Interframe Prediction Based Stationary Block Revalidation VBSME Using Full Search and Vector-driven Techniques

M. Vinutha[1,3,*] and T. M. Manu[2,3]

[1]*Nitte (Deemed to be University), NMAM Institute of Technology (NMAMIT), Department of Electronics Engineering (VLSI Design & Technology), Nitte, Karkala, Udupi, Karnataka 574110, India*
[2]*Department of Electronics and Communication Engineering, KLE Institute of Technology (KLEIT), Hubballi, Karnataka 580027, India*
[3]*Visvesvaraya Technological University, Belagavi, Karnataka 590018, India*
E-mail: m.vinutha24@gmail.com; manutmece@kleit.ac.in
[*]*Corresponding Author*

Received 14 July 2025; Accepted 05 November 2025

Abstract

Full search (FS) motion estimation provides high accuracy but at high computational cost, while fast search methods introduce irregular, hardware-unfriendly patterns. This work proposes four adaptive FS-based VBSME algorithms that preserve FS regularity while reducing complexity through direction-driven search, adaptive block sizes, stationary block revalidation (SBR), and early termination using a spiral pattern.On CIF sequences, vector-driven VBSME with early termination reduces SAD computations by 53.9—94.5%, while SBR improves PSNR by up to 1.39 dB. For 1080p video, SBR-based FS VBSME achieves higher PSNR than conventional FSBME, while SBR-based vector-driven VBSME delivers nearly 60% fewer SAD evaluations.

Keywords: Motion estimation, VBSME, hardware-friendly search, real-time compression.

Journal of Mobile Multimedia, Vol. 21_6, 1195–1220.
doi: 10.13052/jmm1550-4646.2168

1 Introduction

With advances in technology and increasing demands in the entertainment industry, memory and bandwidth requirements have increased tremendously. Many video compression techniques have been introduced to meet the increasing demands of storage and bandwidth. A video is always stored in terms of the number of frames per second. In any video, there is huge redundancy in both spatial and temporal domains. The similarity of content within the frame in the 2D image contributes to spatial redundancy. The similarity between frames along the time domain leads to temporal redundancy. In any video, redundancy in the temporal domain is very high due to common scenic content between frames [1, 2]. Compression in the temporal domain involves a technique called motion estimation (ME).

ME can be pixel-based, where for each pixel in the current frame, a motion vector (MV) is found with respect to the pixels in the reference frame (previous frame) [3]. However, this is an exhaustive process leading to higher computational overhead. Therefore, block search ME is a widely adopted technique for ME. This involves dividing the current frame, whose MV has to be found, into non-overlapping blocks of a fixed size or variable sizes. For every block in the current frame, the best matching block in the reference frame is searched within the search window. The process is repeated for all consecutive pairs of frames.

Widely used metrics for ME include SAD, mean squared error (MSE), and mean absolute difference (MAD), which help us find the MV. The displacement of the best matching block with respect to the candidate block (current frame block) is considered the MV. Figure 1 illustrates finding the MV for a block in the current frame, between a pair of frames. If S is the search size, $(2S + 1)^2$ is the total number of search positions.

The pattern in which blocks in the search window are searched can also be varied. They include a spiral search pattern as in Figure 2(a) and a raster scan pattern as in Figure 2(b), with their own merits based on the objectives

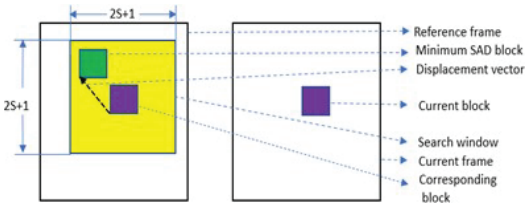

Figure 1 Finding the displacement vector or MV using the block search method.

57	56	55	54	53	52	51	50	81
58	31	30	29	28	27	26	49	80
59	32	13	12	11	10	25	48	79
60	33	14	3	2	9	24	47	78
61	34	15	4	1	8	23	46	77
62	35	16	5	6	7	22	45	76
63	36	17	18	19	20	21	44	75
64	37	38	39	40	41	42	43	74
65	66	67	68	69	70	71	72	73

a

1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54
55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81

b

Figure 2 (a) Spiral search pattern, (b) raster scan pattern.

of the ME. If early termination of SAD calculations is one of the objectives, the suitable search pattern would be the spiral search pattern. The raster scan pattern is a more systematic way of searching. Once the MV for all current frame blocks is found for a pair of frames, the process repeats for the next corresponding pair of frames. To transfer or store a video, only necessary information along with the MV can be used for encoding the video and transmitting it, thus resulting in reduced bandwidth requirements and storage requirements.

Despite the availability of many fast search algorithms such as three-step search, diamond search, and hexagon-based search, none achieve the balance between computational efficiency and hardware suitability. Full search block matching (FSBME) remains the most accurate method but is computationally prohibitive, making it impractical for real-time or energy-constrained hardware systems. On the other hand, fast algorithms reduce complexity but rely on irregular and adaptive search patterns that make them hardware-unfriendly and difficult to parallelize. Thus, the central problem is: how can we retain the systematic nature and accuracy of FSBME while reducing its computational burden and making it hardware-friendly for real-time use?

An additional challenge arises in the handling of stationary blocks. In many video sequences, large regions remain unchanged for several frames. Conventional ME algorithms, including FSBME and fast search methods such as three-step search, diamond search, and hexagon search, process every block of every frame independently. This repeated estimation wastes computations in regions that are truly stationary. On the other hand, permanently assuming stationary status is also problematic, since motion is dynamic and blocks initially classified as stationary may later become active. This tension between avoiding redundant calculations and maintaining accuracy prompted our work. By introducing periodic revalidation of stationary blocks, our approach skips unchanged regions to save computations while re-checking them at fixed intervals to ensure motion changes are captured. This strategy

complements the adaptive block-size and vector-driven search mechanisms, together enabling both reduced complexity and improved video quality.

This work aims to develop adaptive full-search-based algorithms that significantly reduce the number of computations while preserving video quality. The expected outcome is an approach that provides predictable search patterns, lower complexity, and near-FSBME accuracy, enabling practical deployment in hardware for energy-efficient real-time video compression.

1.1 Novelty of the Work

The novelty of this research lies in the following contributions:

1. Four new adaptive algorithms are proposed that combine the systematic full-search strategy with adaptive block-size motion estimation:
   - Interframe prediction based FS VBSME
   - Vector-driven VBSME
   - Stationary block revalidation based FS VBSME
   - Stationary block revalidation based vector-driven VBSME.
2. Stationary block revalidation is introduced to re-check earlier stationary blocks, improving accuracy and PSNR compared to existing approaches.
3. Vector-driven adaptive search windows are developed to eliminate unlikely search positions, reducing the number of candidate points from 81 to as few as 9 while maintaining accuracy.
4. Integration of a spiral search with early termination ensures systematic, predictable, and hardware-friendly computation while significantly cutting down on SAD calculations.
5. Extensive evaluation on 12 CIF sequences demonstrates reductions in computation (up to 94.54%) with negligible loss in PSNR, confirming a favorable trade-off between efficiency and fidelity.

