Implementation of four step search algorithm of motion estimation using FPGA

Ms. Maya Gaikwad.

Assistant Professor, E& TC Dept. BVCOEW, Pune

Maharashtra, India.

Mob.No.9890411435

Abstract— Motion estimation is one of the key elements of many video compression schemes. Motion estimation is the process of determining motion vectors that describe the transformation from one image to another; usually from adjacent frames in a video sequence. In this paper we discuss four step search algorithm, which is a type of block matching algorithm and use for finding motion vector. Four-step search algorithm estimate the amount of motion on a block-by-block basis, i.e. for each block in the current frame, a block from the previous frame is found, that is said to match this block based on a certain criterion. We consider 64*64 pixel images and find motion vector. This paper presents a feasible study of “Motion Estimation by using VHDL for four-step search algorithm”.

Index Terms— VHDL, Compression, frame, Pixels, BDM.

I. INTRODUCTION

Motion estimation is one of the key elements of many video compression schemes is. A video sequence consists of a series of frames. To achieve compression, the temporal redundancy between adjacent frames can be exploited. That is, a frame is selected as a reference, and subsequent frames are predicted from the reference using a technique known as Motion Estimation.

Compression is a process intended to yield a compact digital representation of a signal. There are many applications that benefit when image, a video stream and audio signal are available in compressed form. Our paper is organized as follows: section 2 gives literature survey, section 3 discusses Block matching algorithms; section 4 is a four-step search algorithm. Finally section 5 deals with the summary of important results and conclusion of the paper. Future scope will be given in next section. [4].

If the temporal redundancy reduction processor employs motion compensation, then we can express its output as

\[ e(x, y, t) = I(x, y, t) - I(x-u, y-v, t-1) \]

Where \( I(x, y, t) \) are pixel values at spatial location \((x, y)\) in frame \((t)\) and \( I(x-u, y-v, t-1) \) are corresponding pixel values at spatial location \((x-u, y-v)\) in frame \((t-1)\).

Fig. 1(a) Block diagram of system

![Block Diagram of Motion Estimation](image1)

Fig. 1(b): Block Diagram of Motion Estimation [4]

II. LITERATURE SURVEY:

Motion Estimation (ME) is an important part of any video compression system, since it can achieve significant compression by exploiting the temporal redundancy existing in a video sequence. Different work done related to motion estimation and its algorithms. Firstly in 1990 a 2D motion estimation algorithm is proposed. This algorithm estimates accurately the motion parameters in the case of linear 2-D motion and gives a very good linear approximation in the case of a nonlinear 2-D motion. It assumes that the images have been previously segmented. Its main advantage in comparison to other motion estimation algorithms is its robustness in the presence of noise. An adaptive technique
for Block-Based Motion Estimation (BBME) which leads to significantly higher compression over non-adaptive application of Block Matching Algorithms (BMAs) was presented by using the Adaptive Search Window Method (ASWM) in 1996 [3].

In 1998 by considering parameters like computational complexity, requirement of on-chip memory size, amount of data fetch and times of data fetch, a large-scale-sub sample (4:1 horizontally and vertically sub sampling) hierarchical motion estimation algorithm is proposed. [5]

The full search motion estimation algorithm for video coding is a procedure of high computational cost. For this reason, in real-time low-power applications, low-cost a novel reduced complexity motion estimation algorithm is proposed in April 2002. [2]

In 2003 an efficient motion estimation algorithm for MPEG-4 video coding in real-time application by using Sum of Absolute Difference (SAD) was proposed. After that the first detailed instruction-level simulation results on motion estimation. They analyze various aspects of the selected seven typical motion estimation algorithms, such as 1.3 speed, instruction frequencies, branch behavior, and power distribution.

In 2006, in most block-based video coding systems, block-matching algorithms (BMAs) use the origin as the initial search center, which may not track the motion very well. To improve the accuracy of the fast BMAs, a new adaptive tracking search algorithm is proposed. [6]

The study of papers concluded that there are a large number of motion estimation algorithms for interframe and intraframe predictive coding. Different kinds of algorithms use different criteria for comparison of blocks.

