EURASIP Journal on Advances in Signal Processing

RESEARCH **Open Access**

An enhanced video compression approach through RLAH encoding and KDENN algorithms

D. V. Manjunatha[1], Dattathreya[1], Umair Khan[2,3,4*], G. K. Siddesh[1], S. V. Prabhakar[5], B. R. Sreenivasa[6], Taseer Muhammad[7] and Ahmed M. Hassan[8]

*Correspondence: umair.khan@lau.edu.lb

[1] Department of Electronics & Communication Engineering, Alva's Institute of Engineering & Technology, Moodbidri, Dakshina Kannada, India
[2] Department of Mathematical Sciences, Faculty of Science and Technology, Universiti Kebangsaan Malaysia, UKM, Bangi 43600, Selangor, Malaysia
[3] Department of Mathematics, Faculty of Science, Sakarya University, Serdivan/Sakarya 54050, Turkey
[4] Department of Computer Science and Mathematics, Lebanese American University, Byblos 1401, Lebanon
[5] Department of Electronics, Maharani's Science College for Women (Autonomous), Mysore 570005, India
[6] Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere, India
[7] Department of Mathematics, College of Science, King Khalid University, Abha, Saudi Arabia
[8] Mechanical Engineering, Future University in Egypt, New Cairo 11835, Egypt

Abstract

Recently, video transmission is going through many failures because of the limited size of the top-notch technique for storing large volume videos. Thus, video compression (VC) techniques are introduced, which try to eradicate various sorts of redundancies within or betwixt video sequences. However, the VC often falls short to maintain a good quality of compression if motion discontinuities are present in the video frames (VF). To trounce this challenge, this paper proposes an enhanced VC approach via run length-based ASCII Huffman (RLAH) encoding, Kernel-based deep Elman neural network (KDENN), together with modified Kalman filters (MKF) algorithms. Initially, the video is transmuted into frames, and the frame's color space model (CSM) is changed as of RGB to YC_bC_r. Next, the frames are bifurcated into [8 × 8] blocks, and the significant features are extracted as of every block. On account of these features, the KDENN identifies the motion of every block. Those blocks directly undergo a compression process in case of a single motion. Otherwise, MFK smoothens those blocks in order to eradicate the jitters and undesired movements, and then, it goes through compression. After that, RLAH encoding compresses the VF. Then, on the other side, the RLAH decoding algorithm decomposes the video. The results exhibit that the proposed work renders good quality videos with high PSNR value and also it trounces the prevailing compression techniques concerning compression ratio (CR).

Keywords: Video compression, Block motion, Kernel-based deep Elman neural network (KDENN), Run length-based ASCII Huffman (RLAH) encoding, Compression ratio

1 Introduction

A sequence of orthogonal bitmap digital images called frames displayed in fast sequence at a stable rate for providing the delusion of a motion picture is digital video [1]. Strict requirements on video storage time are possessed by multiple areas, namely the railway transportation industry, civil aviation industry, schools, along with banks [2]. The usage of ever-augmenting bandwidth in addition to extensive smartphones has caused a wide flood of video streams on public along with private networks in recent years [3]. In YouTube, each minute video of about 300 h is uploaded, and almost five billion videos are seen on YouTube daily as per the latest statistics [4]. But most videos are necessary to be coded first for decreasing the data amount on account of the restricted quantity of transmission bandwidth along with storage capacity [5]. A crucial part is played by VC in giving higher-quality video services under the restricted capabilities of transmission networks along with storage [6].

The bit rate is reduced for transmitting and storing information while maintaining the video quality is the basic objective of VC [7]. Further, the transmission costs are lessened, and thus, the energy resources are saved and increased the effective network capacity [8]. During compression, a redundancy that occurred across disparate regions in a frame could be considerably decreased through the efficient exploration of the characteristics of the region being coded [9]. Compression techniques comprise of '2' types: lossless compression (LLC) along with lossy compression (LC). The LC is irrevocable and inexact approximations are utilized for the depiction of original images [10]. After compression, the information is not lost in lossless image compression [11]. The best one for obtaining identical information is LLC. This is specifically vital for a few applications, like automated diagnostic systems, where each pixel is important [12].

Popular compression techniques, namely run length encoding (RLE), arithmetic encoding, Lempel–Ziv–Welch (LZW), along with Huffman coding (HC), are examples of LLC techniques [13]. Currently, the most often utilized LLC algorithms have developed from an amalgamation of the algorithms explained above with some other particular algorithms [14]. The data's quality is not affected by these techniques. If there are jitters or undesired movements like a vibration within the frames, then they ought to be removed. However, the algorithms could not eliminate those distortions. Severe degradation is caused by such distortions in the quality of experience on the decoder [15]. The performance along with the accuracy of a few video processing applications, like target recognition along with video classification, is affected by the video quality's severe degradation on the decoder along with the compression effects. Therefore, the compressed video's quality is needed to be improved efficiently [16].

The utmost apt filter algorithm is vital to be found for blocks in a frame for improving the video compressibility while maintaining the perceived quality to achieve top-quality images or videos at the decoder [17, 18]. For generating distortion-free compressed videos, many compression distortion reduction algorithms are proposed lately. However, distortion-free videos are not provided by any of them along with that excellent CR and improvements are still needed to attain good results. Therefore, an enhanced VC approach was proposed by this research methodology via RLAH encoding, KDENN, and MKF algorithms.

This paper is categorized as: Sect. 2 talks about the existing works with their outcomes and the limitations. Section 3 describes the proposed lossless VC method with suitable diagrams. Section 4 examines the proposed methodology's outcomes and evaluates the existing techniques. Lastly, the paper is concluded by Sect. 5 with future guidelines.

2 Literature survey

Antony and Sreelekha [19] proffered 2 selective intra-prediction plans (SIP) that choose the top prediction design as of the block-centered along with sample-centered predictions at the pixel level. The SIP plan was useful in angular prediction modes in the SIP-A, whereas the combined SIP method (SIP-C) utilized the SIP approach in the angular along with planar prediction modes. The present modern SIP algorithm's performance was ameliorated by the SIP-C method. Aimed at evading the enormous overrides needed to report forecast selection as of the encoder to the decoder, SIP algorithms utilized an extremely low bit (LSP) pick backpacking strategy. Though better compression was offered by the SIP, SIP's run time was not reduced to the expected level.

Irannejad and Mahdavi [20] established a VC technique centered on sparse along with redundant illustration utilizing the K-singular values decomposition (K-SVD) along with iterative least square dictionary learning algorithm (ILS-DLA). The VFs were split into current frames along with reference frames. Concerning the frames of reference, the motion-compensated residuals (MCR) along with the motion vectors of current frames were assessed. The reference frames along with MCR's sparse codes were attained utilizing entropy-coded, learned dictionaries, and it was saved or sent to the motion coding domain together with the receiver. Here, the ILS-DLA along with K-SVD techniques was utilized in the discrete cosines transformation (DCT) field. However, aimed at lossless VC, the field of DCT was not well suitable.

Pal and Bit [21] offered a lightweight VC method, the low overhead spatio-temporal VC method which was made by compression of intra- along with inter-frame. Utilizing an interpolation search-centered method along with an edge detection scheme, redundant frames were eliminated in compression of inter-frame. Utilizing the adaptive column dropping method, compression of intra-frame was carried out. For ameliorating reconstruction quality, two reconstruction filters were utilized at the end of the receiver. Regarding energy consumption, experiential outcomes evinced that the method was better while obtaining an acceptable reconstruction quality. However, energy consumption was its only concentration and does not ponder the CR.