In addition, while modern video compression standards such as H.264/AVC, H.265/HEVC, and VVC/H.266 employ advanced block-based motion estimation with fast search heuristics, they still perform redundant computations in stationary regions. Our proposed stationary block revalidation combined with vector-driven search and early termination complements these standards by reducing unnecessary block-matching operations while preserving high PSNR, making it suitable for real-time hardware-friendly applications.

The proposed strategies have strong practical relevance. Applications such as surveillance, video conferencing, and remote monitoring can tolerate

minor quality loss but demand faster response and low latency. By skipping redundant SAD computations in stationary regions and revalidating only at intervals, the methods reduce block-matching operations compared to FSBME, with the potential to lower execution time in time-critical multimedia applications.

The remainder of this paper is organized as follows. In Section 2, we present the related work. The methodology including the proposed algorithms, an interframe prediction based full search variable block size ME (FS VBSME), a vector-eriven VBSME, a stationary block revalidation based FS VBSME and a stationary block revalidation based vector-driven VBSME are presented in Section 3. The experimental results and comparative analysis are presented in Section 4. In Section 5, we have presented a discussion on the attained results, highlighting the hardware compatibility of parallelism and the limitations. We have drawn conclusions and present future work in Sections 6 and 7.

2 Related Work

Early fast search algorithms:

There are various ME algorithms based on the search patterns. Many fast search algorithms have been introduced, which use selected search positions instead of an exhaustive search process in FSBME. 2D-logarithmic search, three-step search, four-step search, cross-search, diamond search, hexagon-based search are few of the initially introduced algorithms [4–11]. In all these algorithms the search positions are chosen in a particular pattern to find the optimal MV. Since these algorithms involve very limited search positions, the possibility of the best matching position being among the positions that are not searched may be high thus resulting in suboptimal solutions [12]. This can degrade the video quality of compressed image and thus PSNR. Although such algorithms significantly reduce complexity, their main drawback lies in the risk of missing the global optimum motion vector, which limits their applicability when high fidelity is required.

Adaptive/predictive algorithms:

Many algorithms follow adaptive search positions and patterns based on some prediction criteria [12–14] This includes finding a range of motion initially and then concentrating on region of motion to find the MV. In these algorithms, though computational overload is reduced and the quality is compromised to a greater extent. These adaptive approaches thus offer a

better balance between speed and accuracy, but their dependence on prediction makes them vulnerable to error propagation, especially in complex or irregular motion scenarios.

Full search improvements/successive elimination:

Full search block motion estimation (FSBME) is one of the most efficient techniques of ME. Compression techniques involving FSBME are highly efficient [22]. However, due to computational overhead, many algorithms were introduced that involve successive elimination of search blocks thus reducing the number of blocks searched to find MV [15–19]. Such pruning strategies still retain substantial computational burden since they essentially attempt full search in a restricted manner rather than fundamentally avoiding redundant block evaluations.

Variable block size motion estimation (VBSME):

In order to improve the efficiency of ME, VBSME algorithms have been introduced. Some of these algorithms involve block size selection for different regions based on criteria applied for macroblocks [20,21]. H.264/AVC uses VBSME for higher efficiency. However, the computational overhead is very high. While VBSME improves coding efficiency and adaptability, its heavy computational requirements remain a bottleneck, particularly when applied to every block across multiple frames.

Recent hardware/coding-level approaches:

Recent studies in 2024 have advanced motion estimation with different emphases. Hardware-oriented designs for VVC motion estimation [23] leverage parallel architectures for high-resolution video, while motion-free B-frame coding [24] eliminates explicit block matching via coding-level modifications. Although such approaches improve efficiency in their respective domains, they do not address redundant block-level computations across frames. Our work focuses on this gap by introducing algorithmic strategies such as stationary block revalidation and a vector-driven search, offering a complementary direction to hardware- and coding-level efforts.

Our work:

In this work we introduce VBSME algorithms based on FS and vector driven fast FS algorithms. We identify the stationary blocks based on the intra video predictions and set the varying block sizes for different segments of the image that is unique for each video. Experiments are conducted for the CIF and

1080p resolution sequences. We use spiral search pattern in all algorithms making it compatible for early termination. Our approach directly addresses the redundancy challenge, reducing unnecessary SAD computations while maintaining near-FSBME accuracy. This distinguishes our method from prior efforts that either compromise quality for speed or focus on hardware scaling.

3 Methodology

In this section four different algorithms for VBSME are presented, where variable block sizes are selected based on interframe prediction techniques. The two proposed algorithms, Interframe prediction based FS VBSME and vector-driven VBSME do not revalidate stationary blocks. The algorithms, stationary block revalidation based FS VBSME and stationary block revalidation based vector-driven VBSME include revalidation techniques for stationary blocks.

3.1 Interframe Prediction Based FS VBSME

In this algorithm, interframe prediction is used for dividing frames (F) into stationary and non-stationary blocks. Non-stationary blocks are further divided into sub blocks based on predicted MV for the macroblock. The algorithm below explains the steps followed.

Step 1: Perform FSBME for frame pairs F1 F2, F2 F3, F3 F4 with block size of 16. This provides the MV for the blocks in F2, F3, F4 correspondingly. We use SAD as the metric for ME.

Step 2: The sum of MVs, $Sum\_MV(p,q)$ of all corresponding blocks of F2, F3, F4 is found. For a frame with H number of rows and W number of columns, the number of blocks along the rows will be $\frac{H}{Bsiz}$ and along the columns will be $\frac{W}{Bsiz}$, where $Bsiz$ is the block size which is 16. The X and Y coordinate positions of a block are represented as (p,q).

$$\sum_{p=1}^{H/Bsiz}\sum_{q=1}^{W/Bsiz} Sum\_MV(p,q)$$

$$= \sum_{p=1}^{H/Bsiz}\sum_{q=1}^{W/Bsiz} \Big(MV1(p,q) + MV2(p,q) + MV3(p,q)\Big) \quad (1)$$

Step 3: The VBSME starts here which is based on Sum_MV. For frame pairs F4 F5 to F$(N-1)$ FN, where FN is the last frame, the cases below define the selection of different block sizes:

Case 1: If $Sum\_MV(p,q) = 0$, the block at position (p,q) is declared as a stationary block as there is no movement at all in three consecutive frames. The MV is considered $(0,0)$ and no further processing is done.

Case 2: If $Sum\_MV(p,q) \leq 2$, the block at position (p,q) is declared to have very slight movement in three consecutive frames. Therefore, a block size of 16×16 is considered for these blocks.

Case 3: If $2 < Sum\_MV(p,q) \leq 4$, the block at position (p,q) is declared to have considerable movement in three consecutive frames. Therefore, a block size of 8×8 is considered for these blocks. In this case a 16×16 macroblock is divided into four sub-blocks of size 8×8 each.

Case 4: If $Sum\_MV(p,q) > 4$, the block at position (p,q) is declared to have higher displacement between three consecutive frames. Therefore, a block size of 4×4 is considered for these blocks. In this case the 16×16 macroblock is divided into 16 sub-blocks of size 4×4 each.

