Effects of Block Matching Algorithm and Block Sizes on Motion Estimation and Compensation in Video Compression

BY: JED WONG, MATTHEW GU

Aim:

This study aims to investigate the performance of motion compensation (MC) in video compression through varying block sizes and explore the differences from matching algorithms implemented in fixed size block matching.

Introduction:

Video compression has become essential for effectively storing and transmitting video data in the modern digital era, one way of which is through Motion Compensation (MC). [3] Frames in a video, compared to separate static images, exhibit sequential properties, such that each frame has little differences to the previous frame. MC leverages this property. Assuming that it is possible to obtain a close match of the next frame via matching pixels in a previously sent reference frame, MC employs a two-stage process to generate a matching prediction of the next frame from the previous frame. Firstly, the frames are segmented into blocks, where block matching algorithms (BMA) are applied to search in a selected range to output vectors indicating the coordinate of the block in the reference frame that closely matches the block in the actual frame. This is the most computationally heavy task in MC. [6] An estimated frame is then computed using the motion vectors, and the sender only sends the motion vectors and the difference between the estimated frame and the actual frame, where the actual frame can be then reconstructed by the receiver with the information, achieving significant bit compressions in video transmissions and storage.

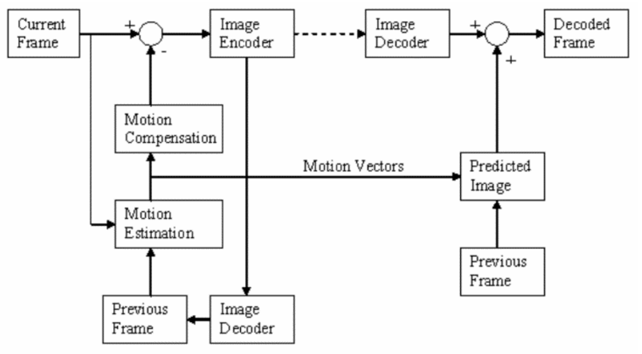

*Figure 1: SEQ Figure * ARABIC 1 Process Flow of an MPEG / H.26x Video Compression [1]*

This investigation involves importing a short video into MATLAB to perform motion compensation on multiple frames. The study includes comparing the amount of the prediction error signal with and without MC whilst also analysing alternative fixed size block matching algorithms employed to identify the closest match with a neighbouring block of pixels in the previous frame. During the analysis, the block sizes and block matching algorithms from a selected few will be varied to observe their corresponding effects on the MC processing complexity and the accuracy of estimated frame generated.

Block Matching Algorithm Implemented:

1. *Exhaustive Search*

Exhaustive Search is a block-matching technique for video compression that seeks the best matching block in a reference frame for each block in the current frame. The technique compares the current block to every conceivable block in the reference frame's search window, which is a predetermined rectangular region centred on the current block. It computes a cost function for each block in the search window by measuring the difference between the pixels in the current and reference blocks. The block with the lowest cost function value is picked as the best match and will be exemplified through the motion vector. This search guarantees that the optimal choice is taken.

2. *Three Step Search (TSS)*

TSS is one of the earliest attempts to reduce the computational cost of exhaustive search whilst maintaining the accuracy of the estimated frame by performing the search in a coarse-to-fine manner. [1] The original algorithm searches in three steps. It searches the centre and all blocks that are 4 pixels vertically, horizontally, and diagonally from the centre. The step size of 4 pixels is then halved, where the same search is performed taking the least cost block seen so far as the new centre. This halves and the search process are repeated until the step size becomes 1 when the last search is performed. This corresponds to a search range of 7, but can be extended to further ranges, by making the starting step size greater than 4. This approach exploits the correlation between neighbouring blocks. TSS offers considerable positive trade-offs between computational complexity and prediction accuracy in video encoding applications. With a search range of 7, the number of search points is 25, one ninth of that of exhaustive search while achieving comparable results. [1] However, more sophisticated motion estimation algorithms have been developed in recent years that outperform TSS in terms of the cost-to-accuracy ratio. [4]

3. *New Three Step Search (NTSS)*

NTSS is an extension and improvement of TSS. In addition to the first step, it searches the neighbouring points of the centre. Early termination of the search is introduced if the best match is found at the centre or in one of the neighbours, such that it wastes no computation power when the motion is slight or zero. [2] It is hence more sensitive to local motion.

