Comparison of Motion Estimators Selected Topics in Computer Vision Engineering

Muhammad Asad
Reg No. 110128953
University Of Sheffield
masad1@sheffield.ac.uk

I. Objective

The objective of this project is to present the comparison between full search motion estimator and an improved method. For the improved method three step search is chosen. The *Three Step Search* motion estimator is compared with *Full Search* motion estimator. Both techniques are based on block matching algorithms, however the latter is more efficient as the number of comparisons required for this method is far less than the former method. Whereas the former is much more accurate, because it searches for every possible motion with in a search area. The comparison results are shown in detail.

II. Introduction

In the recent years, the introduction of many digital technologies which use application of digital videos such as High Definition TV, smart phones, video conferencing, online streaming and use of digital cameras, has increased the need to store and transmit video sequences efficiently. This has been a dominant factor for the research and development in this field. Over the years, many different video formats have been proposed, which have the capability of storing and transmitting the video sequences using as minimum resources as possible. These include formats like MPEG, H.261, H.263 which rely on compression using different coding techniques.

Coding is dependant on both spatial and temporal redundancies in an image. For video sequences the latter is more important as it has much more redundancy in video sequences because most of the video sequences have some objects moving in foreground with some background. Most of the information is same and can be regenerated using motion vectors and some reference frames. For some of the coding techniques the first frame is always transmitted without compression and is used as a reference frame for future motion compensated frame generation, for others the reference image is transmitted from time to time for example every 6th image [1].

Generally all these coding techniques exploit the statistical redundancy and the way that human visual system works. In general there are two types of coding techniques based on the type of redundancies used for compression. Coding techniques which are based on spatial redundancies are called intraframe coding techniques and are mostly used for single image compression. On the other hand, coding techniques which rely on temporal redundancies are called interframe coding and are used in video sequences [2].

The most significant part of interframe coding is motion estimation as it takes a lot of computational power. Over the years, there has been a lot of research for developing different accurate and efficient motion estimation techniques. All these techniques can be classified into following four categories [2]:

1) **Gradient Techniques:** Gradient technique have been developed to analyse a video sequence. The result from this technique is dense motion field, which can then be used to analyse the content of the video sequence.
2) **Pel-Recursive:** Pel-Recursive techniques are derived from gradient techniques, however they are specifically developed for coding different format. These techniques constitute an important part for coding.
3) **Block Matching:** These techniques make use of the concept of minimizing certain cost function and then calculating the corresponding motion vector. Due to the simplicity of this method, there has been a lot of research for using this method in coding. This has resulted in a number of video formats that use DCT for compression to use block matching to compute motion compensated frame[3].
4) **Frequencey-Domain:** These techniques are based on the relation of transformed coefficient. However they are not as widely used as the above methods [2].

This report presents the comparison between two early block matching methods. Since there development there has been a lot of research, which has resulted in much more efficient and accurate block matching algorithms. This report first explains the common constraints in a block matching algorithms, and then explains the implementations of the two block matching motion estimators used. Then the results are presented along with comparisons, and finally this report concludes with a conclusion and output result sequences at the end.

III. Motion Estimators

This project makes comparison between two motion estimators, namely full search and three step search block motion estimation. Both of these methods depend on the underlying fact that in a given video sequence, the objects within a given frame move in front of a background to result in a sequence of images.

All block matching techniques have the some basic similarities in functionality and principles on which they are based. First of all, the block matching techniques divide the current image into a matrix of blocks. These blocks are picked up one by one and compared to blocks in different surrounding location from previous image in the sequence. The surrounding search area is defined by a search area parameter. For sequences which have really small motion, a comparatively low value of search area parameter can be used. On the contrary, if the image has a lot of motion then a high value of this parameter is used. As the number of comparisons increase with increasing this parameter, therefore selecting an optimum value for this parameter is really crucial for a good block search motion estimation technique. Most implementations use a value of 7 for search area parameter and a block size of 16x16 [1] [3].