Figure 3 illustrates how the macroblock (MB) is divided into sub-blocks of different sizes. Each square block in the first image is block of size 16×16. All gray blocks represent blocks that are stationary according to Case 1. Red boxes indicate the blocks which have slight movement according to Case 2. Therefore, the block size is retained as 16×16. Green boxes indicate blocks with considerable displacement between the frames as in Case 3. Therefore, the macro block is divided into 4 sub blocks, each of size 8×8. Yellow boxes indicate blocks with higher displacement between the frames as in Case 4. Therefore, the MB is divided into 16 sub blocks, each of size 4×4.

Step 4: Find the MV for all current blocks of varied block sizes for consecutive pair of frames using the SAD metric. This way VBSME is conducted for the frame pairs F4 F5 to F$(N-1)$ FN. For each MB in position (p,q) with block size, $Bsiz = 16$, the MV for the sub-blocks within, of block size, B_Size is found. Based on the following cases U, V values are selected.

Case 1: If $Sum\_MV(p,q) = (0,0)$, the ME for the macroblock of size 16 is terminated and next block is loaded.

Case 2: If $Sum\_MV(p,q) \neq (0,0)$ and $Bsiz = B\_Size$, then $(U,V) = (1,1)$.

The block ranges from $(B\_Size)(p-1)+U$ to $(B\_Size)(p)$ along the rows $(B\_Size)(q-1)+V$ to $(B\_Size)(q)$ along the columns.

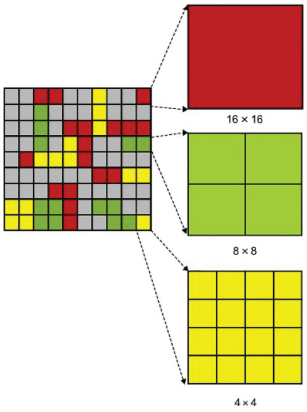

Figure 3 Sub-division of macroblocks into sub-blocks of different sizes.

Case 3: If $Sum\_MV(p,q) \neq (0,0)$ and $B\_size = 8$, then $(U,V) = (1,1),(1,2),(2,1),(2,2)$.

The sub-block ranges from $(Bsiz)(p-1)+(U-1)(B\_Size)+1$ to $(Bsiz)(p-1)+U(B\_Size)$ along the rows and from $(Bsiz)(q-1)+(V-1)(B\_Size)+1$ to $(Bsiz)(q-1)+V(B\_Size)$ along the columns.

Case 4: If $Sum\_MV(p,q) \neq (0,0)$ and $B\_size = 4$, $(U,V) = (1,1),(1,2),(1,3),(1,4),(2,1),(2,2),(2,3),(2,4),(3,1),(3,2),(3,3),(3,4),(4,1),(4,2),(4,3),(4,4)$.

The sub-block ranges from $(Bsiz)(p-1)+(U-1)(B\_Size)+1$ to $(Bsiz)(p-1)+(U)(B\_Size)$ along the rows $(Bsiz)(q-1)+(V-1)(B\_Size)+1$ to $(Bsiz)(q-1)+(V)(B\_Size)$ along the columns.

Step 5: Once FSBME for all frame pairs is conducted and MV is found, the total number of calculations is found using Equations (2) to (9). H is the height and W is the width of the frame. Block size can be $Bsiz$ or B_size based on the block for which SAD is being evaluated. Block size is $Bsiz$ for MB and B_size for subblock and S, is the search size.

The total number of search positions per block is given by:

$$T_{SP} = (2S+1)^2 \tag{2}$$

The total number of subtractions per block:

$$T_{SB} = (block_{size})^2 \tag{3}$$

The total number of absolute calculations per block:

$$T_{AB} = (block_{size})^2 \tag{4}$$

The total number of additions per block:

$$T_{AD} = (block_{size})^2 - 1 \tag{5}$$

The total number of blocks in a frame of size $H \times W$:

$$T_B = \left(\frac{H}{Block\ size}\right)\left(\frac{W}{Block\ size}\right) \tag{6}$$

The total minimum computations per block:

$$T_{\min} = (2S+1)^2 - 1 \tag{7}$$

The total number of minimum SAD comparisons per frame:

$$T_{\min \text{frame}} = T_B \cdot \left((2S+1)^2 - 1\right) \tag{8}$$

The total number of SAD operations required per frame is:

$$\begin{aligned} \text{Total}_{SAD} = T_B\,[T_{SP}(T_{SB} + T_{AB} + T_{AD})] \\ + T_{\min} \cdot T_B \end{aligned} \tag{9}$$

Step 6: Reconstruct the frame blocks using the MVs found with respect to their reference frames. Find the PSNR.

The interframe prediction based FS VBSME algorithm effectively reduces computational complexity by adaptively selecting block sizes based on motion characteristics. By categorizing blocks into stationary ($Sum\_MV = 0$), slight movement ($Sum\_MV \leq 2$), considerable movement ($2 < Sum\_MV \leq 4$), and high movement ($Sum\_MV > 4$) categories, the algorithm optimizes the trade-off between computational efficiency and video quality. This approach eliminates unnecessary processing for stationary blocks while appropriately sizing sub-blocks for areas with varying motion intensities, resulting in significant computational savings compared to traditional fixed-block FSBME.

3.2 Vector-driven VBSME

In the proposed algorithm, we have vector-driven VBSME with and without early termination techniques. In both these algorithms, like in the interframe prediction based FS VBSME algorithm, the interframe prediction is used for dividing the frames into stationary and non-stationary blocks. The non-stationary blocks are further divided into sub-blocks based on the predicted

MV for the MB. For a candidate block, search positions in the direction opposite to the MV of the corresponding block in the previous frames are eliminated. The least probable search positions are eliminated.

Interframe prediction is conducted for the first three frame pairs to decide the block sizes to be used for different segments in the image. Since decisions on the direction of displacement have to be made based on MVs of variable block sizes, full search using VBSME is conducted for the F4 F5 pair. Based on the MVs for varied block sizes of the F4 F5 pair, vector-driven VBSME starts from the F5 F6 pair. The vector-driven VBSME algorithm follows the steps below.

Step 1 and *Step 2* are similar to the proposed interframe prediction based FS VBSME.

Step 3: The VBSME starts here which is based on the Sum_MV. Conduct VBSME for the F4 F5 pair based on the FS to obtain the MV for F5. The block size selections for frame pairs F4 F5 up to F$(N-1)$ FN is explained clearly in Step 3 of the interframe prediction based FS VBSME. The MVs are found for all blocks of different sizes.

Step 4: Load the F5 F6 pair. The vector-driven VBSME starts from here. Evaluate MVs found using VBSME for the F4 F5 pair, to confine the search positions, based on the direction of displacement between the blocks across the frames. The below cases explain the selection of search positions in the search window.

Figure 4 shows the search windows selected based on the MV of the reference block. If the MV of the blocks in F4 (reference frame) corresponding to the candidate block in F5 (current frame) shows column displacement in the forward or backward direction, row displacement in the upward or downward direction, no displacement.

Step 5: The step 4 is repeated until MVs for last frame pair is found.

Step 6: Find the total number of calculations due to SAD.

Step 7: Reconstruct the frames based on the MVs found and evaluate the PSNR.

