HARDWARE ARCHITECTURE FOR GLOBAL MOTION ESTIMATION FOR MPEG-4 ADVANCED SIMPLE PROFILE

Ching-Yeh Chen, Shao-Yi Chien, Wei-Min Chao, Yu-Wen Huang, and Liang-Gee Chen

DSP/IC Design Lab
Graduate Institute of Electronics Engineering and Department of Electrical Engineering
National Taiwan University
1, Sec. 4, Roosevelt Rd., Taipei, Taiwan

ABSTRACT

Global motion estimation and compensation (GME/GMC) is a new and important coding tool in MPEG-4 Advanced Simple Profile (ASP). The coding gain of GME/GMC is 1-1.6 dB compared to that without GMC. However, because of the irregular memory access and huge memory bandwidth of GME/GMC, few hardware architectures have been proposed. In this paper, we proposed an algorithm which can save 87.27% memory bandwidth compared to that of the original algorithm. By the proposed arrangement of memory, the impact of the irregular memory access is reduced, and four neighboring pixels, which are used for the interpolation of GMC, can be accessed in one cycle. Finally, we also proposed a hardware architecture for GME in MPEG-4 ASP@L5. The total gate count is 131K, at 100MHz and the internal memory size is about 10.8Kb. It is reasonable to be integrated into MPEG-4 ASP encoders.

1. INTRODUCTION

In video coding standards, MPEG-4 Advanced Simple Profile (ASP) [1] undoubtedly is a powerful video coding standard. It can save 50% bit-rate compared to MPEG-4 Simple Profile(SP). This is because that several advanced motion compensation tools such as quarter-pixel motion estimation and compensation(QME/QMC), global motion estimation and compensation(GME/GMC), and B-frame are included in MPEG-4 ASP. In a video sequence, all motions are composed of individual object motion and the motion of background, camera motion. In MPEG-4 ASP, QME/QMC is adopted to compensate the individual object motion and GME/GMC is employed to compensate the camera motion.

Since GMC becomes a new and important coding tool, many global motion estimation algorithms have been proposed. They can be simply classified into three types: feature-point based[2], differential technique[3], and frame matching algorithms[4]. In feature-point based algorithms, some feature points are selected to represent the whole frame. And by motion vectors of feature points, global motion parameters are estimated. The Taylor series is applied to expand a criterion function and further approaches to the global motion parameters in the differential method. Frame matching matches the whole frame with the candidate global motion parameters to select the optimal one. Obviously, frame matching and differential method have large computation complexity, and feature-point based algorithm cannot get accurate global motion parameters.

Besides global motion estimation algorithms, global motion model is also an important factor to describe the camera motion. Panning, zooming, and rotation are three major camera motions. There are many global motion models to be used to describe the camera motion. MPEG-4 ASP supports four global motion models, translation, isotropic, affine, and perspective models. Among of them, the affine model is usually used, which is

$$x' = m_0 x + m_1 y + m_2 \quad (1)$$

$$y' = m_3 x + m_4 y + m_5, \quad (2)$$

where (x, y) is the position of a pixel in the current frame, (x', y') is the position of the corresponding pixel in the reference frame, and $(m_0 m_1 m_2 m_3 m_4 m_5)$ are global motion parameters. m_0, m_4 are scaling factors, and m_1, m_3 are rotation factors, and m_2, m_5 are translation factors.

GME not only has a large computation complexity, but also needs a very large memory bandwidth because of many iterations of GME. Moreover, since GME supports the deformation of scaling and rotation, the irregular memory access of GME is necessary, and it becomes a very difficult problem for hardware implementation. Up to now, there are almost no algorithms suitable for hardware implementation. In this paper, we overcame the above-mentioned problems. The global motion estimation algorithm in MPEG-4 Verification Model(VM) is adopted and analyzed. According to our analysis, we effectively reduced computation complexity and saved 87.27% memory bandwidth of the GME algorithm. We also provided a solution for irregular memory access and memory access of interpolation. Finally, a hardware architecture for GME in MPEG-4 ASP is proposed. The structure of this paper is as follows. In Sec. 2, we introduce the GME algorithm and show our analysis including computation complexity and memory bandwidth. Based on the analysis, the optimized algorithm is also proposed. Next, a hardware architecture for GME is proposed in Sec. 3. And a conclusion is given in Sec. 4.

2. GLOBAL MOTION ESTIMATION ALGORITHM IN MPEG-4 VM AND THE PROPOSED ALGORITHM

This section would introduce the global motion estimation algorithm and its analysis. Based on the analysis, we proposed a GME algorithm for hardware implementation.

The flowchart of GME in MPEG-4 is shown in Fig. 1. There are three parts, *3-tap filter*, *Initial Matching*, and *Gradient Descent* in this algorithm. It is applied on a three-level pyramid and

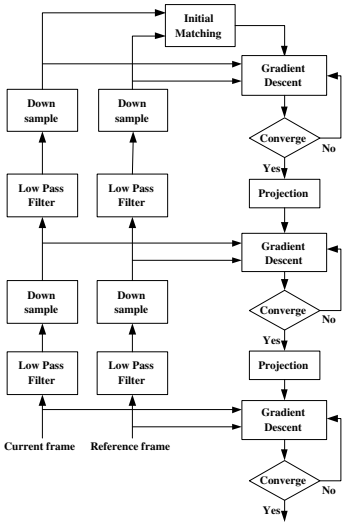

Fig. 1. The flowchart of GME algorithm in VM.

3-tap filter is used for this propose. At the top level, the predictive translation vector is estimated in the Initial Matching. After *Initial Matching*, the global motion parameters are calculated by the Levenberg-Marquardt iterative minimization in *Gradient Descent*, which is an iterative method. Therefore, the iterative process does not finish until it converges or the number of iteration is maximum at each level.

2.1. Gradient Descent

In *Gradient Descent*, the method is Levenberg-Marquardt iterative minimization. Global motion parameters are estimated by minimizing the mean square error, *E* ,

$$E = \frac{1}{N}\sum_{i\in N}|e(i)|^2, \tag{3}$$

$$e(i) = I(x,y) - I'(x',y'), \tag{4}$$

where (x, y) is the position of pixel i, $I(x, y)$ is the luminance of pixel i in the current frame, (x', y') is the corresponding position of pixel i in the reference frame, $I(x', y')$ is the luminance of the corresponding pixel, N is the total number of effective pixels, whose errors are smaller than the error threshold. The error threshold is used to excluded those pixels which result in the top 20% of the distribution of $|e(i)|$. The iterative procedure is shown as follows.

$$M_{t+1} = M_t + A^{-1}B, \tag{5}$$

$$M = (m_0 m_1 .... m_{n-2} m_{n-1})^T, \tag{6}$$

where M_t are global motion parameters at iteration t, *A* is an $n \times n$ matrix, *B* is an $n \times 1$ matrix, n is the number of global motion parameters. The coefficients of the matrix *A* and *B* are given by

$$A_{kj} = \sum_{i\in N}\frac{\partial e(i)}{\partial m_k}\frac{\partial e(i)}{\partial m_j}, \tag{7}$$

$$B_k = \sum_{i\in N} -e(i)\frac{\partial e(i)}{\partial m_k}. \tag{8}$$

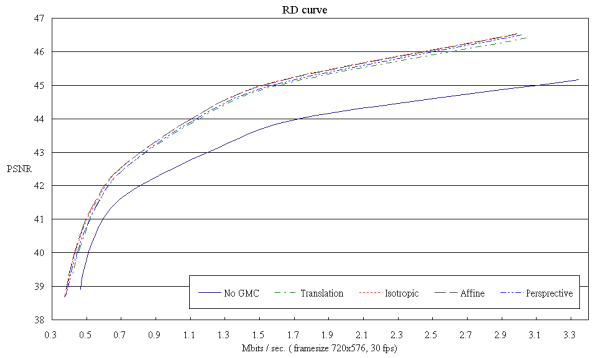

Fig. 2. The RD curve of no GMC, and GMC with translation, isotropic, affine and perspective model.

The *Gradient Descent* starts after *Initial Matching* at the top level of the pyramid, and repeats at the subsequent levels. The process would iterate until the change of each parameters is small enough or the number of iteration is above 32.