The comparison of these blocks is usually done using some cost function. The idea is to match current block with blocks in previous image using cost function and define a motion vector for the block which minimizes this cost. Cost function is implemented using mean absolute difference algorithm which is defined as:

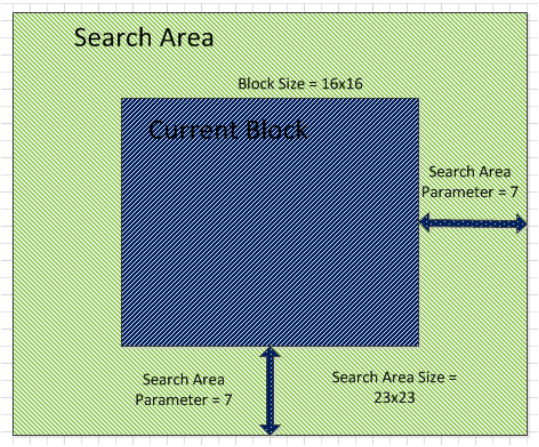


Figure 1: Basic block matching concept

$$MAD = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}|, \qquad (1)$$

where N is the block size, C_{ij} are the pixel values in the current block and R_{ij} are the pixel values in the reference block with which the current block is being compared.

The basic implementation of each of these block matching techniques differ by the algorithms adapted to find the reference block with minimum cost within the search area,

65537	65537	65537	65537	65537	65537	65537	65537	65537	0	0
65537	65537	65537	65537	65537	65537	65537	65537	65537	0	0
65537	65537	65537	65537	65537	65537	65537	65537	65537	0	0
65537	65537	65537	65537	65537	65537	65537	65537	65537	0	0
65537	65537	65537	65537	672	719	705	737	837	1037	1249
65537	65537	65537	65537	469	509	593	660	792	896	1100
65537	65537	65537	65537	532	478	491	587	689	825	986
65537	65537	65537	65537	658	537	466	507	596	716	888
65537	65537	65537	65537	753	647	510	504	540	602	748
0	0	0	0	868	733	608	511	498	538	628
0	0	0	0	953	791	622	538	459	468	557
0	0	0	0	1014	860	674	550	464	441	478
0	0	0	0	1096	922	771	574	452	386	434

Figure 2: Three Step Search working shown using cost for each comparison

which is then used to compute the corresponding motion vector. These algorithms are explained below:

A. Full Search Block Matching

In full search block matching algorithm, current block is compared with every possible reference block within the search area. This is described in Figure 1. This method has maximum number of comparisons, and hence is computationally very expensive. With search area size equal to 7 the number of cost comparison for each block are 49, assuming that the block has all of the search area defined around it and it is not one of the blocks from the edge of the image where only a part of the search area is defined. These 49 costs are then compared with each other to find the minimum cost, whose location is then used to compute motion vector.

B. Three Step Block Matching

Three step block matching is a relatively fast block matching algorithm as compared to full search block matching. This algorithm is explained in Figure 2. In the figure actual values calculated for the cost function by the algorithm are shown along with the minimum cost selected after each step. This matching technique works on the assumption that the motion of current block is based on a unimodal surface. This means that for a given current block, there exists only one minimum cost in the surrounding area and can be found near the block whose cost is minimum for a specific comparison. As the name suggests, this approach has three major steps. A step size for each step is defined. For comparison purposes, the search area size is kept same as for full search implementation, and therefore step size for first, second and third step is 4, 2 and 1 respectively.

In the first step, this algorithm computes cost with 9 reference blocks all at +/- step size from current block location with one central reference block exactly in the same position as current block. This comparison is then used to calculate the location of the block with the minimum cost for current step. For the second step, step size is decreased by a factor of 2 and the whole process is repeated at the location of the minima from step one. In the last step, the process is again repeated to find the location of

the reference block with the most minimum cost, which is then used to compute the motion vector for current block. A total of 27 costs are computed in this algorithm, comparisons within the cost calculations are also limited to comparing 9 costs at a time for all three steps, which is far less than comparing 49 costs as was in the case of full search block matching.

IV. Results and Comparisons

The algorithms were tested on the Caltrain and Flower Garden sequences. In the caltrain sequence there is a lot of small motion, and the camera also has some movement with respect to the background. There is also some moving shadows and the background is repetitive in some parts and textured in others. This sequence checks the accuracy of a motion estimator with its ability to estimate small motions in different directions. It also check the ability of a motion estimator to perform well for repetitive background. As compared to this, the Flower Garden Sequence has different types of motion; both large and small. It also checks motion estimator against moving texture and moving repetitive background in the form of flower bed and sky respectively.

