> 102 <

Uniform-reference Threshold-dynamic skipping for Video Compressive Sensing

***Abstract*—Compressive Sensing (CS) technology completes the compression of the original signal during the sampling process, and reduces the computational burden at the encoder. For video signals, the adjacent frames are highly similar, so some researchers have utilized the temporal correlation of video frames to further compress the data. The existing algorithms mark highly similar blocks in video frames as skip blocks. The reference and selection of skip blocks are related to the recovery quality of the decoder and the compression rate at the encoder. This paper proposes a Uniform-reference Threshold-dynamic Skipping (UTS) algorithm. Firstly, the proposed algorithm sets a dynamic threshold selection skip block, which is suitable for video sequences with different motion variations. Secondly, in the general GOP framework, keyframes and non-keyframes in the middle are used as reference frames, so that the reference frames are uniformly distributed, which can provide accurate skip block references for more non-keyframes. At the same time, a high threshold is set for the intermediate frame for block skipping to ensure its reliability as a reference frame and further improve the compression rate. Experimental results show that the proposed UTS algorithm has a higher compression rate without significantly increasing the computational complexity at the encoder. Meanwhile, the recovery quality of the UTS algorithm is comparable to that of existing algorithms at the decoder.**

***Index Terms*—skip block, dynamic threshold, uniformly distributed, intermediate frame.**

I. INTRODUCTION

As multimedia communication technology advances, individuals can access a variety of information anywhere, anytime from their mobile devices and computers. The expression of information has shifted from text and sound to more vivid and illustrative images and videos. Various multimedia applications have become indispensable parts of daily life. At present, network data traffic is experiencing explosive growth, making it critical to enhance signal transmission capabilities. Traditional Nyquist sampling theorem posits that to preserve all the information in the original signal after sampling, the sampling frequency must be more than twice the highest frequency in the signal. Compress Sensing theory demonstrates that if a signal is sparse or sparse in its transformed domain, it can be reconstructed from a number of sampling points far fewer than required by the Nyquist sampling theorem[1-4]. Compressive sensing (CS) technology performs compression during the sampling process, reducing wastage of computational resources of the encoder and lowering costs.

A、Research Background

For video encoding and decoding systems, due to the substantial amount of video signal data, the characteristic of CS technology to reduce computational complexity of the encoder bears significant meaning[5]. In practical applications of video compressive sensing, especially for high-resolution images, each frame of the video sequence contains a large number of pixels, necessitating sufficient hardware resources. To reduce the computational complexity of video compressive sensing, GAN[6] proposed a block-based Compressive Sensing (BCS) sampling scheme, simultaneously dividing the video sequence into multiple groups of pictures (GOP) with the same number of frames. Currently, most sampling schemes in video and image compressive sensing technology are based on the GOP-BCS system.

To better apply BCS theory to video transmission systems, a distributed coding and streaming system was proposed in [7], which significantly reduces computational complexity of the encoder. However, this system employs traditional H.26X intra-frame coding and decoding technology for keyframes. A Distributed Video Compressive Sensing (VCS) system entirely based on the GOP-BCS sampling scheme was proposed in [8], representing the mainstream architecture in current research on video compressive sensing.

The VCS system greatly simplifies the computations of the encoder, which is of great significance for the development of upstream media streaming applications[9]. Under the architecture of the VCS system, current research at the decoder mainly aims to enhance the recovery quality of non-keyframes. Several effective reconstruction algorithms for the decoder have been proposed to date, such as Motion Compensated (MC) [10], Multihypothesis Prediction (MH)[11], and the superior quality Reweighted Residual Sparsity (RRS)[12]. For the encoder, non-keyframes can utilize low sampling rates, dramatically reducing the amount of data that needs to be transmitted by the encoder. However, experiments have found that observational data between adjacent frames often exhibit a high degree of similarity, indicating temporal redundancy in the

Received: 2024-xx-xx **Accepted:** 2024-xx-xx; **Published online:** 2024-xx-xx xxx
Foundation item: xxx
The authors are with the Department of xxx

data. Therefore, there is potential to further decrease the volume of data transmitted by the encoder. This paper primarily discusses relevant research on the encoder in the context of the VCS system.

B. Related Works

The image information of adjacent frames in a video sequence is highly similar, as mentioned in [13]. Therefore, in the encoder under the BCS framework, certain blocks can be non-sampled based on temporal correlation to reduce encoder energy consumption. In [14], the similarity between the current frame and the reference frame is measured at the encoder using the sum of absolute differences of image signals. A fixed threshold is set to mark blocks with similarity higher than the threshold as skip blocks. This algorithm reduces the amount of signal transmission and is applicable for end-to-cloud video transmission. In [15], keyframes are used as references, and different sampling rates are applied to blocks of varying similarity, categorizing encoding blocks into HIGH, LOW, and SKIP types to minimize redundant information. In this algorithm, the similarity between the current frame and the reference frame is measured using the mean of absolute differences. In [16], it is proposed that under the BCS framework, non-intersecting blocks in the encoder can utilize adaptive sampling rates based on statistical feature estimation. Such uneven sampling requires corresponding adjustments in the decoder and renders the use of a generic decoder unfeasible. An adaptive sampling rate allocation algorithm based on context features is proposed in [17]. This algorithm calculates contextual features based on preliminary observations and then sets adaptive sampling rates to re-observe the original image. The encoder observing the image signal twice greatly increases the calculation amount and the burden of the encoder.

Yuan et al. [18] proposed the Motion-adaptive Adjacent-reference Skipping (MAS) algorithm. This algorithm no longer only uses keyframes as references but selects adjacent frames as reference frames while incorporating a motion-adaptive module, effectively improving the energy efficiency of the encoder. Another High-compression Secondary Compression (HSC) algorithm is proposed in [19], more suitable for wireless visual sensor networks. In this algorithm, it is suggested that in practical CS scenarios, sensors directly sense sampling and compression values from the scene. Therefore, the approach of judging inter-frame correlation based on original image data to select skip blocks still requires a large amount of storage space for storing original image data at the encoder. To address this, they further compress the data in the encoder based on the correlation between measurements. However, the proposed algorithm designates non-keyframes adjacent to keyframes as reference frames without skipping processing, which wastes observational information from keyframes.

C. Contributions

Under the framework of the VCS system, we propose a Uniform-reference Threshold-dynamic Skipping (UTS) algorithm which set a dynamic threshold of the correlation between measurements to select skip blocks. Compared with the most state-of-the-art algorithms, this solution can achieve higher compression without reducing the recovery quality of the decoder. The main contributions include the following:

1. The UTS algorithm calculates the cosine correlation of the measurements of all corresponding position blocks between the current frame and the reference frame, and set dynamic thresholds based on the correlation of the measurements. Dynamic threshold can adaptively adjust the block skip ratio according to the motion changes of the video sequence.