2.2. GME Performance and Analysis

In this section, the performance and memory bandwidth of GME are analyzed in MPEG-4 ASP. According to the analysis, global motion model could be selected for hardware implementation. In Fig. 2, the rate-distortion curves without and with various global motion models are shown. The test sequence is Fight, whose frame size is 720x576 with 300 frames. There is no B-frame, and searching range is -64 - 63.75 with QME/QMC. The coding gain is 1 dB at low bit rate (1Mbps, frame size 720x576, 30 fps) and the gain is 1.6 dB at high bit rate (3Mbps, frame size 720x576, 30 fps) compared to that without GMC. Because translation model cannot deal with camera scaling and rotation, the performance of translation model is lower 0.2 dB than others. Perspective model is too sensitive to have a stable performance. The isotropic and affine models have the similar performances, but affine model is more complicated than isotropic model. For this reason, the isotropic model is a better choice. Besides, global motion information also supports high level video signal processing, such as video segmentation, global motion descriptor in MPEG-7, and scene change detection. Therefore, GME should be included in the implementation of MPEG-4 ASP encoder.

Fig. 3 shows the memory bandwidth and run-time profile of GME with isotropic model in frame size 720x576, 30 fps. The memory bandwidth of GME is 908 MB/sec in the average case and 1439 MB/sec in the worst case. That is, how to effectively reduce memory bandwidth of GME is an important thing for hardware implementation. Moreover, GME with isotropic model takes 17% computation time in MPEG-4 ASP, and the run-time profile shows that the major computation complexity is *Gradient Descent*. According to the analysis, the hardware implementation of GME is necessary.

2.3. Proposed Algorithm and Simulation Result

In the above-mentioned analysis, the huge memory bandwidth of GME results from the iterations of *Gradient Descent*. The number of iteration is 32 in the worst case at each level. In the average

Function	Memory Bandwidth (MB/sec)		Run-time Profile (ms)	
	Original	Proposed	Original	Proposed
3-tap Filter & Subsample	14.82	14.82	1.4%	7.5%
Initial Mathcing	379.8	12.66	9.9%	1.5%
Gradient Descent (average case)	513.6	103.5	88.1%	87.7%
Gradient Descent (worst case)	1044.15	155.7		
Total (average / worst)	908 / 1439	131 / 183.2	830520	155700

Fig. 3. Memory bandwidth and run-time profile of original and proposed GME algorithm with isotropic model.

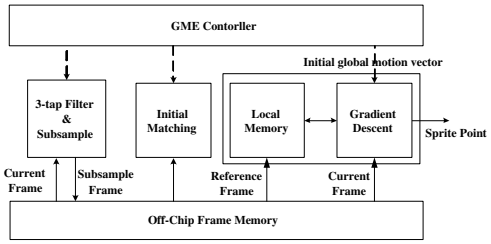

Fig. 4. Hardware architecture of GME.

case, there are also 17 iterations at each level. However, important improvements have been estimated in the first several iterations. There are few improvements in the later iterations. Therefore, the prediction, the previous global motion vector, is applied to reduce the number of iteration. And the maximum of the number of iteration is set as five. In order to avoid the problem of error propagation, the initial global motion vector is set as the identity matrix, and *Initial Matching* would be executed once in each 30 frames. The number of error histogram bins, which are used to calculate the error threshold, is also reduced from 256 to 9 because of the consideration of on-chip memory size. By these methods, the memory bandwidth is saved 87.27% in the worst case, and the PSNR only drops 0.04 dB. In the proposed GME algorithm, the memory bandwidth is 183.2 MB/sec in the worst case and 131 MB/sec in the average case. At the same time, the runtime is also speeded up five times.

3. PROPOSED HARDWARE ARCHITECTURE

In this section, based on the proposed GME algorithm, a hardware architecture for GME in MPEG-4 ASP is proposed, in Fig. 4. There are four major components, *GME controller*, *3-tap filter & Subsample*, *Initial Matching*, and *Gradient Descent with local memory*. An off-chip frame memory is required in this architecture to store the reference and current frames. GME controller controls other modules and decides which module takes action. Each module is described in detailed in the following subsections.

3.1. 3-tap filter & Subsample

3-tap filter and frame downsampling of GME algorithm are implemented in this module. 3-tap filter can be decomposed to two separated filters, vertical and horizontal filters. The data is filtered by horizontal filter and subsampled before the vertical filter. Because of the vertical filter, two delay lines are required to store temporary data after horizontal filter. In general cases, the delay line is implemented by dual ports RAM. However, because the data rate

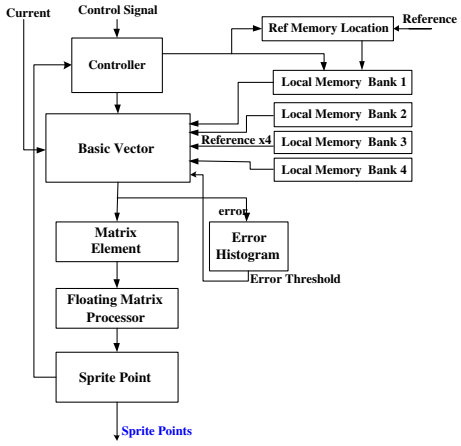

Fig. 5. Hardware architecture of Gradient Descent and Local Memory.

of vertical filter is only half of the horizontal filter, the single port RAM is enough to be used to implement the delay line. After vertical filter, the data are filtered and outputted to the off-chip RAMs.

3.2. Initial Matching

Since *Initial Matching* is executed once in each 30 frames, one processing element is sufficient. The architecture of *Initial Matching* is similar to other conventional motion estimation architectures. The absolute difference accumulator accumulates the total error of the whole frame for each motion vector. The searching range is -8 - 7 at the top level. Besides the absolute difference accumulator, there is an error histogram which is used to calculate the error threshold.

3.3. Gradient Descent

The detailed architecture of *Gradient Descent* is shown in Fig. 5. Controller generates the address of the current pixel and calculates the corresponding pixel in the reference frame according to global motion parameters at the last iteration. The corresponding pixel is usually not an integer pixel because of the scaling and rotation factors. Therefore, the luminance of the reference pixel is interpolated in Basic Vector. At the same time, the related differential data are generated. The distribution of errors between the current and reference pixels is estimated in Error Histogram, and then the error threshold is estimated. The elements of matrix *A* and *B* are estimated and accumulated in Matrix Element. Because the variance and magnitude of the matrix element are very large, a two stage accumulator is applied. In the first stage, a fixed-point accumulator is used to accumulate the matrix element. When the partial result of the first stage is larger than the accumulated threshold, a floating-point accumulator is adopted to accumulate the partial results of the first stage in the second stage. After finishing the processing of the accumulation, the inverse matrix A^{-1} and matrix multiplication $A^{-1}B$ are calculated in Floating Matrix Processor. There are an adder, a subtractor, a multiplier and a multi-cycle-shift-and-subtract divider in Floating Matrix Processor. Sprite point checks if the global motion parameters converge or not and calculates the reference points for GMC mode in MPEG-4 ASP .

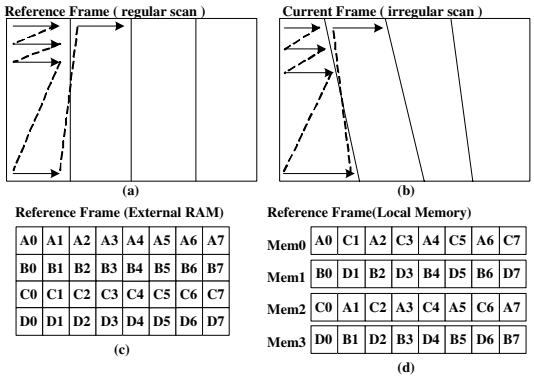

Fig. 6. The relationship between current frame, reference frame, and local memory. (a)The scan order and sections in reference frame. (b)The corresponding scan order and region of each section in current frame. (c) The arrangement of reference data in external memory. (d)The arrangement of reference data in Local Memory.

3.4. Local Memory

In GME algorithm, the irregular memory access and the interpolation of GMC are necessary. The former results in the difficulty of memory access, the latter results in huge memory bandwidth because it has to access four neighboring pixels in one cycle. In traditional cases such as motion estimation, the data in Local Memory is loaded based on the current data. Because of the irregular memory access, this method cannot be applied. Fig. 6 (a) and (b) show the proposed scan orders of reference and current frames, respectively. Because the irregular memory access is required in the reference frame, the proposed scan order of the reference frame is regular to simplify the memory access, and that of the current frame is irregular. By this way, not only the impact of irregular memory access can be reduced, but also the data in the reference frame can be as reused as possible. Fig. 6 (c) and (d) show the relationship of external and local memory. Because of the interpolation of GMC, the reference data are allocated into the Local Memory in the interleaved way. There are four dual port RAMs in Local Memory. By proposed arrangement of local memory, we can access four neighboring pixels in one cycle. The flow is as follows. The reference frame is segmented into several sections, and the data of each section are loaded in Local Memory line by line. According to the range of each section, there is a certain corresponding region in the current frame. All pixels in the corresponding region of each section are calculated at the same schedule. After all pixels in the corresponding region of the section have been calculated, the process could deal with the next section.

3.5. Hardware Simulation Results

In Fig. 7, the hardware implementation of proposed GME architecture is shown. The target specification is MPEG-4 ASP@L5, that is, the frame size of input frame is 720x576 with 30 fps. And the target operating frequency is 100MHz. The hardware is implemented and simulated with Verilog-HDL and synthesized with SYNOPSYS Design Compiler. ARTISAN 0.18*um* cell library is adopted to design hardware. The total gate count is about 131K and the internal memory size is about 10.8Kb. It is reasonable to be integrated into MPEG-4 ASP encoders. Compared with the other GME hardware architecture, SPM[5], whose target is MPEG-4@(L2, L3), the gate count is 66K and internal memory is 31Kb with AVANT! 0.35*um* cell library. Although the gate count of the proposed architecture is two times that of SPM, the memory size is much smaller, and the processing capability is much higher.

Module	Gate Count	Internal Memory
3-tap Filter & Subsample	1144	5760 b
Initial Mathcing	3718	0 b
Sprite Point	3348	0 b
Gradient Descent		
Basic vector	2215	0 b
Matrix Element	72911	0 b
Error Histogram	2370	0 b
Controller	18921	0 b
Floating Matrix Processor	26308	0 b
Local Memory	0	5120 b
Total gate count	130935	10880 b
SPM	66116	30880 b

Fig. 7. The results of GME hardware implementation in MPEG-4 ASP@L5 (frame size 720x576, 30frames/sec.) and comparison.

4. CONCLUSION

In MPEG-4 ASP, global motion estimation and compensation is an important coding tool. The coding gain of GME/GMC is 1 dB at low bit rate (1Mbps, frame size 720x576, 30 fps), and the gain is 1.6 dB at high bit rate (3Mbps, frame size 720x576, 30 fps) compared to that without GMC. However, there are few hardware architectures. In this paper, we overcame difficult problems of GME/GMC for hardware implementation, irregular memory access and huge memory bandwidth. The memory bandwidth is saved 87.27% by skipping the redundant computation. And an interleaved arrangement of local memory is adopted to reduce the memory access of interpolation. A hardware architecture for GME in MPEG-4 ASP@L5 is also proposed. The gate count is 131K and the internal memory size is about 10.8Kb. It is reasonable to be integrated into MPEG-4 ASP encoders.

5. REFERENCES

[1] MPEG Video Group, *AMENDMENT 4: Streaming Video Profile*, ISO/IEC JTC 1/SC 29/WG11 N3904, 2001.

[2] A. Smolic, T. Sikora, and J.-R. Ohm, "Long-term global motion estimation and its application for sprite coding, content description, and segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 8, pp. 1227–1242, Dec. 1999.

[3] MPEG Video Group, *The MPEG-4 Video Standard Verification Model version 18.0*, ISO/IEC JTC 1/SC 29/WG11 N3908, 2001.

[4] D. Adolph and R. Buschmann, "1.15Mbit/s coding of video signals including global motion compensation," *Signal Processing: Image Communication*, vol. 3, no. 2-3, pp. 259–274, June 1991.

[5] S.Y. Chien, C.Y. Chen, W.M. Chao, Y.W. Huang, and L.G. Chen, "Analysis and hardware architecture for global motion estimation in mpeg-4 advanced simple profile," in *Proc. of International Symposium on Circuits and Systems 2003*, May 2003, vol. 2, pp. 720–723.