In Figure 4(a–d), it can be observed that the number of search positions are reduced to 45, which was 81 in case of interframe prediction based FS VBSME for a search size, $S = 4$. For the zero movement MV, as in Figure 4(e), the number of search points is only 9. As can be observed, we have used spiral search pattern in our estimation process making the

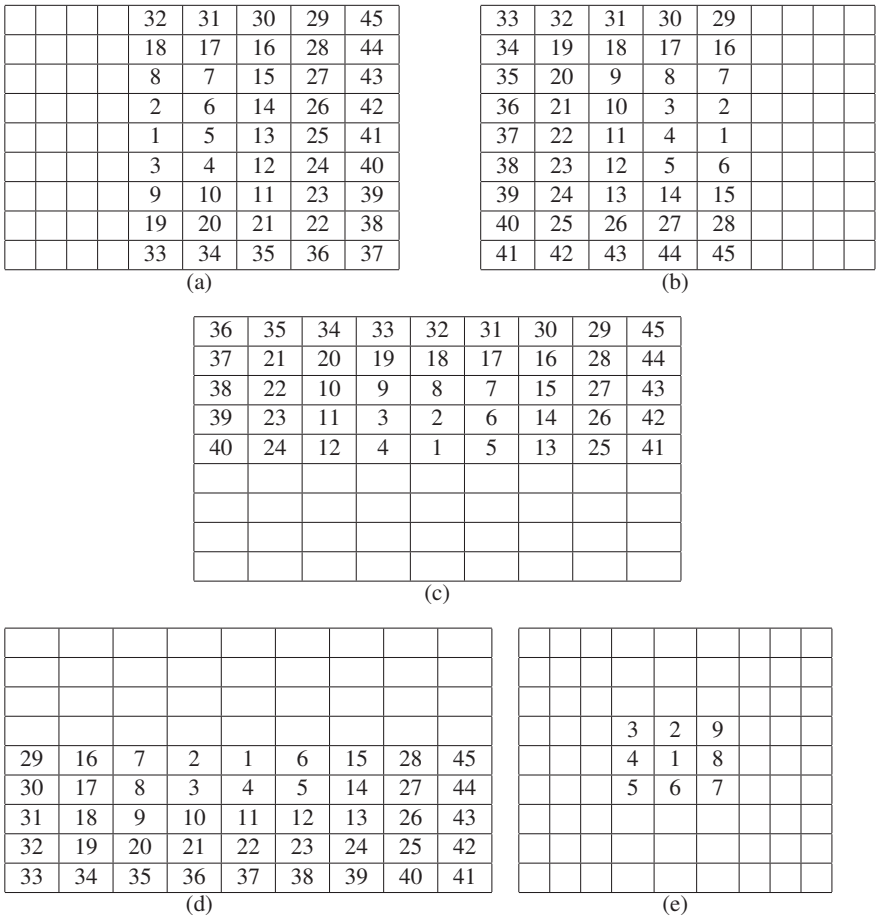

Figure 4 Adaptive search window (a) forward displacement, (b) backward displacement, (c) upward displacement, (d) downward displacement, (e) no displacement.

algorithm compatible for early termination of SAD calculations. This is based on the idea that the search block positions closest to the zero-motion position have the highest probability of becoming the best matching position. In this way, the minimum SAD value may be obtained in the beginning of the search process, helping us to terminate the early SAD.

The proposed algorithms make use of the spiral search pattern in the process of ME to find the best matching position. Taking advantage of the fact that the probability of best matching position being very close to zero motion

position is high, the search starts from the zero motion position. Therefore, in order to reduce the total number of calculations further, the early termination of SAD is incorporated in the algorithm. Since, in most of the cases the minimum SAD position is obtained at the beginning of the search process, for the rest of the positions, SAD calculations can be terminated as soon as the current value of SAD exceeds the minimum SAD. Therefore, a new feature of the early termination of SAD is added to the proposed vector driven algorithm, where the SAD calculations are terminated when they exceed the minimum found SAD so far. This additional feature reduces the number of calculations further.

3.3 Stationary Block Revalidation based FS VBSME

Based on the fact that initially identified stationary blocks may not remain stationary across the frames for a video, we introduce this new algorithm. It modifies the interframe prediction based FS VBSME by periodic revalidation of the stationary blocks. If the earlier identified stationary blocks have a non-zero MV when it is tested for revalidation, the block size to be used is evaluated and converted to a non-stationary block from the stationary block.

The steps below clearly explain the algorithm.

Steps 1–3 are exactly same as that followed for interframe prediction based FS VBSME.

Step 4: Perform VBSME for the frame pairs F4 F5 to F9 F10. For each macroblock in position (p, q) with block size, $Bsiz = 16$, the MV for the sub-blocks within, of block size B_Size is found. Based on the cases exactly like in step 4 of interframe prediction based FS VBSME, U and V values are selected.

Step 5: For the F10 F11 pair, we check all the MBs which were earlier identified to be stationary. For revalidation of stationary blocks, all the stationary blocks are converted to non-stationary blocks for one frame pair. The rest of the block's sizes for different segments of the frame remain unchanged. Case 1 explains how stationary blocks are changed to non-stationary blocks and Cases 2–4 remain unchanged, as shown below. For the F10 F11 pair, there are no stationary blocks. Therefore, FS VBSME has to be conducted for the entire frame.

Case 1: If $Sum\_MV(p, q) = (0, 0)$, we set $Bsiz = B\_Size$, $(U, V) = (1, 1)$.

The block ranges from $(B\_Size)(p-1)+U$ to $(B\_Size)(p)$ along the rows $(B\_Size)(q-1)+V$ to $(B\_Size)(q)$ along the columns.

Case 2: If $Sum\_MV(p,q) \neq (0,0)$ and $Bsiz = B\_Size$, $(U,V) = (1,1)$.

The block ranges from $(B\_Size)(p-1)+U$ to $(B\_Size)(p)$ along the rows $(B\_Size)(q-1)+V$ to $(B\_Size)(q)$ along the columns.

Case 3: If $Sum\_MV(p,q) \neq (0,0)$ and $B\_size = 8$, $(U,V) = (1,1),(1,2),(2,1),(2,2)$.

The sub-block ranges from $(Bsiz)(p-1)+(U-1)(B\_Size)+1$ to $(Bsiz)(p-1)+(U)(B\_Size)$ along the rows and $(Bsiz)(q-1)+(V-1)(B\_Size)+1$ to $(Bsiz)(q-1)+(V)(B\_Size)$ along the columns.

Case 4: If $Sum\_MV(p,q) \neq (0,0)$ and $B\_size = 4$, $(U,V) = (1,1)$, $(1,2),(1,3),(1,4),(2,1),(2,2),(2,3),(2,4),(3,1),(3,2),(3,3),(3,4),(4,1),(4,2),(4,3),(4,4)$.

The sub-block ranges from $(Bsiz)(p-1)+(U-1)(B\_Size)+1$ to $(Bsiz)(p-1)+(U)(B\_Size)$ along the rows and $(Bsiz)(q-1)+(V-1)(B\_Size)+1$ to $(Bsiz)(q-1)+(V)(B\_Size)$ along the columns.

