hardware extensibility should not be left behind. It is worth to integrate with other descriptors with small overhead.

Operational analysis of software simulation is shown in Table 1. "Accumulation" comprises related operations of moving SW and CSD histogram accumulation. For a video sequence with frame size 256×256, 30 frames per second (fps), 4.5 GIPS and 6.2 giga bytes per second (GB/s) of memory bandwidth are required in one second. Such computational cost is the reason why CSD can not be applied to real-time products without a hardware accelerator. And there is no good solution at present.

In this paper, we first describe briefly the algorithm of CSD in section 2. Before going into implementation details of each functional block, computational complexity is discussed in section 3 and then each block design. Section 4 shows indexing and retrieval results of CSD and scalable color descriptor using different color spaces. Section 5 is dedicated to concluding remarks and future research.

2. COLOR STRUCTURE DESCRIPTOR

CSD represents an image by color accumulation and the local spatial distribution of colors. The procedure of CSD histogram uses a

0-7803-8554-3/04/$20.00 ©2004 IEEE.

Table 1. MIPS and memory bandwidth of CSD generator. 4.5 giga instructions per second (GIPS) and 6.2 GB/s of memory bandwidth is the reason why CSD is not suitable for real-time application on software platform.

Operation	1 fps		30 fps	
	Number of instructions (MIPS)	Memory bandwidth (MBytes)	Number of instructions (MIPS)	Memory bandwidth (MBytes)
HMMD	5.625	3.585	168.750	107.550
Accumulation	143.657	202.456	4309.710	6073.680
Quantization	0.051	0.001	1.517	0.039
Others	0.990	0.697	29.713	20.901
Total	150.323	206.739	4509.690	6202.170

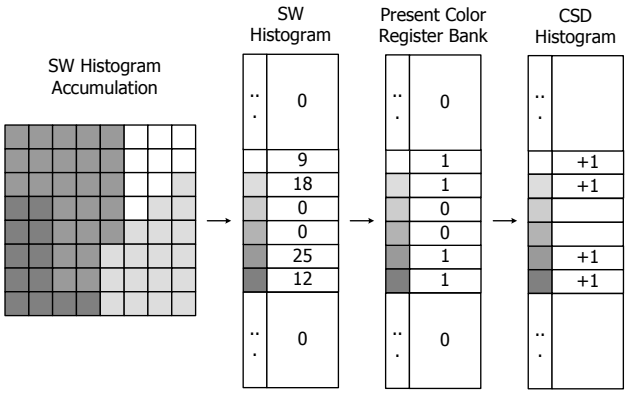

Fig. 1. SW histogram accumulation. Present color registers use 1-bit for each color to indicate which color presents in the SW. After completing a SW, CSD histogram is updated according the result of the register bank.

8×8 SW, which shifts one row or one column at a time, to observe which colors are presented in it, and then updates those color bins by only adding one, no matter how many same color pixels exist. This procedure is shown in Fig. 1. Figure 2 shows that two images have different CSD description with the same traditional histogram [6]. Right image looks more scattered than left one. Such situation causes gray pixels exist in more SWs and reflects on gray bin in CSD description. This advantage let us easily distinguish those images with similar dispersion.

Figure 3 depicts CSD extraction procedure [7]. Our design chose highest number of bins for more precise CSD description in real-time applications. The top path directs the flow of 256-bin CSD who starts with color transformation from RGB to HMMD. Next step is histogram accumulation which is followed by a decision of number of bins needed. After a nonlinear quantization, CSD description is derived.

3. ANALYSIS AND PROPOSED ARCHITECTURE

As described in Section 1, we focus on real-time applications of MPEG-7 like video coding assistance and surveillance systems. Besides, generated CSD descriptions still can be used for search of multimedia contents. And for supporting comparison with descriptions generated by other tools, 256 levels of color quantization is adopted for downscale comparison.