2. Adjacent frames are usually more valuable for reference. In the general GOP framework, keyframes and intermediate non-keyframes are preferred as reference frames. This method makes the reference frames evenly distributed in the GOP and can provide accurate block-skipping references for more non-keyframes.

3. The intermediate frame uses the forward keyframe as a reference and sets a higher threshold to select skipped blocks. A high threshold ensures the reliability of intermediate non-key frames as reference frames for other non-key frames. At the same time, for video sequences with small motion changes, intermediate frames can further reduce the temporal redundancy of data.

II. VCS SYSTEM

Fig.1 shows the VCS system architecture, which is the mainstream architecture adopted by current compressed sensing technology. Current research is mainly divided into two aspects: encoder and decoder.

A. Encoder

The encoder of the VCS system is entirely based on GOP-BCS sampling, where keyframes use a high sampling rate and non-keyframes use a low sampling rate. After the video frame is observed, the amount of data is greatly reduced compared to the original image, effectively reducing the spatial redundancy of the data. But for video sequences, especially high frame rate videos, where the changes between adjacent frames are weak and the vector differences of measurements are small, we can use the temporal correlation of video frames to further compress the data.

As shown in the encoder block diagram on the left side of Fig.1, in actual CS application scenarios, the sensor directly senses the original image in blocks, that is, BCS sampling. measurements of keyframes are directly transmitted to the decoder; measurements of non-keyframes can be further compressed based on the temporal correlation between frames. Then the skip blocks are selected by calculating the similarity between all corresponding position blocks of the current frame and the reference frame. Skip blocks only needs to transfer the mark position information to the decoder, effectively reducing the time redundancy of the data.

For the selection of skip blocks, the traditional method is to

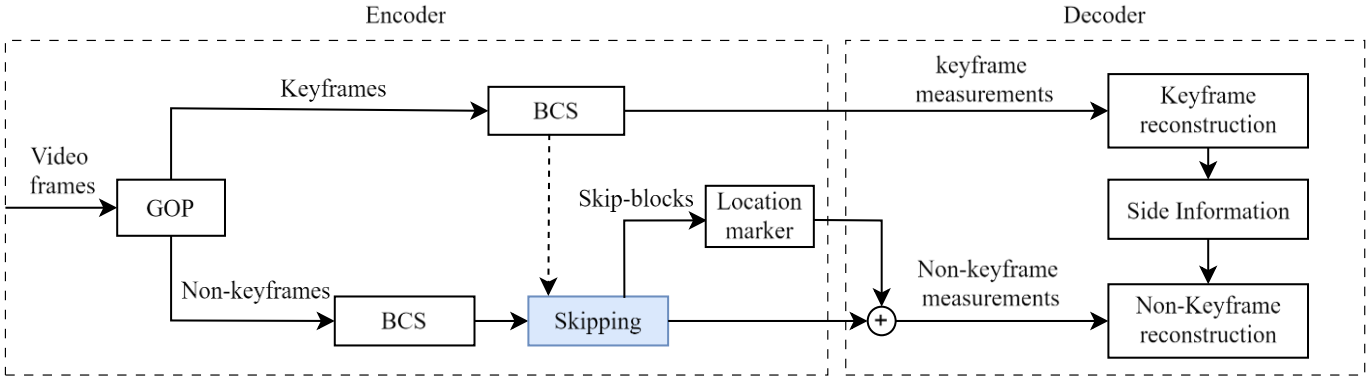


Fig.1 VCS system with skipping

type information is transmitted to the encoder through the feedback channel[20-22], which further increases the pressure on the channel. In the current mainstream skip block coding framework, image blocks are classified on an encoder with low computational complexity, and keyframes or adjacent frames are usually selected as reference frames.

The average skipping ratio in the GOP is specified in the MAS algorithm, and the skipping ratio of each frame is adaptively allocated according to the motion relationship between frames. It also recommends using the adjacent forward frame as a reference. Compared with representative skip coding algorithms such as ES[13], RS[23] and KAT[24], the skip block reference obtained by the MAS algorithm is more accurate. However, the algorithm calculates the correlation between the current frame and the forward neighboring frames. When the corresponding block of the forward adjacent frame is a skip block, only the corresponding block of the previous frame can be selected to replace the skip block, which may accumulate errors. At the same time, the algorithm uses the original image to calculate the correlation between adjacent frames, which means that the encoder requires a lot of storage space.

An encoder-end compression algorithm HSC for wireless visual sensor networks proposed in [19]. HSC essentially uses skip blocks to further compress the data processed by CS. This algorithm directly uses the measurements of the video frame sensed by the sensor to perform correlation analysis, which is more in line with actual application scenarios. This algorithm sets non-keyframes adjacent to keyframes as prime non-keyframes with a skipping ratio of 0. The prime non-keyframe is the reference frame for other non-keyframes. This algorithm wastes keyframe measurements. At the same time, the middle non-keyframe is far away from the prime non-keyframe in time. If the prime non-keyframe is used as the reference frame, the selected skip block error will be large, which will cause the decoder's image recovery quality to be reduced.

B. Decoder

As shown in Fig.1, the measurements generated by the encoder is transmitted to the decoder of the VCS system. The sampling rate of keyframes is high, so intra-frame reconstruction is used; the sampling rate of non-keyframes is low, so the side information provided by keyframes needs to be used to assist reconstruction. Improving the recovery quality of non-keyframes in VCS system decoders is a current research hotspot and difficulty. Classic reconstruction algorithms include MC, MH, RRS, etc. The reconstruction algorithm of the decoder is not the main content of this paper.

III. PROPOSED ALGORITHM

Under the framework of the VCS system, the proposed UTS algorithm set a dynamic threshold according to the correlation, and then selects some skip blocks. This solution enables higher compression rate compared to state-of-the-art algorithms without degrading the decoder's recovery quality.

A. Dynamic threshold settings

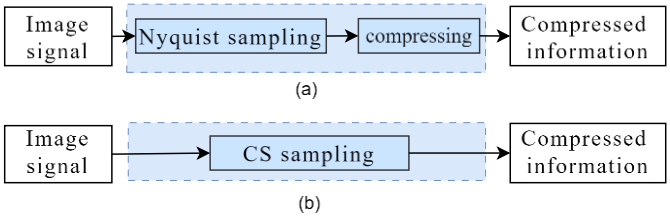


Fig. 2. Comparison of signal sampling methods: (a) Nyquist sampling. (b) CS sampling.

Fig.2 shows the advantages of using the compressive sensing algorithm in the encoder compared with the traditional Nyquist sampling scheme to obtain compressed information. As shown in Fig.2(a), after using Nyquist to sample the source data, it still contains a large amount of redundant information, which requires further data compression processing. The redundant information obtained by spending a lot of computing resources in the sampling stage is directly discarded in the compression stage. This processing method wastes a lot of computing resources and storage resources. As shown in Fig.2(b), CS technology completes the compression of the original signal during the sampling process, reducing the computational burden on the encoder and avoiding the problem of invalid information resource occupation.

