Pratheepa.A, Anita Titus / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 3, May-Jun 2012, pp.1272-1277

A High Sensitive and Fast Motion Estimation for One Bit Transformation Using SSD

Pratheepa.A[1] and Anita Titus[2]
[1] ME-VLSI Design
[2] Dept of ECE
Easwari Engineering College, Chennai-89

Abstract-In this paper, architectures implementing fixed block size (FBS) and variable block size (VBS) for One bit transform (1-BT) using SSD (Sum of Squared Differences) to find motion between frames have been proposed. In general Motion estimation (ME) requires a huge amount of computation, and hence consumes the largest amount of power. The technique used to reduce the usage of power is 1-BT based ME algorithms. They have low computational complexity. In this less On-chip memory used than the previous 1BT based ME hardware by using a data reuse scheme and memory organization. Here the Sum of Absolute Difference (SAD) matching method is followed with the application of Diamond Search (DS). The 1-BT based ME is usually performed by applying full search (FS) algorithm. The proposed architecture reduce the power by performing the application of toroidal path in the pipelining approach on processing element with Sum Of Squared differences technique(SSD). The simulation results reveal that the application of DS on 1-BT based ME using SSD technique can significantly reduce the computational complexity and the power which is observed in SAD based 1-BT ME.

Index terms-One Bit Transformation, Motion estimation, Sum of Absolute Differences, Sum of Squared Differences, diamond search algorithm, fixed block size, variable block size.

I. INTRODUCTION

Motion estimation is the process of determining motion vectors that describe the transformation from one 2D image to another, usually from adjacent frames in a video sequence. It is an ill-posed problem as the motion is in three dimensions. The motion vectors may relate to the whole image (global motion estimation) or specific parts (local motion estimation), such as rectangular blocks, arbitrary shaped patches or even per pixel in the images. The motion vectors may represented by other models that can approximate the motion of a real video camera, such as rotation and translation in all three dimensions and zoom.

In general, motion of objects is specified in 3-D real world. In motion estimation, the 'projected motion' has been concerned with the 3-D objects onto the 2-D plane of an imaging sensor. It means the estimation of the displacement (or velocity) of image structures from one frame to another in a time sequence of 2-D images, by motion estimation. This projected motion is referred to as 'apparent motion', '2-D image motion', or 'optical flow'. A possible first approach to **motion estimation** is to assume that the motion is locally translational, and that the image intensity is invariant to motion, i.e., it is conserved along motion trajectories. Smoothness (slow spatial variation of the displacement vector field along the motion) can be imposed either implicitly, when formulating the motion estimation algorithm, or explicitly during implementation.

Motion estimation (ME), which is the most essential part of any video coding technique, exploits and tries to minimize the temporal redundancy present between successive frames. ME, which is computationally intensive, involve about 80% of the total computational power of the encoder. So there is a **block matching** required. There are two mainstream techniques of motion estimation: pixel (pel)-recursive algorithm (PRA) and block-matching algorithm (BMA). PRAs are iterative refining of motion estimation for individual pels by gradient methods. BMAs assume that all the pels within a block has the same motion activity.

BMAs estimate motion on the basis of rectangular blocks and produce one motion vector for each block. PRAs involve more computational complexity and less regularity, so they are difficult to realize in hardware. In general, BMAs are more suitable for a simple hardware realization because of their regularity and simplicity. Here a BMA method has been taken place for the proposed technique.

Motion Estimation Algorithms vary with respect to the information, as well as the method of computation they utilize to obtain the estimate. The idea of using a block of pixels and assuming a common displacement for them in the matching process corresponds to a local smoothness (coherence). In this paper an efficient method for block matching motion estimation is considered for interframe motion compensated prediction of video signals. In contrast to other block matching methods, this approach takes into consideration the behavior of individual pels in the search to find the best match. This is achieved by classifying each pel in the block into following categories: matching pel and

mismatching pel [3].The method existing is sum of absolute difference (SAD) technique [1]. **Block matching motion estimation** (BMME) technique is one of the efficient methods that remove the temporal redundancy present between the successive frames. In this method video frame is partitioned into 16×16 frames known as macro blocks.

II ONE BIT STRATEGY WITH SAD(EXISTING)

Now a transform has been constructed that maps a frame of multivalued pixels to a frame of binary value pixels. That transform is defined with respect to a convolution kernel K and the transform is denoted by Q_k. Let F denote a frame and let ^ F denote the filtered version of F obtained by applying the convolution kernel K to F. Let B = Q_k (F) be the frame obtained by applying QK to F. The pixels of B are given by,

$$B(i,j) = \begin{cases} 1, & if\ F(i,j) \geq \hat{F}(i,j) \\ 0, & otherwise \end{cases}$$

Here, *i* and *j* are the spatial coordinates of the pixel. The foregoing process by which an original frame with 8 bits/pixel representation is converted into a binary frame with 1 bit/pixel representation is known as one-bit transformation.