Rabie and Baziyad [22] proffered a VC technique that makes exploit of the temporary reluctant in segments video employing the concept of pixogram. The pixogram possessed the capability to transform the unrelated spatial areas of individual VFs into extremely relevant temporary vectors and thus augmenting the redundancy in the transition domain. Utilizing this strategy, high CR in video stream could be achieved. This compression method was the first to utilize the pixogram concept in VC, and it intended to dare the conventional trade-off related to high CRs which causes decreased visual quality.

Hemanth and Anitha [23] established a pattern-centered artificial bees colony (ABC) method aimed at block motion (BM) estimation in VC applications. Every VF was subpartitioned into macroblocks in the BM process. The present frame's every macroblock was analogized with the preceding frame. The method's major intend was reducing the sum absolute difference. Aimed at decreasing the computational expense, the ABC algorithm utilized an initial pattern. Concerning the search points along with convergence time, the computational expenditure was exhibited. Regarding the performance metrics, experiential outcomes evinced the method's amelioration when analogized with other block matching methods. However, more compression time was utilized by it.

Siddeq and Rodrigues [24] recommended a method aimed at the compression of the image along with video centered on 2-level DCT utilizing hexadata coding. In the transformation, the proffered HD coding technique was considerably disparate as of JPEG. Various additional steps at the stage of compression were incorporated. In this, the compression of higher-frequency data via the hexadata algorithm was the significant step that leads to the augmented CR, and also the data coding by utilizing 5 disparate keys which are created by a key generator. For recovering the large-frequency matrix, matching of binary feature method's utilization was made at the decompression stage employing 5 symmetric keys. Better compression was given by this method; however, execution time was very high aimed at large images.

Afonso et al. [25] modeled a VC frame centered on spatio-temporal resolution adaptation (ViSTRA) that resampled the input video temporally along with spatially during encoding which was centered on a quantization resolution decision, along with the reconstruction of full-resolution video at the decoder carried out. Utilizing frame repetition, temporal upsampling's execution was carried out. Aimed at spatial resolution upsampling, utilization of convolutional neural networks (CNN) super-resolution design was carried out. ViSTRA had been incorporated the higher-efficiency videos coding (HEVC) along with software. The major coding gains could be attained by applying the ViSTRA and were evinced as of the experiential outcomes. High complexity was this method's major limitations.

3 RLAH encoding-based enhanced video compression

VC is a method that lessens the data utilized to encode the video content. This lessening in data transcribes to benefits like small storage as well as lower transmission bandwidth needs, aimed at a clip of video content. To attain the best video watching experience, it is important to compress the videos devoid of losing the original quality; however, it is an extremely challenging task. To overcome this challenge and achieve enhanced VC, this paper proposes three different novel algorithms, say RLAH encoding, KDENN, and MKF. Certain steps are performed to design an effectual VC methodology. The proposed work encompasses '8' steps: (i) video to frame conversion, (ii) color transform, iii) dividing frames into $[8 \times 8]$ blocks, (iv) attribute extraction, (v) block motion classification, (vi) smoothening, (vii) compression, and, (viii) decompression. These steps are elucidated below in detail. Figure 1 exhibits the proposed VC method's workflow.

3.1 Video to frame conversion

The VC is carried out via VF. Thus, initially, the video is transmuted into frames. The sequence of frames is written as

$$Input\ video \rightarrow \left(\hat{F}_r; sequence(video\_frames)\right) \tag{1}$$

$$\hat{F}_r = \left\{\hat{f}_{r1}, \hat{f}_{r2}, \hat{f}_{r3} \ldots \hat{f}_{rN}\right\} \tag{2}$$

wherein $\hat{F}_r$ represent the sequence of frames set and $\hat{F}_{rN}$ implies the $_{N}$- number of frames.

3.2 Color transform

VF has significant correlations betwixt every signal, and it is composed of 'R,' 'G,' and 'B' (red, green, as well as blue) signals. RGB color space is not well fit for autonomous coding on account of the higher correlation. Thus, the RGB-CSM is transmuted into the

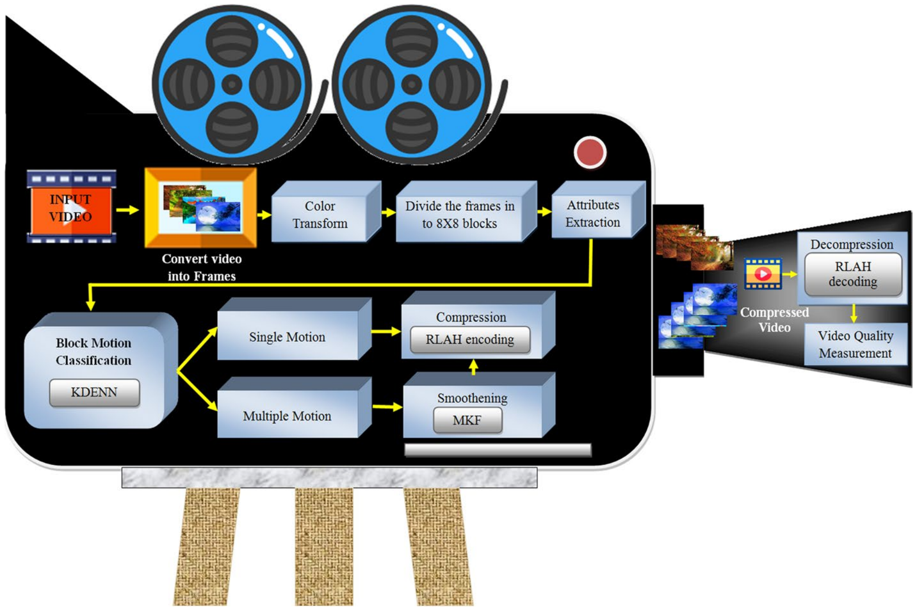


Fig. 1 Work flow of the proposed VC

CIE-Lab (Commission International d'Eclairage Lab) CSM. CIE-Lab is basically a perceptual uniform color space in which perceptual uniformity alludes to how '2' colors vary when a person notices those '2' colors. Thus, it can well be utilized to make precise color balance corrections via altering output curves on a$'$ as well as b$'$ components, or to regulate the lightness contrast utilizing the '3' component. In CIE-Lab, the '3' components L' signifies luma component that is illumination information and a$'$b$'$ implies the chroma information. L$'$ = 0 renders the black color and L$'$ = 0 renders white color. The a$'$, the values a$'$ > 0 imply magenta. The b$'$, the values b$'$ > 0 imply yellow.

Let consider the RGB-CSM of video frames $\hat{F}_r$ is $\hat{F}_r(r, g, b)$. As of these RGB values, the CIE-Lab values are attained via a nonlinear mapping of the $x'y'z'$ coordinates as:

$$x' = 0.4303\hat{F}_r(r) + 0.3416\hat{F}_r(g) + 0.1784\hat{F}_r(b) \tag{3}$$

$$y' = 0.2219\hat{F}_r(r) + 0.7068\hat{F}_r(g) + 0.0713\hat{F}_r(b) \tag{4}$$

$$z' = 0.0202\hat{F}_r(r) + 0.1296\hat{F}_r(g) + 0.939\hat{F}_r(b) \tag{5}$$

Utilizing Eqs. (3), (4), and (5), the CIE-Lab can well be derived as

$$L' = \left\{ \begin{array}{ll} 116h\left(\frac{y'}{y_m} - 16\right) & if\left(\frac{y'}{y_m}\right) > 0.00856 \\ 903.3 & otherwise \end{array} \right\} \tag{6}$$

$$a' = 500\left(h\left(\frac{x'}{x_m}\right) - h\left(\frac{y'}{y_m}\right)\right) \tag{7}$$

$$b' = 200\left(h\left(\frac{y'}{y_m}\right) - h\left(\frac{z'}{z_m}\right)\right) \tag{8}$$

wherein x_{m}, y_{m}, and z_{m} signify the tristimulus values of the illuminant. The conversion of RGB-CSM to CIE-Lab CSM is expressed as

$$\hat{F}_r(r,g,b) \overset{RGB\ to\ CIE-Lab}{\longrightarrow} \hat{F}_r(L',a',b') \tag{9}$$

wherein $\hat{F}_r(L',a',b')$ signifies the CIE-Lab CSM frame. CIE-Lab includes more color (even more compared to a person eye could see) than other color spaces.

