# Method and architecture design for motion compensated frame interpolation in high-definition video processing

**Conference Paper** · June 2009

DOI: 10.1109/ISCAS.2009.5118085 · Source: IEEE Xplore



**2 authors**, including:

Truong XUAN Nguyen
Deakin University
**234** PUBLICATIONS **3,815** CITATIONS

SEE PROFILE

# Method and Architecture Design for Motion Compensated Frame Interpolation in High-Definition Video Processing

Yen-Lin Lee and Truong Nguyen
ECE Dept., UCSD, La Jolla, CA 92093-0407
Email: yel004@ucsd.edu, nguyent@ece.ucsd.edu
http://videoprocessing.ucsd.edu

***Abstract*— In this paper, a novel, fast, and low complexity method is proposed for motion compensated frame interpolation (MCFI). Unlike previous works involving high complexity and complicated time-consuming iterations, the proposed method adopts one-pass and low-complexity approach without any repeated iteration and is capable of dealing with high definition (HD) video processing. The proposed method employs a unique true motion engine that explores at most nine motion candidates with different motion directions and then determines one true motion by referring to neighboring spatial information as well as temporal information. Besides, the proposed method introduces an adaptive overlapped block matching algorithm to enhance the searching accuracy and visual quality. An architecture design is also proposed, which employs a modified multi-level successive eliminate algorithm (MSEA) and has capability to reduce the heavy computation of full search. Experimental results show that the proposed algorithm provides better video quality than conventional methods and shows satisfying performance for HD1080p 30fps at 180 MHz or HD720p 30fps at 83 MHz.**

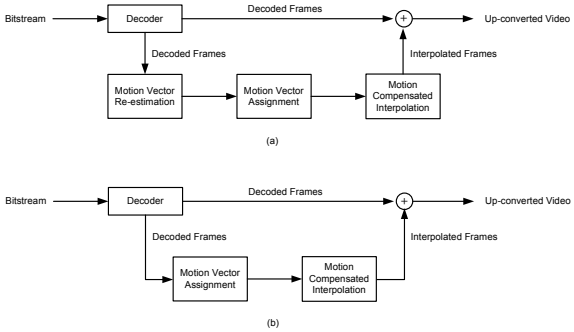


Fig. 1. (a) Motion re-estimation method (b) Motion vector post-processing

## I. Introduction

Digital displaying devices are widely prevalent in recent years, such as liquid crystal display (LCD) and plasma televisions because consumer electronics, sports, and movies are the prime factors in the popularity of large screen display. However, motion blur and jerkiness appear as objects move rapidly or color dramatically changes on a wide range of LCD devices because of displaying devices' slow response time and sample-and-hold drive nature. Frame rate up conversion (FRUC) is a well-known method that is used to eliminate or reduce devices' physical disadvantages. Hence, motion compensated frame interpolation (MCFI) has recently been widely studied to enhance the quality of reconstructed video, which explores block motions of the interpolated frames. For this reason, many researchers started to work on how to accurately estimate true motions, which considers spatial and temporal correlation and searches with block matching algorithm (BMA) [1]. Contemporary proposals and methods of MCFI are classified into two categories: a motion re-estimation method and a motion vector post-processing method. Fig. 1 shows basic block diagrams of these two approaches.

In the past ten years, numerous methods are proposed to find the true motions. However, these methods usually involve complex computation, complicated time-consuming iterations, and are difficult to work on a real-time high-definition (HD) quality video. Considering these problems, the proposed method and architecture adopt one-pass and low-complexity design without any repeated iteration to deal with HD or non-HD video processing. Besides, this approach has no restriction in video compressed format due to motion re-estimation search. Techniques of the proposed method are composed of bilateral estimation, multi-directional enlarge matching algorithm (MDEMA), one-pass motion vector selection with multi-grids classification, temporal/spatial object information and localized global motion vector, motion compensated frame interpolation, and deblocking filter with block difference. For real-time HD requirements, the proposed architecture embeds a true motion engine that employs a modified multi-level successive eliminate algorithm (MSEA) [2], a method of probable sum of absolute difference (PSAD), true motion selecting, and a refinement of motion vector processing. MSEA is a fast full-search block matching algorithm (FFSBMA) [3] following a successive elimination algorithm (SEA) [4], which have come into notice for their capabilities to reduce the heavy computation of full-search BMA (FSBMA) [5] and to maintain the similar quality. Although MSEA has originally been developed for video compression, the proposed architecture introduces this technique to enhance the accuracy of true motion search and reach a balance between complexity and performance. These techniques will be discussed in the following sections

978-1-4244-3828-0/09/$25.00 ©2009 IEEE

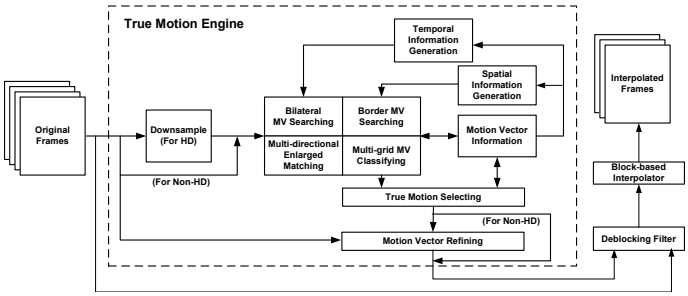


Fig. 2. Processing flow of the proposed method

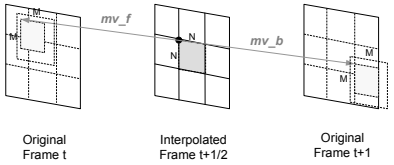


Fig. 3. The proposed bilateral search with MDEMA

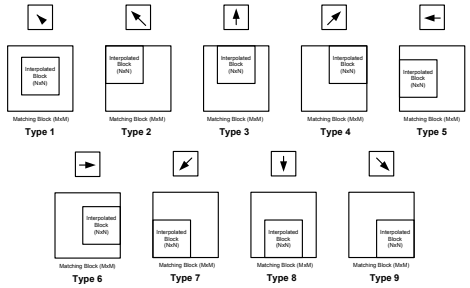


Fig. 4. Multi-directional enlarge matching algorithm (MDEMA)

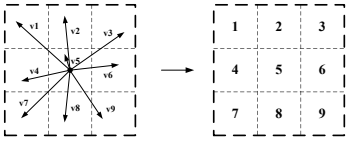


Fig. 5. Multi-grids classification in a searching window

## II. The Proposed Method

The proposed scheme adopts a motion-compensated approach to insert one or several interpolated frames between any two contiguous original frames. Fig. 2 shows the processing flow of the proposed method. The entire processing procedure is composed of three major parts: true motion engine, block-based interpolator, and deblocking filter. True motion engine takes charge of motion estimation and true motion decision, and it also outputs true motion vectors to block-based interpolator. Block-based interpolator obtains motion information and performs a motion-compensated procedure with weighted values. At last, a deblocking filter smooths the blocking artifact where a neighboring interpolated block with larger sum of absolute difference (SAD) shows up. Key techniques are explained in following sub-sections.

### A. Bilateral Search

For acquiring more accurate true motions, we adopt motion re-estimation with bilateral search with original uncompressed frames from a decoder rather than process and correct motion vector field extracted from a compressed bitstream. Naturally, a method without motion re-estimation should have lower complexity, but it is arduous to look for true motions when motion vector field is inaccurate and irregular. Besides, repeated iterations may increase difficulty on high-definition video with high frame rate. Fig. 3 shows the proposed method using a bilateral search. The proposed bi-directional search directly seeks two similar blocks from a zero distance of the interpolated block. $mv\_f$ and $mv\_b$ represent two different directional motion vectors, a forward motion and a backward motion.

### B. Multi-directional Enlarged Matching Algorithm

Compared to same block size as the interpolated block, larger matching block size could improve the accuracy of searching true motion because larger part of a particular object is considered. Based on our experiments, the results suggest that using smaller matching size is worse and easily obtain wrong motion with better pixel difference instead of a true motion, especially for blocks with less texture. In accordance with different motion direction, nine types of enlarged matching directions are defined and shown in Fig. 4. Small squares in the figure mean N×N block size, and this area will be used for interpolation as we find the most suitable true motion vector by matching with large size M×M. The reason for using different enlarged directions is to keep the completeness of the front edge for the moving object. From the observation, viewers are more sensitive to be aware of the artifacts and fractures of the moving object's front edge.

### C. One-pass Process with Multi-grids Classification

The proposed approach adopts one-pass processing method, which means no iteration is needed to perform when determining true motion vector field. This method gains faster processing time and less memory access. It only counts on information of motion vectors from previous processed blocks, which scheme is similar with advanced techniques in video compression standards, such as H.264 and VC-1, and these data would assist to make the right decision for the current interpolated block. The proposed method also introduces a multi-grids classification to simplify the process and reduce the storage size for motion vector field. It divides a searching window into multiple areas shown in Fig. 5, and each area has one motion candidate.

### D. Temporal Object Information

Although one-pass processing would not provide complete information, such as motion types from the right or below neighboring blocks relative to the current interpolated frame, it can acquire some temporal information from the previous processing result to realize if an object moves. Using temporal information is an efficient motion compensated method for objects with constant speed, especially in panning scenes. When the camera pans slowly across a scene, most interpolated

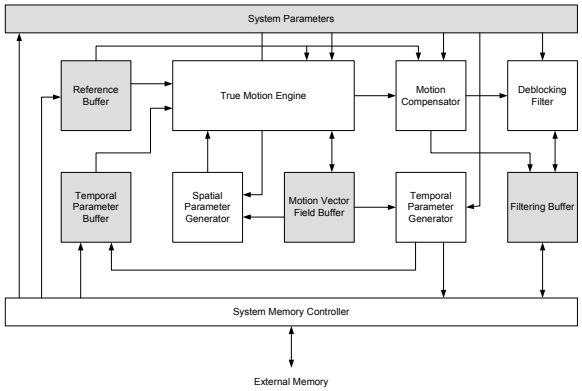


Fig. 6. System block diagram of the proposed architecture

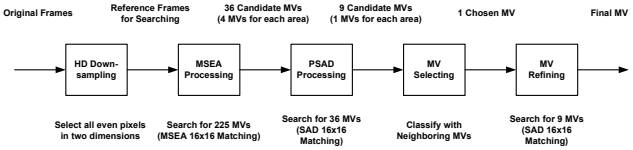


Fig. 7. Processing flow of the proposed true motion engine

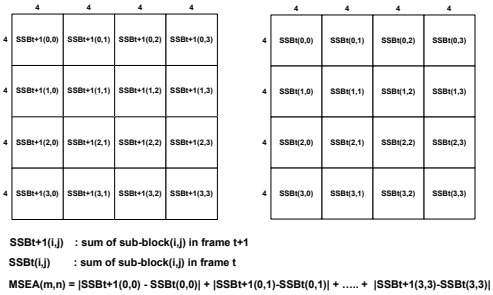


Fig. 8. Multi-level successive elimination algorithm (MSEA)

blocks would move consistently except real moving objects. However, not all information of previous interpolated blocks is considered for global motion in our proposed method. In other words, only interpolated blocks with a larger pixel variance are considered, which means that there are more textures. By means of this method, it would remove those inaccurate motion vectors to take into account.

## III. The Proposed Architecture

In order to lower the computation complexity, the proposed architecture introduces a fast full search method, multi-level successive elimination algorithm (MSEA) [2]. The proposed system architecture is shown in Fig. 6. In this figure, there are two kinds of elements: the blocks in white color are functional elements, including motion estimator, motion compensator, spatial parameter generator, temporal parameter generator, deblocking filter, and system memory controller; the blocks in gray color are storage elements, including system parameters, reference buffer, temporal parameter buffer, motion vector field buffer, and filtering buffer.

### A. True Motion Engine

Five steps of processing would be performed in the engine. In our implementation, it sets nine-grids classification, searching window from -7 to +7 for a down-sampled frame, 8×8 block size, 16×16 enlarged block size, and four candidates for each MSEA sub-block. The processing flow of the proposed true motion engine is shown in Fig. 7. HD down-sampling is the first step, which drops all odd pixels in two dimensions if input video has high resolution. There is no low-pass filter applied here when downsampling. It is because there are several searching stages in the proposed architecture and it would apply to a refinement at last. The second step is MSEA, which is a compromise between full search and fast search. Based on our experiments, fast search methods, such as three-step search or diamond search, are all unsuitable for the MCFI due to serious local trap problem. An MSEA example is shown in Fig. 8. The third step of true motion engine is to search by probable sum of absolute difference (PSAD). Since MSEA engine decides four highly possible candidates for each directional area, PSAD engine only needs to check SAD values for these four vectors instead of a full search with SAD. Multi-directional enlarged matching method is used for MSEA and PSAD searching. The fourth step is true motion selection from these nine motion candidates by pre-defines conditions, and neighboring and temporal information. At the last stage, the proposed architecture would perform a refining procedure by probable refining engine (PREF) if input video sequence has high resolution and is down-sampled before searching.

### B. Implementation

The specification of the proposed architecture has capability to double the frame rate for an HDTV1080p (1920×1080) 30fps video at 180MHz or a 720p (1280×720) 30fps video at 83MHz. To verify the accuracy and efficiency between the proposed method and architecture, the architecture is designed in VHDL and implemented with TSMC 90-nm technology. The implemented architecture operates with our VHDL- and Matlab-model, and the result has been verified with our prototype by a pure Matlab-model. Table I shows logic gate count including all functional blocks of the proposed architecture synthesized with Cadence RTL compiler at 180 MHz, and the total logic gate count without on-chip memory is about 27.997K. TABLE II summarizes the distribution of the on-chip memory resources, and the proposed architecture needs 22.67 K byte on-chip memory when operating.

## IV. Performance and Experimental Result

Based on our implementation, we simulate this architecture with real video sequences so as to calculate the cycle count and verify the processing time that meets timing requirement for HD1080p 30fps. Since the processing cycle count is fixed on the proposed design, the cycles in TABLE III can be analyzed for any sequence with the same resolution.

Here, we also compare the performance by two high-resolution and 60fps test sequences with three other methods: bilinear interpolation, 3-D recursive algorithm (3DRS) [1], and Phase-Plane Correlation algorithm (PPC) [6] with the same searching block size 16×16. This test drops all even

TABLE I

FRONT-END HARDWARE COST OF THE PROPOSED TRUE MOTION ENGINE

| Modules | MSEA | PSAD | MV Sel | PREF | Total |
|---|---|---|---|---|---|
| Gate Counts | 19,559 | 4,055 | 2,125 | 1,466 | 27,997 |

TABLE II

ON-CHIP MEMORY BUFFERS FOR HD1080P VIDEO

| Name | Memory Size (KByte) |
|---|---|
| Reference Buffer | 14.256 |
| Temporal Parameter Buffer | 0.026 |
| Motion Vector Field Buffer | 7.684 |
| Filtering Buffer | 0.704 |
| Total | 22.670 |

TABLE III

PROCESSING TIME OF THE PROPOSED ARCHITECTURE

| Type | Cycles | Frequency(MHz) | Time (ms) |
|---|---|---|---|
| 16×16 Block | 683 | 180/83 | - |
| 1 Frames@1080p | 5,581,795 | 180 | 30.69 |
| 30 Frames@1080p | 167,453,850 | 180 | 921.99 |
| 1 Frames@720p | 2,462,755 | 83 | 29.55 |
| 30 Frames@720p | 73,882,650 | 83 | 886.59 |

frames and generates these missing frames by four different methods, and these interpolated frames are compared with the original frames so as to calculate PSNR results. The results are organized in TABLE IV and also shown in Fig. 9 and Fig. 10. Experimental results show that the proposed algorithm provides better video quality than conventional methods. (Also check other visual tests and comparisons from our website: http://videoprocessing.ucsd.edu/∼yenlinlee/iscas2009/)