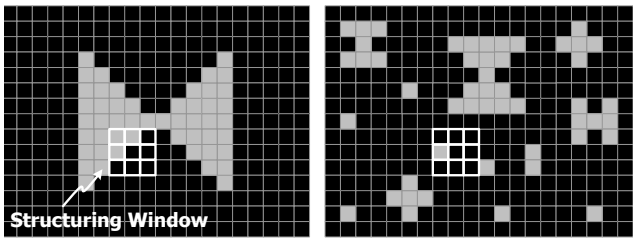

Fig. 2. Two images have the same traditional histogram, but right one has much more gray components in CSD description.

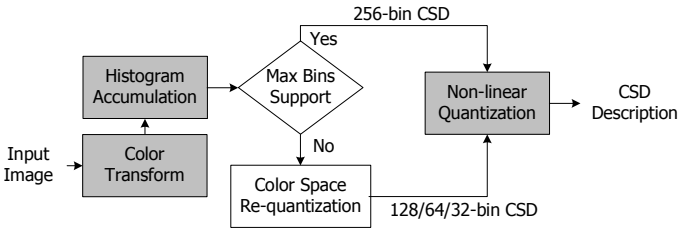

Fig. 3. CSD extraction flow.

The block diagram of the proposed architecture is shown in Fig. 4. After color transformation, pixels are sent to the corresponding local histogram observing (LHO) block to index which colors exist in current SW. Then, the output of LHO updates CSD histogram. Finally, CSD description is obtained via non-linear quantizing the final histogram.

3.1. Analysis

Since a sub-sample factor is defined in CSD for large images, we choose 256×256 as input image size without losing of generality. The defined sub-sample factor $K = \max\{1, 2^{\lfloor \log_2 \sqrt{W \cdot H} - 7.5 \rfloor}\}$, where W and H are the width and height of image. For example, $K = 2$ implies an image is sub-sampled by 2 horizontally and vertically. Note that the SW size is always 8×8.

Specification of our CSD generator is for the video sequence running at 30 fps. The operating frequency is 60 MHz if one SW (8×8 pixels) is buffered for data sharing. Approximately, in the situation of no local buffer of SW, each pixel in every window has to be scanned again even though it has been scanned during the period of operations of last neighboring window. The memory bandwidth is about 357 megabytes/sec (MB/s) and the required operating frequency is 119 MHz. In fact, we assume histogram can be updated once in one cycle to make this chip running at 119 MHz. But according to the problem described in subsection 3.2, it takes four cycles to update one pixel data on average and forces the required operating frequency to 476 MHz. Buffering one SW will reduce memory bandwidth to 46 MB/s and operating frequency to 60 MHz.

The way to record which colors exist in a SW efficiently is another main issue. It is unrealistic to query all pixels at the same time and inefficient to query one pixel per cycle. The former method will make interconnection of decision circuit become very large and inconvenient to handle. The letter one has to be realized by

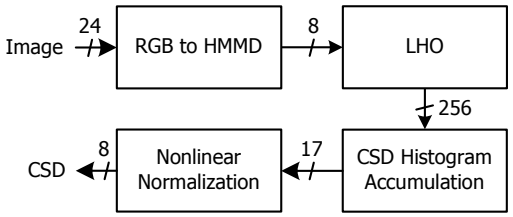

Fig. 4. CSD block diagram.

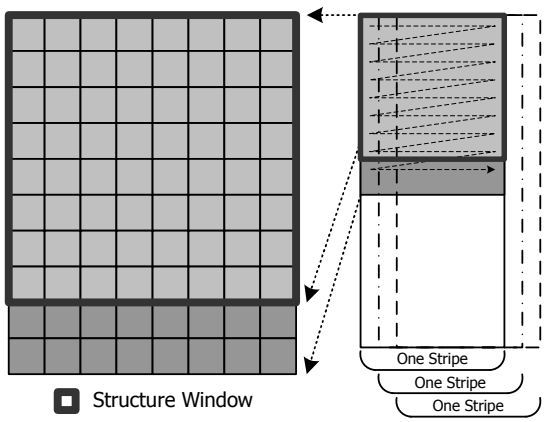

Fig. 5. Pixel scan order of a structuring window. The raster scan order in a stripe provides data reuse merit, which only eight pixels are needed to update a new SW. After finishing one stripe, index SW colors in next one until all stripes in a frame are visited.

increasing operating frequency. In order to solve the problem, we proposed a LHO architecture.