3.3 Block motion identification

The target's movement regarding the preceding frame is determined in each successive video frame for eliminating the undesirable translational camera motions. Thus, a stabilized video is generated. The captured videos are simple to encompass a higher-frequency motion like vibration. The image's quality is affected by this high-frequency motion (multiple motions). Thus, for eliminating the motions from the frames, motion identification is conducted to identify the high-frequency motions within the frames. The finding of motion vectors (single motion, multiple motions) from neighboring frames in a video sequence is motion identification. The motion vectors might be the entire frame or an explicit part, for example, random-shaped patches, rectangular blocks, or else a single pixel. Each VF is separated into 8×8 blocks aimed at providing an accurate motion identification process. The obtained blocks are mathematically expressed as

$$b_j = \left\{(b_1, b_2, ..., b_{64})_1, ..., (b_1, b_2, ..., b_{64})_N;\ list(frame\_blocks)\right\} \tag{10}$$

where b_j indicates total blocks obtained from N-number of frames, $(b_1, b_2, ..., b_{64})_1$ and $(b_1, b_2, ..., b_{64})_N$ denotes the list of blocks of frame 1 and frame N, respectively. Few features are initially extracted from every block for training the classifier to distinguish the block motion.

3.4 Attribute extraction

Here, the vital features are taken as of every VF's block. Attributes like standard deviation, kurtosis, mean, skewness, block correlation, and also gray-level co-occurrences matrix (GLCM) are chosen aimed at the block motion detection procedure. The attributes are detailed as:

3.4.1 Mean ($\overline{b_j(u,v)}$)

This is an average that signifies the VF block's general brightness and is articulated as

$$\overline{b_j(u,v)} = \frac{1}{m \times n} \sum_{u=1}^{m} \sum_{v=1}^{n} b_j(u,v) \tag{11}$$

where $\sum_{u=1}^{m} \sum_{v=1}^{n} b_j(u,v)$ implies the addition of the block's every pixel value (PV); $m \times n$ is the VF's size.

3.4.2 Standard deviation ($\overline{S_d}$)

The measurement of the frequency distributions of a block's PV is termed as that block's standard deviation. It is articulated as

$$\overline{S_d} = \sqrt{\frac{1}{m \times n} \sum_{u=1}^{m} \sum_{v=1}^{n} \left(b_j(u,v) - \overline{b_j(u,v)} \right)^2} \tag{12}$$

3.4.3 Kurtosis ($\overline{K_s}$)

The parameter that determines the shape of a random variable's probability distribution is termed kurtosis that is enumerated as

$$\overline{K_s} = \frac{1}{m \times n} \sum_{u=1}^{m} \sum_{v=1}^{n} \frac{\left(b_j(u,v) - \overline{b_j(u,v)} \right)^4}{\overline{S_d}^4} \tag{13}$$

3.4.4 Skewness ($\overline{S_w}$)

It is stated as the measurement of symmetry or the nonexistence of symmetry. The VF's skewness is computed utilizing Eq. (14) as

$$\overline{S_w} = \sqrt{\frac{1}{m \times n} \sum_{u=1}^{m} \sum_{v=1}^{n} \frac{\left(b_j(u,v) - \overline{b_j(u,v)} \right)^3}{\overline{S_d}^3}} \tag{14}$$

3.4.5 Correlation ($C_{u,v}$)

Correlation computes the specific pixel pairs' joint occurrence probability.

$$C_{u,v} = \frac{\sum \left(u - \overline{b_j(u,v)} \right) \left(v - \overline{b_j(u,v)} \right)}{\sqrt{\sum \left(u - \overline{b_j(u,v)} \right)^2 \sum \left(v - \overline{b_j(u,v)} \right)^2}} \tag{15}$$

3.4.6 GLCM

The GLCM functions describe an image's texture by computing how frequently the pixel pairs comprising certain values in a certain spatial relationship prevail inside an image, generating a GLCM and then taking statistical measures as of this matrix. The GLCM's textural attributes are detailed as:

3.4.6.1 Energy ($\overline{E_n}$) Energy is determined centered on the block's normalized histogram.

$$\overline{E_n} = \frac{1}{m \times n} \sum_{u=1}^{m} \sum_{v=1}^{n} \left(b_j(u,v) \right)^2 \tag{16}$$

3.4.6.2 Entropy ($\overline{E_y}$) Entropy is stated as the randomness's statistical measure, which may be utilized aimed at characterizing the block's texture.

$$\overline{E_y} = \frac{1}{m \times n} \sum_{u=1}^{m} \sum_{v=1}^{n} \left(b_j(u,v)\right) \log_2 \left(b_j(u,v)\right) \tag{17}$$

3.4.6.3 Homogeneity ($\overline{H_y}$) It signifies the closeness of the elements' distribution as of the GLCM toward the GLCM's diagonal.

$$\overline{H_y} = \sum_{u=1}^{m} \sum_{v=1}^{n} \frac{\left(b_j(u,v)\right)}{1 + (u-v)^2} \tag{18}$$

3.4.6.4 Dissimilarity ($\overline{D_y}$) Dissimilarity is stated as the local intensity variation's measurement determined as the mean absolute differentiation betwixt the neighboring pairs.

$$\overline{D_y} = \sum_{u=1}^{m} \sum_{v=1}^{n} \left(b_j(u,v)\right) |u-v| \tag{19}$$

3.4.6.5 Cluster tendency ($\overline{C_t}$) Cluster tendency is stated as the measurement of voxels' groupings comprising equivalent gray-level values.

$$\overline{C_t} = \sum_{u=1}^{m} \sum_{v=1}^{n} \left(u + v - \overline{b_j(u)} - \overline{b_j(v)}\right)^2 \left(b_j(u,v)\right) \tag{20}$$

where $\overline{b_j(u)}$ implies $b_j(u)$'s mean intensity; $\overline{b_j(v)}$ implies $b_j(v)$'s mean intensity. The total extracted attributes are signified as $Z_{f,N}$.

$$extracted\_attributes \xrightarrow{\left(\overline{b_j(u,v)}, \overline{S_d}, \overline{K_s}, \overline{S_w}, C_{u,v}, \overline{E_n}, \overline{E_y}, \overline{H_y}, \overline{D_y}, \overline{C_t}\right)} Z_{f,N} \tag{21}$$

3.5 Block motion classification

Here, the classifier extracts a block's features and categorizes them as one amidst the '2' types either comprising single or multiple motions. Multiple motions signify speedy changes betwixt the image's pixels; the single motion signifies the steady alterations betwixt the image's pixels. The proposed work employs a KDENN classifier aimed at block motion categorization. The conventional Elman neural network is defined as a partial recurrent network design that comprises '4' layers: the inputted layer (IL), the hidden layer (HL), the context layer, and then the outputted layer (OL). The difference betwixt this network and the feed-forward neural network is the context layer's presence in the Elman network. The context layer is utilized to memorize the HL's output that is observed as a step delay operator. Aimed at decrementing the training error, greater than n-number of HLs are utilized in the proposed classifier that studies the data intensely by filtering the information via this n-number of the HLs. This deep learning provides an accurate categorization. Figure 2 exhibits the KDENN's structure.