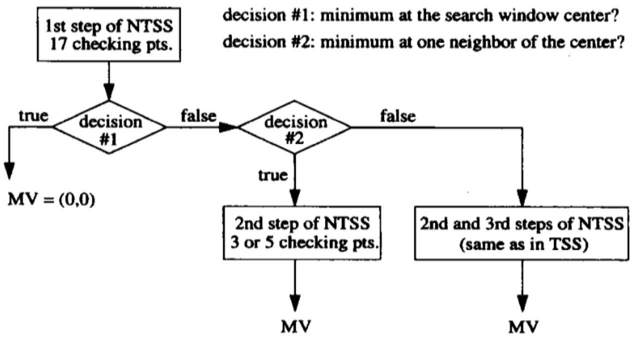
1st step of NTSS 17 checking pts.
decision #1: minimum at the search window center?
decision #2: minimum at one neighbor of the center?
true
decision #1
false
decision #2
false
MV = (0,0)
true
2nd step of NTSS 3 or 5 checking pts.
2nd and 3rd steps of NTSS (same as in TSS)
MV
MV

Figure 2: Flow Chart for New Three Step Search [2]

4. *No Compensation*

Additionally, the effects of applying no search, meaning that the block at the same coordinate is taken as the close match, is also included to show the difference between motion compensation and no motion compensation in the ability to compress video size.

In this study, the cost functions implemented to compute the differences between blocks in the algorithms is the mean absolute difference (MAD). The equation of which is shown below:

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|C_{ij} - R_{ij}\right|$$

where N in the size of blocks compared, C_{ij} is the pixel of the current block and R_{ij} is the pixel of the reference block.

The Peak-Signal-to-Noise-Ratio (PSNR) is calculated as the measure for power of error to compare the accuracy of the estimation from the various search algorithms implemented in this study. Note that MSE is the mean squared error between the predicted frame and the actual frame.

$$PSNR = 10 * log_{10}\left[\frac{255}{MSE}\right]$$

A high PSNR suggests a small distortion in compressed video relative to the original video, and hence indicating a matching frame output quality and compression rate and vice versa. [5]

Simulation Results:

In this study, a 480 by 640 resolution, 25 frames per second video sequence containing fast motion is used for simulation to measure the performance, cost and thus efficiency of the above algorithms. The simulation for algorithm comparison fixed parameter search range of 7 and a block size of 16 by 16 pixels. Further trend on the effect of block size is explored, where simulation is run with block sizes of 2, 4, 8, 16, 32, 64 and a fixed search range of 7.

Table 1: Average PSNR and Cost for First 200 Frames

Algorithm	PSNR (dB)	Approximate FLOP Per Motion Estimation (*10^6)
No MC	23.9301	0
ES	30.0751	65.4070
TSS	29.8323	7.39404
NTSS	29.8269	7.28759

With no block matching algorithm used, the output shows the worst accuracy from the significantly lower PSNR, equivalent to worst compression performance at zero computational cost. This observation demonstrates the necessity of some level of computation to achieve meaningful compression. ES proves to be the most accurate algorithm in terms of estimation, as it guarantees the optimum in the current search range. On the other hand, it requires nearly 9 times the computation needed for TSS and NTSS. TSS reduces the computation cost, in exchange for 0.8% worse accuracy. It suggests that TSS is an excellent option when accuracy is essential, but computational cost needs to be minimized. NTSS improves upon TSS results, achieving a 1.4% decrease in average computational cost at the expense of 0.018% accuracy decrease.

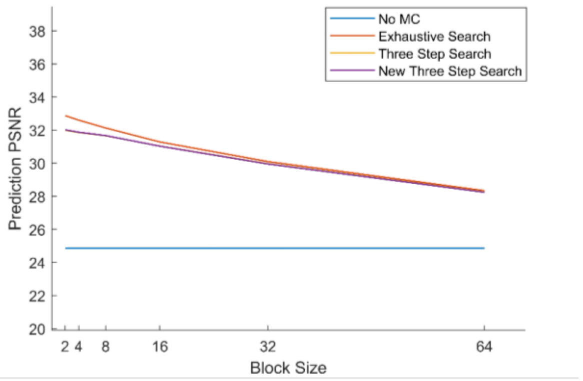

Figure 3: Average PSNR Against Block Size for First 200 Frame

Furthermore, the project studies the effect of varying block sizes. As expected, an inversely associated trend between accuracy and block size is observed in Figure 3. This can be attributed to the higher block sizes missing the finer details of the smaller, local motion that occurs between the frames.