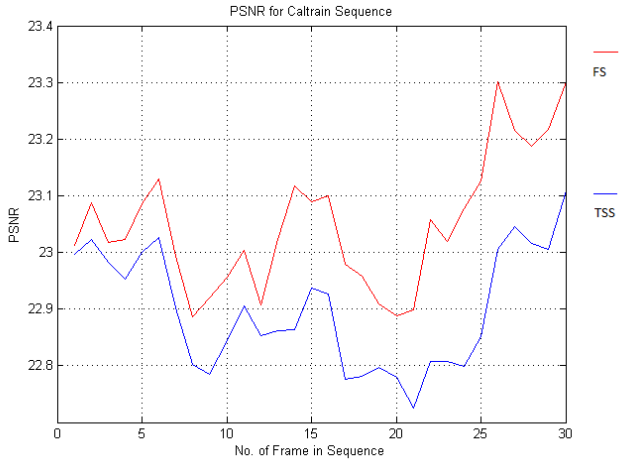


Figure 3: PSNR for FS and TSS using Caltrain Sequence

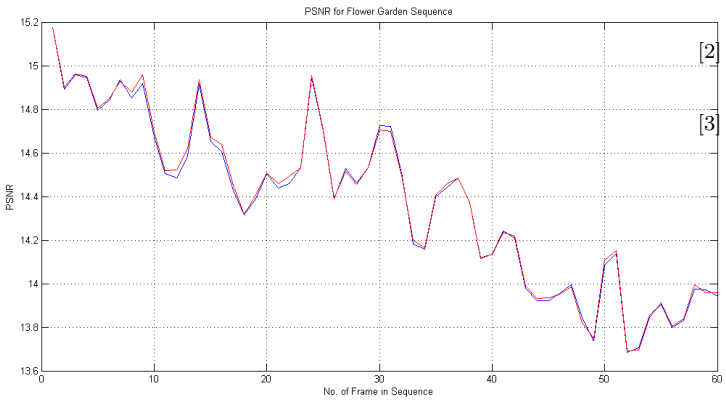


Figure 4: PSNR for FS and TSS using Garden Sequence

Using the motion vectors, motion compensated images were computed for each reference image. These motion compensated images were then compared to actual frame at that step and PSNR values were calculated. Figure 3 4 shows the PSNR values for these two sequences using both FS and TSS. The plot in red is for FS and blue is for TSS. These plots show that of the two techniques, full search motion estimator is the most accurate motion estimation technique, however computationally it is the most expensive one. As compared to this, TSS is computationally less complex and is very accurate for large motion estimation, as it has almost same PSNR for Flower Garden Sequence. TSS is not good for small motion estimation as it is totally based on the assumption of unimodal solution surface, which in the case of small motion does not contribute towards optimum solution. In fact TSS end up on a different minimum, as the small motion is missed by large step size of first step.

Figure 5 - 8 show some frames from both sequences with computed motion vectors and motion compensated frames for both FS and TSS. The accuracy of FS can be observed by the less number wrong motion vectors, whereas TSS has few error vectors (large arrows in stationary background for caltrain sequence). The flower garden sequence has almost same motion vectors for both algorithms, this is because of the fact that the motion in that image is large as compared to caltrain sequence.

V. Conclusion

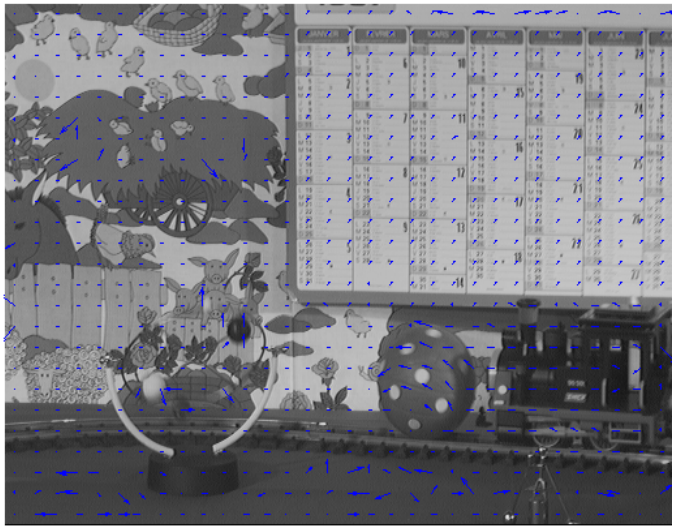
A comparison between full search motion estimation and three step search motion estimation. The results show that the full search motion estimator is highly accurate motion estimator, however it is not very efficient due to its exhaustive search algorithm. Whereas three step search compensates some accuracy for implementing a much more efficient algorithm. TSS comes close to accuracy of FS for comparatively large motions, however it fails to estimate small motions. This is evident by the blocking artifacts introduced in the motion compensated images for caltrain sequence.