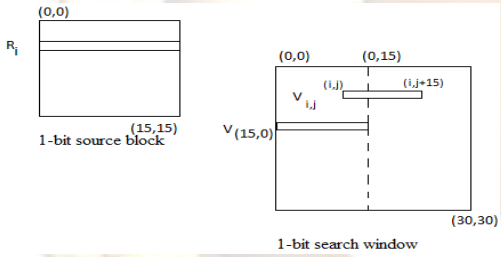


Fig.1. Pixel co-ordinates for a 16×16 source block and a one bit search window.

And for sum of absolute differences technique, the equation is

$$SAD = \sum_{(i,j)\in W} | I_1(i,j) - I_2(x+i, y+j) |$$

The similarity measure which is calculated by applying the simplest SAD method, subtracting pixels within a square neighborhood between the reference image I_1 and the target image I_2 followed by the aggregation of absolute differences within the square window, and optimized. If the left and right images are exactly matched then the resultant will be zero otherwise the window size will be increased.

else SAD_{ij} = 1, then threshold will be

$$V_{th} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} SAD_{ij}$$

A key to accurate motion estimation is the observation on the edges of the images. A simple way to extract the edges is to carry out a high-pass thresholding, that is, compare the frame pixel by pixel to a high-pass filtered version of the frame, and threshold the pixels to zero or one, depending on the outcome of the comparison [4]. The convolution kernel that we propose is motivated by this consideration, as well as the need to minimize the number of arithmetic operations.

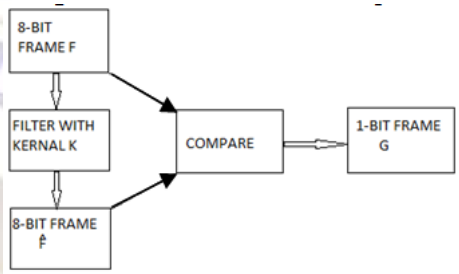


Fig.2. Operation for thresholding 8-bit frames to 1-bit frames

The operations for the one-bit transform are shown in Fig.- 2.Note that there is no global threshold for all pixels in a frame. For video coding, the one-bit motion estimation strategy consists of the following steps: 1) apply the one-bit transform Q to both the current frame and the reference frame; 2) use any motion-vector search strategy in combination with the metric defined in (fig 4).

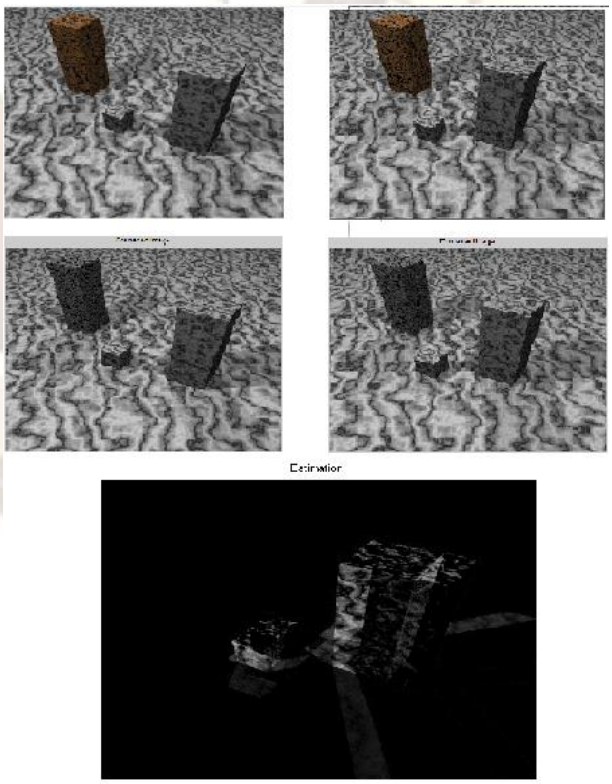


Fig.3. Flow of operations of converting to 1-b based block motion estimation.

After this operation, the number of non-matching points (*NNMP*) at any point (*m, n*) for a MB of size *N*×*N* is found as:

$$\text{NNMP}(m,n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} B^{t}(i,j) \oplus B^{t-1}(i+m, j+n)$$

Where $-S \leq (m,n) \leq S$

Here,'s' is the maximum search range and $\oplus$ denotes XOR operation. Also, B^t and B^{t-1} represent the current and the reference 1-BT frames respectively.

III THE DIAMOND SEARCH and MOTION VECTOR PREDICTION

The proposed DS algorithm employs basically for a search pattern for easy prediction of motion vector present which is originally deviated from the frames, and it is having two search patterns, the first pattern, called large diamond search pattern (LDSP), and comprises nine checking points from which eight points surround the center one to compose a diamond shape (♦). The second pattern consisting of five checking points forms a smaller diamond shape, called small diamond search pattern (SDSP). In the searching procedure of the DS algorithm, LDSP is repeatedly used until the step in which the minimum block distortion (MBD) occurs at the center point. The search pattern is then switched from LDSP to SDSP as reaching to the final search stage.

Among the five checking points in SDSP, the position yielding the MBD provides the motion vector of the best matching block. After the search pattern analyses the pattern has been given to for FBS the frame size will not be change. For example if the frame size is 4x4 means, it will be maintained the same till the last frame check. For the next search only the frame size will be changed to next criteria. And for the VBS the frame size will be changed even during the run time also. The frame size can be 2x2, 4x4, 8x8, and 16x16.

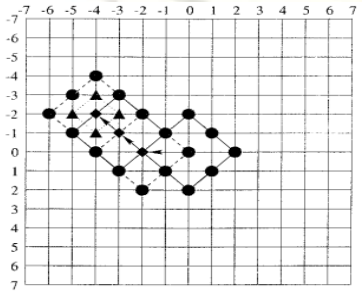


Fig .4. Example of motion of vector

Generally, a motion estimation algorithm includes two important components for the MV search: search center and search pattern. In previous reported ME algorithms, search center is generally predicted from spatial and temporal neighbors motion vector (MV).

IV. SUM OF SQUARED DIFFERENCES

The tracking method presented as the enhancement of the base work in this paper. It is based on minimizing the sum-of-squared differences (SSD) between a selected set of pixels obtained from a previously stored image of the tracked patch (image template) and the current image of it.

$$\text{SSD} = \sum_{(i,j)\in w} (I_1(i,j) - I_2(x+i, y+j))^2$$

In Sum of Squared Differences (SSD), the differences are squared and aggregated within a square window and later optimized by search strategy. This measure has a lower computational complexity compared to SAD algorithm as it involves less numerous multiplication operations.

Steps for SSD method:

Step 1: For the first P(previously coded)frame, FS is executed. The SSD Value (SSD_{ij}) is stored in a data table.

Step 2: According to the existing SSD value table, a threshold value V_{th} is computed and SSD table is updated:
If SSDij < V_{th} then $SSD_{ij} = 0$

Step 3: For the following P frames, motion estimation for the macroblock in current P frame will refer to the value in corresponding position of the obtained data in the table.

If SSDij = 0, a 16x16 matching search is performed.

Else, the macroblock will be split into four 8x8 blocks for further matching.

Step 4: SSD value results from the motion estimation of current P frame are stored. Go to step 2 if there is remaining P frames or terminate the loop otherwise.

V. PIPELINING ON TOROIDAL PATH AND PROCESSING ELEMENT

A. PIPELINING WITH TOROIDAL STRUCTURE

Pipelined computer architecture has received considerable attention since when the need for faster and more cost-effective systems became critical. The merit of pipelining is that it can help to match the speeds of various subsystems without duplicating the cost of the entire system involved. In this paper in pipelining a FIFO (first in first out) method is applied. In that the architecture has been changed with the application of toroidal structure. A toroidal approach to an orbit path filter was presented that provides more versatility than a simple distance function. This method uses the primary orbit to define a focus-centered elliptical ring torus

with separate, user-defined, in-plane and out-of-plane bounds. Through toroidal path a data storage, retrieve, and to track the value easily from one PE to another PE with shortest minimum path calculation and this would keep the throughput at high rates also.