Interestingly, it is discovered that the rate of decrease in accuracy is not consistent across all BMAs at smaller block sizes. The decrease in accuracy is greater for ES than both NTSS and TSS. This is possibly due to TSS algorithms being further from optimal with the increased search space. As a result, the differences between the accuracy of ES and TSS algorithms are greater, and it is likely that the impacts of larger block sizes reduce such differences.

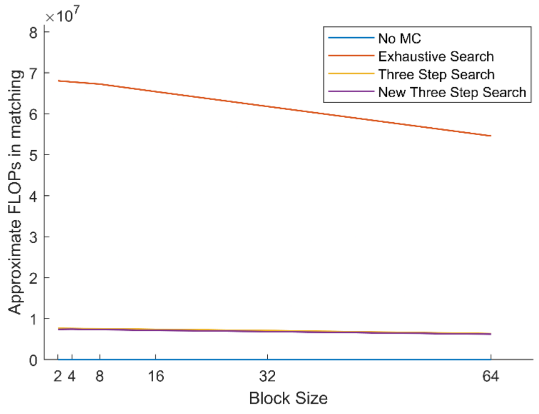

Figure 4: Average FLOPs Against Block Size for First 200 Frames

While Figure 4 does underpin the expected decrease in computation power as the block size is increased, the reduction is most prevalent in exhaustive search and close to indistinguishable in TSS algorithms. The decrease in block size does reduce the total number of blocks to be matched by a quadratic relationship, but for each comparison, the number of floating-point comparisons also increases by the same factor. The search points for ES are much reduced, yet the search points for TSS algorithms remain the same 3 steps. Hence, the effect is negligible. However, it is worth noting that the running time is much greater for smaller block sizes, even with the same number of floating-point operations, possibly due to matrix comparison optimization in the hardware, and overheads that come with each matrix assignment and comparison, which becomes significant when low block sizes are used.

Conclusion:

In the first part of this study, the motion compensation for video encoding is explored, where various block matching algorithms are compared in relation to the trade-off between accuracy and computation power. It is observed that while ES guarantees the best accuracy, it requires significant computational cost, which may not be practical for real-time applications. Fast BMAs such as TSS and one of its successors NTSS improve in accuracy to cost ratio and are more cost-efficient than ES, with NTSS being the most cost-effective option.

Additionally, this investigation found a negative correlation between the increase in block size and the computation cost and accuracy. In particular, it is noted that while FLOPs obtained are similar, the real runtime is drastically longer for smaller block sizes. Hence, the true computation cost for various block sizes is subject to other software and hardware specifications, such as hardware optimization for matrix operations.

References:

[1] R. Singh and P. Singh, "A Study on Block Matching Algorithms for Motion Estimation," ResearchGate, 2012. [Online]. Available: https://www.researchgate.net/figure/MPEG-H26x-video-compression-process-flow_fig1_50235332. [Accessed: May 1, 2023].

[2] Reoxiang Li, Bing Zeng and M. L. Liou, "A new three-step search algorithm for block motion estimation," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 4, no. 4, pp. 438-442, Aug. 1994, doi: 10.1109/76.313138.

[3] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: a review and a new contribution," in Proceedings of the IEEE, vol. 83, no. 6, pp. 858-876, June 1995, doi: 10.1109/5.387089.

[4] W. Hassen and H. Amiri, "Block Matching Algorithms for motion estimation," 2013 7th IEEE International Conference on e-Learning in Industrial Electronics (ICELIE), Vienna, Austria, 2013, pp. 136-139, doi: 10.1109/ICELIE.2013.6701287.

[5]“Peak Signal-to-Noise Ratio as an Image Quality Metric,” www.ni.com. https://www.ni.com/en-au/shop/data-acquisition-and-control/add-ons-for-data-acquisition-and-control/what-is-vision-development-module/peak-signal-to-noise-ratio-as-an-image-quality-metric.html [accessed May 11, 2023].

[6] Rahul Gonawala and Prof. Nehal Shah, “Full Search and Fast Search Algorithms for Motion Estimation in Video Compression”, International conference on Information, Knowledge & Research in Engineering, Technology & Sciences, March 2012.