Step 6: Frame pair F11 F12 is loaded and the MV for blocks in F12 is found after evaluating F11 MVs. If F11 MVs for all stationary blocks converted to non-stationary blocks in the previous step is $(0,0)$, then it is again declared a stationary block as in case 1. If it's not $(0,0)$, then we evaluate the F11 MVs based on Cases 2–4 to decide the block sizes. The rest of the cases are defined for the remaining blocks.

Case 1: If $Sum\_MV(p,q) = 0$ and $MV(p,q) = 0$, the block at position (p,q) is declared a stationary block again. The MV is considered $(0,0)$ for these blocks and no further processing is done.

Case 2: If $Sum\_MV(p,q) = 0$ and $MV(p,q) \leq 2$, the block at position (p,q) is declared to have very slight movement between frames. Therefore, a block size of 16×16 is considered.

Case 3: If $Sum\_MV(p,q) = 0$ and $2 < MV(p,q) \leq 4$, the block at position (p,q) is declared to have considerable movement. Therefore, a block size of 8×8 is considered. In this case, a 16×16 macroblock is divided into four sub-blocks of size 8×8 each.

Case 4: If $Sum\_MV(p,q) = 0$ and $MV(p,q) > 4$, the block at position (p,q) is declared to have higher displacement between frames. Therefore, a block size of 4×4 is considered. In this case a 16×16 macroblock is divided into 16 sub-blocks of size 4×4 each.

Case 5: If $Sum\_MV(p,q) \leq 2$, a block size of 16×16 is considered for these blocks.

Case 6: If $2 < Sum\_MV(p,q) \leq 4$, a 16×16 macroblock is divided into four sub-blocks of size 8×8 each.

Case 7: If $Sum\_MV(p,q) > 4$, a 16×16 macroblock is divided into 16 sub-blocks of size 4×4 each.

Step 7: FS VBSME for frame pair F11 F12 to F14 F15 is conducted. From this point, for every fifth frame the stationary frames are revalidated.

Step 8: Step 5 to Step 7 is repeated for different frame pairs until last frame pair.

Step 9: Total number of calculations due to SAD are finally found.

Step 10: The PSNR is evaluated based on the original and reconstructed frames.

For every fifth frame pair F10 F11, F15 F16, F20 F21, F25 F26...FN 5 FN 4, the stationary blocks are changed to non-stationary blocks with block size 16. The F5 F6 pair is not considered as there is just one frame difference after finding the initial stationary blocks. For frame pairs F11 F12, F16 F17, F21 F22, F26 F27...FN 4 FN 3, revalidation of the MVs is done to test if the blocks are still stationary. This way, though the number of calculations as compared to interframe prediction based FS VBSME is slightly higher, we can observe improved PSNR for most of the CIF sequences.

3.4 Stationary Block Revalidation Based Vector-driven VBSME

Like in the stationary block revalidation based FS VBSME algorithm, the interframe prediction is used for dividing the frames into stationary and non-stationary blocks. The non-stationary blocks are further divided into sub-blocks based on the predicted MV for the MB. For a candidate block, search positions in the direction opposite to the previous displacement vector are eliminated. The least probable search positions are thus eliminated. Based on the fact that earlier identified stationary blocks may not remain stationary across the frames, periodic revalidation of the stationary block is done. The steps below explain the algorithm.

Step 1: Perform FSBME for a fixed block size of 16 for F1 F2, F2 F3, F3 F4. Find $Sum\_MV(p,q)$.

Step 2: If $Sum\_MV(p,q) = 0$, declare corresponding blocks to be stationary until further changes are made. If $Sum\_MV(p,q) \neq 0$, divide the macroblocks into sub-blocks of sizes as already explained in stationary block revalidation based FS VBSME.

Step 3: For F4 F5 pair, perform FS VBSME to obtain the MV for F5.

Step 4: Load the F5 F6 pair. Evaluate the MVs of F5 to find the direction of displacement for the varied block sizes as explained in the vector-driven VBSME. Vector-driven VBSME starts from here, thus eliminating least probable search positions. Conduct vector-driven VBSME for frames pairs F5 F6–F9 F10.

Step 5 and *Step 6* are exactly the same as that of stationary block revalidation based FS VBSME.

Step 7: Vector-driven VBSME is conducted from F11 F12–F14 F15.

Step 8: Step 3 to Step 8 is repeated for different frame pairs till ME is complete for all the frames.

Step 9: Find the number of calculations due to SAD.

Step 10: Find the PSNR based on the original and reconstructed frames.

FS FSBME is conducted for F4 F5, F10 F11, F15 F16, F20 F21,...FN 5 FN 4. Based on the immediate previous MVs, vector-driven VBSME is conducted for frames F5 F6–F9 F10, F11 F12–F14 F15, F16 F17–F19 F20, F21 F22–F24 F25, FN 4 FN3–FN1 FN, once the revalidation of the stationary blocks is done.

The revalidation technique for the stationary block proves to improve the PSNR with slight increase in the number of computations.

4 Experimental Results and Comparative Analysis

A set of 12 CIF and 3 1080p resolution sequences were evaluated for all the four proposed algorithms for 30 frames. A search size of 4 is used. Comparison between the algorithms is made based on the PSNR, total number of calculations due to SAD and percentage reduction in the number of calculations.

Figure 5 shows the total number of calculations across all the proposed algorithms along with the standard FSBME algorithm. Figure 6 shows the percentage of calculations reduced for three pairs of algorithms. For FSBME,

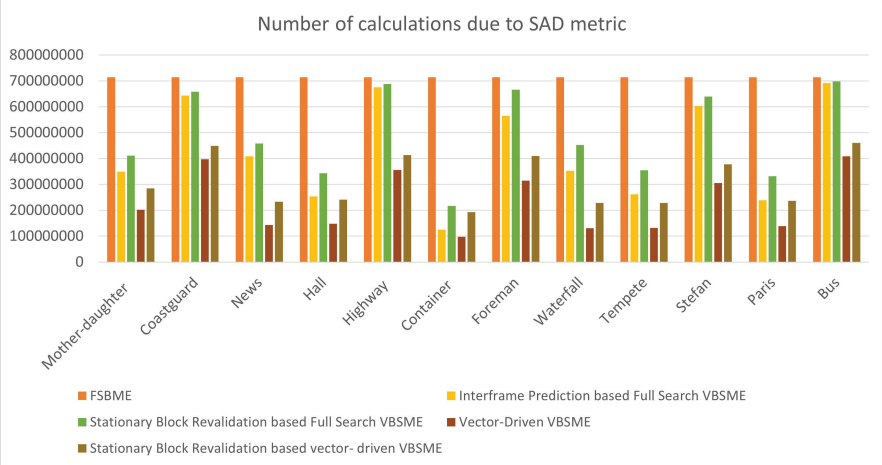

Figure 5 Total calculations due to SAD for all the proposed algorithms.