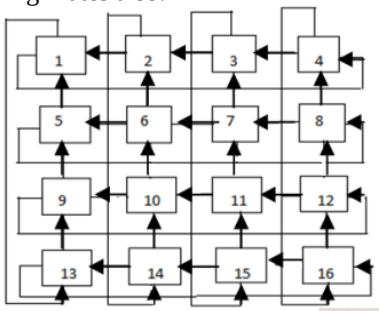


Fig.5. Toroidal Structure

B. PROCESSING ELEMENT

The architecture of the PE for both VBS and FBS is shown in Fig.7&8. The PE possesses two input ports namely, C and S for reading two 16-bit vectors from the CB and the SW respectively. There are two 8- bit XOR arrays in the PE. One of the XOR arrays operates on the eight most significant bits of the 16-bit vectors C and S, and the other operates on the remaining eight least significant bits. The number of 1^s as a result of the XOR operation is obtained by using two look-up tables (LUTs) with 2^8 entries. The outputs of the LUTs are then applied to a 4-bit adder. The output of the adder is then applied at the input of the accumulator. The output of the accumulator provides the value of the NNMP for a particular location after 16 clock cycles for a MB of size 16×16.

VI ARCHITECTURES

The proposed architecture is,

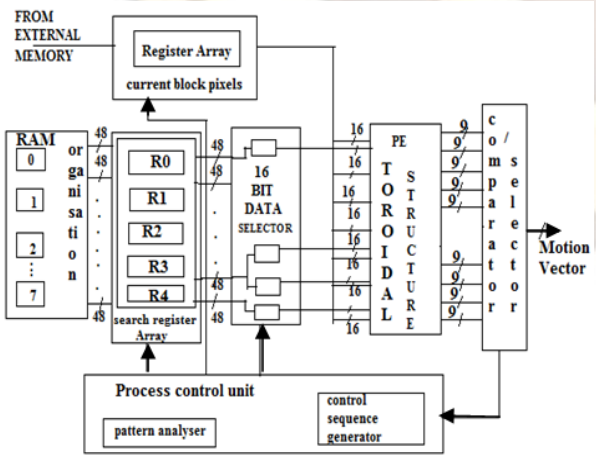


Fig .6.proposed architecture

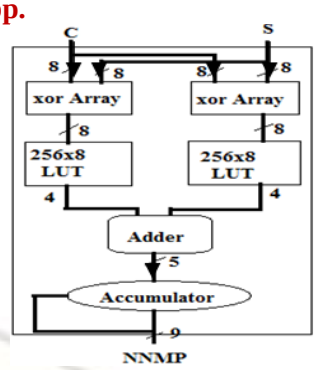


Fig.7. PE Architecture for one bit FBS ME

In recent years, the variable block-size (VBS) motion estimation has been widely employed to improve the performance of the block matching algorithm. In this paper a comparison for both FBS and VBS is shown on power.

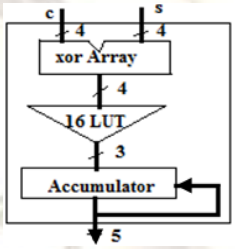


Fig.8. PE Architecture for one bit VBS ME

In the past algorithms, many fast BMAs were proposed for reducing the computational complexity for the FS, such as three-step search (TSS), four-step search (FSS), orthogonal search, and hexagon based search (HEXBS) algorithms, etc. Compared to these, diamond search algorithm on VBS and FBS reducing the computational complexity, usage of power, PSNR value compared to other algorithms. And the diamond search is applied on proposed SSD technique with toroidal pipelined architecture has reduced the power further and the speed of process is also increased.

VII IMPLEMENTATION RESULTS

A VBS motion estimation algorithm based on 16x16, 8x8, 8x4 and 4x8, for FBS 4x4,4x8 has been proposed which can efficiently reduce the computational cost while achieving similar or better visual quality as compared with other 1-BT based ME architectures, the proposed architectures involve lowest latency. The PSNR value is 30.81db, thus signal to noise ratio is increased compared to previous papers. The proposed fast binary FBS ME architecture consumes 932 slices (1533 LUTs). The on-chip memory of the proposed architecture is 2296 bits for storing the SW for a single MB.

PSNR COMPARISON:

SCHEME	PSNR
MAD	23.79db
2-BT	23.43db
MF-1-BT	23.32db
1-BT SAD	25.29db
1-BT SSD	30.81db

VIII CONCLUSION

The proposed DS based binary ME architecture for FBS and VBS are described in VHDL. The proposed architectures are implemented on ALTERA. In this paper, low power ME architectures have been developed for implementing DS on 1-BT frames with FBS and VBS support for SSD. As compared with other 1-BT based ME architectures, the proposed architectures involves the lowest latency. The clock frequency, I/O pins used, gates required for the proposed ME architectures therefore can be effectively reduced for low power designs.