Assuming that the original signal is $\boldsymbol{x} \in \mathbb{R}^{N\times 1}$, The CS sampling process is as follows:

$$\boldsymbol{y} = \boldsymbol{\Phi x} \tag{1}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N}$ is the measurement matrix, M is much smaller than N, and $\boldsymbol{y} \in \mathbb{R}^{M \times 1}$ is the measurement from the observation.

Because the number of columns N of the measurement matrix is the length of the observed signal $\boldsymbol{x}$, when the amount of image data is large, a measurement matrix with a large amount of data is needed. BCS sampling is proposed for this purpose, which divides the image into non-overlapping square blocks. BCS sampling uses the same measurement matrix $\boldsymbol{\Phi}_B$ to independently sample each block. The data volume of the block measurement matrix is smaller, which can effectively reduce the cache pressure on the encoder. Assume that the original input video frame is divided into a series of non-overlapping square blocks with fixed sizes. At this time, the original measurement matrix $\boldsymbol{\Phi}$ can be expressed as a block diagonal matrix composed of the measurement matrix $\boldsymbol{\Phi}_B$ of each image block, as shown in the formula:

$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\Phi}_B & & 0 \\ & \ddots & \\ 0 & & \boldsymbol{\Phi}_B \end{pmatrix} \tag{2}$$

In the reference [19], the similarity of the measurements of all corresponding position blocks of the current frame and the reference frame is calculated to determine the correlation between images, where the current frame and the reference frame are both non-keyframes. According to formula 1, all non-keyframes must use the same measurement matrix to determine the similarity of the image signal $\boldsymbol{x}$ through the measurements $\boldsymbol{y}$. Most existing compressive sensing frameworks use Gaussian random matrices as measurement matrices. All elements of the matrix are independently and identically distributed and obey a Gaussian distribution with a mean of 0 and a variance of $1/\sqrt{M}$, as shown in Formula 3.

$$\boldsymbol{\Phi} \sim N(0, \frac{1}{\sqrt{M}}) \tag{3}$$

A row vector in the measurement matrix is multiplied by the original signal to obtain the measurements. Since the sampling rate of keyframes is higher than that of non-keyframes, the length of the measurements M_1 of keyframes is greater than the length of the measurements M_2 of non-keyframes. When the first M_2 rows of the measurement matrix of keyframes are identical to the measurement matrix of non-keyframes, the keyframes can also be considered as reference frames, and the measurements in the first M_2 rows are valid data.

We propose to judge the degree of motion changes between frames based on the observed values and set a dynamic threshold. First, calculate the cosine similarity of all corresponding position block measurements of the current frame and the reference frame. Second, select a value within an appropriate range to calculate the average, and set this value as the threshold. Finally, when the similarity between the current block and the corresponding position block in the reference frame is higher than the threshold value, the current block can be marked as a skip block, and the mark position is only transmitted to the decoder, where the measurements of the corresponding position block in the reference frame can replace the mark position.

Cosine similarity is used to measure the consistency of the directions of two vectors and can reflect the similarity of the original image signals. The calculation formula of cosine similarity is:

$$s^j = \frac{\boldsymbol{y}_{cur}^j \cdot \boldsymbol{y}_{ref}^j}{\| \boldsymbol{y}_{cur}^j \| \times \| \boldsymbol{y}_{ref}^j \|} \tag{4}$$

Where $\boldsymbol{y}_{cur}^j$ represents the measurements of the j^{th} block in the current frame. $\boldsymbol{y}_{ref}^j$ represents the measurements of the j^{th} block in the reference frame. $\|*\|$ represents the modulus of the calculated vector.

The dynamic threshold needs to reflect the overall degree of motion change. We take the average cosine similarity as the threshold. When there is a significant change in motion at a certain position between frames, the calculated cosine similarity is very small. When calculating the threshold, it is necessary to first remove this outlier to prevent lowering the threshold. Define an indicator function $f(x)$, where $f(x)$ is 1 when x is within the range of (0.95, 1), otherwise it is 0.

$$f(x) = \begin{cases} 1 & 0.95 \leqslant x \leqslant 1 \\ 0 & x < 0.95 \end{cases} \tag{5}$$

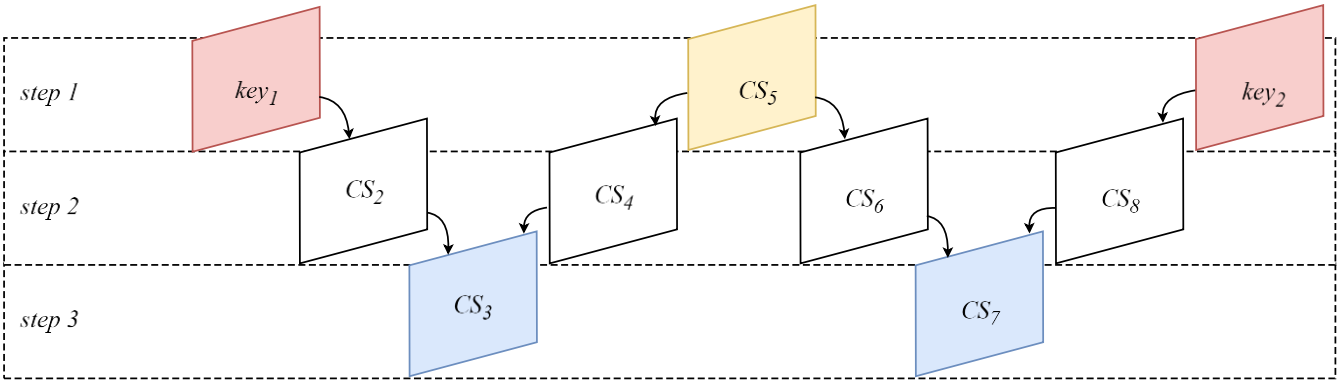


Fig.3 Illustration for reference frame selection

The threshold calculation formula is as follows:

$$\mathrm{T} = \frac{\sum_{j=1}^{num}(f(s^j)\cdot s^j)}{\sum_{j=1}^{num} f(s^j)} \tag{6}$$

Where *num* represents the number of non-overlapping blocks in a frame of image, the calculation formula is:

$$num = \frac{M \times N}{B^2} \tag{7}$$

Where B represents the pixel side length of the block. Dynamic thresholding can adaptively change the threshold based on the degree of motion changes in video frames, ensuring a certain skip ratio and labeling blocks with high similarity as skip blocks.