The feature values extracted are inputted to the KDENN classifier's IL. IL comprises N -dimensional external input vectors $Z_{f,N}$. At $Z_{f,N}$'s arrival in IL, equivalent weight values

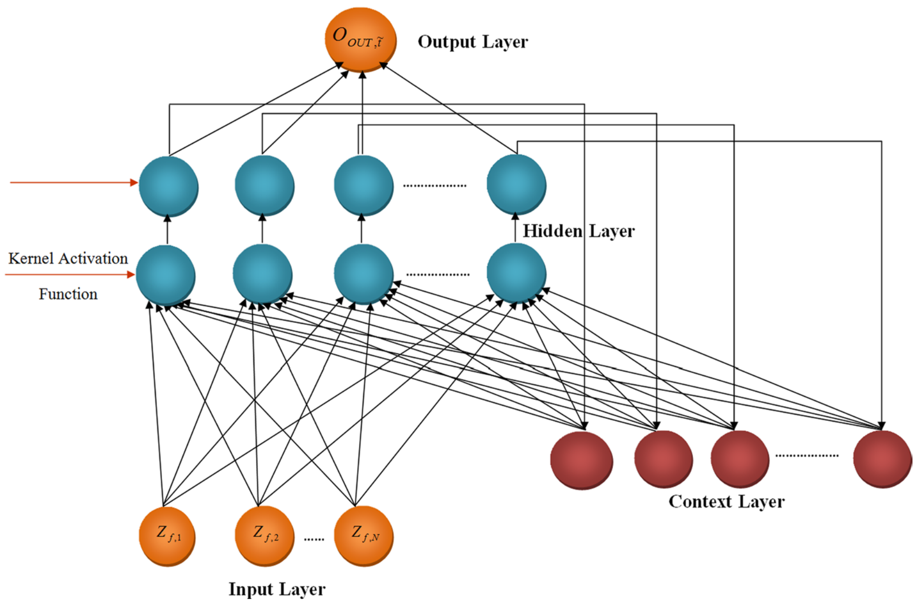


Fig. 2 Structure of KDENN

aimed at these inputted feature values are produced arbitrarily. Next, the inputted values and their equivalent weight values are sent to the HL and the HL's output can be equated as

$$O_{HID,\tilde{t}} = \kappa_H\left(W_{IH}\ O_{CON,\tilde{t}} + W_{HC}\left(Z_{f,N}\right)\right) \tag{22}$$

where $O_{HID,\tilde{t}}$ signifies the HL's output at $\tilde{t}$ iteration; $O_{HID,\tilde{t}}$ signifies the HL unit's output at $\tilde{t}$ iteration; $O_{CON,\tilde{t}}$ implies the context layer unit's output at $\tilde{t}$ iteration, correspondingly. The context layer's output is formulated as

$$O_{CON,\tilde{t}} = O_{HIDD,\tilde{t}-1} \tag{23}$$

where $O_{HIDD,\tilde{t}-1}$ implies the HL's output at $\tilde{t}-1$ iteration; W_{IH} implies weight of the IL to HL; W_{HC} signifies the weight of HL to context layer; κ_H symbolizes the kernel activation function. The κ_H is employed in the KDENN in place of the Gaussian activation function. The κ_H is revealed to comprise a strong performance in the deep learning neural networks. The κ_H is equated as

$$\kappa(\alpha) = \sum\nolimits_N \beta_N k(\alpha, d_N) \tag{24}$$

where $\kappa(\alpha)$ signifies the κ_H aimed at the input α; β_N implies the mixing coefficients; d_N symbolizes the dictionary elements; k implies kernel coefficient. Past the κ_H, the HL provides its output $O_{HID,\tilde{t}}$ as an input onto the OL. Gr on HL's output, the OL generates the end output as

$$O_{OUT,\tilde{t}} = \kappa_O\left(W_{HO}O_{HID,\tilde{t}}\right) \tag{25}$$

where $O_{OUT,\tilde{t}}$ signifies the OL unit; κ_O implies the OL's activation function; W_{HO} signifies the weight of HL to the OL. The final categorization outcome comprises '2' classes

('0,' '1'). Herein, '0' signifies a single motion; '1' implies multiple motions. In the single motion block, the VF directly undergoes a compression procedure, or else in the multiple motion blocks, a smoothing process is carried out aimed at those multiple motion blocks and that is implemented to the compression procedure past smoothing. The motion smoothening is executed aimed at eliminating the undesired movements and jitters, generally pondered as multiple motions, like a vibration in the captured video.

3.6 Smoothing

Here, the undesired movements and jitters prevalent in the captured videos are eliminated through the MKF. Kalman filter (KF) is stated as an algorithm, which smooths the VFs by the filtering of numerous motions. It offers a few unknown variables' estimates, provided the measures are noted over time. KFs are indicating their effectiveness in the noise smoothing process. The proposed work does a few alterations to the standard KFing procedure utilizing the variances–covariance matrix (VCM). The measurement error matrix (MEM) is utilized in the standard KF aimed at observation vector computation that provides the means for every variable. In MKF, a VCM is implemented in the MEM's place. VCM provides the variables' variances along the main diagonal and the co-variances betwixt every variable pair prevalent in the other matrix positions aimed at incrementing the smoothing performance with not affecting the frames' significant information.

KFing procedure comprises '2' phases: prediction and updation. Ponder a linear discrete-time dynamic system defined by the linear stochastic difference equation,

$$S(q+1) = \theta_q S_q + d_q \tag{26}$$

where S_q signifies the state vector; θ_q implies the state transition matrix; d_q symbolizes the disturbance vector; q signifies time index. Next, the smoothing block is attained as of various motions as

$$S_m(q) = \Phi_q S_q + n_q \tag{27}$$

where $S_m(q)$ signifies the smoothed blocks, Φ_q symbolizes the VCM; n_q implies the noise measurement. Utilizing Eqs. (26) and (27), the undesired movement and jitters are eliminated as of various motion blocks. Next, the VFs are implemented to the subsequently pace aimed at furthermore processing.

3.7 Video compression

The VF undergoes a major step of lessening the file size in this step. The video is compressed using a run length-centered ASCII Huffman (RLAH) encoding algorithm. The sequence of PV is initially suppressed using an RLE algorithm in RLAH, and then, corresponding ASCII values are calculated for those PV. The HC compresses the video grounded on these ASCII values. The step-by-step explanation of RLAH centered VC is explained as follows,

Step 1: Run length encoding

RLE is a very easy compression technique, which runs on the sequence encompassing the same PV occurring many successive times and the sequence is encoded for storing merely a single value along with its count. The frequency is not utilized for the single occurrence of a specific value as that would produce an overhead influencing the compression's efficiency.

The same PV in the frames is initially counted in RLE. After that, the encoded PV is written as PV with a count of PV, which is shown below.

Example for RLE

Input pixel values: 44 44 44 56 56 56 56 25 81 81

Encoded pixel values: 344 456 25 281.

A series of frequency-PV pairs are generated by the output PV, which is indicated as S_k. This new representation was forwarded to the next section for the ASCII value calculation.

$$\hat{F}(u,v)_i \xrightarrow{Suppress\,pixel\,values} S_k \tag{28}$$

where $\hat{F}(u,v)_i$ indicates the PV of i-th VF and S_k signifies the new PV generated using RLE.

Step 2: ***ASCII value calculation:*** The corresponding ASCII value is produced for the output PV in this step, which is obtained as of the preceding step.

$$S_k \xrightarrow{ASCII\ code\ conversion} A_c \tag{29}$$

where A_c symbolizes the ASCII code of S_k PV.