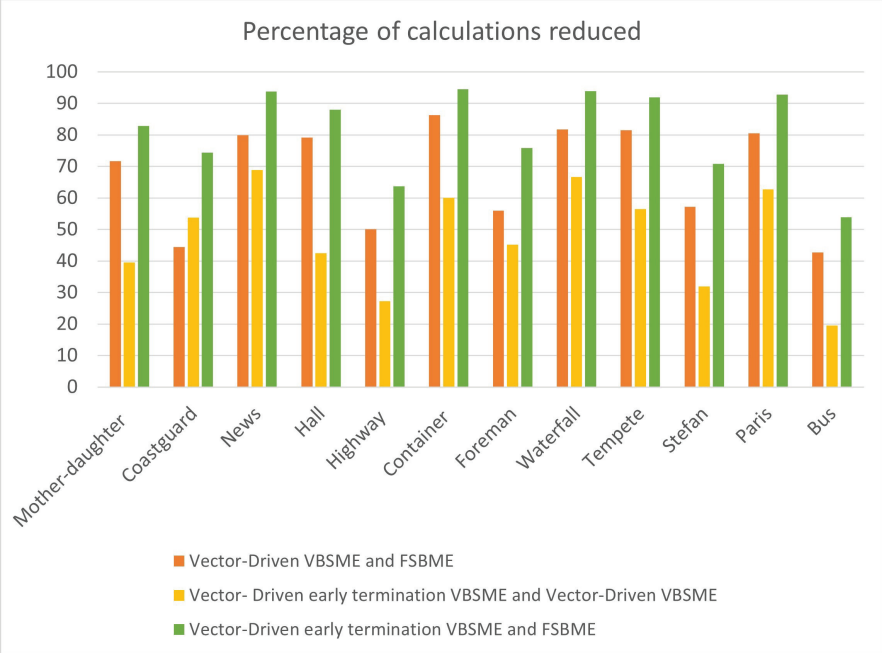

Figure 6 Percentage of calculations reduced due to SAD across the algorithms, FSBME, and vector-driven VBSME with and without early termination.

Table 1 Total calculations due to SAD for all the proposed algorithms for 1080 × 1920 sequences

1080p Sequences	FSBME 16×16	Interframe Prediction Based FS	Stationary Block Revalidation Based FS VBSME	Vector-driven VBSME	Stationary block Revalidation Based Vector-driven VBSME
ducks_take_off_1080p50	1.4721×10^{10}	1.4046×10^{10}	1.4392×10^{10}	7.8863×10^{9}	9.0828×10^{9}
life_1080p30	1.4721×10^{10}	7.3363×10^{9}	9.2418×10^{9}	4.1336×10^{9}	6.1032×10^{9}
park_joy_1080p50	1.4721×10^{10}	1.4718×10^{10}	1.4718×10^{10}	8.9493×10^{9}	9.8812×10^{9}

the number of calculations is fixed and these calculations are compared with other proposed algorithms. When the vector-driven VBSME is compared with FSBME, the overall percentage of calculations reduced for 12 CIF sequences range from 42.70% to 86.32%. A percentage reduction of 27.22% to 68.87% is attained when vector-driven VBSME with and without early termination techniques is compared. 53.87% to 94.54% of calculations is reduced on comparing vector-driven VBSME with early termination and FSBME. It can be observed that number of calculations is minimum for vector-driven VBSME with an early termination technique.

To further validate the scalability of the proposed algorithms, experiments were extended to high-resolution 1080p sequences, namely ducks_take_off_1080p50, life_1080p30, and park_joy_1080p50. Table 1 highlights the computational advantage of the proposed techniques at 1080p resolution. The stationary block revalidation based vector-driven VBSME achieves the largest reduction, lowering the number of SAD calculations from 1.47×10^{10} (FSBME) to 6.10×10^{9} for life_1080p30, corresponding to nearly 60% savings. A similar trend is seen for ducks_take_off_1080p50 and park_joy_1080p50, where reductions of more than 35–45% are achieved compared to conventional FSBME. The results demonstrate that out proposed algorithms reduce the computational overhead to a greater extent for different resolutions.

The PSNR for the proposed algorithms across 12 CIF resolution datasets are recorded as shown in Table 2. In the stationary block revalidation based FS VBSME and stationary block revalidation based vector-driven VBSME algorithms, due to the addition feature of stationary revalidation, there is an improvement in the PSNR (dB). The stationary block revalidation based FS VBSME algorithm achieves 0–1.39 dB improvement in the PSNR as compared to the interframe prediction based FS VBSME algorithm. The stationary block revalidation based vector-driven VBSME algorithm achieves 0–1.32 dB increase in the PSNR. Experiments conducted on high-resolution 1080p sequences, namely ducks_take_off_1080p50, life_1080p30,

Table 2 Average PSNR (dB) comparison for CIF sequences

CIF Sequences	FSBME 16×16	Interframe Prediction Based FS	Stationary Block Revalidation Based FS VBSME	Vector-driven VBSME	Stationary Block Revalidation Based Vector-driven VBSME
Mother-daughter	42.36	42.59	42.74	42.51	42.67
Coastguard	30.80	31.16	31.24	31.13	31.17
News	38.51	38.82	38.97	38.65	38.78
Hall	35.53	34.98	35.72	34.92	35.37
Highway	35.16	36.32	36.32	35.71	35.80
Container	38.38	38.39	38.39	38.36	38.38
Foreman	33.28	32.81	34.20	32.51	33.83
Waterfall	35.39	35.43	35.48	35.42	35.46
Tempete	26.23	25.36	26.15	25.75	25.97
Stefan	23.39	25.74	26.05	24.94	25.30
Paris	31.31	31.76	31.73	30.61	31.57
Bus	19.54	22.13	22.19	21.38	21.57

Table 3 Average PSNR (dB) comparison for 1080×1920 sequences

1080p Sequences	FSBME 16×16	Interframe Prediction Based FS	Stationary Block Revalidation Based FS VBSME	Vector-driven VBSME	Stationary Block Revalidation Based Vector-driven VBSME
ducks_take_off_1080p50	29.06	30.09	30.13	29.51	29.98
life_1080p30	32.53	29.59	32.36	28.60	31.44
park_joy_1080p50	23.24	23.77	23.77	23.24	23.33

and park_joy_1080p50, for the proposed methods maintain PSNR levels comparable to FSBME. The average PSNR values for these sequences are summarized in Table 3. It can be observed that, in ducks_take_off_1080p50, the stationary block revalidation based FS VBSME achieves a PSNR of 30.13 dB against 29.06 dB for FSBME, reflecting a noticeable quality improvement. Similarly, park_joy_1080p50 records consistent performance across all proposed methods, with a maximum gain of about 0.5 dB. Although life_1080p30 shows slight variations across algorithms, the proposed revalidation and adaptive schemes continue to provide reliable reconstruction quality.

Figure 7 presents a comparative analysis of original and reconstructed frames for all four proposed algorithms at frames 5, 14, and 24 (randomly selected), showing stationary blocks that were excluded from reconstruction. Red rectangles indicate blocks that deviated from their initial stationary classification in revalidation based algorithms.

Figures 8 and 9 show that there is no visible difference between the original frames and the reconstructed frames for all the proposed ME algorithms. The reconstructed Frame 4, Frame 11 and Frame 22 (randomly selected) of the CIF NEWS sequence is shown in Figure 8 and that of life_1080p30 sequence in Figure 9. It can be observed that the smallest details in the frames are also efficiently reconstructed.

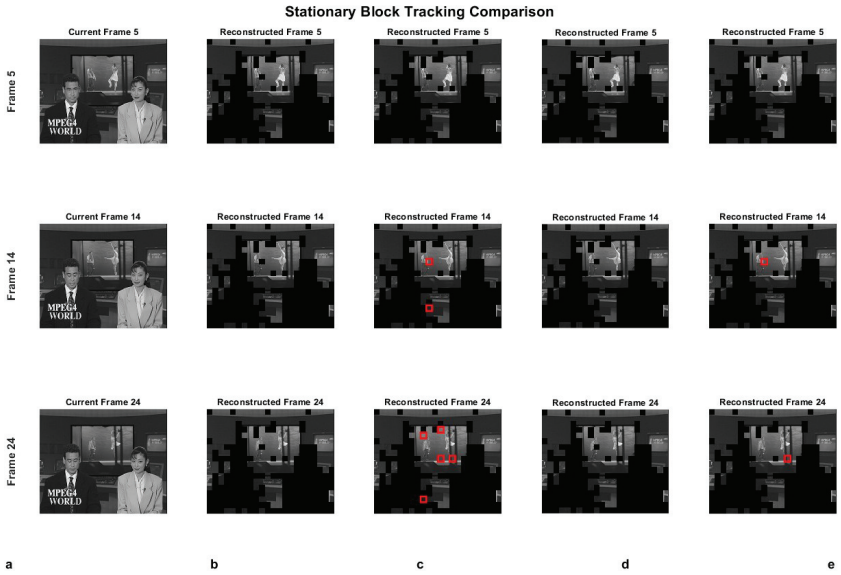

Figure 7 Stationary block analysis in VBSME-based video frame reconstruction. (a) Original video frames, (b) interframe prediction based FS VBSME, (c) vector-driven VBSME, (d) stationary block revalidation based FS VBSME, (e) stationary block revalidation based vector-driven VBSME.

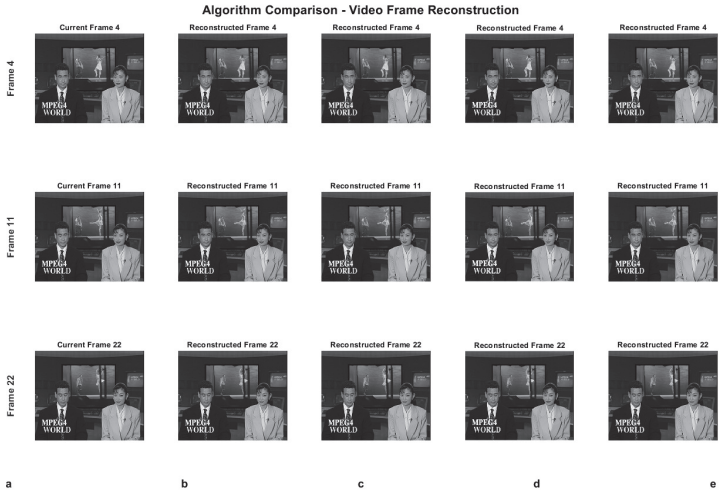

Figure 8 (a) Original video frames, (b) frames reconstructed from interframe prediction based FS VBSME, (c) frames reconstructed from vector-driven VBSME, (d) frames reconstructed from stationary block revalidation based FS VBSME, (e) frames reconstructed from stationary block revalidation based vector-driven VBSME.

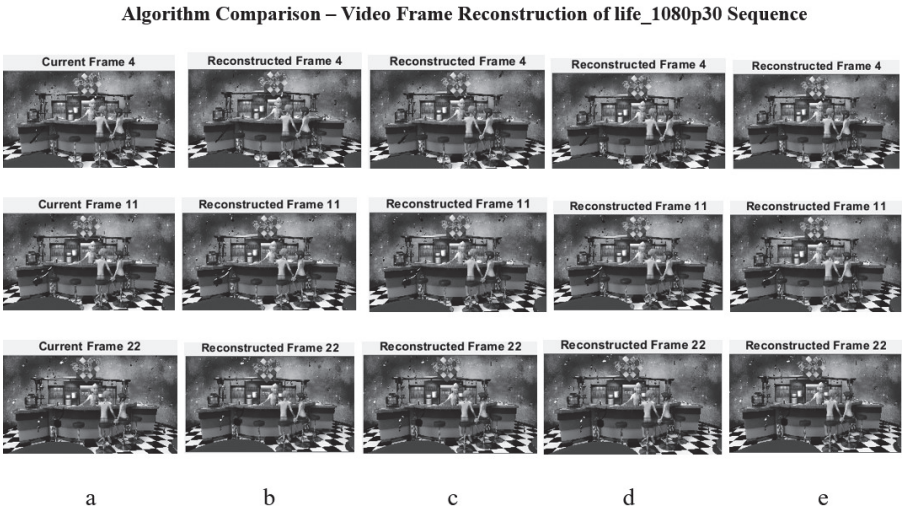

Figure 9 (a) Original video frames, (b) frames reconstructed from interframe prediction based FS VBSME, (c) frames reconstructed from vector-driven VBSME, (d) frames reconstructed from stationary block revalidation based FS VBSME, (e) frames reconstructed from stationary block revalidation based vector-driven VBSME.

These results confirm that the proposed algorithms scale effectively to higher resolutions, retaining near-FSBME quality while offering significant reductions in computational load. This shows their suitability for modern video content, where high-definition formats dominate.

Recent studies [23] have reported hardware-level full search implementations for high-resolution video, evaluated mainly in terms of resource usage and processing throughput. In contrast, the present work focuses on algorithmic enhancements, where performance is assessed using PSNR and the number of SAD computations. The proposed methods achieve significant reductions in computation while maintaining near-FSBME quality. These algorithm-level improvements are complementary to hardware-centric approaches and can be integrated into such frameworks in future.

5 Discussion

The proposed vector-driven VBSME algorithm demonstrates substantial computational efficiency across high-resolution video sequences. For the three 1080p sequences considered, the number of SAD calculations is reduced by 39.2–71.9%, with an average reduction of approximately 52%,

and the maximum reduction of 71.9% observed for the life_1080p30 sequence. These savings indicate that the algorithm significantly lowers the computational load compared to conventional FSBME. Despite this reduction, the reconstructed frames show no perceptible difference from the original video (Figure 9), demonstrating that visual quality is well maintained. This confirms that the proposed algorithm achieves an effective balance between computational efficiency and reconstruction quality.

The algorithm is specifically designed to support parallelization, allowing multiple motion estimation computations to be performed concurrently. This feature makes it particularly suitable for hardware platforms such as FPGA or GPU where efficient parallel execution can further enhance processing speed.

The current evaluation is performed in Matlab R2024a, which establishes the algorithmic benefits; however, real-time performance and memory utilization can only be fully assessed through hardware implementation. With proper parallelization and memory strategies, the method shows strong potential for deployment in practical high-resolution video processing systems.