B. Reference frame selection

the non-keyframe adjacent to the keyframe is used as the prime non-keyframe, which is directly transmitted to the decoder after CS processing, and no skipping processing is used to further compress the data. The prime non-keyframe serves as a reference frame for selecting skip-blocks for other non-keyframes. As the time distance increases, the similarity between the current frame and the reference frame will decrease, and selecting only the prime non-keyframe as the reference frame for selecting skip blocks will bring errors.

We propose a new reference frame selection algorithm at the encoder, which is suitable for VCS systems with a GOP of 8. Table I gives the main pseudo code of the proposed UTS algorithm. As shown in Fig.3, key_1 and key_2 are keyframes. CS_i represents the i^{th} non-keyframe. The specific process of the algorithm is as follows:

Step 1: First, obtain an measurements $\{\boldsymbol{y}_1、\boldsymbol{y}_2、\cdots、\boldsymbol{y}_9\}$ in GOP based on the preset sampling rate, where $\boldsymbol{y}_9$ is the measurement of the keyframe in the next GOP. Set CS_5 as the prime non-keyframe without performing skipping processing. Firstly, keyframes and prime non keyframes can be used as reference frames for other non-keyframe skipping processing. At this point, the reference frames are uniformly distributed in the GOP, and the non-keyframes adjacent to the reference frames are CS_2, CS_4, CS_6 and CS_8. These non-keyframes can obtain more accurate block skipping references.

Step 2: Referring to the method of calculating dynamic thresholds in the previous text, calculate the thresholds between CS_2, CS_4, CS_6, CS_8,and adjacent reference frames. If the similarity of the corresponding position block exceeds the threshold, it can be marked as a skip block. When recovering skip block data at the decoder, there are:

$$\boldsymbol{y}_i^j = \boldsymbol{y}_{ref}^j[1{:}M_2] \ \text{ for} \begin{cases} i = 2,4,6,8 \\ s_i^j > \mathrm{T}_i \end{cases} \tag{8}$$

The subscript i represents the i^{th} frame, and the superscript j represents the j^{th} non-overlapping block, so s_i^j is the cosine similarity between the j^{th} block of the i^{th} frame and the corresponding position block of the reference frame. T_i is the threshold between CS_i and the reference frame. $\boldsymbol{y}$ is the

TABLE I

PSEUDO CODE OF THE PROPOSED UTS ALGORITHM

Algorithm 1: Uniform-reference Threshold-dynamic Skipping (UTS)

Input: A GOP $\boldsymbol{x}_i(i=1,2,\cdots,n)$ with the GOP size 8,
Keyframe Sensing Rate: R_k,
Non-Keyframe Sensing Rate: R_{cs},
the block size $B \times B$.

Output: Compressed vectors: $Y = \{Y_1, Y_2, \cdots, Y_n\}$,
Skip- blocks position marker: $P_{i,ref}^j$.

Initialization: Keyframe Measurement matrix: $\boldsymbol{\Phi}_{BK}$,
Non-keyframe Measurement matrix: $\boldsymbol{\Phi}_{BN}$,
No. of measurements corresponding to a keyframe block: M_1, No. of measurements corresponding to a non-keyframe block: M_2, No. of Blocks: num, Fixed high threshold: T_{high}
Skip- blocks position marker: $P_{i,ref}^j$.

1: Calculate the $\boldsymbol{x}_i$ measurement vectors $\boldsymbol{y}_i$ according to the BCS framework
Keyframe: $\boldsymbol{y}_i^j = \boldsymbol{\Phi}_{BK}\boldsymbol{x}_i^j : j \in [1, num], \boldsymbol{\Phi}_{BK} \in \mathbb{R}^{M_1 \times N}$
Non-keyframes: $\boldsymbol{y}_i^j = \boldsymbol{\Phi}_{BN}\boldsymbol{x}_i^j : j \in [1, num], \boldsymbol{\Phi}_{BN} \in \mathbb{R}^{M_2 \times N}$
2: **while** $i = 5$ **do**
3: Find cosine similarity $(\boldsymbol{y}_1, \boldsymbol{y}_5)$;
4: **for** $j = 1{:}\,num$ **do**
5: **if** Correlation $(\boldsymbol{y}_1^j, \boldsymbol{y}_5^j) > \mathrm{T}_{high}$ **then**
6: $\boldsymbol{y}_5^j = \boldsymbol{y}_1^j$; $P_{i,ref}^j = j$
7: **end if**
8: **end for**
9: **end while**
10: **for** $i = 2、4、6、8$ **do**
11: Find cosine similarity $(\boldsymbol{y}_{ref}, \boldsymbol{y}_i)$;
$ref = 1$ if $i = 2$; $ref = 5$ if $i = 4,6$; $ref = 9$ if $i = 8$
12: Calculate dynamic threshold T_i;
13: **for** $j = 1{:}\,num$ **do**
14: **if** Correlation $(\boldsymbol{y}_{ref}^j, \boldsymbol{y}_i^j) > \mathrm{T}_i$ **then**
15: $\boldsymbol{y}_i^j = \boldsymbol{y}_{ref}^j$; $P_{i,ref}^j = j$
16: **end if**
17: **end for**
18: **end for**
19: **for** $i = 3、7$ **do**
20: Find cosine similarity $(\boldsymbol{y}_{ref}, \boldsymbol{y}_i)$;
$ref = (\mathrm{T}_{i-1} > \mathrm{T}_{i+1})?\, i-1 : i+1$
21: Calculate dynamic threshold T_i;
22: **for** $j = 1{:}\,num$ **do**
23: **if** Correlation $(\boldsymbol{y}_{ref}^j, \boldsymbol{y}_i^j) > \mathrm{T}_i$ **then**
24: $\boldsymbol{y}_i^j = \boldsymbol{y}_{ref}^j$; $P_{i,ref}^j = j$
25: **end if**
26: **end for**
27: **end for**

measurement. Since the measurement vector length M_1 of the keyframe is larger than the sampling signal M_2 of the non-keyframe, the first M_2 items of $\boldsymbol{y}_{ref}^{j}$ are taken. When i is equal to 4 or 6, ref is 5. When i equals 2, ref is 1. When i is equal to 8, ref is 9.

Step 3: For CS_3 and CS_7, there are two options when selecting adjacent frames as reference frames. We expect to select the reference frame that is more similar to the current frame. If the similarity between the current frame and the previous frame is calculated separately, and then the reference frame with high similarity is selected, the calculation amount of the encoder is bound to increase. Taking CS_3 as an example, before processing CS_3, we calculated the threshold between key_1 and CS_2, and the threshold between CS_4 and CS_5. The size of the threshold reflects the degree of motion changes between frames. Therefore, we can compare the sizes of T_2 and T_4, and then choose the neighboring frame with the higher threshold as the reference frame.