Step 3: ***Huffman coding:*** HC is an LLC algorithm. A variable-length code is allotted to input different characters in this algorithm. The code length is related to how the characters are utilized frequently. The smallest codes are possessed by the most frequent characters along with longer codes meant for the least frequent ones.

The Huffman encoding compression operates by means of generating a binary tree of nodes. A table was initially drawn, which comprises an ASCII column and frequency columns. Each ASCII value is entered in the ASCII column and the total occurrences of that value into the frequency column. All the nodes are grouped in ascending order centered on the frequency values. Next, a parent node is formed by joining 2 least frequency value nodes. Then, these 2 frequency values are added and labeled as the parent node. This step is recurred till all the nodes are formed. The resultant parent nodes are merged for generating a new parent node subsequent to node creation; then, child node values are added, and also it is labeled as the parent node. If any node remains, then it will be added to the uppermost parent node. All the values are added and also it is named the root node. Each left arrow is named as 0, and the right arrow is labeled as 1 on a binary tree. For compressing the values, start as of the root node and trace the character by reading off 1's and 0's.

$$A_c \xrightarrow{\text{Frame compression}} C(\hat{F}_r) \tag{30}$$

where $C(\hat{F}_r)$ indicates the compressed video frames. The video sequence, using the above 3 steps, is compressed frame by frame. Each frame of the video is provided to the RLAH encoder, and each frame's output is concatenated at the output. The pseudocode of the RLAH encoding algorithm is given in Fig. 3.

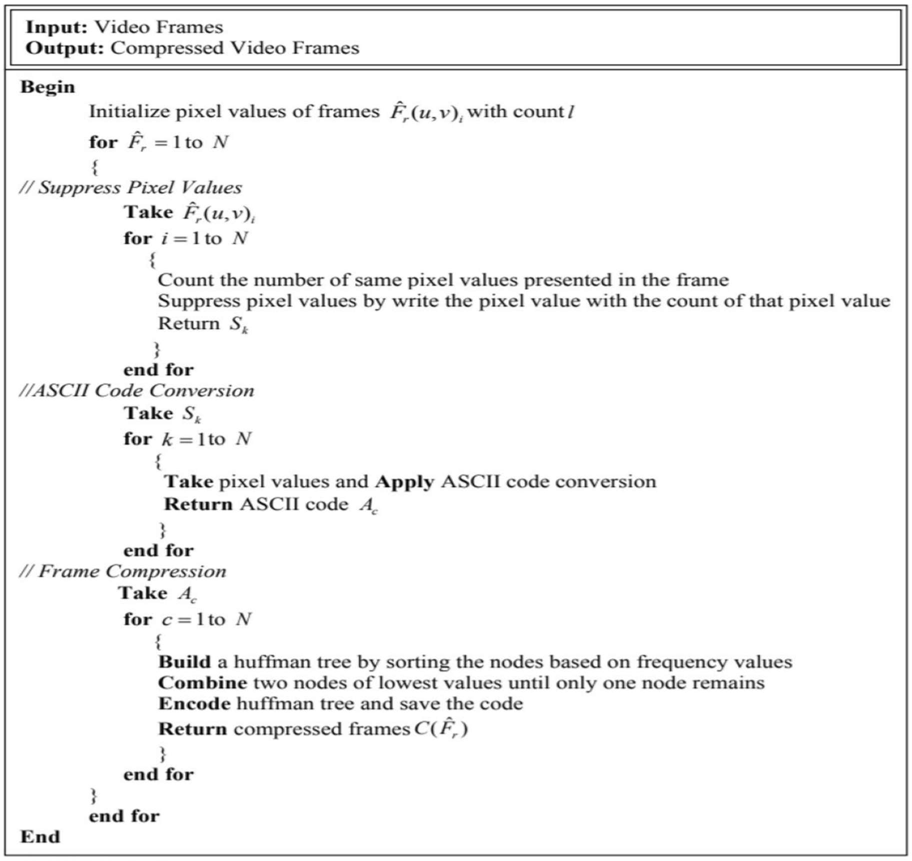
The RLAH is a reversible process. The above three steps are executed in reverse order for decompressing the video. The video's quality is calculated among the original video in addition to the decompressed video centered on some quality metrics after decompression.

4 Result and discussion

In this section, performance analysis of the model is discussed. The proposed process is carried out using MATLAB tools and validates the performance using the permanence parameters as peak signal-to-noise ratio (PSNR).

4.1 Performance evaluation

Peak signal-to-noise ratio (**PSNR**) is used as the performance metric in this analysis. It is an engineering term for the ratio between the maximum possible power of



```
Input: Video Frames
Output: Compressed Video Frames

Begin
        Initialize pixel values of frames $\hat{F}_r(u,v)_i$ with count $l$
        for $\hat{F}_r = 1$ to $N$
        {
// Suppress Pixel Values
            Take $\hat{F}_r(u,v)_i$
            for $i = 1$ to $N$
              {
               Count the number of same pixel values presented in the frame
               Suppress pixel values by write the pixel value with the count of that pixel value
               Return $S_k$
              }
            end for
//ASCII Code Conversion
            Take $S_k$
            for $k = 1$ to $N$
              {
                Take pixel values and Apply ASCII code conversion
                Return ASCII code $A_c$
              }
            end for
// Frame Compression
            Take $A_c$
            for $c = 1$ to $N$
              {
               Build a huffman tree by sorting the nodes based on frequency values
               Combine two nodes of lowest values until only one node remains
               Encode huffman tree and save the code
               Return compressed frames $C(\hat{F}_r)$
              }
            end for
        }
        end for
End
```

Fig. 3 Pseudocode of proposed RLAH encoding algorithm

a signal and the power of corrupting noise that affects the fidelity of its representation. Because many signals have a very wide dynamic range, PSNR is usually expressed as a logarithmic quantity using the decibel scale.

PSNR is most easily defined via the mean squared error (MSE). Given a noise-free m × n monochrome image I and its noisy approximation K, MSE is defined as:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left[I\left(i,j\right) - K\left(i,j\right)\right]^2 \tag{31}$$

The PSNR (in dB) is defined as

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20\log_{10}\left(MAX_I\right) - 10\log_{10}(MSE)$$

where MAX_I is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255. More generally, when samples are represented using linear PCM with B bits per sample, MAX_I is $2^B - 1$.

The raw video is sent to the proposed encoder, and the encoded H.264 streams are received as the output. The encoder only PSNR (PSNR enc) is calculated also. The H.264/AVC video streaming apps are now supported by the framework, new video codecs can be readily added. A quality measurement based on PSNR is presented, which works well for both short and lengthy video sequences. The experiments are executed using five disparate standard videos (foreman_cif.y4m, grandma_qcif.y4m, akiyo_qcif.y4m, coastguard_qcif.y4m, and bowing_qcif.y4m) with disparate sizes included by the standard database. The detailed results are mentioned below.

4.2 Experimental results

Here the analysis outcomes regarding the compression process are given, and in this, one sample frame is pondered as of every video.

The clear depiction of every stage is given by the work which is exhibited in Fig. 4. The video sequence's input frames are exhibited in Fig. 4A. Figure 4b exhibits the conversion of RGB into CIE-Lab for adjusting the frame's lightness for attaining a better-quality image. Finally, utilizing RLAH decoding, image decompression is performed and is exhibited in Fig. 4C. Regarding diverse quality metrics, the frame quality is analyzed by analogizing the actual frame with the achieved decompressed frames.

4.3 Comparative results

Here, aimed at the proposed design's verification, the proposed compression technique's CR is analyzed, verified, and analogized with the prevailing compression techniques. In the proposed RLAH encoding along with prevailing arithmetic encoding (AE), run length encoding (RLE), Lempel–Ziv–Welch encoding (LZWE), 10 adjacent frames in every video together with the 10 frames' average CR are picked which are exhibited in Table 1.