3.2. Color Appearance Recording in LHO

The main idea of LHO is to record SW histogram to indicate which colors exist in the SW. Along with updating histogram, LHO observes the value of changing color bin and saves this information into color appearance register bank. Nonzero bin means this color belongs to the window. After update, this information is sent to CSD histogram accumulation block.

Color transform in Fig. 4 sends the HMMD value of a new pixel to LHO and the value is saved in SW buffer. The pixel scan order of a frame is shown in Fig. 5. HMMD values in a SW are complete updated after discarding top row pixels from last SW and reading in eight new bottom pixels in current SW. After finishing indexing SW colors in one stripe, we start to index SW colors in next stripe.

Figure 6 shows the LHO architecture. The LHO contains a SRAM to record color histogram of a SW and a color appearance register bank to indicate which colors exist in the SW according to the values of the color bins.

When LHO receives new HMMD value, LHO updates the bin of this value in SW histogram by adding one. At the same time, the bit that stands for the existence of the value in the register bank turns into one. On the contrary, when the HMMD value of an old pixel is just replaced by that of a new pixel in SW buffer, LHO updates the bin of old value by subtracting one. Meanwhile, LHO

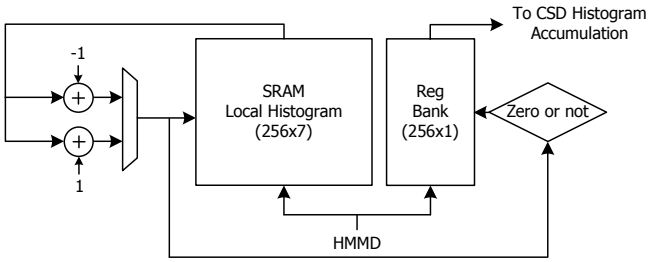

Fig. 6. Structuring window histogram updating architecture. HMMD values from color transform and SW buffer indicate which bin in the SW histogram needed to be added or subtracted by one.

checks if the bin is zero or not and updates the related bit in register bank.

Using SRAM to record histogram of SW is an area efficient method. But histogram updating cycles are directly restricted by SRAM specification. Single port SRAM provides one read or one write in a cycle. That means, when we get an address from the color which needs to update corresponding color bin, we read the bin value in one cycle, add or subtract the value by 1, and write it back to SRAM in another cycle. With an appropriate design for dual port SRAM, the throughput of updating histogram can achieve one update per cycle at the expense of double SRAM area and power. With area and power consideration, we choose single port SRAM as buffer of SW histogram. Single port SRAM takes four cycles to refresh histogram for each pixel. Two cycles are for removing accumulation from previous pixel and the others are for addition of incoming pixel. To update a SW by refreshing eight pixels will take 32 cycles.

3.3. CSD Histogram Accumulation

According to cycle analysis in section 3.2, it requires 32 cycles to complete 256-bin CSD histogram accumulation. If we store CSD histogram in single port SRAM and wish to refresh it in time, that means we have to update 16 color bins in two cycles. Similar to updating SW histogram, one cycle is for retrieving corresponding color bin, the other is for writing the updated value back. Such parallelism forces the bit-width of SRAM become 16×16 to access 16 bins at a time. Improper bit-width and number of addresses of SRAM will cause this SRAM occupies large area and waste much power. Here we divide this SRAM into four to lower the unreasonable bit-width. The bit-width is equal to four color bins. Each SRAM is 16×64 bits.

4. EXPERIMENTAL RESULT

In this section, we show that indexing and retrieval results of CSD and scalable color descriptor (SCD) [7] for comparison. The indexing and retrieval database contains 526 images in 78 categories. Those images are collected from Internet and manual categorized. The quantity of images per category varies from 2 to 28. The semantic categories include landscapes, animals, transportation, night scenes, buildings, paintings, cartoons, and so forth. Nine sample images are showed in Fig. 7.