C. High threshold for processing intermediate frames

In the above algorithm, the prime non-keyframes need to be used as reference frames and do not undergo block skipping processing. If the motion change of the video sequence is small or no motion change occurs. There will still be information redundancy in the prime non-keyframes. Therefore, it is proposed to use high threshold processing for intermediate frames, that is, to perform block skipping operations for CS_5 as well. Using keyframe key_1 as the reference frame, set a high fixed threshold so that only the corresponding position blocks with extremely high similarity can be marked as skip blocks, ensuring that the prime non- keyframe remains reliable as the reference frame. Meanwhile, for video sequences with small motion changes, the encoding information can be further compressed.

IV. EXPERIMENTAL RESULTS AND COMPARISON

In this section, the proposed UTS algorithm is compared with the state-of-the-art HSC algorithm[19], MAS algorithm[18], and the common KAT algorithm[24] to evaluate the performance of the proposed algorithm. The video sequences used in the experiments are Aikyo, Forman, Coastguard, and Football, with increasing levels of motion variations in these four sets of video sequences. To ensure the reliability of the conclusions, tests are conducted on different video sequences and sampling rates to examine the skip block rate in the encoder and the recovery quality in the decoder under different scenarios using the proposed algorithm. Additionally, the algorithm complexity of the existing algorithms and the proposed algorithm is compared and analyzed. For parameter settings, the GOP is set to 8. In the encoder, each frame image is divided into non-overlapping blocks of size 16×16. The sampling rate R_k for keyframes is set to 0.7, and the sampling rate R_{cs} for non-keyframes varies from 0.1 to 0.4. Peak Signal-to-Noise Ratio (PSNR) is used to evaluate the image recovery quality at the decoder for different algorithms. To ensure fairness, all experiments use the measurements for decision-making at the encoder and employ the RRS algorithm for image reconstruction at the decoder. All experiments were conducted on a computer equipped with an Intel Core i7-7700HQ CPU, 8GB RAM, and Windows 10 operating system, using MATLAB R2023a.

A. Peak Signal-to-Noise Ratio, and Skipping Ratio

We conducted tests on each algorithm using video sequences with different sampling rates, varying levels of motion variations, and different scene complexities. UTS- represents that the intermediate frame will not be set to a high threshold for block skipping processing. We conducted experiments using UTS and UTS- to evaluate the impact of setting a high threshold for block skipping of intermediate frames. KAT is a skipping algorithm with fixed threshold. In order to reflect the performance of UTS, we first use the UTS algorithm to conduct experiments, record the average threshold of the encoder, and set it as the fixed threshold to be used in KAT algorithm. UTS-, UTS and HSC algorithms all use dynamic thresholds, and the skip ratio is related to the degree of motion change. Table 2 shows the average block skipping ratio of non-keyframes in the video sequence of these four algorithms. The block skipping ratio of MAS is fixed, so no comparison is made. It can be seen from the table that UTS always has the highest block skipping ratio for any video sequence and sampling rate. Compared with the state-of-the-art HSC algorithm, this algorithm has a higher block skipping ratio, that is, it removes more redundant information. For the video sequence Aikyo with small motion

TABLE II
AVERAGE SKIPPING RATIOS FOR DIFFERENT ALGORITHMS

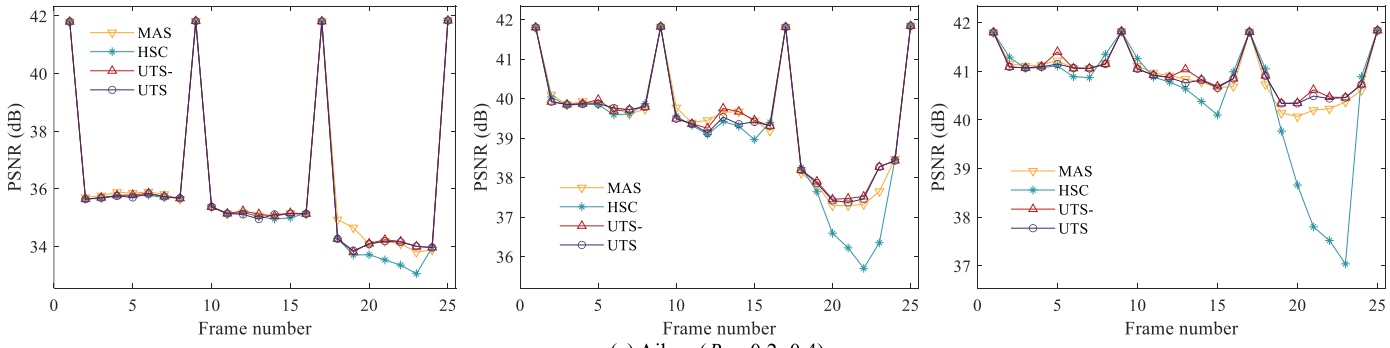
Subrate pair	Algorithm	Video sequence			
		Aikyo	Foreman	Coastguard	Football
(0.7,0.4)	KAT	61.6%	49.3	52.3	42.8
	HSC	61.8%	52.5%	56.5%	45.8%
	UTS-	61.1%	50.8%	57.9%	47.1%
	UTS	**75.5%**	**61.1%**	**63.7%**	**53.2%**
(0.7,0.3)	KAT	59.8%	49.1%	53.7%	42.7%
	HSC	61.3%	52.7%	56.8%	46.2%
	UTS-	61.2%	50.9%	58.7%	47.8%
	UTS	**75.6%**	**60.6%**	**64.9%**	**54.1%**
(0.7,0.2)	KAT	59.9%	50.3	53.2%	42.9%
	HSC	61.4%	53.5%	56.7%	47.4%
	UTS-	61.3%	49.2%	59.0%	48.4%
	UTS	**75.4%**	**59.1%**	**65.6%**	**54.8%**
(0.7,0.1)	KAT	59.2%	50.7%	54.9%	41.8%
	HSC	60.2%	**54.7%**	**58.1%**	49.1%
	UTS-	54.8%	42.0%	53.0%	43.5%
	UTS	**67.1%**	49.9%	58.0%	**49.2%**

changes, UTS has a significantly higher block skipping ratio than UTS-. For football, the jump rate improvement is weak. It shows that for video sequences with small motion changes, the information of prime non-keyframes is redundant, and the UTS algorithm can effectively improve the encoder compression rate. Table III shows the average PSNR of different algorithms in decoders and non-keyframes. Although UTS further compresses the measurements of the intermediate frames at the encoder compared to UTS-, at the decoder, for the same video sequence, UTS does not significantly reduce the recovery quality. The PSNR difference between the two algorithms is mostly 0.1dB. The block skipping ratio of MAS algorithm is a fixed value set artificially, and KAT is a fixed threshold algorithm. In order to ensure the fairness of the experiment, the block skipping ratio of MAS algorithm is set to the average block skipping ratio of UTS in the corresponding scene, and the KAT threshold is set to the average threshold of UTS in the corresponding scene. As shown in table III. It can be observed that UTS performs better than MAS and KAT in terms of recovery quality at the decoder under the same block skipping ratio.

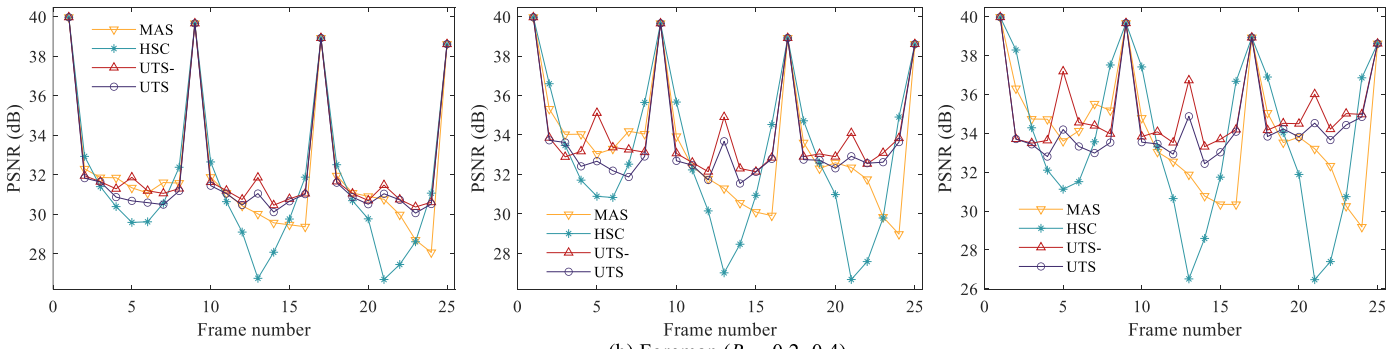
Fig.4 shows the comparison of frame-by-frame PSNR (Peak Signal-to-Noise Ratio) when different algorithms are used in the encoder. When using the HSC algorithm, the prime non-keyframes adjacent to keyframes have higher recovery quality, but as the distance from the keyframe increases, the recovery quality significantly decreases. This indicates that the reference value of prime non-keyframes adjacent to keyframes gradually diminishes over time. When using the MAS algorithm, in most cases, the recovery quality of non-keyframes shows a decreasing trend, but the rate of decrease is lower than that of

TABLE III
AVERAGE PSNR FOR DIFFERENT ALGORITHMS UNDER DIFFERENT NON-KEYFRAME SUBRATES

Subrate pair	Algorithm	Video sequence			
		Aikyo	Foreman	Coastguard	Football
(0.7,0.4)	KAT	40.36	32.14	29.43	24.27
	MAS	40.77	33.10	30.19	24.89
	HSC	40.26	32.74	30.21	24.43
	UTS-	40.86	34.46	31.16	25.35
	UTS	40.82	33.69	31.07	25.33
(0.7,0.3)	KAT	38.51	30.58	28.77	23.10
	MAS	39.02	32.35	29.31	23.85
	HSC	38.71	31.80	29.31	23.51
	UTS-	39.06	33.20	30.06	24.17
	UTS	39.02	32.66	29.98	24.15
(0.7,0.2)	KAT	34.29	29.64	26.39	22.04
	MAS	35.06	30.71	27.03	22.53
	HSC	34.83	30.12	26.95	22.36
	UTS-	34.99	31.17	27.34	22.79
	UTS	34.97	30.88	27.32	22.76
(0.7,0.1)	KAT	19.25	18.82	18.93	17.86
	MAS	19.79	19.91	19.67	18.19
	HSC	19.33	20.01	19.66	18.07
	UTS-	19.33	20.08	19.70	18.17
	UTS	19.34	19.89	19.72	18.17



(a) Aikyo (R_{cs}=0.2~0.4)



(b) Foreman (R_{cs}=0.2~0.4)

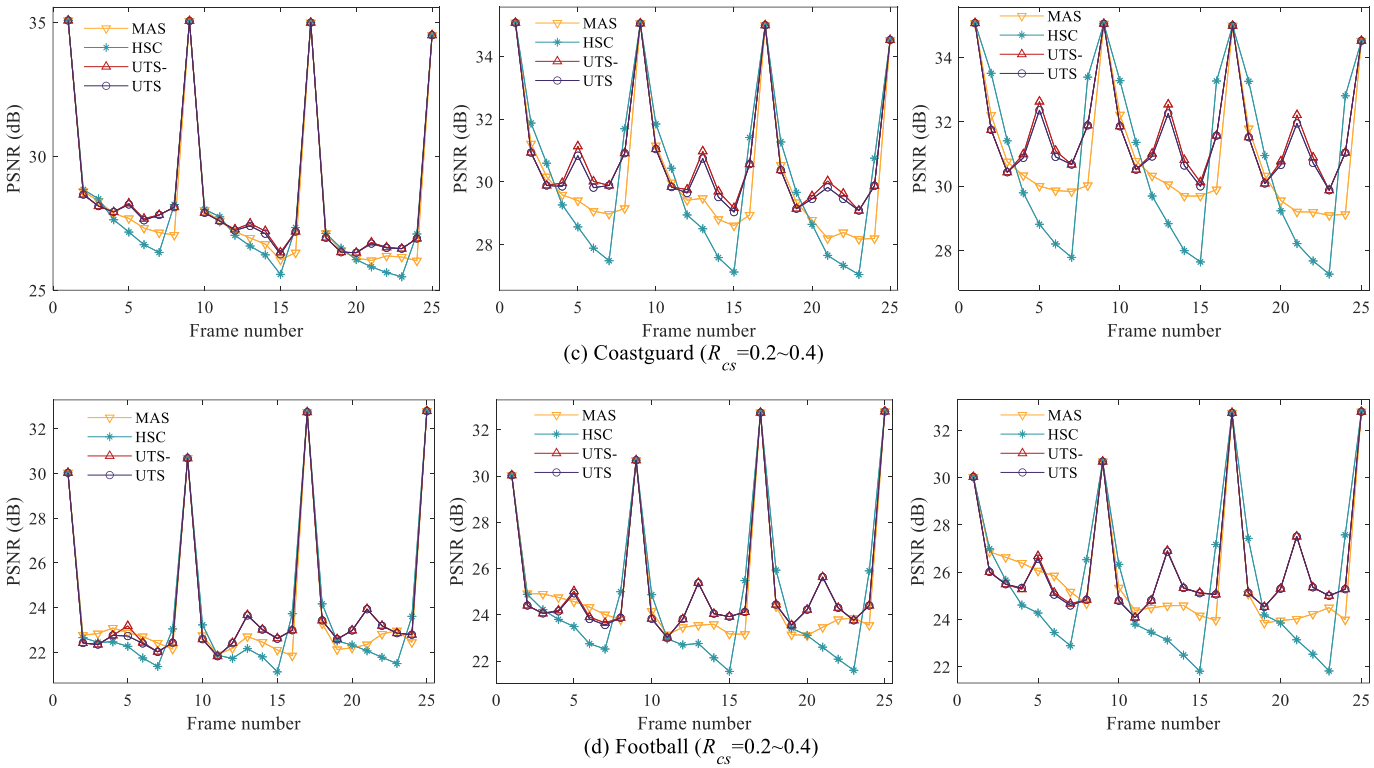
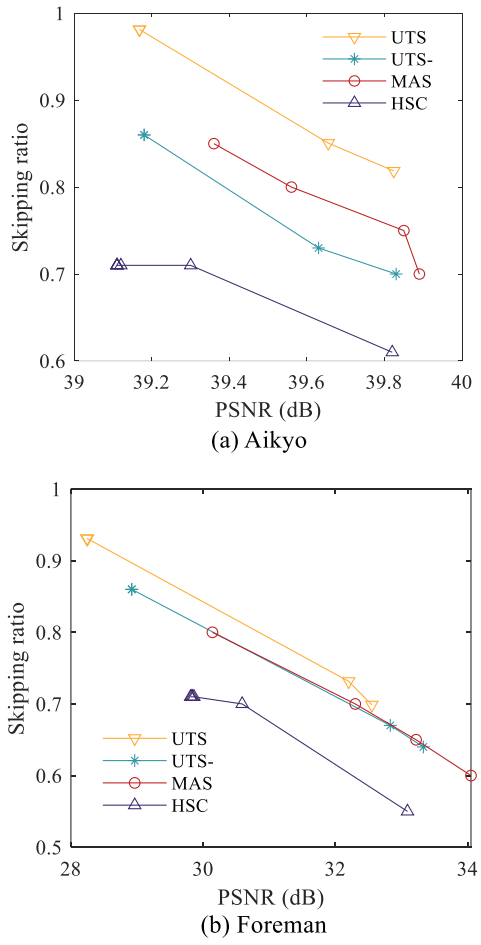


Fig.4 Frame-by-frame PSNR results for different algorithms

the HSC algorithm. This suggests that using neighboring frames as reference frames provides more accurate results. The recovery quality of UTS and UTS- algorithm is similar in most cases, and the recovery quality of the middle frame is very high, forming a small peak value. The non-keyframes are referenced by adjacent frames, which makes the overall recovery quality better than other algorithms.

Fig.5 shows the relationship between the average recovery quality and sampling rate of non-keyframes of the decoder using different algorithms. As the sampling rate increases, the observation vectors become longer and the selection of skip blocks is usually more reliable. It can also be seen from the picture that as the non-keyframe sampling rate increases, the advantages of UTS become more obvious. For the video sequence Aikyo with minimal motion changes, there will not be a big difference in whether the reference frame is a keyframe or a neighboring frame, so the relative gain of UTS is smaller.

The block skipping ratio and PSNR are inversely related. For the same video sequence, if the block skipping ratio is set high, or the threshold is lowered to obtain a higher block skipping ratio, the PSNR of the decoder will decrease accordingly. Fig.5 shows the relationship between PSNR and block skip ratio in different video sequences for several existing better algorithms and the proposed algorithm. From left to right, the degree of motion change of the video sequence gradually becomes larger, and the slope of the curve becomes larger and larger. It shows that the greater the change in the motion of the video sequence, the greater the impact of the block skipping ratio on the recovery quality of its decoder, which is consistent with



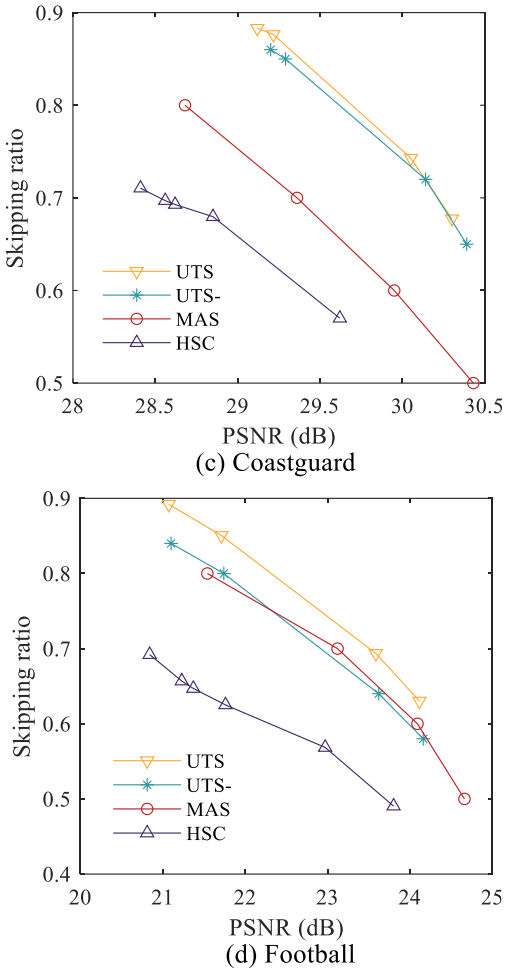


Fig.5 Variation curves of block skipping ratio and PSNR of different algorithms

cognitive logic. It shows that the greater the change in the motion of the video sequence, the greater the impact of the block skipping ratio on the recovery quality of its decoder, which is consistent with cognitive logic. Among the four video sequences, under the same recovery quality, the UTS algorithm has the highest block skipping ratio. This shows that under the same decoder recovery quality requirements, the UTS algorithm needs to transmit less information, and information redundancy is effectively reduced.

B. Computational Complexity of Encoder

Table 4 describes the encoding time of non-keyframes of the encoder. The running time of encoder CPU has nothing to do with the type of signal, but with the order of magnitude of the signal. We select the Foreman video sequence, and calculate the total running time of CPU processing a GOP signal at the encoder under different sampling rates of the proposed algorithm and the existing algorithm. With the increase of sampling rate, the observation vector length of non-keyframes will increase. Therefore, when calculating the correlation between the current frame and the reference frame according to the observation vector, the calculation time with high sampling rate is longer. The MAS algorithm first calculates the similarity of all non-intersecting blocks between the current frame and the reference frame, and then sorts them according to the similarity, so the running time is relatively longer. Compared with UTS, UTS has higher threshold processing in the middle frame, so the relative running time is higher. Compared with MAS and HSC, UTS algorithm has more advantages in complexity. Compared with KAT, the running time is slightly increased.

TABLE IV
CPU RUNTIME OF DIFFERENT ALGORITHMS (s)

Algorithm	non-keyframe subrates			
	0.1	0.2	0.3	0.4
KAT	0.102	0.115	0.127	0.146
HSC	0.157	0.168	0.174	0.175
MAS	0.192	0.201	0.209	0.214
UTS-	0.119	0.122	0.137	0.154
UTS	0.137	0.138	0.149	0.169

V. CONCLUSION

Our UTS algorithm improves the reliability of skip block reference by using the intermediate frame as reference. At the same time, the high threshold method is used to process the intermediate frame, which further improves the compression rate of the encoder. Several video sequences with different motion degrees are analyzed experimentally at different sampling rates. The results show that the recovery quality of the decoder and the compression rate of the encoder are improved compared with the existing algorithms.

REFERENCES

[1] E. J. Candes, J. Romberg and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489-509, Feb. 2006, doi: 10.1109/TIT.2005.862083.

[2] L Hao, Z Haoran, HUANG Ronget, "Region-hierarchical predictive coding for quantized block compressive sensing," *Journal of Beijing University of Aeronautics and Astronautics*, vol. 48, no. 8, pp. 1376-1382, 2022, doi: 10.13700/j.bh.1001-5965.2021.0511

[3] Y Ji, Z Kang, X Zhang, L Xu, "Model recovery for multi-input signal-output nonlinear systems based on the compressed sensing recovery theory." *Journal of the Franklin institute*, pp. 2317-2339. 2022, doi: 10.1016/j.jfranklin.2022.01.032

[4] E. J. Candes and M. B. Wakin, "An Introduction To Compressive Sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21-30, March 2008, doi: 10.1109/MSP.2007.914731.

[5] S. Wang, L. Yu and S. Xiang, "A Low Complexity Compressed Sensing-Based Codec for Consumer Depth Video Sensors," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 4, pp. 434-443, Nov. 2019, doi: 10.1109/TCE.2019.2929586.

[6] Lu Gan, "Block Compressed Sensing of Natural Images," *2007 15th International Conference on Digital Signal Processing*, Cardiff, UK,

2007, pp. 403-406, doi: 10.1109/ICDSP.2007.4288604.

[7] T. T. Do, Yi Chen, D. T. Nguyen, N. Nguyen, L. Gan and T. D. Tran, "Distributed compressed video sensing," *2009 16th IEEE International Conference on Image Processing (ICIP)*, Cairo, Egypt, 2009, pp. 1393-1396, doi: 10.1109/ICIP.2009.5414631.

[8] L. -W. Kang and C. -S. Lu, "Distributed compressive video sensing, "*2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan, 2009, pp. 1169-1172, doi: 10.1109/ICASSP.2009.4959797.

[9] H. Liu, R. Huang and H. Yuan, " Survey on compressive sensing video stream for uplink streaming media," *Journal of Image and Graphics*, vol. 26, no. 7, pp. 1545-1557, July 2021, doi: 10.11834/jig.200487

[10] S. Mun and J. E. Fowler, "Residual Reconstruction for Block-Based Compressed Sensing of Video," *2011 Data Compression Conference*, Snowbird, UT, USA, 2011, pp. 183-192, doi: 10.1109/DCC.2011.25.

[11] E. W. Tramel and J. E. Fowler, "Video Compressed Sensing with Multihypothesis," *2011 Data Compression Conference*, Snowbird, UT, USA, 2011, pp. 193-202, doi: 10.1109/DCC.2011.26.

[12] C. Zhao, S. Ma, J. Zhang, R. Xiong and W. Gao, "Video Compressive Sensing Reconstruction via Reweighted Residual Sparsity," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 6, pp. 1182-1195, June 2017, doi: 10.1109/TCSVT.2016.2527181.

[13] JE Fowler, S Mun and EW Tramel, "Block-Based Compressed Sensing of Images and Video, " *Foundations & trends in signal processing*, 2012, pp, 297-416. https://doi.org/10.1561/2000000033

[14] S. Zheng, X. -P. Zhang, J. Chen and Y. Kuo, "A High-Efficiency Compressed Sensing-Based Terminal-to-Cloud Video Transmission System," *IEEE Transactions on Multimedia*, vol. 21, no. 8, pp. 1905-1920, Aug. 2019, doi: 10.1109/TMM.2019.2891415.

[15] A. S. Unde and D. P. Pattathil, "Adaptive Compressive Video Coding for Embedded Camera Sensors: Compressed Domain Motion and Measurements Estimation," *IEEE Transactions on Mobile Computing*, vol. 19, no. 10, pp. 2250-2263, 1 Oct. 2020, doi: 10.1109/TMC.2019.2926271.

[16] J. Wang, W. Wang and J. Chen, "Adaptive Rate Block Compressive Sensing Based on Statistical Characteristics Estimation," *IEEE Transactions on Image Processing*, vol. 31, pp. 734-747, 2022, doi: 10.1109/TIP.2021.3135476.

[17] R Li, Y Yang and F Sun, " Green visual sensor of plant: an energy-efficient compressive video sensing in the internet of things," *Frontiers in plant science*, 2022, 13: 849606, https://doi.org/10.3389/fpls.2022.849606

[18] W Yuan, H Liu, "Motion-adaptive adjacent-reference skipping for distributed video compressive sensing with general decoders." *Optoelectron. Lett. 18*, 2022, pp. 755–762 https://doi.org/10.1007/s11801-022-2069-7

[19] G. L. Priya and D. Ghosh, "An Effectual Video Compression Scheme for WVSNs Based on Block Compressive Sensing," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 2, pp. 1542-1552, March-April 2024, doi: 10.1109/TNSE.2023.3325279.

[20] Z. Liu, A. Y. Elezzabi and H. V. Zhao, "Maximum Frame Rate Video Acquisition Using Adaptive Compressed Sensing," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 11, pp. 1704-1718, Nov. 2011, doi: 10.1109/TCSVT.2011.2133890.

[21] J. Zhu, Y. Zhang, G. Han and C. Zhu, "Block-based adaptive compressed sensing with feedback for DCVS," *9th International Conference on Communications and Networking in China*, Maoming, China, 2014, pp. 625-630, doi: 10.1109/CHINACOM.2014.7054371.

[22] X Zhang, A Wang, B Zeng, L Liu, " Adaptive distributed compressed video sensing," *J. Inf. Hiding Multim. Signal Process*., 2014, vol. 5, no. 1, pp. 98-106.

[23] C. Zhao, S. Ma, J. Zhang, R. Xiong and W. Gao, "Video Compressive Sensing Reconstruction via Reweighted Residual Sparsity," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 6, pp. 1182-1195, June 2017, doi: 10.1109/TCSVT.2016.2527181.

[24] A. S. Unde and D. P. Pattathil, "Adaptive Compressive Video Coding for Embedded Camera Sensors: Compressed Domain Motion and Measurements Estimation," *IEEE Transactions on Mobile Computing*, vol. 19, no. 10, pp. 2250-2263, 1 Oct. 2020, doi: 10.1109/TMC.2019.2926271.