The ratio betwixt the original frame size and the compressed frame size is articulated as CR, and it alludes to how much compression a specific frame attains and the frame quality in general. The frame's CR must be high aimed at effectual compression. The

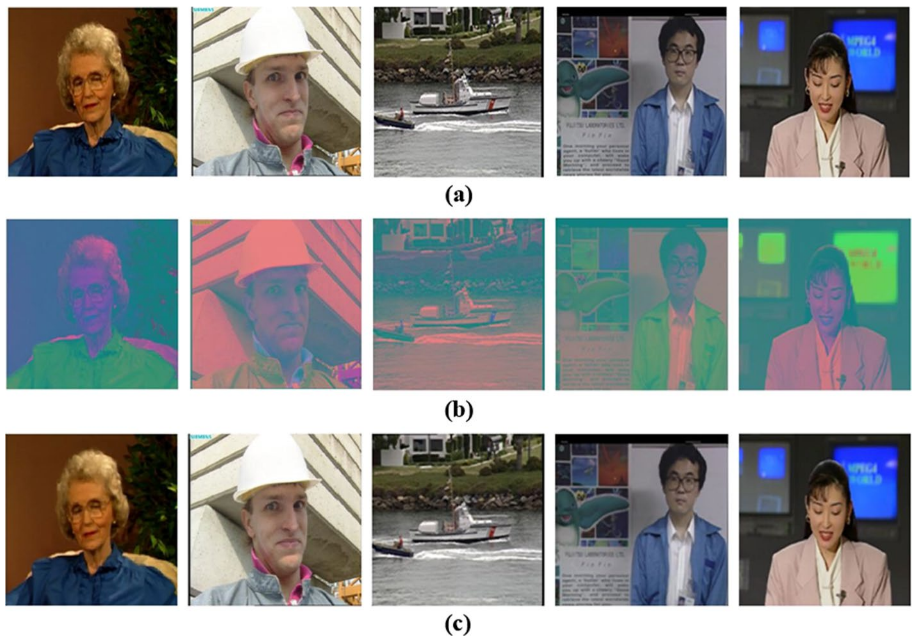


Fig. 4 Visual results of proposed video compression model **a** Input VFs, **b** RGB to CIE-Lab conversion, **c** decompressed image

Table 1 Average CR of 10 frames

Video	Arithmetic encoding	Run length encoding	LZE encoding	Proposed RLAH encoding
Foreman_cif.y4m	0.2936	0.3966	0.0221	0.6770
Grandma_qcif.y4m	0.1241	0.1362	0.0218	0.6044
Akiyo_qcif.y4m	0.3666	0.4033	0.0225	0.5677
Coastguard_qcif.y4m	0.3743	0.4243	0.0221	0.7643
Bowing_qcif.y4m	0.3676	0.3993	0.0054	0.5143

proposed RLAH encoding is analogized with the prevailing compression techniques and is exhibited in Fig. 5, which exhibits that the highest CR for every disparate video is attained by the proposed RLAH encoding. The proposed RLAH encodings' average CR is 0.6770 for foreman_cif.y4m, whereas average CR of 0.2936, 0.3966, and 0.0221 is attained by the prevailing AE, RLE, along with LZW encoding, respectively, and is low when analogized with the RLAH's CR. The CR of 0.6044, 0.5677, 0.7643, along with 0.5143 aimed at grandma_qcif.y4m, akiyo_qcif.y4m, coastguard_qcif.y4m, along with bowing_qcif.y4m videos, respectively, is attained by the proposed RLAH. And when analogized with the prevailing techniques, the proposed RLAH attains better outcomes regarding CR. In general, the proposed RLAH is more effectual at compressing the disparate types of videos and is exhibited by the graphical analysis.

4.4 Performance analysis

The proposed RLAH attains top performance in compression of the video when analogized with the other compression techniques which are evinced as of the

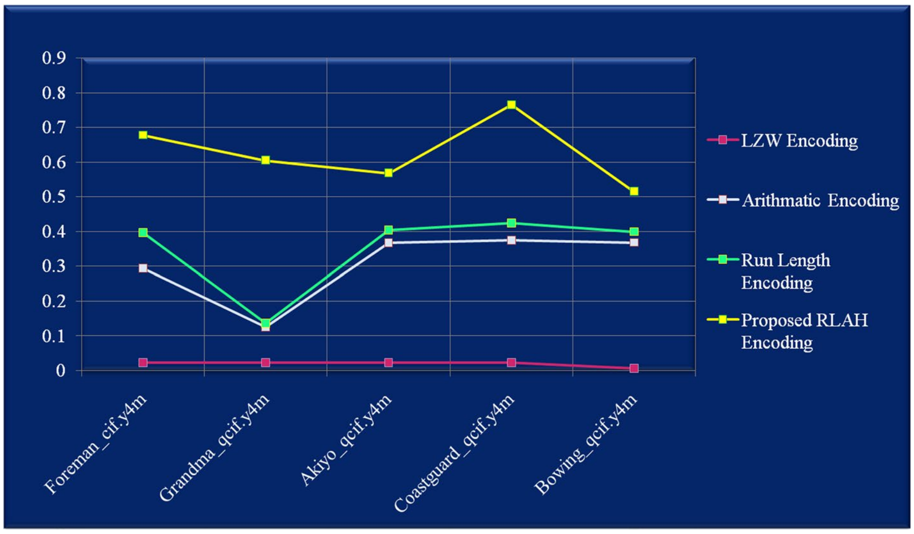


Fig. 5 CR analysis

Table 2 Performance of proposed model (average of 10 frames)

Video	PSNR	SNR	Corr	SSIM	IQI	MD	MAE	MSE	RMSE
Foreman_cif.y4m	34.35	31.24	0.99	0.97	0.84	0.75	0.02	0.0003	0.033
Grandma_qcif.y4m	39.42	28.33	0.99	0.99	0.90	0.33	0.01	0.0004	0.018
Akiyo_qcif.y4m	39.20	32.30	0.99	0.99	0.92	0.27	0.01	0.0001	0.018
Coastguard_qcif.y4m	35.59	29.78	0.99	0.95	0.93	0.64	0.02	0.0027	0.028
Bowing_qcif.y4m	39.25	34.33	0.99	0.98	0.91	0.49	0.01	0.0001	0.018

aforesaid comparison. Here, regarding some performance metrics, namely correlation value (CORR), root mean square errors (RMSE), indifference quality level (IQI), maximum difference (MD), peak signal-to-noise ratio (PSNR), mean absolute errors (MAE), mean square error (MSE), signal-to-noise ratio (SNR), along with structural similarity index measure (SSIM), the proposed design's decompressed image's quality is assessed. Here, 10 adjacent frames are picked in each video and the average of MAE, SSIM, MD, CORR, PSNR, SNR, MSE, IQI, along with RMSE of proposed RLAH encoding aimed at 10 frames is exhibited in Table 2 which is the same as the CR analysis.

Regarding the PSNR along with SNR, Fig. 6 exhibits the proposed design's performance. The decompressed image's quality will be superior if the PSNR value is high. The PSNR value of 34.35, 39.42, 39.20, 35.59, and 39.25 for foreman_cif.y4m, grandma_qcif.y4m, akiyo_qcif.y4m, coastguard_qcif.y4m, and bowing_qcif.y4m video, respectively, is attained by means of the proposed design. The PSNR must be in the range betwixt 30–50 aimed at better image quality, and the proposed design proffers a PSNR value of above 34. The proposed design offers outstanding performance regarding grandma_qcif.y4m and akiyo_qcif.y4m videos. Therefore, it is evinced that the proposed model's decompressed image quality is high. The proposed design offers 31.24, 28.33, 32.30, 29.78, and 34.33 for the five disparate videos regarding SNR value.