III. BLOCK MATCHING ALGORITHMS:

Block-matching methods are the most widely used motion estimation methods for the low computation complexity compared with other methods such as optical flow based methods and pel recursive methods. [1]

The current frame is divided into many rectangle blocks. For each blocks the motion displacement is achieved by finding the displaced coordinate of a match block within the search window of a reference frame. The ‘F’ block is the block of the current frame and ‘G’ block is the block of the reference frame. The maximum number of the checking points pattern on a 5*5 window located at the center of the window is summarized as follows: [4]

Step 1: A minimum BDM point is found from a 9 checking points pattern on a 5*5 window located at the center of the 15*15 searching area as shown in Fig. If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 2.

Step 2: The search window size is maintained in 5*5. However, the search pattern will depend on the position of the previous minimum BDM point.

a. If the previous minimum BDM point is located at the corner of the previous search window, 5

Fig.2: Block matching algorithm [1]

Following are the different types of block matching algorithms. [4]

A. Exhaustive Search (ES)
B. Three Step Search (TSS)
C. New Three-Step Search (NTSS)
D. Simple and Efficient Search (SES)
E. Four Step Search (4SS)
F. Diamond Search (DS)
G. Adaptive Ruod Pattern Search (ARPS)

Each algorithm has its advantages and disadvantages. We are going to implement four-step search algorithm in this paper for finding the motion vector. Motion vector is a two-dimensional vector used for inter prediction that provides an offset from the coordinates in the test picture to the coordinates in a reference picture.

IV. FOUR STEP SEARCH ALGORITHM:

The four-step search algorithm (FSS) has been proposed by L. M. Po and W. C. Ma in 1996 [4]. This algorithm also exploits the center-biased characteristics of the real world video sequences by using a smaller initial step size compared with TSS. The initial step size is fourth of the maximum motion displacement d (i.e. d/4). Due to the smaller initial step size, the FSS algorithm needs four searching steps to reach the boundary of a search window with d=7. Same as the small motion case in the NTSS algorithm, the FSS algorithm also uses a halfway stop technique in its second and third step search.

For the maximum motion displacements of 7*7, the proposed 4SS algorithm utilizes a center-biased search pattern with 9 checking points on a 5*5 window in the first step instead of a 9*9 window in the 3SS. The center of the window is then shifted to the point with minimum block distortion measure (BDM). The search window size of the next two steps is depended on the location of the minimum BDM points. If the minimum BDM point is found at the center of the search window, the search will go to the final step (Step 4) with 3*3 search window. Otherwise, the search window size is maintained in 5*5 for step 2 or step 3. In the final step, the search window is reduced to 3*3 and the search stops at this small search window. The 4SS algorithm is summarized as follows: [8]
additional checking points as shown in Fig. are used.

b. If the previous minimum BDM point is located at the middle of horizontal or vertical axis of the previous search window, 3 additional checking points as shown in Fig. are used.

If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 3.

Step 3: The searching pattern strategy is the same as Step 2, but finally it will go to Step 4.

Step 4: The search window is reduced to 3*3 as shown in Fig. and the direction of the overall motion vector is considered as the minimum BDM point among these 9 searching points.

Fig. 3. Search patterns of the FSS.[8]
(a) First step (b) second/third step
(c) second/third step (d) fourth step

V. SIMULATION AND SYNTHESIS RESULTS:
VI. DISCUSSION

Two grey scale images are given as inputs. These images are converted into digital images by using matlab. Hence images are with either ‘0’ or ‘1’. Pixel values of both images are compared block by block. If there is change in the pixel values of both images, then motion vector is ‘1’ otherwise ‘0’ for ‘clk=1’ and ‘Reset=1’.

VII. CONCLUSION

This paper presents the implementation of four step search algorithm of motion estimation using VHDL. It is implemented on Spartan-III FPGA device. Motion vectors between two images found in this paper is limited to 32*32 images and it can be extended to 512*512 and to further big numbers. In this paper we are taking the digital values of the image not the actual image or the pixel is fed to memory. It can be done for the real time images. The synthesis report shows that the area of utilization on FPGA is about 4% that saves more power.
References:


