DOI: 10.19884/j.1672-5220.202512016

Motion-Texture Joint Threshold-Adaptive Skipping for Distributed Video Compressive Sensing

SU Jinhao[1], LIU Hao[1,2*], HUANG Rong[1,2]
1. School of Information and Intelligent Science, Donghua University, Shanghai 201620, China
2. Engineering Research Center of Digitized Textile & Fashion Technology, Ministry of Education, Donghua University, Shanghai 201620, China

Abstract: The traditional distributed video compressive sensing (DVCS) system effectively reduces transmission costs and encoder complexity for wireless visual sensor networks (WVSN), but struggles with dynamic scenes involving rapid motion or complex textures. Their reliance on temporal correlation and skipping causes reconstruction artifacts and loss of structural details, especially at high skip ratios. This paper proposes the motion-texture joint (MTJ) algorithm to enhance DVCS performance, where the motion module first preliminarily screens those candidate skip-blocks using the sum of absolute differences (SAD) and then generates more robust side information by combining frame interpolation with motion vectors for redundant-content identification and skip-block decision. Further, a texture feature module based on local binary pattern (LBP) analysis jointly exploits LBP texture features, variance textures, and gradient energy textures from adjacent frames, which improves inter-frame correlations. On this basis, an MTJ adaptive weight module dynamically allocates coefficients to motion information and texture information according to their spatio-temporal contributions and controls a skip ratio of DVCS. Experiments on standard video sequences demonstrate that the MTJ algorithm achieves an improvement in average peak signal-to-noise ratio (PSNR) of 0.12 dB over state-of-the-art uniform-reference threshold-dynamic skipping (UTS) with minimal encoder complexity increase. Especially at a 70% skip ratio, the maximum PSNR improvement over UTS reaches up to 0.20 dB on the Foreman sequence. MTJ algorithm effectively balances reconstruction quality and computational complexity, making it suitable for WVSN applications.

Keywords: distributed video compressive sensing (DVCS); skip ratio; motion estimation; local binary pattern (LBP) texture feature; adaptive weight
CLC number: TN919.8 **Document code**: A
Article ID: 1672-5220(2026)04-0067-16

0 Introduction

The proliferation of wireless communication technologies and the vision of the Internet of Things have led to an increase in mobile monitoring applications, where typical front-end equipment possesses limited processing capabilities. Distributed video coding (DVC), which independently compresses video frames by processing solely within individual frames and without requiring inter-frame correlations, offers a relatively low encoding complexity. While this approach circumvents complex inter-frame dependencies, it remains constrained by the Nyquist theorem. Compressive sensing (CS) overcomes this limitation by surpassing the Nyquist sampling rate, effectively addressing the aforementioned challenges[1]. The basic idea of CS is to leverage the sparsity or compressibility of signals. By acquiring a small amount of data, high-quality images can be reconstructed through iterative optimization of a series of nonlinear reconstruction algorithms under certain constraints. Since CS enables signal compression simultaneously with sampling, it significantly reduces the complexity of data processing and storage requirements. Thus, its superior performance in signal acquisition and transmission has facilitated its adoption in imaging systems. CS also significantly enhances the energy efficiency of an encoder, where a subrate is defined as the ratio of the number of measurements to the signal dimension. Zeng et al.[2] proposed a block compressive sensing (BCS) solution that involved grouping video sequences and partitioning images into blocks. Grouping refers to dividing the video sequence into multiple groups of pictures (GOP) with the same number of frames, while block partitioning entails decomposing each image into non-overlapping blocks of uniform size. Each frame is measured block by block, which corresponds to the strategy of sparse sampling performed in small batches.

In recent years, with the rapid advancement of artificial intelligence, deep learning has emerged as a research hotspot in the field of video processing, bringing new opportunities for optimizing the GOP-BCS system. By leveraging the powerful capabilities of deep learning

Received date: 2025-12-26
Foundation items: National Natural Science Foundation of China (Nos. 62372100 and 62571110)
* Correspondence should be addressed to LIU Hao, email: liuhao@dhu.edu.cn
Citation: Su J H, Liu H, Huang R. Motion-texture joint threshold-adaptive skipping for distributed video compressive sensing[J]. *Journal of Donghua University (English Edition)*, 2026, 43(4): 67-82.

in automatic feature learning and complex correlation modeling, researchers have explored deep learning-based approaches to enhance the reconstruction quality of video sequences. Building on temporal dependencies, Unde et al.[3] proposed a long short-term memory (LSTM) enhancement framework that utilized temporal correlations between adjacent high-resolution frames to refine the reconstruction quality of compressed video content through cross-frame feature propagation. To complement these advancements, Guan et al.[4] introduced an integrated sampling-reconstruction framework employing convolutional neural network architectures to simultaneously optimize both spatial and temporal data correlations. This end-to-end system achieved improved performance by co-training measurement acquisition and signal recovery components through a unified learning process. Chen et al.[5] developed a hybrid architecture named CSVideoNet, which synergistically combined the capabilities of convolutional neural networks and LSTM modules to facilitate effective reconstruction of video sequences. Since deep learning undermines the generality of the GOP-BCS decoder, it results in limitations for resource-constrained application scenarios.

Under the GOP-BCS framework, Song et al.[6] combined DVC with CS to propose a distributed video compressive sensing (DVCS) system, which has become the mainstream system in the field of mobile monitoring. To enhance DVCS performance, Zhang et al.[7] proposed a method that categorizes intra-frame blocks in non-keyframes into static and motion blocks based on temporal correlation, assigning different measurement rates to each category to optimize the information capture efficiency of compressive sensing. Song et al.[8] introduced a skip-block coding approach that adapts to video sequences with varying motion speeds by applying different skip ratios to non-keyframes. Furthermore, the reweighted residual sparsity (RRS) algorithm, as described in Ref. [9], leveraged spatial correlation within video frames to achieve good initial reconstruction results and improved the reconstruction quality by exploiting temporal correlation. Yuan et al.[10] proposed the motion-adaptive adjacent-reference skipping (MAS) algorithm. This MAS algorithm reduces errors caused by the increased distance between non-keyframes and keyframes by using adjacent frames as reference frames. Meanwhile, different skip ratios are allocated to non-keyframes according to their motion speeds. Priya et al.[11] proposed the high-reduction secondary compression (HSC) algorithm. Based on the correlation of observation vectors, the HSC algorithm achieves secondary compression without the need for a feedback channel, featuring low encoder complexity. However, it underutilizes keyframe information, leaving room for optimization. Moreover, the decoder reconstruction relies on iterative optimization, leading to relatively high decoder complexity. Liu et al.[12] proposed the uniform-reference threshold-dynamic skipping (UTS) algorithm. The UTS algorithm improves the accuracy of skip-ratio allocation by designating prime non-keyframes among non-keyframes as references. However, at low subrates, the measurement vectors fail to accurately judge image similarity, leading to poor performance. UTS algorithm mainly focuses on the impact of adaptive subrate on the reconstruction quality of video sequences, and its skip ratios lack consistency when compared with those of other algorithms. This study primarily investigates the scenario with a unified subrate and improves the reconstruction quality by accurately allocating skip ratios, thus modifying the UTS algorithm.

However, these existing algorithms primarily focus on temporal continuity and fail to fully exploit spatio-temporal correlations. By relying predominantly on information from keyframes, the reconstruction quality of non-keyframes degrades as their distance from keyframes increases, leading to noticeable fluctuations in the reconstruction quality across the video sequence[13]. Moreover, neglecting inter-frame relationships limits the algorithm's ability to accurately characterize both the motion characteristics and textures of video sequences. Specifically, when there are changes in static textures such as illumination and shadows in video sequences, motion variables may show little variation, but the actual content of the video has already changed. This kind of content change is often overlooked due to insufficient exploration of inter-frame content correlations. Although these algorithms can process low-sampling-rate data under resource-constrained conditions and achieve relatively good reconstruction quality, they face challenges in accurately predicting fast motion. Moreover, the complexity of their decoders prevents them from meeting real-time processing requirements. This can lead to detail loss and blocking artifacts, ultimately limiting reconstruction performance. To address these limitations, this paper proposes a novel algorithm that comprehensively considers inter-frame correlations to further improve reconstruction quality based on spatio-temporal dependencies. In summary, the key contributions of this paper are as follows.

1) A motion estimation module is designed for the DVCS system. This module first uses the sum of absolute differences (SAD) to preliminarily screen those candidate skip-blocks, and then generates side information by combining motion vectors with frame interpolation from adjacent keyframes, and further computes the side-information error for more refined skip-block decisions. By adopting the coarse-to-fine strategy, this module can more effectively identify redundant content and improve video reconstruction quality.

2) An LBP texture feature module is incorporated into the encoder. This module exploits LBP texture features, variance texture features, and gradient energy texture features to enhance inter-frame correlations and leverage frame similarities to improve reconstruction

quality.

3) A joint motion-texture adaptive weight module conducts spatio-temporal decomposition of input video sequences to establish the proportional relationship between motion energy distribution and textural information density. By formulating these attributes through the proportional relationship, the system derives the joint weighted coefficient C_{jw} that governs frame-wise measurement retention thresholds. This optimization mechanism improves reconstruction quality through intelligent redistribution of skip-ratio resources across temporal frames based on scene complexity characteristics.

1 Methods

1.1 DVCS system with skipping

Figure 1 depicts the architecture of the DVCS system with skipping under a general RRS decoder, with the shaded region highlighting the primary focus of this research. The skip-block algorithm partitions a frame into smaller blocks, analyzes the similarity between them, and designates them as either skip-blocks or non-skip-blocks[14].

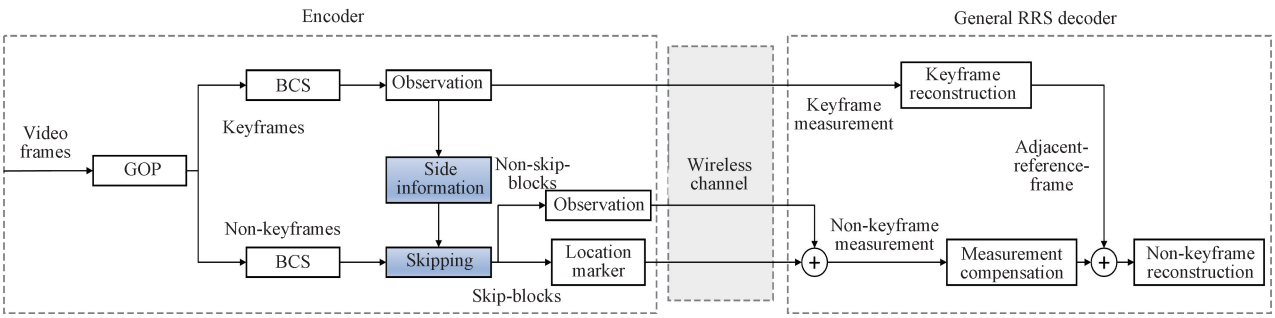


Fig. 1 Architecture of DVCS system with skipping under a general RRS decoder

Location markers are then used to convey block information. The measurement compensation module is employed for the reconstruction of skip-blocks. Existing algorithms utilize keyframes as reference frames, mitigating the issue of significantly reduced inter-frame correlations and subsequent performance degradation that arises when relying solely on keyframes as references, especially as the temporal distance between keyframes and non-keyframes increases[15].

However, to ensure seamless integration with a universal decoder in the DVCS system, these algorithms often rely on simple SAD calculations to assess motion between frames[16]. This approach can lead to inaccurate similarity measurements and suboptimal skip-block decisions, particularly when dealing with complex motion patterns.

This adaptive skip-block algorithm, characterized by low encoding complexity, an efficient dual-stage compression strategy, and substantial transmission cost optimization, has become a preferred solution for resource-constrained wireless visual sensor networks[17]. However, its applicability to real-time high-precision video systems remains limited due to inherent constraints in dynamic scene adaptation and reconstruction fidelity. In operational environments such as security surveillance or adaptive streaming scenarios—where video sequences frequently exhibit rapid motion patterns or intricate texture variations—the algorithm demonstrates critical shortcomings. While its iterative multi-hypothesis reconstruction framework effectively stabilizes low-sampling-rate data processing, this approach introduces motion blur artifacts during fast-moving sequences and imposes computational demands during decoding that challenge real-time operational requirements. Moreover, the algorithm's underutilization of spatial texture features often results in the loss of fine-grained texture details, a limitation that may degrade performance in target recognition tasks for surveillance applications. These combined factors constrain its deployment in precision-sensitive video systems requiring both temporal responsiveness and spatial accuracy. Consequently, frames exhibiting rapid motion may be excessively skipped, resulting in detail loss and heightened errors, whereas frames with slow motion may be insufficiently skipped, leading to the transmission of redundant information and reduced compression efficiency[18].

1.2 Proposed algorithm

To address these limitations, this paper proposes a novel motion-texture joint (MTJ) algorithm that incorporates three new modules into the conventional DVCS framework. Figure 2 presents the module diagram of the MTJ algorithm. The mathematical symbols defined therein are summarized in Table 1.

The first module is a motion estimation module. In reference to the side information functional unit at the decoding end, and to ensure the algorithm's compatibility with a universal decoder, the information produced by this module is used exclusively during the encoding stage. Specifically, motion information is employed to facilitate a more accurate skip-block decision.

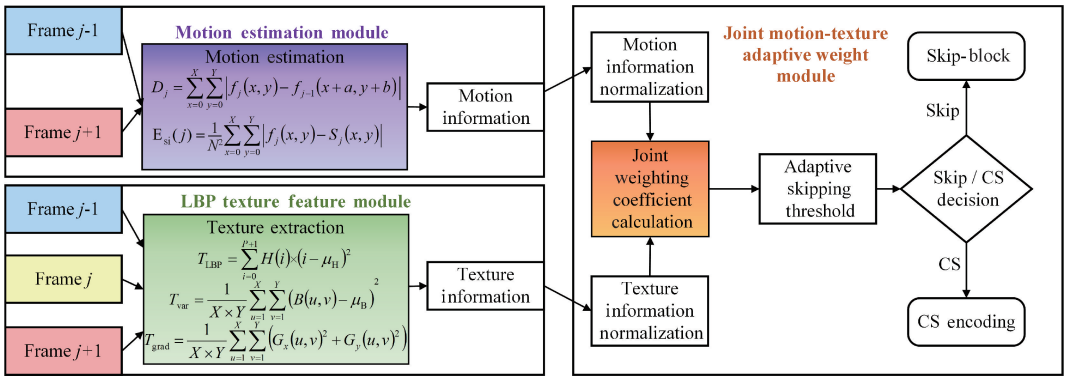


Fig. 2 Module diagram of the proposed MTJ algorithm

Table 1 Mathematical symbols and definitions

Symbol	Definition
i	Block number
j	Frame number
$X \times Y$	Block size
$f_j(x, y)$	Current pixel value
$f_{j-1}(x+a, y+b)$	Pixel value of the decoded keyframe
a, b	Relative displacements
P	Number of neighboring pixels
T_{var}	Variance texture features
T_{grad}	Gradient energy texture features
T_{LBP}	LBP texture features
D_j	SAD of adjacent frames
$H(i)$	Frequency count for LBP
μ_{H}	Mean of the LBP histogram
$B(u,v)$	Gray intensity value at pixel (u,v)
$G_x(u,v)$	Gray-level gradient in x-direction at pixel (u,v)
$G_y(u,v)$	Gray-level gradient in y-direction at pixel (u,v)
μ_{B}	Mean gray intensity of the block
S_j	Side-information frame
$E_{\text{si}}(j)$	Side-information error metric

Frame interpolation (FI) and motion estimation (ME) are well-established side information generation algorithms[19]. FI generates side information by calculating a weighted average of the previous and subsequent keyframes, while ME determines motion trajectories by calculating the motion displacement of each pixel between two adjacent frames in the video sequence[20]. SAD is commonly used as a matching criterion in ME of frame j:

$$D_j = \sum_{x=0}^{X} \sum_{y=0}^{Y} |f_j(x,y) - f_{j-1}(x+a, y+b)|. \quad (1)$$

In the DVCS system, the side information functional unit is usually deployed at the decoder, which predicts and reconstructs the current non-keyframe with the aid of the previously decoded adjacent keyframes. Determining skip-blocks solely based on SAD tends to overlook the influence of motion vectors. Therefore, this functional unit adopts FI as the side-information generation method at the encoder. Specifically, motion vectors are estimated from adjacent keyframes, and bidirectional motion-compensated prediction is performed to obtain the side information for the current non-keyframe. SAD is first used to preliminarily screen those candidate skip-blocks in the reference frame. The side-information error metric $E_{\text{si}}(j)$ is calculated based on the side information, which measures the discrepancy between the original frame and the side-information frame, where the formal definition is given by

$$S_j = P_{j-1\to j}(x + u_{j-1}(x, y), y + v_{j-1}(x, y)) + P_{j+1\to j}(x + u_{j+1}(x, y), y + v_{j+1}(x, y)), \quad (2)$$

$$E_{\text{si}}(j) = \frac{1}{N^2} \sum_{x=0}^{X} \sum_{y=0}^{Y} |f_j(x,y) - S_j(x,y)|, \quad (3)$$

where $P_{j-1\to j}$ denotes forward motion-vector prediction frame; $P_{j+1\to j}$ denotes backward motion-vector prediction frame; $u_{j-1}(x, y)$ and $v_{j-1}(x, y)$ represent the horizontal and vertical motion displacements from frame $j-1$ to the target frame j at pixel coordinate (x, y), respectively; $u_{j+1}(x, y)$ and $v_{j+1}(x, y)$ correspond to the horizontal and vertical motion displacements from frame $j+1$ toward frame j at pixel position (x, y), respectively.

A smaller value indicates that the side information is closer to the original frame, which makes this block more suitable for skipping; otherwise, this block should not be skipped. If the D_j value of the block is small, it is marked as a preferred skip-block and assigned a high skip ratio; if the D_j value of the block is large and $E_{\text{si}}(j)$ exceeds the skip-block threshold, the corresponding block is marked as a candidate block for subsequent adaptive skip ratio decision and assigned a skip ratio in the second-highest range. This approach improves reconstruction fidelity, particularly in video sequences characterized by rapid or

erratic motion, thus enhancing the performance of the motion-adaptive skipping algorithm.

The second module is an LBP texture feature module. Most existing skipping algorithms do not consider the similarity between frames and texture features, so the LBP texture feature module extracts texture information from adjacent frames. This texture information is then combined with motion features to enable a more refined calculation of the skip-block threshold. Furthermore, because it only extracts the texture features of adjacent frames without utilizing information from the decoding end, this approach maintains the universality of the decoder. LBP texture features are an operator employed to characterize local image features. They are noted for their simplicity, computational efficiency, and invariance to grayscale variations[21]. The fundamental principle of the algorithm involves generating a binary code for each pixel by comparing its grayscale value to those of its surrounding neighbors. A neighbor with a grayscale value greater than or equal to that of the central pixel is assigned a value of 1; otherwise, it is assigned a value of 0. The formal definition is given by

$$\mathrm{LBP}(x_c, y_c) = \sum_{q=0}^{P-1} 2^q \times s(l_n - l_c), \tag{4}$$

where (x_c, y_c) is the coordinate of the central pixel; q denotes the serial number of the neighboring pixel ($0 \leqslant q \leqslant P-1$); s is a sign function; l_n is the gray intensity of the neighboring pixel; l_c is the gray intensity of the central pixel. The formula for the LBP texture features T_{LBP} is

$$\mu_H = \sum_{i=0}^{P+1} i \times H(i), \tag{5}$$

$$T_{\mathrm{LBP}} = \sum_{i=0}^{P+1} H(i) \times (i - \mu_H)^2. \tag{6}$$

In addition, we introduce the variance texture features T_{var} and the gradient energy texture features T_{grad}, and combine them with LBP texture features to obtain comprehensive texture features. The formulas for the variance texture features and gradient texture features are

$$\mu_B = \frac{1}{X \times Y} \sum_{u=1}^{X} \sum_{v=1}^{Y} B(u, v), \tag{7}$$

$$T_{\mathrm{var}} = \frac{1}{X \times Y} \sum_{u=1}^{X} \sum_{v=1}^{Y} (B(u, v) - \mu_B)^2, \tag{8}$$

$$T_{\mathrm{grad}} = \frac{1}{X \times Y} \sum_{u=1}^{X} \sum_{v=1}^{Y} (G_x(u, v)^2 + G_y(u, v)^2). \tag{9}$$

In practical application scenarios such as security surveillance and adaptive streaming, video sequences often exhibit significant differences in content complexity or intricate texture variations. If only SAD is used to determine the skip-block threshold without integrating other motion information and texture features, the application effect of the algorithm in high-precision video systems will be severely constrained. This leads to two major issues: frames with rapid motion may be skipped excessively, thereby causing the loss of visual details and increased errors; while frames with slow motion may be insufficiently skipped, ultimately resulting in the transmission of redundant information, which reduces compression efficiency. To mitigate these issues, we introduce a third module: a joint motion-texture adaptive weight module. Unlike previous algorithms that determined the skip-block threshold—and consequently the skip ratio—solely based on motion estimation, the proposed algorithm integrates both motion estimation and texture features[22]. By leveraging spatial and temporal information simultaneously, it achieves a more accurate estimation of the threshold. To facilitate a fair comparison between motion estimation and texture features, both are first normalized to the same scale. The algorithm employs the joint weighted coefficient C_{jw} to represent the relative contributions of motion estimation and texture features. Given the greater significance of motion estimation in the skip-block decision, the C_{jw} values adapt dynamically under varying conditions to reflect this importance[23]. The algorithm jointly weighs motion estimation and texture features through C_{jw} to determine adaptive skip-block thresholds, establishing a precision-optimized skipping mechanism[24].

The implementation first normalizes motion information and texture information into comparable numerical ranges through spatio-temporal characteristic mapping, eliminating dimensional disparities via the following normalization equations:

$$V_n = \frac{V - V_{\min}}{V_{\max} - V_{\min}}, \tag{10}$$

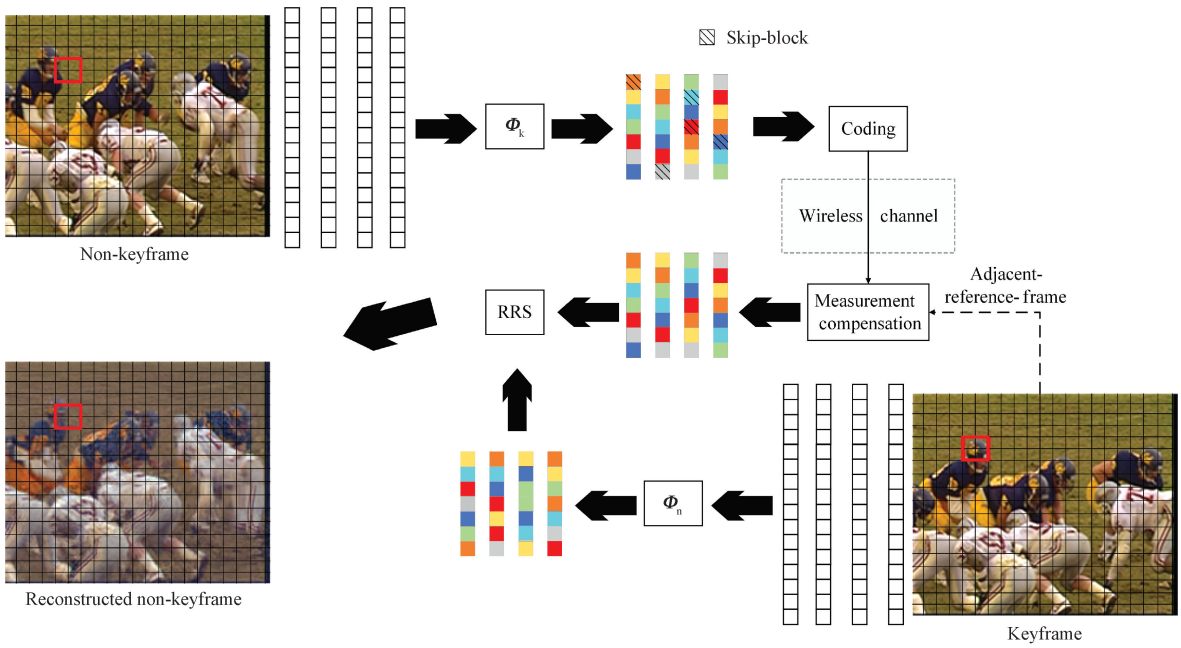
$$T_n = \frac{T - T_{\min}}{T_{\max} - T_{\min}}, \tag{11}$$

$$C_{\mathrm{jw}} = \frac{1}{1 + e^{-m \times (V_n - T_n) + c}}, \tag{12}$$

where V_n denotes the normalized motion information; T_n denotes the normalized texture information; V denotes the raw value of motion information; $V_{\max}$ denotes the maximum value of motion information among all blocks in the current frame; $V_{\min}$ denotes the minimum value of motion information among all blocks in the current frame; T denotes the raw value of texture information; $T_{\max}$ denotes the maximum value of texture information among all blocks in the current frame; $T_{\min}$ denotes the minimum value of texture information among all blocks in the current frame; m and c denote the slope coefficient and the intercept coefficient in the linear fitting model of Eq. (12), respectively. The integration of these three modules enhances the accuracy of skip-ratio allocation. Furthermore, the application of motion information and texture information improves the precision of the joint motion-texture adaptive weight module, leading to a

more significant overall performance improvement.

Figure 3 details the algorithm's diagram of adjacent-reference-frame analysis, explicitly outlining the step-by-step mechanism of the skip-block decision. It mainly illustrates the transmission and reconstruction processes of keyframes and non-keyframes, where keyframes serve as adjacent-reference-frames to assist in the reconstruction of non-keyframes. Among them, the colored blocks represent measurements, which are obtained by multiplying the grouped original images by the measurement matrix; the squares with grid patterns in their centers denote skip-blocks.



Φ_k —keyframe measurement matrix; Φ_n —non-keyframe measurement matrix.

Fig. 3 Diagram of adjacent-reference-frame analysis

The algorithm also employs C_{jw} to adaptively adjust the skip-block threshold. Initially, a base threshold is preset according to the complexity of the video sequence, and an adjustment coefficient is applied to control the sensitivity of C_{jw} in modifying the threshold, as expressed by

$$T_a = T_b \times (1 + \alpha \times (1 - C_{jw})), \tag{13}$$

where T_a and T_b denote the adaptive threshold and the base threshold, respectively; α denotes an auxiliary adjustment coefficient.

Furthermore, C_{jw} is used to optimize the skip-block threshold by considering the relative magnitudes of motion and texture, thereby enhancing the accuracy of the threshold. Figure 4 provides a detailed illustration of the joint motion-texture adaptive weight module introduced in Fig. 2. By comparing the magnitudes of the combined score S_c and the adaptive threshold T_a, this module determines whether the block should be skipped and calculates the initial skip ratio r_{si} for the skipped blocks.

To address this requirement and ensure the adaptive skip ratios maintain a predetermined mean value, our algorithm incorporates a post-calibration phase as outlined in the pseudocode's adjustment module. The implementation first calculates M_d and T_d between the current frame and its reference frame. Here, M_d represents the difference in motion between adjacent frames; T_d represents the difference in texture between adjacent frames. These M_d and T_d values are nonlinearly mapped to skip-ratio adjustment coefficients through a piecewise logistic function. The algorithm aggregates skip ratios across all non-keyframes, computes the deviation between the empirical mean and the target value, and derives a global adjustment scaling factor. This factor uniformly adjusts per-frame skip ratios to achieve statistical equilibrium while preserving scene-adaptive characteristics. The process formula is

$$r_{su} = r_{si} \times f_s = r_{si} \times \frac{r_{ss}}{r_{sa}}, \tag{14}$$

where r_{su}, r_{si}, r_{ss}, and r_{sa} represent the ultimate skip ratio, the initial skip ratio, the specified skip ratio, and the average skip ratio, respectively; f_s is a scale factor.

The initial skip ratio r_{si} is determined by combining the adaptive adjustment function with M_d, T_d, and r_{sf}. Here, r_{sf} represents the fixed skip ratio. The specific process is illustrated in Fig. 4.

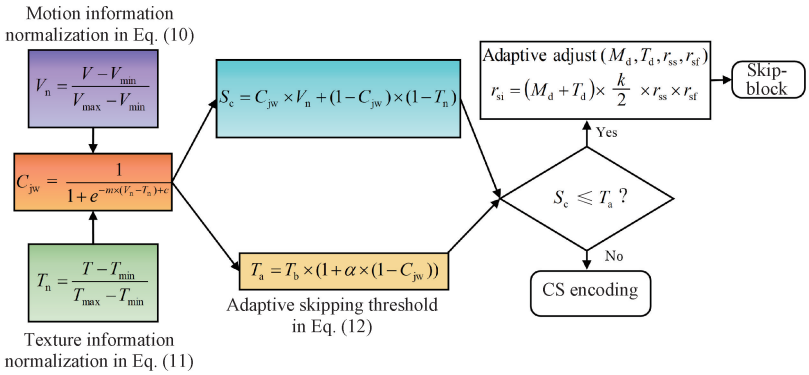


Fig. 4 Flowchart of skip-block decision for joint motion-texture adaptive weight module

The main pseudocode of the proposed MTJ algorithm is given in Algorithm 1, where the specific procedures or some other details are explained as follows.

Algorithm 1: MTJ

Initialization: Keyframe measurement matrix: $\boldsymbol{\Phi}_k$,
Non-keyframe measurement matrix: $\boldsymbol{\Phi}_n$,
Count of non-keyframes: C_n,
Number of frames per GOP: k,
Total number of blocks per frame: num,
Fixed skip ratio: r_{sf},
Index mapping function for outputting the column index corresponding to the i block in the frame matrix: $\boldsymbol{I}[i]$

for each GOP in the video sequence **do**
  for $j = 1$ to k **do**
    X_j = Frame$[j]$;
    x_j = im2col$(X_j, [X, Y])$;
    if j mod $k == 1$ **then**
      $y_j = \boldsymbol{\Phi}_k \times x_j$;
    else
      $r_j = |x_j - x_{j-1}|$;
      D_j = sum$(r_j, a = b = 0)$;
      $V_n = \frac{V - V_{min}}{V_{max} - V_{min}}$
      $T_n = \frac{T - T_{min}}{T_{max} - T_{min}}$
      $S_c = C_{jw} \times V_n + (1 - C_{jw}) \times (1 - T_n)$
      $T_a = T_b \times (1 + \alpha \times (1 - C_{jw}))$
      r_{si} = Adaptive adjust $(M_d, T_d, r_{ss}, r_{sf})$;
      for $i \in [1, num]$ **do**
        if $S_c[i] < T_a$
          $y[j][:, I[i]] = y[ref][:, I[i]]$;
        else
          $y[j][:, I[i]] = \boldsymbol{\Phi}_n \times x_j[i][:, I[i]]$;
        end if
      end for
      result-PSNR$[j]$ = PSNR (Frame$[j]$, $x[j]$);
end for
if $C_n > 0$ **then**
  r_{sa} = mean(r_{si});
  $f_s = r_{ss}/r_{sa}$;
  for each non-keyframe j **do**
    $r_{su}[j] = r_{si} \times f_s$;
  end for
end if

Note: result-PSNR$[j]$ denotes the recorded peak signal-to-noise ratio (PSNR) between the original frame$[j]$ and the reconstructed frame $x[j]$.

2 Results and Comparison

2.1 Quantitative comparison of different algorithms

In this section, we compare the proposed algorithm with existing algorithms. All experiments are conducted in the MATLAB R2023b environment and executed on Windows 10, an Intel(R) Core(TM) i7-10750H CPU operating at 2.59 GHz, and 16 GB RAM. The experiments utilize a DVCS system with a standard decoder, employing a Gaussian random matrix as the measurement matrix. For the experimental videos, we use the first 25 frames from standard test video sequences (Coastguard, Soccer, Foreman, and Football). The video sequences vary in speed, encompassing a wide range of motion characteristics. Consistent with existing algorithms, the subrate for keyframes is set at 0.7, while the subrate for non-keyframes ranges from 0.2 to 0.4. The number of frames per GOP k is set to 8, and the block size is configured at 16×16. And α is 0.5, m is 5, and c is 0.3. These parameters facilitate a balance between computational overhead at the decoding end and reconstruction quality, and are obtained by fitting the data of the first three frames.

We compare the performance of the proposed algorithm against that of MAS[10], HSC[11], and UTS[12] algorithms. The encoder skip ratios are set to 30%, 50%, and 70%. To ensure a fair comparison, all algorithms are evaluated under identical skip ratios, with the decoder employing the general RRS decoder for reconstruction[9]. Table 2 presents the average PSNR results for non-keyframes at a 70% skip ratio under various subrates for the compared algorithms. Maximum average PSNR values are indicated in bold.

Table 2 Average PSNR for different algorithms at 70% skip ratio

Subrate pair	Algorithm	PSNR under different test video sequences/dB			
		Coastguard	Soccer	Foreman	Football
(0.7, 0.4)	MAS	29.04	27.08	29.70	23.09
	HSC	29.21	27.10	29.86	23.29
	UTS	29.37	27.27	30.53	23.49
	MTJ	**29.51**	**27.46**	**30.77**	**23.63**
(0.7, 0.3)	MAS	28.49	26.86	29.40	22.84
	HSC	28.61	26.85	29.53	22.99
	UTS	28.75	26.98	30.12	23.14
	MTJ	**28.86**	**27.13**	**30.32**	**23.22**
(0.7, 0.2)	MAS	27.46	26.39	28.77	22.37
	HSC	27.54	26.37	28.89	22.50
	UTS	27.62	26.46	29.35	22.61
	MTJ	**27.68**	**26.58**	**29.51**	**22.68**

As can be seen from Table 2, the MTJ algorithm consistently achieves the highest average PSNR values, demonstrating its superior performance across the tested subrate conditions. Specifically, for the Foreman sequence under the subrate pair (0.7, 0.3), the PSNR of MTJ over UTS reaches 0.20 dB.

Figure 5 presents the average PSNR results for different algorithms under the conditions of a keyframe subrate of 0.7 and a non-keyframe subrate of 0.4 across various skip ratios. It is evident that as the skip ratio increases, the average PSNR decreases more significantly. The average PSNR of the MTJ algorithm remains consistently close to the maximum value, demonstrating a greater capacity to mitigate the decline in PSNR as the skip ratio rises. The LBP texture feature module utilizes inter-frame similarity to extract texture information, thereby enhancing the correlation between frames and compensating for details lost when the skip ratio is excessively high, ultimately improving reconstruction quality.

Figure 6 illustrates the average PSNR results for four distinct test video sequences. In each group, every individual curve plot presents the average PSNR of competing algorithms with a fixed skip ratio. It is evident from Fig. 6 that the average PSNR of the MTJ algorithm is consistently the highest.

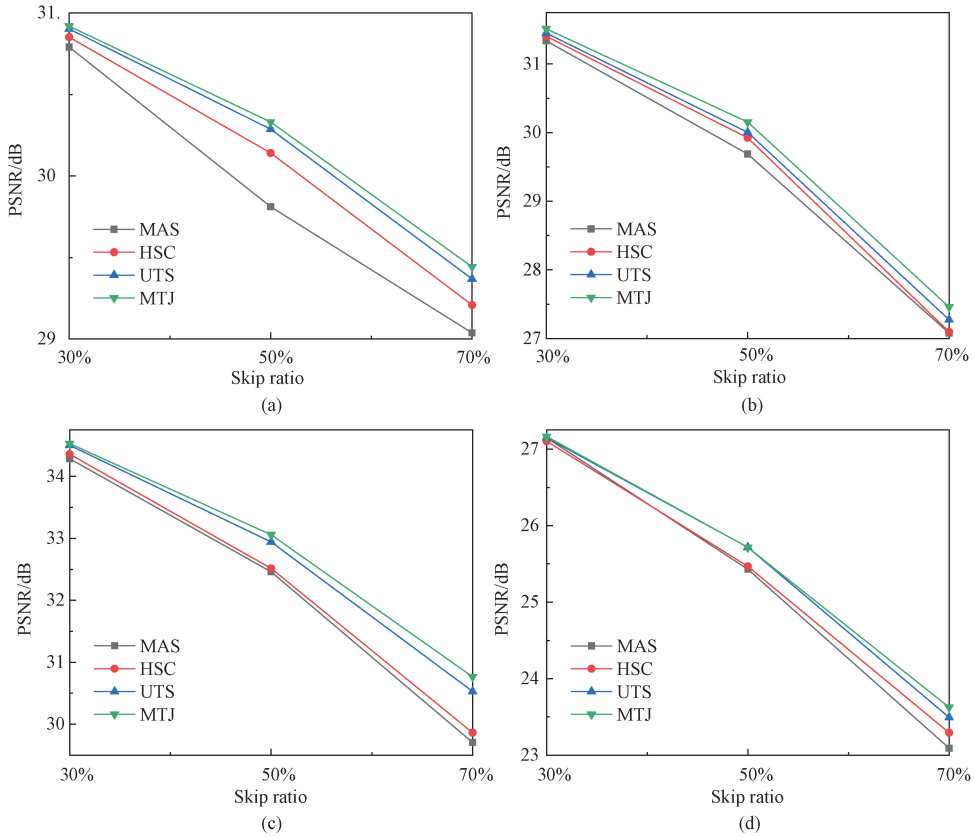
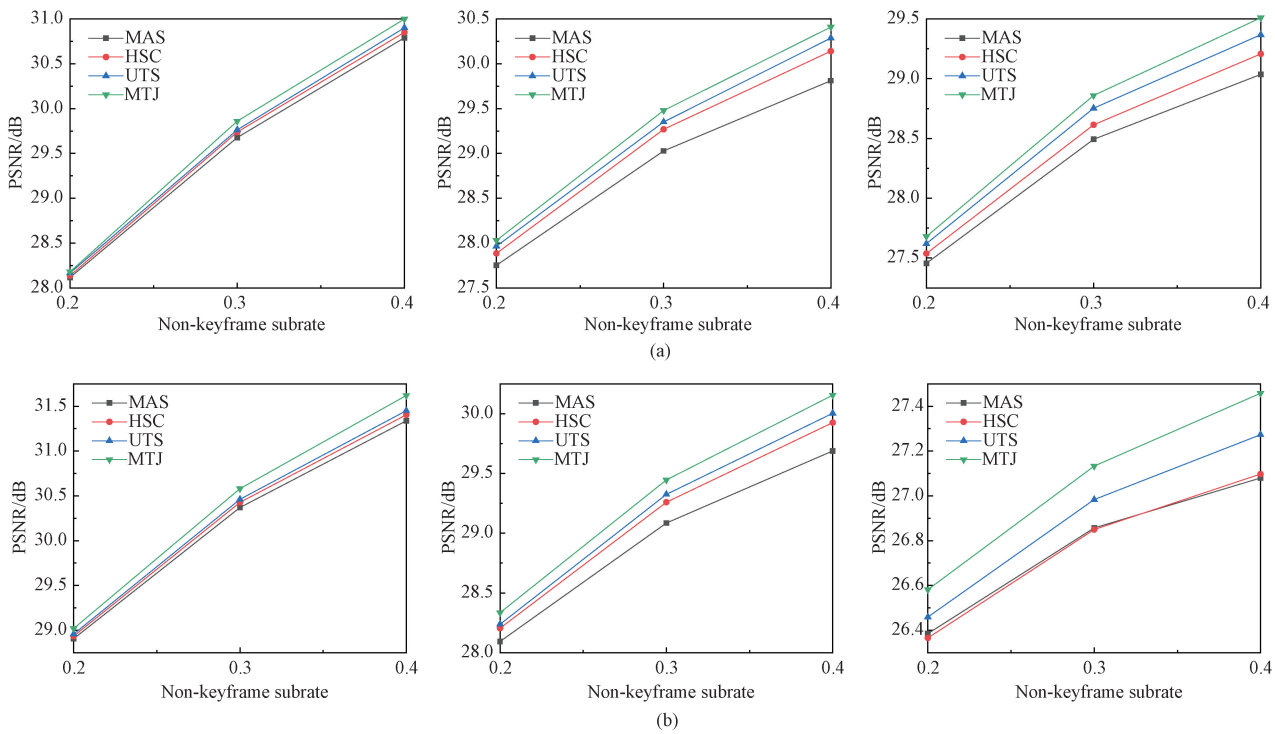


Fig. 5 Average PSNR results for different algorithms at different skip ratios and test video sequences: (a) Coastguard; (b) Soccer; (c) Foreman; (d) Football



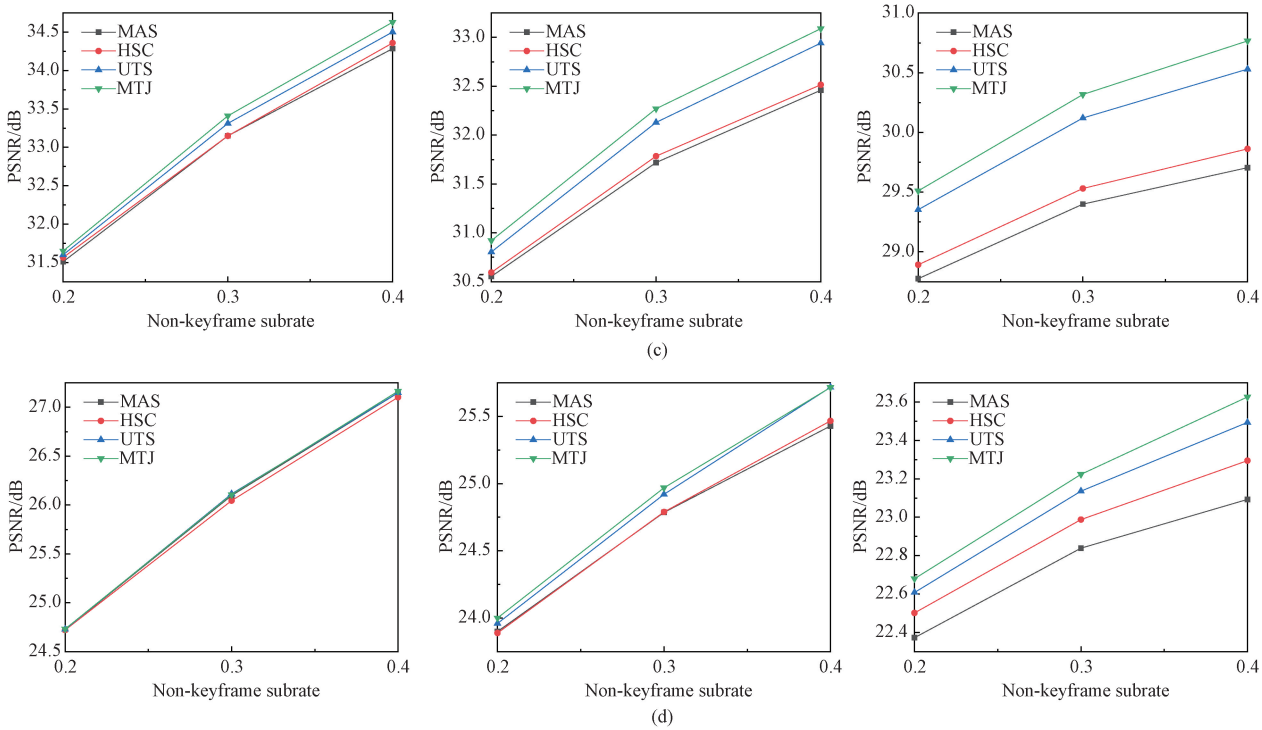
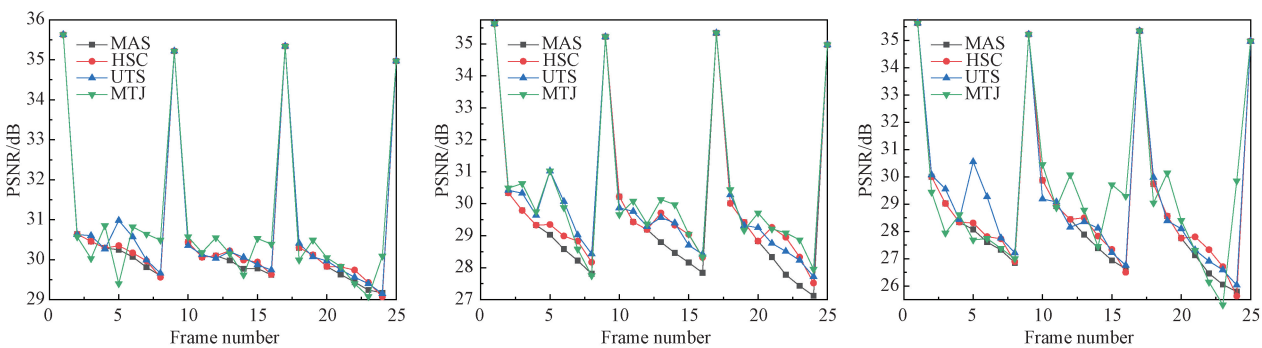


Fig. 6 Average PSNR results for different algorithms under different non-keyframe subrates and test video sequences (left, middle, and right plots correspond to skip ratios of 30%, 50%, and 70%, respectively): (a) Coastguard; (b) Soccer; (c) Foreman; (d) Football

The MTJ algorithm achieves an improvement in average PSNR of 0.12 dB compared to the UTS algorithm. Furthermore, as the skip ratio increases, the performance enhancement of the MTJ algorithm becomes more pronounced. This is attributed to the UTS algorithm's reliance on uniform-reference motion estimation, which becomes less accurate at high skip ratios, leading to suboptimal skip-block decisions and excessive skipping of motion-intensive frames. Therefore, it can be concluded that incorporating an adaptive weighted skip module can effectively address this issue.

Figure 7 illustrates the frame-by-frame performance for different algorithms across various video sequences and skip ratios, given a keyframe subrate of 0.7 and a non-keyframe subrate of 0.4. The MTJ algorithm exhibits significantly different skip ratios for certain frames compared to other algorithms. This is attributed to the incorporation of the motion-texture adaptive weight module, which expands the range of adaptive skipping and enables a more precise allocation of skip ratios, thereby enhancing performance at the same average skip ratio.



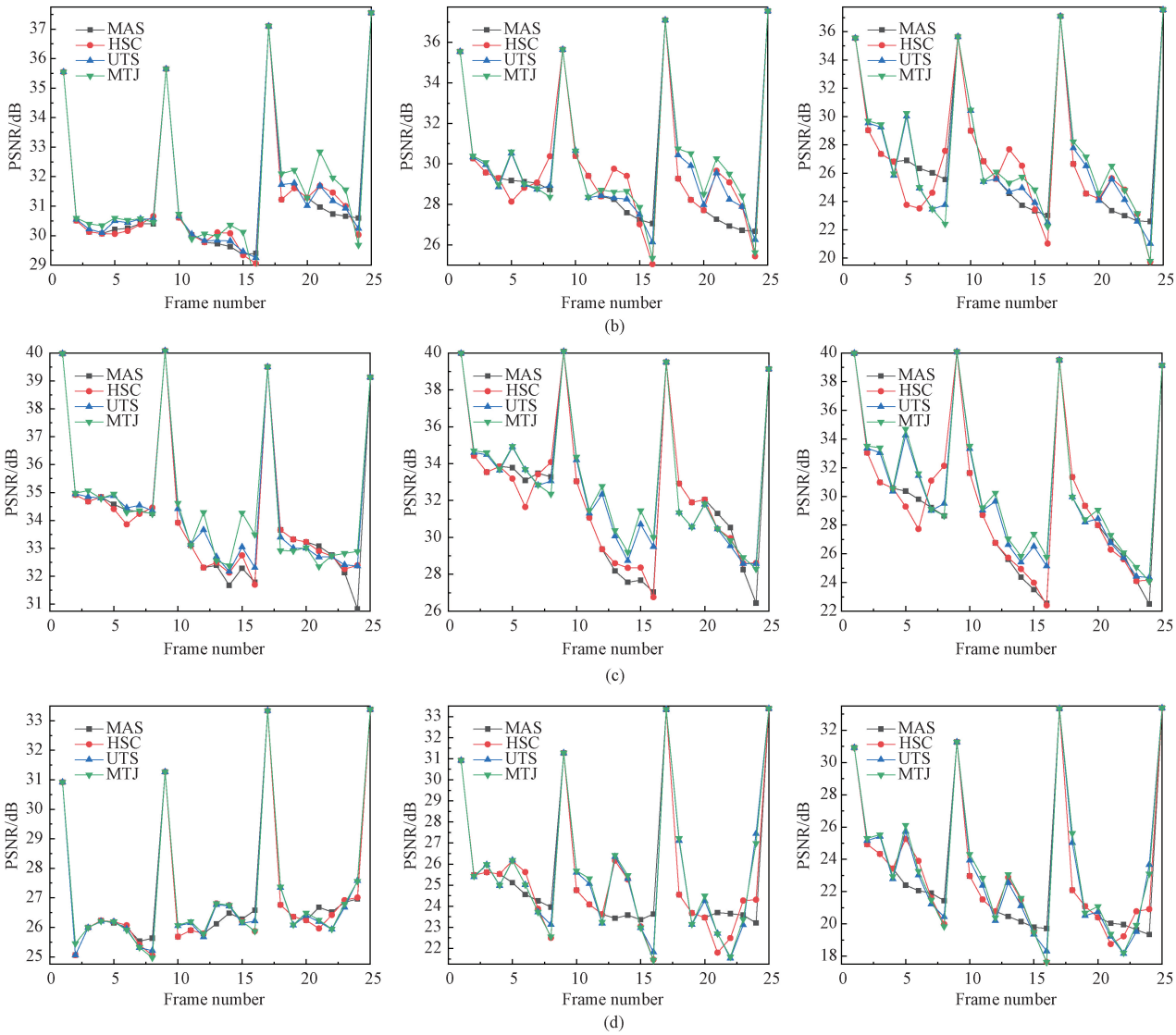


Fig. 7 Frame-by-frame PSNR results for different algorithms at different skip ratios and test video sequences (left, middle, and right plots correspond to skip ratios of 30%, 50%, and 70%, respectively): (a) Coastguard; (b) Soccer; (c) Foreman; (d) Football

Table 3 presents the average PSNR results for non-keyframes at different subrates and a 50% skip ratio for the compared algorithms. Table 4 presents the average PSNR results for non-keyframes at various subrates and a 30% skip ratio for the compared algorithms. Maximum average PSNR values are indicated in bold. As can be seen from Tables 3 and 4, the MTJ algorithm shows modest gains over UTS at lower skip ratios, which is expected since lower skip ratios preserve more measurements and limit optimization potential. The algorithm's advantages become more pronounced at higher skip ratios, where precise block decisions are critical.

Table 3 Average PSNR for different algorithms at 50% skip ratio

Subrate pair	Algorithm	PSNR under different test video sequences/dB			
		Coastguard	Soccer	Foreman	Football
(0.7, 0.4)	MAS	29.81	29.69	32.46	25.43
	HSC	30.14	29.93	32.52	25.47
	UTS	30.29	30.00	32.94	**25.72**
	MTJ	**30.41**	**30.16**	**33.09**	**25.72**
(0.7, 0.3)	MAS	29.03	29.09	31.72	24.79
	HSC	29.27	29.26	31.78	24.79
	UTS	29.35	29.32	32.13	24.92
	MTJ	**29.48**	**29.45**	**32.27**	**24.97**
(0.7, 0.2)	MAS	27.75	28.09	30.55	23.90
	HSC	27.88	28.21	30.59	23.89
	UTS	27.97	28.24	30.80	23.96
	MTJ	**28.03**	**28.34**	**30.92**	**24.00**

Table 4 Average PSNR for different algorithms at 30% skip ratio

Subrate pair	Algorithm	PSNR under different test video sequences/dB			
		Coastguard	Soccer	Foreman	Football
(0.7, 0.4)	MAS	30.79	31.34	34.29	27.15
	HSC	30.85	31.40	34.36	27.10
	UTS	30.90	31.45	34.50	27.15
	MTJ	**31.00**	**31.62**	**34.63**	**27.16**
(0.7, 0.3)	MAS	29.68	30.37	33.15	26.10
	HSC	29.74	30.43	33.15	26.04
	UTS	29.76	30.46	33.31	**26.11**
	MTJ	**29.86**	**30.58**	**33.41**	**26.11**
(0.7, 0.2)	MAS	28.12	28.91	31.51	24.73
	HSC	28.14	28.94	31.57	24.72
	UTS	28.17	28.96	31.60	**24.73**
	MTJ	**28.18**	**29.02**	**31.65**	**24.73**

2.2 Ablation experiments

To test the ablation performance, we remove the side information functional unit from the MTJ algorithm, and name it MTJ-s. Meanwhile, we remove the LBP texture feature from the MTJ algorithm and name this variant MTJ-t. Tables 5, 6, and 7 show the performance differences between MTJ, MTJ-s, and MTJ-t at various skip ratios and different video sequences.

Table 5 Average PSNR for MTJ ablation variants at 70% skip ratio

Subrate pair	Algorithm	PSNR under different test video sequences/dB			
		Coastguard	Soccer	Foreman	Football
(0.7, 0.4)	MTJ-s	28.87	26.92	29.66	23.32
	MTJ-t	29.32	27.30	30.41	23.55
	MTJ	**29.51**	**27.46**	**30.77**	**23.63**
(0.7, 0.3)	MTJ-s	28.33	26.69	29.43	23.05
	MTJ-t	28.71	27.00	30.04	23.18
	MTJ	**28.86**	**27.13**	**30.32**	**23.22**
(0.7, 0.2)	MTJ-s	27.35	26.32	28.84	22.60
	MTJ-t	27.59	26.51	29.31	22.65
	MTJ	**27.68**	**26.58**	**29.51**	**22.68**

Table 6 Average PSNR for MTJ ablation variants at 50% skip ratio

Subrate pair	Algorithm	PSNR under different test video sequences/dB			
		Coastguard	Soccer	Foreman	Football
(0.7, 0.4)	MTJ-s	29.69	29.47	32.22	25.31
	MTJ-t	30.18	29.95	32.72	25.58
	MTJ	**30.41**	**30.16**	**33.09**	**25.72**
(0.7, 0.3)	MTJ-s	28.94	28.89	31.67	24.72
	MTJ-t	29.32	29.28	32.09	24.89
	MTJ	**29.48**	**29.45**	**32.27**	**24.97**
(0.7, 0.2)	MTJ-s	27.78	28.00	30.51	23.89
	MTJ-t	27.94	28.25	30.77	23.97
	MTJ	**28.03**	**28.34**	**30.92**	**24.00**

Table 7 Average PSNR for MTJ ablation variants at 30% skip ratio

Subrate pair	Algorithm	PSNR under different test video sequences/dB			
		Coastguard	Soccer	Foreman	Football
(0.7, 0.4)	MTJ-s	30.30	30.84	34.02	26.84
	MTJ-t	30.79	31.36	34.45	27.04
	MTJ	**31.00**	**31.62**	**34.63**	**27.16**
(0.7, 0.3)	MTJ-s	29.39	29.92	33.11	25.85
	MTJ-t	29.72	30.33	33.27	26.02
	MTJ	**29.86**	**30.58**	**33.41**	**26.11**
(0.7, 0.2)	MTJ-s	28.05	28.71	31.55	24.63
	MTJ-t	28.16	28.94	31.62	24.72
	MTJ	**28.18**	**29.02**	**31.65**	**24.73**

2.3 Computational complexity and operational efficiency analyses

Table 8 shows the average per-frame running time for different algorithms at the fixed subrate pair (0.7, 0.4), with results evaluated across different skip ratios and test video sequences, where the minimum running time is indicated in bold. It can be observed that MTJ achieves a relatively low overall running time at a skip ratio of 30%. Under other conditions, UTS achieves the minimum runtime in most test cases, while MTJ mostly ranks second in computational complexity and also attains the shortest running time on several sequences, but yields a significantly higher average PSNR.

Table 8 Average per-frame running time for different algorithms at different skip ratios

Skip ratio	Algorithm	Average per-frame running time under different test video sequences/s			
		Coastguard	Soccer	Foreman	Football
70%	MAS	11.596 6	9.150 0	**10.332 8**	7.018 3
	HSC	11.145 3	8.632 8	11.175 8	7.542 2
	UTS	**8.481 0**	**6.396 1**	10.580 5	7.600 8
	MTJ	8.734 0	7.164 7	11.044 5	**6.951 3**
50%	MAS	10.594 2	8.283 4	10.446 7	7.524 4
	HSC	9.980 0	8.734 3	11.162 9	7.969 3
	UTS	**8.457 7**	**7.212 2**	**9.250 9**	6.347 5
	MTJ	8.735 4	7.341 9	9.438 6	**6.301 3**
30%	MAS	10.293 7	8.554 5	10.481 2	7.331 7
	HSC	10.286 9	8.372 7	9.591 5	6.200 7
	UTS	10.144 1	8.236 0	10.788 3	7.044 9
	MTJ	**9.977 5**	**7.649 2**	**9.215 2**	**5.988 6**

Furthermore, Tables 9, 10, and 11 present the experimental analyses of the complexity of each module. The average per-frame coding time of individual modules under the fixed subrate pair (0.7, 0.4) is presented, with assessments conducted across different skip ratios and test video sequences. Experimental results show that the added lightweight texture module has little impact on the algorithm's complexity. Other computationally expensive components are common, including the SAD calculation in the motion estimation module and the skip ratio computation in the joint motion-texture adaptive weight module. Therefore, the overall complexity of the proposed algorithm does not increase significantly compared with other algorithms.

Table 9 Average per-frame coding time for different modules at 70% skip ratio

Module	Average per-frame coding time under different test video sequences/ms			
	Coastguard	Soccer	Foreman	Football
Motion estimation	4.73	3.21	3.12	2.82
LBP texture feature	0.25	0.10	0.10	0.10
Joint motion-texture adaptive weight	9.65	8.93	10.51	10.10

Table 10 Average per-frame coding time for different modules at 50% skip ratio

Module	Average per-frame coding time under different test video sequences/ms			
	Coastguard	Soccer	Foreman	Football
Motion estimation	2.41	2.56	2.40	2.81
LBP texture feature	0.10	0.10	0.10	0.10
Joint motion-texture adaptive weight	7.64	7.82	8.88	9.49

Table 11 Average per-frame coding time for different modules at 30% skip ratio

Module	Average per-frame coding time under different test video sequences/ms			
	Coastguard	Soccer	Foreman	Football
Motion estimation	4.06	2.40	2.25	2.44
LBP texture feature	0.18	0.10	0.10	0.10
Joint motion-texture adaptive weight	10.21	8.50	7.93	7.60

From the ablation study and complexity analysis of each module, it can be observed that modules with smaller computational overhead generally contribute less to performance improvement. This indicates a good trade-off between complexity and effectiveness in the proposed MTJ algorithm.

3 Conclusions

The proposed MTJ algorithm enhances the accuracy of skip-block decisions by effectively leveraging inter-frame similarity through the integration of a motion estimation module, an LBP texture feature module, and a joint motion-texture adaptive weight module. Experimental analysis comparing the MTJ algorithm with other algorithms across video sequences of diverse speeds reveals that it consistently achieves superior reconstruction quality. MTJ algorithm incurs only a negligible increase in computational complexity compared with conventional DVCS systems. Motion estimation serves as a shared baseline component across all algorithms. The introduced modules, including motion vector refinement, lightweight LBP texture extraction, and joint weighting operations, follow the same linear order without introducing iterations or high-dimensional transforms. Notably, the proposed MTJ algorithm achieves a considerable improvement in reconstruction quality at the cost of only a slight increase in complexity, demonstrating promising engineering practicability.

References

[1] Spagnolo F, Lal B, Corsonello P, et al. A novel compressive sensing method for secure and energy efficient ECG signal transmission applications [J]. *IEEE Journal of Biomedical and Health Informatics*, 2025, 29(5): 3439-3450.

[2] Zeng C Y, Zou F, Xia S Y, et al. SAU-Net: saliency-based adaptive unfolding network for interpretable high-quality image compressed sensing in Internet of Things [J]. *IEEE Internet of Things Journal*, 2025, 12(14): 28672-28688.

[3] Unde A S, Pattathil D P. Adaptive compressive video coding for embedded camera sensors: compressed domain motion and measurements estimation [J]. *IEEE Transactions on Mobile Computing*, 2020, 19(10): 2250-2263.

[4] Guan Z Y, Xing Q L, Xu M, et al. MFQE 2.0: a new approach for multi-frame quality

enhancement on compressed video [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021, 43(3): 949-963.
[5] Chen C, Wu Y T, Zhou C, et al. JsrNet: a joint sampling-reconstruction framework for distributed compressive video sensing [J]. *Sensors*, 2020, 20(1): 206.
[6] Song X D, Peng X L, Xu J Z, et al. Distributed compressive sensing for cloud-based wireless image transmission [J]. *IEEE Transactions on Multimedia*, 2017, 19(6): 1351-1364.
[7] Zhang Y F, Zhang C, Fan R, et al. Recent advances on HEVC inter-frame coding: from optimization to implementation and beyond [J]. *IEEE Transactions on Circuits and Systems for Video Technology*, 2020, 30(11): 4321-4339.
[8] Song Z, Chen J H. Adaptive rate compression for distributed video sensing in wireless visual sensor networks [J]. *The Visual Computer*, 2025, 41(10): 7909-7923.
[9] Zhao C, Ma S W, Zhang J, et al. Video compressive sensing reconstruction via reweighted residual sparsity [J]. *IEEE Transactions on Circuits and Systems for Video Technology*, 2017, 27(6): 1182-1195.
[10] Yuan W Y, Liu H. Motion-adaptive adjacent-reference skipping for distributed video compressive sensing with general decoders [J]. *Optoelectronics Letters*, 2022, 18(12): 755-762.
[11] Priya G L, Ghosh D. An effectual video compression scheme for WVSNs based on block compressive sensing [J]. *IEEE Transactions on Network Science and Engineering*, 2024, 11(2): 1542-1552.
[12] Liu H, Sun R H. Uniform-reference threshold-dynamic skipping for video compressive sensing [J]. *Displays*, 2025, 87: 102963.
[13] Yang X, Yang C L. MAP-inspired deep unfolding network for distributed compressive video sensing [J]. *IEEE Signal Processing Letters*, 2023, 30: 309-313.
[14] Wang Q R, Qian L P, Jiang W, et al. Integrated communication and computation resource allocation for the compressive sensing based image transmission [J]. *IEEE Transactions on Mobile Computing*, 2025, 24(10): 10510-10522.
[15] Zhang J G, Xiang Q M, Yin Y G, et al. Adaptive compressed sensing for wireless image sensor networks [J]. *Multimedia Tools and Applications*, 2017, 76(3): 4227-4242.
[16] Wang W, Li J J, Mo H, et al. Side information hybrid generation based on improved motion vector field [J]. *Multimedia Tools and Applications*, 2021, 80(17): 26713-26730.
[17] Li R, Yang Y H, Sun F Y. Green visual sensor of plant: an energy-efficient compressive video sensing in the Internet of Things [J]. *Frontiers in Plant Science*, 2022, 13: 849606.
[18] Song Z, Chen J H, Li K. A hybrid sampling mode design for compressive imaging based on adaptive block compressed sensing [C]//2025 5th International Conference on Neural Networks, Information and Communication Engineering (NNICE). Piscataway, NJ: IEEE, 2025: 301-304.
[19] Park M, Bae Y. Complex motion-aware splatting for video frame interpolation [C]//2023 14th International Conference on Information and Communication Technology Convergence (ICTC). Piscataway, NJ: IEEE, 2023: 1872-1876.
[20] Rup S, Majhi B, Padhy S. An improved side information generation for distributed video coding [J]. *AEU-International Journal of Electronics and Communications*, 2014, 68(3): 201-209.
[21] Ji Y K, Ge H Y, Meng Y Q, et al. MLGIA: recognition of traffic panel information based on PaddlePaddle [J]. *Journal of Donghua University (English Edition)*, 2025, 42(5): 494-502.
[22] Zhou J, Liu H. Video compressive sensing reconstruction via long-short-term double-pattern prediction [J]. *Optoelectronics Letters*, 2020, 16(3): 230-236.
[23] Chen C, Zhou C, Liu P Y, et al. Iterative reweighted Tikhonov-regularized multihypothesis prediction scheme for distributed compressive video sensing [J]. *IEEE Transactions on Circuits and Systems for Video Technology*, 2020, 30(1): 1-10.
[24] Zeng X Y, Xu W J, Zhang X N. DrownACB-YOLO: an improved YOLO for drowning detection in swimming pools [J]. *Journal of Donghua University (English Edition)*, 2025, 42(4): 417-424.

面向分布式视频压缩感知的运动-纹理联合阈值-自适应跳块算法

苏金昊[1]，刘　浩[1,2*]，黄　荣[1,2]

1. 东华大学 信息与智能科学学院，上海 201620

2. 东华大学 数字化纺织服装技术教育部工程研究中心，上海 201620

摘　要：传统的分布式视频压缩感知（distributed video compressive sensing，DVCS）系统有效降低了无线视觉传感器网络（wireless visual sensor network，WVSN）的传输成本和编码器复杂度，但在涉及快速运动或复杂纹理的动态场景中表现不佳。它们对时域相关性和跳块的依赖导致重建伪影和结构细节丢失，尤其是在高跳块率下更加明显。本文提出采用运动-纹理联合（motion-texture joint，MTJ）阈值跳块算法来提升DVCS 性能，其中运动模块先通过绝对差值和（sum of absolute differences，SAD）对候选跳块进行粗筛，再结合帧插值与运动矢量生成稳健性更好的边信息，用于识别冗余内容并辅助跳块决策。此外，基于局部二值模式（local binary pattern，LBP）分析的纹理特征模块，利用相邻帧的 LBP 纹理特征、方差纹理和梯度能量纹理，提升帧间相关性。在此基础上，运动-纹理联合自适应权重模块根据其时空贡献程度来动态分配运动信息和纹理信息的权重系数，并控制 DVCS 的跳块率。在标准视频序列上的实验表明，所提 MTJ 算法相较最先进的统一参考阈值动态跳块（uniform-reference threshold-dynamic skipping，UTS）算法，平均峰值信噪比（peak signal-to-noise ratio，PSNR）提升 0.12 dB，编码器复杂度仅微增。尤其在 70%跳块率下，该算法在 Foreman 序列上相对 UTS 的 PSNR 最大提升达到 0.20 dB。MTJ 算法有效平衡了重建质量和计算复杂度，使其更适用于 WVSN。

关键词：分布式视频压缩感知；跳块率；运动估计；局部二值模式纹理特征；自适应权重