6 Conclusions

Our proposed algorithms, vector-driven VBSME and stationary block revalidation based vector-driven VBSME reduce the computational overhead more effectively than FS VBSME algorithms, interframe prediction based FS VBSME, and stationary block revalidation based FS VBSME. Vector-driven VBSME with early termination achieves the maximum reduction in the number of calculations, while the stationary block revalidation algorithms improve PSNR values compared to interframe prediction based FS VBSME and vector-driven VBSME for CIF sequences. This work demonstrates significant improvements in computational efficiency while maintaining or improving video quality metrics. The adaptive block size selection based on motion characteristics provides an effective approach to optimizing motion estimation algorithms for practical video compression applications.

While the proposed algorithms show substantial computational savings and quality preservation, further improvements could be achieved by considering both spatial and temporal neighboring motion vectors for adaptive block size selection, and by incorporating more temporal frame pairs depending on application requirements. From a hardware perspective, using both spatial and temporal neighbors may increase memory requirements and implementation complexity, which should be carefully managed in FPGA or GPU designs.

7 Future Work

The revalidation process in the current algorithm is performed only for stationary blocks to improve the PSNR. As the dynamics in the video change with time, the revalidation technique can also be extended to block size revision for different segments of the frame, similar to how stationary block revalidation is done at regular frame intervals. This may prove to improve PSNR further. Additionally, incorporating illumination changes and developing adaptive revalidation intervals based on video content characteristics could enhance the algorithm's performance across diverse video sequences.

Acknowledgements

The authors thank the reviewers for their valuable comments and suggestions that helped improve the quality of this article.

References

[1] Richardson, I.E., *The H.264 Advanced Video Compression Standard* (Wiley Publishing, New York, 2010).

[2] G. J. Sullivan and T. Wiegand, "Video Compression – From Concepts to the H.264/AVC Standard", *Proceedings of the IEEE* **93**, 18–31 (2005).

[3] Dong, X., Wen, J., "A pixel-based outlier-free motion estimation algorithm for scalable video quality enhancement", *Front. Comput. Sci.* **9**, 729–740 (2015).

[4] Xuan Jing and Lap-Pui Chau, "An efficient three-step search algorithm for block motion estimation", *IEEE Transactions on Multimedia* **6**, 435–438 (2004).

[5] Reoxiang Li, Bing Zeng and M. L. Liou, "A new three-step search algorithm for block motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology* **4**, 438–442 (1994).

[6] Lai-Man Po and Wing-Chung Ma, "A novel four-step search algorithm for fast block motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology* **6**, 313–317 (1996).

[7] M. Ghanbari, "The cross-search algorithm for motion estimation (image coding)", *IEEE Transactions on Communications* **38**, 950–953 (1990).

[8] Chun-Ho Cheung and Lai-Man Po, "A novel cross-diamond search algorithm for fast block motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology* **12**, 1168–1177 (2002).

[9] Shan Zhu and Kai-Kuang Ma, "A new diamond search algorithm for fast block-matching motion estimation", *IEEE Transactions on Image Processing* **9**, 287–290 (2000).
[10] Woong IL Choi, Byeungwoo Jeon and Jechang Jeong, "Fast motion estimation with modified diamond search for variable motion block sizes", *Proceedings 2003 International Conference on Image Processing* (2003), pp. II-371.
[11] Ce Zhu, Xiao Lin and Lap-Pui Chau, "Hexagon-based search pattern for fast block motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology* **12**, 349–355 (2002).
[12] I. Ahmad, Weiguo Zheng, Jiancong Luo and Ming Liou, "A fast adaptive motion estimation algorithm", *IEEE Transactions on Circuits and Systems for Video Technology* **16**, 420–438 (2006).
[13] S. Goel, Y. Ismail and M. A. Bayoumi, "Adaptive search window size algorithm for fast motion estimation in H.264/AVC standard", *48th Midwest Symposium on Circuits and Systems* (2005), pp. 1557–1560.
[14] Yao Nie and Kai-Kuang Ma, "Adaptive rood pattern search for fast block-matching motion estimation", *IEEE Transactions on Image Processing* **11**, 1442–1449 (2002).
[15] W. Li and E. Salari, "Successive elimination algorithm for motion estimation", *IEEE Transactions on Image Processing* **4**, 105–107 (1995).
[16] X. Q. Gao, C. J. Duanmu and C. R. Zou, "A multilevel successive elimination algorithm for block matching motion estimation", *IEEE Transactions on Image Processing* **9**, 501–504 (2000).
[17] Lurng-Kuo Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding", *IEEE Transactions on Circuits and Systems for Video Technology* **6**, 419–422 (1996).
[18] M. Brunig and W. Niehsen, "Fast full-search block matching", *IEEE Transactions on Circuits and Systems for Video Technology* **11**, 241–247 (2001).
[19] Ahn, T.G., Moon, Y.H., Kim, J.H., "Fast full-search motion estimation based on multilevel successive elimination algorithm", *IEEE Trans. Circuits Syst. Video Technol.* **14**, 1265–1269 (2004).
[20] G.-L. Li and M.-J. Chen, "Fast Motion Estimation Algorithm by Finite-State Side Match for H.264 Video Coding Standard", *APCCAS 2006–2006 IEEE Asia Pacific Conference on Circuits and Systems* (2006), pp. 414–417.

[21] Han, K.H., Lee, Y.L., "Fast macro block mode decision to reduce H.264 complexity", *IEICE Trans. Fundamentals* **E88-A**, 800–804 (2005).
[22] Basha, S.M., Kannan, M., "Novel architecture and implementation of low power oriented full search block motion estimation", *Cluster Comput* **22**, 4503–4509 (2019).
[23] Ahmad, W., Mahdavi, H. & Hamzaoglu, I. An efficient versatile video coding motion estimation hardware. J Real-Time Image Proc 21, 25 (2024). https://doi.org/10.1007/s11554-023-01402-8.
[24] Nguyen, V.T. Motion free B-frame coding for neural video compression. arXiv preprint arXiv:2411.17160 (2024).
[25] https://media.xiph.org/video/derf/.

Biographies

M. Vinutha received her bachelor's degree in Electronics and Communication Engineering from Visvesvaraya Technological University, Karnataka, India, in 2012. She obtained her M.Tech. degree in Digital Electronics and Communication Systems from Visvesvaraya Technological University in 2014. She is currently pursuing her Ph.D. under Visvesvaraya Technological University, Belagavi, with KLE Institute of Technology (KLEIT), Hubli, as her recognized research center. She is working as an Assistant Professor in the Department of VLSI at NMAM Institute of Technology, Nitte. Her research interests include video compression, motion estimation algorithms, and VLSI design for multimedia applications.

T. M. Manu received his bachelor of Electronics and Communication Engineering from B.D.T. College of Engineering, Davanagere. He obtained his masters from B.V.Bhoomreddy College of Enggg., Hubli. He obtained his Doctorate from Visvesvaraya Technological University, Belgaum. He is actively involved in grooming young researchers for achieving significant contribution in their field of interest. He is an author of a great deal of research studies published at national and international journals as well as conference proceedings. His research interests include digital signal processing, video compression, and embedded systems.