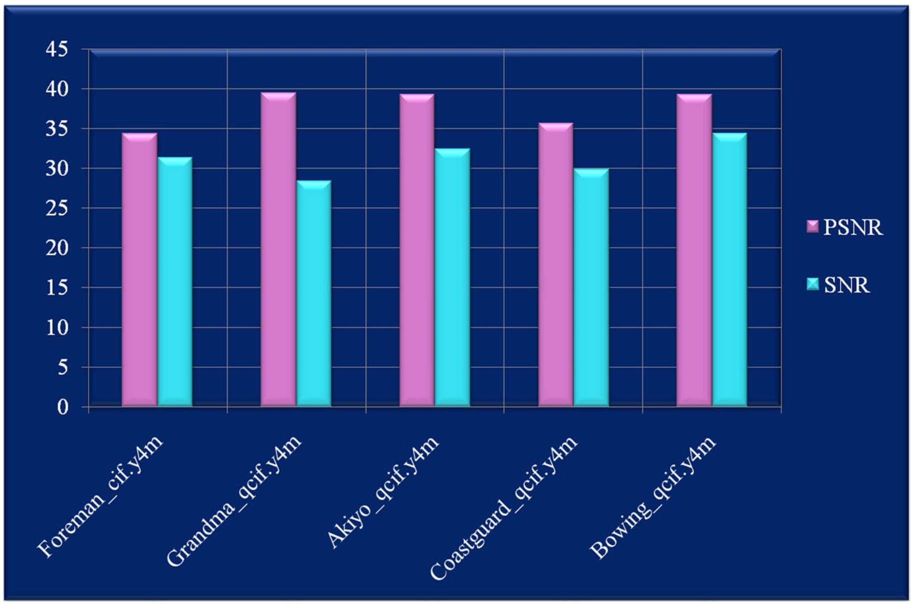


Fig. 6 Performance of proposed model regarding PSNR and SNR

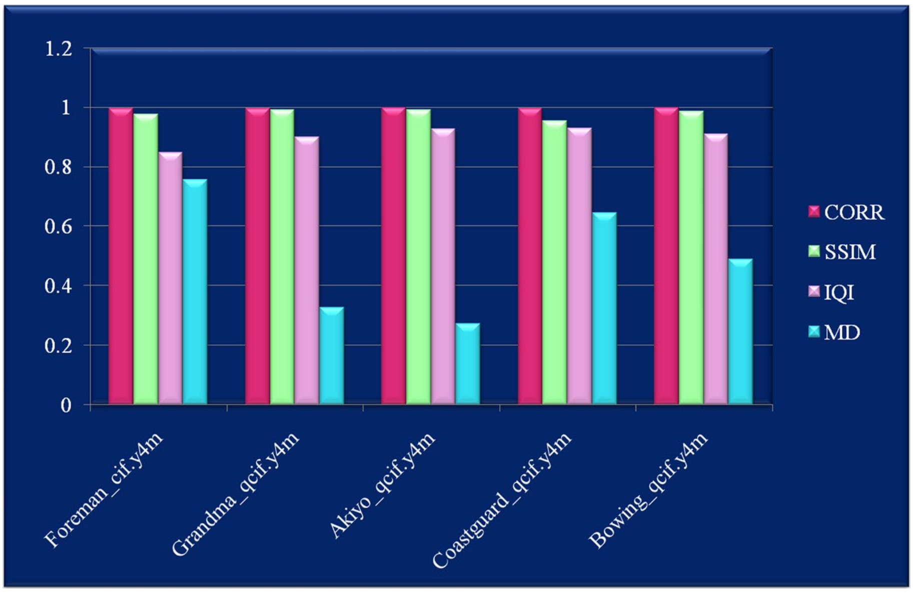


Fig. 7 Performance of proposed model regarding CORR, SSIM, IQI, and MD

If the SNR value equals 20, then the image's quality is acceptable, and the resulting image is of outstanding quality if the SNR value is above 30. The proposed design's SNR value is above 29 aimed at all the videos. The proposed model's SNR value is above 31, especially for Foreman_cif.y4m, Akiyo_qcif.y4m, and Bowing_qcif.y4m videos. Therefore, it is evinced that better video quality is attained by the proposed design.

Regarding, the CORR, SSIM, IQI, along with MD, Fig. 7 exhibits the proposed model's performance. The degree of similarity (or dissimilarity) betwixt 2 videos is evaluated by the CORR. The CORR betwixt the actual and the decompressed video is above 0.99 for every video that means the decompressed video quality is nearly close to the actual video quality. The structural similarity betwixt actual and decompressed videos is measured by SSIM. The proposed design's SSIM is 0.97, 0.99, 0.95, 0.99, along with 0.98, for foreman_cif.y4m, grandma_qcif.y4m, coastguard_qcif.y4m, akiyo_qcif.y4m, along with bowing_qcif.y4m video, respectively, which is nearly close to 1. Thus, it is evinced as of outcomes that the proposed design is highly effectual and appropriate for compression of video, and regarding IQI along with MD also it proffers top performance.

Regarding, MAE, MSE, along with RMSE, Fig. 8 exhibits the proposed model's performance. These 3 metrics are referring to as error metrics. The average squared difference betwixt the actual video and the decompressed video is measured by the MSE. The MAE, RMSE, together with the MSE of any model, must be low for a high-quality video. The MAE, MSE, along with RMSE for foreman_cif.y4m video, are only 0.02, 0.0003, 0.033, for akiyo_qcif.y4m video are 0.01, 0.0001, and 0.018, for grandma_qcif.y4m video are 0.01, 0.0004, and 0.081. The MAE, RMSE, along with MSE values, are extremely low, i.e., below 0.1 for the remaining videos also. Therefore, the error betwixt the actual video and decompressed video is low which is evinced as of the outcomes, and hence, the decompressed video quality is almost near to the actual video.

Aimed at compressing the videos, the proposed design is more effectual and consistent which is evinced as of the entire graphical analysis. The video compression plays an important role in the following applications viz., EDTECH (e-Learning Platform), OTT (over the top) service platforms, and social media. So, this proposed work implies an active involvement in bringing about the results in the area of video compression for many applications including the above-mentioned one.