The reason why we compared CSD with SCD is that many operation units in SCD are similar to those in CSD. This fact indicates that integrating SCD with CSD architecture is possible. Besides,

Fig. 7. Sample images of experimental database. The indexing and retrieval database contains 526 images in 78 categories.

SCD also wins the second place among other MPEG-7 color descriptors [4]. Furthermore, for extending the concepts of these descriptors to image and video coding, we replace default color spaces with YCbCr domain and the performance drops slightly.

Here we use a quantitative measure method called query-by-example (QBE) suggested by MPEG-7 [6]. QBE sorts the distances between description vector of query image and those of images contained in a database. Retrieval rank represents the rank at which certain ground-truth image is retrieved. Normalized modified retrieval rank (NMRR) eliminates the influence of number of ground-truth images. Finally, average normalized modified retrieval rank (ANMRR) is the average of NMRR of each query. The smaller ANMRR means the descriptor provides better indexing and retrieval ability. Table 2 shows the indexing and retrieval results of CSD and SCD with designated and YCbCr color spaces. ANMRR of CSD with HMMD is the lowest as our expectation. And the results of two descriptors with YCbCr are also acceptable. Because of the database characteristic and subjective manual categorization, the values in this table are smaller than the experiment results in [6]. These good results imply that we can apply the concepts of these descriptions to the field of image and video coding which chooses YCbCr as default color space.

This architecture is synthesized and the gate counts (two inputs NAND gate equivalent) are 7440 along with a 64×8 (SW buffer), a 256×7 (SW histogram), and four 16×64 (CSD histogram) SRAMs. The proposed CSD architecture for real-time applications can generate CSD description of 256×256 image at frame rate 30 fps. The computation complexity is about 4.5 GIPS to achieve real-time applications.

5. CONCLUSION

In this paper, an architecture which can generate CSD description of 256×256 image at 30 fps is proposed. LHO approach is used to record SW histogram to indicate which colors exist in a SW. Furthermore, we provide the vision of future MPEG-7 descriptor applications for not only indexing and retrieval, but also for real-time multimedia applications. First analysis of dedicated hardware architecture design for MPEG-7 CSD descriptor is also proposed. Detailed design explorations of the hardware implementation, and practical reference data of prototype is valuable for future researchers. The integration with SCD by sharing much existed resource is ongoing. In the future, descriptors with similar architecture can be integrated into this design.

Table 2. Indexing and retrieval results of CSD and SCD.

Descriptor	Color space	ANMRR
CSD	HMMD	0.00105097
CSD	YCbCr	0.00360790
SCD	HSV	0.00165656
SCD	YCbCr	0.00428604

6. REFERENCES

[1] Kenneth R. Wood Kerry Rodden, "How do people manage their digital photographs," in *Proceedings of the conference on Human factors in computing systems, ACM Press*, 2003, pp. 409–416.

[2] Eiji Kasutani and Akio Yamada, "An adaptive feature comparision method for real-time video identification," in *Proc. International Conference on Image Processing 2003*, September 2003, vol. 2, pp. II – 5–8 vol.3.

[3] Patrick Ndjiki-Nya and Oleg Novychny, "A mpeg-7-aided segmentation tool for content-based video coding," in *Proc. International Symposium on Circuits and Systems 2004*, May 2004, pp. III – 849–852.

[4] T. Ojala, M. Aittola, and E. Matinmikko, "Empirical evaluation of mpeg-7 xm color descriptors in content-based retrieval of semantic image categories," in *Proc. International Conference on Pattern Recognition 2002*, August 2002, vol. 2, pp. 1021–1024.

[5] R.J. Qian, P.J.L. Van Beek, and M.I. Sezan, "Image retrieval using blob histograms," in *Proc. International Conference on Multimedia and Expo 2000*, August 2000, vol. 1, pp. 125–128.

[6] B.S. Manjunath, Philippe Salembier, and Thomas Sikora, *Introduction to MPEG-7*, pp. 204–208, JOHN WILEY and SONS, LED, 2002.

[7] ISO/IEC JTC 1/SC 29/WG11 N4062, *Text of ISO/IEC 15938-3/FCD Information technology - Multimedia content description interface - Part 3 Visual*, pp. 47–52, March 2001.