References

[1] A. Barjatya, "Block matching algorithms for motion estimation," *IEEE Transactions Evolution Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[2] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital tv: A review and a new contribution," *Proceedings of the IEEE*, vol. 83, no. 6, pp. 858–876, 1995.

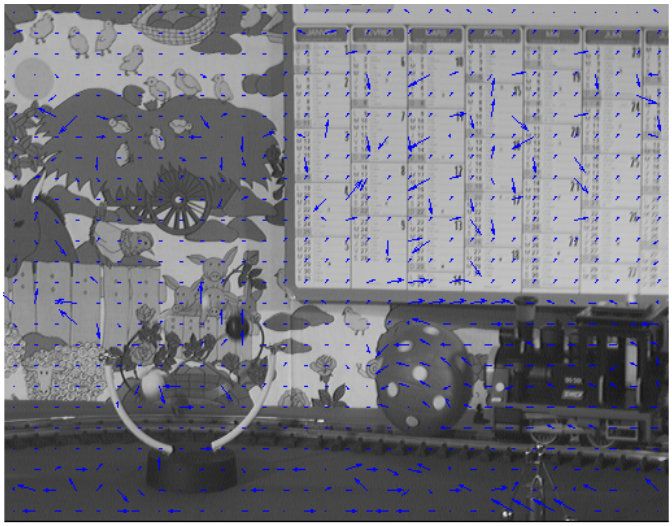
[3] P. Shenolikar and S. Narote, "Different approaches for motion estimation," in *Control, Automation, Communication and Energy Conservation, 2009. INCACEC 2009. 2009 International Conference on.* IEEE, 2009, pp. 1–4.



(a) Frame with Motion Vectors

(b) Motion Compensated Image

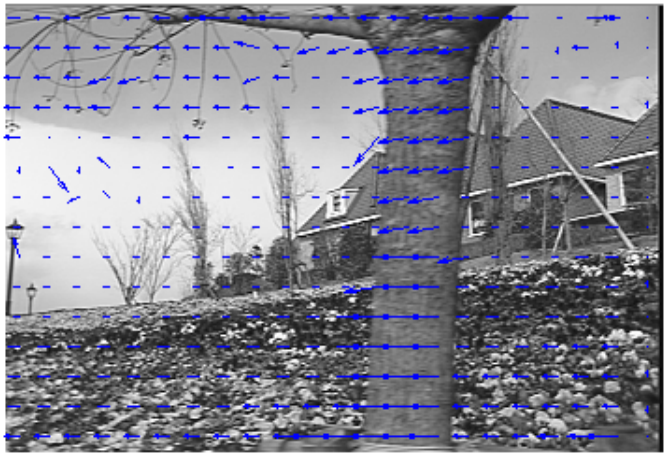
Figure 5: FS Result using Caltrain Sequence



(a) Frame with Motion Vectors

(b) Motion Compensated Image

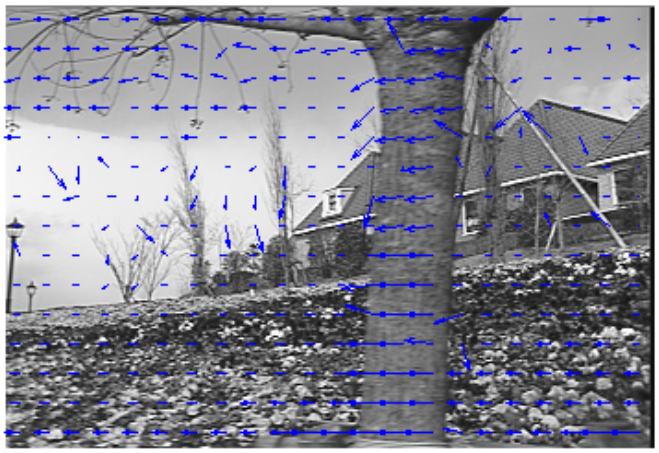
Figure 6: TSS Result using Caltrain Sequence



(a) Frame with Motion Vectors

(b) Motion Compensated Image

Figure 7: FS Result using Garden Sequence



(a) Frame with Motion Vectors

(b) Motion Compensated Image

Figure 8: TSS Result using Garden Sequence