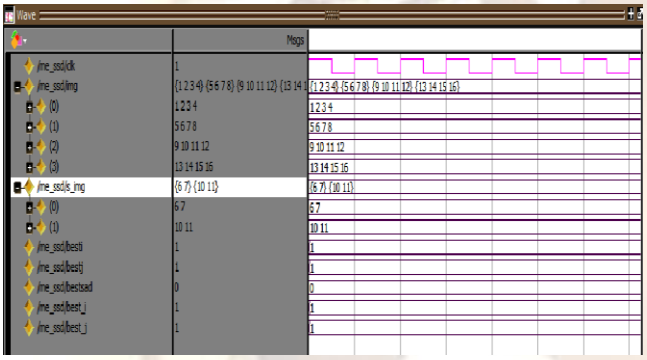


Fig.8.Example wave for ssd calculation

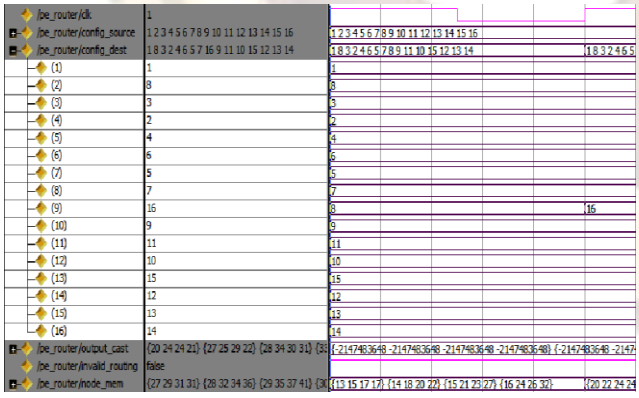


Fig.9.False invalid routing indication wave for PE

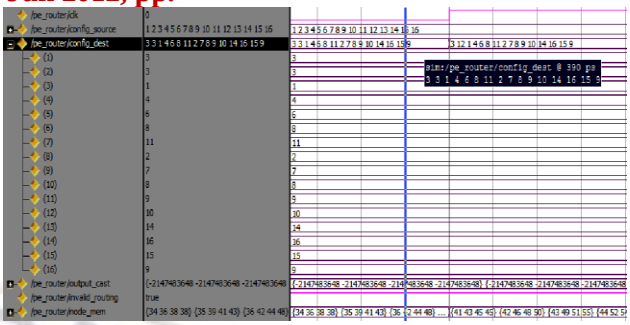
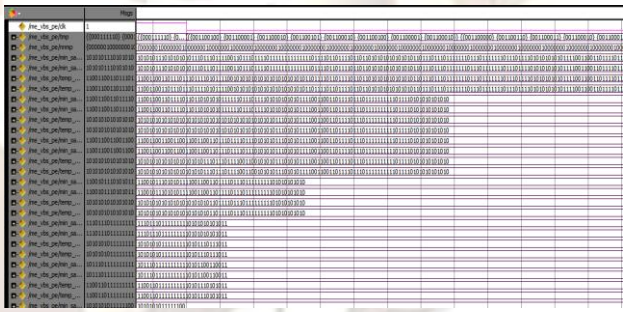


Fig.10.True invalid routing indication wave for PE

Fig.11.Output of PE for FBS (NNMP deviated data)



REFERENCES

[1] Sumit K Chatterjee and Indrajit Chakrabarti, "Low Power VLSI Architectures for One Bit Transformation Based Fast Motion Estimation", IEEE Trans. Consumer Electronics., vol. 56, no. 4, pp. 2652-2660, November 2010.

[2] C. H. Hsieh and T. P. Lin, "VLSI architecture for block-matching motion estimation algorithm," IEEE Trans. Circuits Syst. Video Technol., vol. 2, no. 2, pp. 169-175, June 1992.

[3] H. Gharavi and M. Mills, "Block matching motion estimation algorithms new results," IEEE Trans. on

Circuits and Systems, Vol. 37, no. 5, pp. 649-665, May 1990.

[4] Surin Kittitornkun and Yu Hen Hu, "Frame-Level Pipelined Motion Estimation Array Processor" IEEE Transactions On Circuits And Systems For Video Technology, Vol. 11, No. 2, February 2001.

[5] Peter H. W. Wong and Oscar C. Au, "Modified One-Bit Transform for Motion Estimation", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 9, No. 7, October 1999.

[6] A. Celebi, O. Urhan, I. Hamzaoglu, and S. Ertürk, "Efficient hardware implementation of low bit depth motion estimation algorithms," IEEE Signal Processing Lett., vol. 16, no. 6, pp. 513-516, June 2009.

[7] Rafael C Gonzalez, Richard E Woods, "Digital Image Processing", publication year 2008.