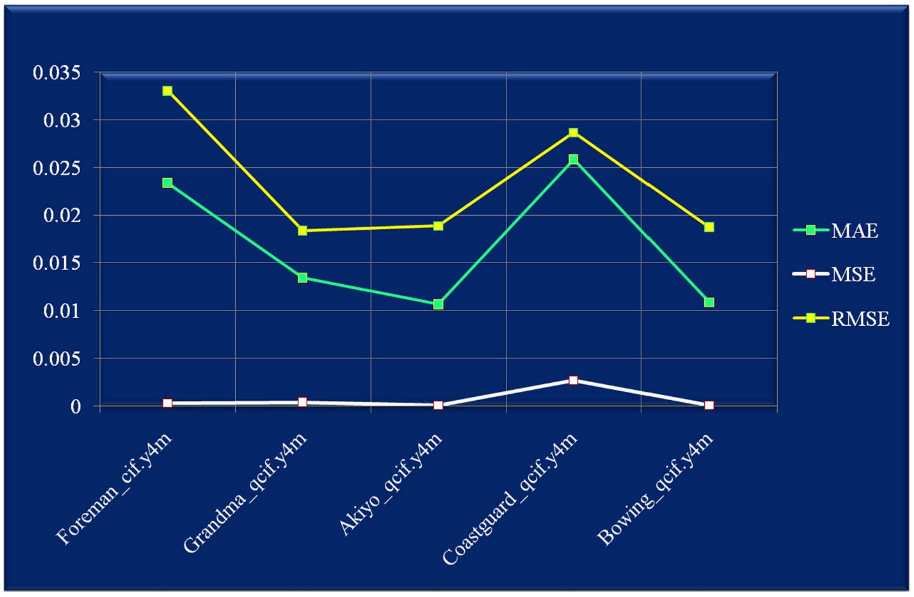


Fig. 8 Performance of proposed model regarding MAE, MSE, and RMSE

5 Conclusion

VC is stated as the technique of decrementing the amount of data needed aiming at representing the digital video while conserving acceptable video quality. This work proffers an enhanced VC technique. This work's core aim is to utilize efficient techniques and model an effective VC method aimed at attaining a higher-quality compressed video with higher PSNR Value and also higher CR. Latest techniques have been utilized in every stage aimed at effective VC. The proposed RLAH encoding technique's performance is analogized with the prevalent techniques concerning the CR. The experiential outcomes exhibited that the proposed RLAH encoding technique gives an outstanding compression performance with a higher CR of 0.76 aimed at Coastguard_qcif.y4m video. Moreover, examining the proposed approach's performance by analogizing the decompressed video's quality with the actual video identified that the RLAH encoding technique offers a high-quality video comprising maximal PSNR value with minimal error values. It attained the 39.42 PSNR value with a minimal MSE of 0.0001 aimed at the Akiyo_qcif.y4m video. The performance examination confirmed that the RLAH encoding approach effectively compresses the videos and offer high-quality videos.

Even though the RLAH encoding approach attains a high CR, it causes important artifacts in the compressed videos, like ringing, blurring, and blocking. The proposed work can be prolonged in the upcoming future by including artifact filtering methods aimed at incrementing its performance.

Acknowledgements
This work has been funded by the Universiti Kebangsaan Malaysia project number "DIP-2023-005". Also, the authors extend their appreciation to the Deanship of Scientific Research at King Khalid University, Abha, Saudi Arabia for funding this work through the Small Groups Project under grant number RGP.1/370/44.

Author contributions
All authors have equally contributed in the paper.

Data availability
All data that support the findings of this study are included within the article.

Declarations

Competing interests
The authors declared that they have no competing interest.

Received: 26 August 2023 Accepted: 9 January 2024
Published online: 22 January 2024

References

1. A.J. Hussain, Z. Ahmed, A survey on video compression fast block matching algorithms. Neurocomputing **335**, 215–237 (2018). https://doi.org/10.1016/j.neucom.2018.10.060
2. Lu. Zonglei, Xu. Xianhong, Deep compression: a compression technology for apron surveillance video. IEEE Access **7**, 129966–129974 (2019). https://doi.org/10.1109/ACCESS.2019.2940252
3. R. Birman, Y. Segal, O. Hadar, Overview of research in the field of video compression using deep neural networks. Multim. Tool Appl. (2020). https://doi.org/10.1007/s11042-019-08572-3
4. T. Tian, H. Wang, Large-scale video compression: recent advances and challenges. Front. Comp. Sci. **12**(5), 825–839 (2018)
5. G.R. Karthik, R. Kanthavel, R. Dhaya, Development of video compression using EWNS linear transformation and un-repetition simulated contrary based resurgence procedure. Multim. Tool Appl. **79**(5), 3519–3541 (2020)
6. S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, S. Wanga, Image and video compression with neural networks: a review. IEEE Trans. Circuits Syst. Video Technol. (2019). https://doi.org/10.1109/TCSVT.2019.2910119
7. S.M. Darwish, A.A.J. Almajtomi, Metaheuristic-based vector quantization approach: a new paradigm for neural network-based video compression. Multim. Tool Appl. (2020). https://doi.org/10.1007/s11042-020-10003-7

8. K. Siva Kumar, S. Sasi Kumar, N. Mohan Kumar, Efficient video compression and improving quality of video in communication for computer endcoding applications. Comput. Commun. **153**, 152–158 (2020)
9. A. Antony, G. Sreelekha, HEVC-based lossless intra coding for efficient still image compression. Multim. Tool Appl **76**(2), 1639–1658 (2017)
10. F. Chunxiao, H. Zhou, J. Lu, M. Hai, A novel lossless compression encoding framework for SAR remote sensing images. Signal, Image Video Process. (2020). https://doi.org/10.1007/s11760-020-01763-8
11. C. Raghavendra, S. Sivasubramanian, A. Kumaravel, Improved image compression using effective lossless compression technique. Clust. Comput. **22**(2), 3911–3916 (2019)
12. S.A. Alshehri, Video compression using frame redundancy elimination and discrete cosine transform coefficient reduction. Multim. Tools Appl. **80**(1), 367–381 (2020)
13. A. Gupta, A. Bansal, V. Khanduja, Modern lossless compression techniques: review, comparison and analysis, In *IEEE Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1–8, 2017, https://doi.org/10.1109/ICECCT.2017.8117850
14. H. Li, X. Tuo, T. Shen, M.J. Henderson, J. Courtois, M. Yan, An improved lossless group compression algorithm for seismic data in SEG-Y and MiniSEED file formats. Comput. Geosci. **100**, 41–45 (2017)
15. R. Yang, Xu. Mai, T. Liu, Z. Wang, Z. Guan, Enhancing quality for HEVC compressed videos. IEEE Trans. Circuits Syst. Video Technol. **29**(7), 2039–2054 (2018)
16. W. Sun, X. He, H. Chen, R.E. Sheriff, S. Xiong, A quality enhancement framework with noise distribution characteristics for high efficiency video coding. Neurocomputing **411**, 428–441 (2020)
17. Lu. Guo, X. Zhang, W. Ouyang, Xu. Dong, Li. Chen, Z. Gao, Deep non-local kalman network for video compression artifact reduction. IEEE Trans. Image Process. **29**, 1725–1737 (2019)
18. M. Saeedi, B. Ivanovic, T. Stolarczyk, I. Amer, G. Sines, Content adaptive pre-filtering for video compression. Signal, Image Video Process. (2020). https://doi.org/10.1007/s11760-019-01625-y
19. A. Antony, G. Sreelekha, Selective intra prediction in HEVC planar and angular modes for efficient near-lossless video compression. Multim. Tools Appl. **77**(1), 1093–1113 (2019)
20. M. Irannejad, H. Mahdavi-Nasab, Block matching video compression based on sparse representation and dictionary learning. Circ. Syst. Signal Process. **37**(8), 3537–3557 (2018)
21. T. Pal, S.D. Bit, Low overhead spatiotemporal video compression over smartphone based Delay Tolerant Network. J. Visual Commun. Image Represent (2020). https://doi.org/10.1016/j.jvcir.2020.102813
22. T. Rabie, M. Baziyad, PixoComp: a novel video compression scheme utilizing temporal pixograms. Multim. Tools Appl. (2020). https://doi.org/10.1007/s11042-020-08660-9
23. D. Jude-Hemanth, J. Anitha, A pattern-based artificial bee colony algorithm for motion estimation in video compression techniques. Circ, Syst, Signal Process. **37**(4), 1609–1624 (2018)
24. M.M. Siddeq, M.A. Rodrigues, A novel method for image and video compression based on two-level DCT with hexadata coding. Sens. Imag. **21**(1), 1–25 (2020)
25. M. Afonso, F. Zhang, D.R. Bull, Video compression based on spatio-temporal resolution adaptation. IEEE Trans. Circuits Syst. Video Technol. **29**(1), 275–280 (2018)

Publisher's Note