TABLE IV

PERFORMANCE COMPARISON - PSNR

| | Bilinear | 3DRS | PPC | Proposed |
|---|---|---|---|---|
| CREW 720p | 28.649 dB | 28.769 dB | 28.400 dB | 28.967 dB |
| CITY 720p | 28.954 dB | 29.441 dB | 29.551 dB | 32.420 dB |

## V. CONCLUSION

In this paper, a novel, fast, and low complexity method with a well-designed architecture is proposed for motion compensated frame interpolation. Our method employs a unique true motion engine with an adaptive overlapped block matching algorithm, multi-directional enlarged matching algorithm, and one-pass process. The proposed architecture employs a modified multi-level successive eliminate algorithm and has capabilities to reduce the heavy computation. Experimental results show that the proposed algorithm provides better video quality than conventional methods and shows satisfying performance to deal with HD1080p 30fps at 180 MHz or HD720p 30fps at 83 MHz.

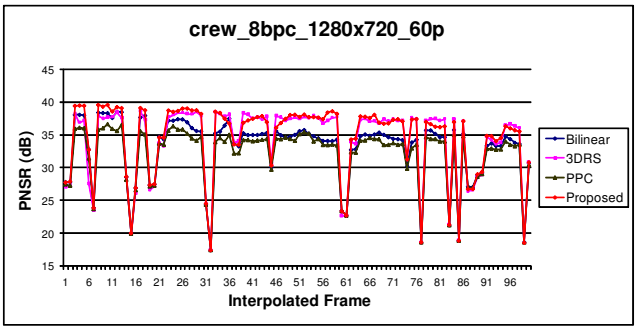


Fig. 9. PSNR comparison 1 - CREW 1280×720 60fps

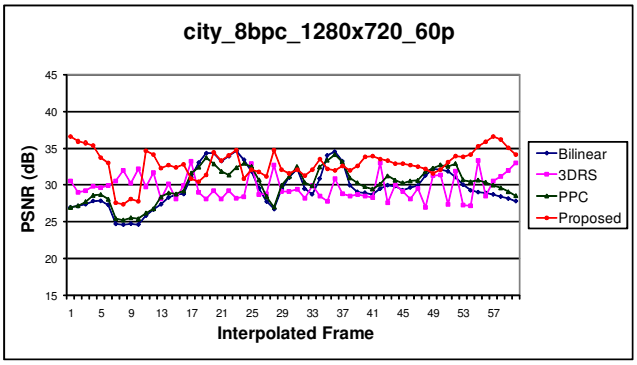


Fig. 10. PSNR comparison 2 - CITY 1280×720 60fps

## REFERENCES


[1] G. de Haan, P.W.A.C. Biezen, H. Huijgen, and O. A. Ojo, "True-motion estimation with 3-D recursive search block matching," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol.3, No.5, pp.368-379, 388, Oct. 1993.

[2] X. Q. Gao, C. J. Duanmu, and C. R. Zou, "A multilevel successive elimination algorithm for block matching motion estimation," *IEEE Trans. on Image Processing*, Vol.9, No.3, pp.501-504, March 2000.

[3] Y. Noguchi, J. Furukawa, and H. Kiya, "A fast full search block matching algorithm for MPEG-4 video," *in Proc. Int. Conf Image Processing (ICIP)*, Vol.1, pp.61-65, 1999.

[4] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Trans. on Image Processing*, Vol.4, No.1, pp.105-107, Jan. 1995.

[5] L. De Vos and M. Stegherr, "Parameterizable VLSI architecture for the full-block matching algorithm," *IEEE Trans. on Circuit and Systems*, Vol.36, pp.1309-1316, Oct. 1989.

[6] M. Biswas and T. Nguyen, "A novel motion estimation algorithm using phase plane correlation for frame rate conversion," *in Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, Vol.1, No.3-6, pp.492-496, 2002.