



# Metaheuristic-based vector quantization approach: a new paradigm for neural network-based video compression

Saad M. Darwish<sup>1</sup> · Ahmed A. J. Almajtomi<sup>2</sup>

Received: 4 April 2020 / Revised: 26 August 2020 / Accepted: 29 September 2020 /  
Published online: 28 October 2020  
© Springer Science+Business Media, LLC, part of Springer Nature 2020

## Abstract

Video compression has great significance in the communication of motion pictures. Video compression techniques try to remove the different types of redundancy within or between video sequences. In the temporal domain, the video compression techniques remove the redundancies between the highly correlated consequence frames of the video. In the spatial domain, the video compression techniques remove the redundancies between the highly correlated consequence pixels (samples) in the same frame. Evolving neural-networks based video coding research efforts are focused on improving existing video codecs by performing better predictions that are incorporated within the same codec framework or holistic methods of end-to-end video compression schemes. Current neural network-based video compression adapts static codebook to achieve compression that leads to learning inability from new samples. This paper proposes a modified video compression model that adapts the genetic algorithm to build an optimal codebook for adaptive vector quantization that is used as an activation function inside the neural network's hidden layer. Background subtraction algorithm is employed to extract motion objects within frames to generate the context-based initial codebook. Furthermore, Differential Pulse Code Modulation (DPCM) is utilized for lossless compression of significant wavelet coefficients; whereas low energy coefficients are lossy compressed using Learning Vector Quantization (LVQ) neural networks. Finally, Run Length Encoding (RLE) is engaged to encode the quantized coefficients to achieve a higher compression ratio. Experiments have proven the system's ability to achieve higher compression ratio with acceptable efficiency measured by PSNR.

**Keywords** Video compression · Intelligent vector quantization · Optimal codebook · Optimization

---

✉ Saad M. Darwish  
saad.darwish@alexu.edu.eg

## 1 Introduction

The extensive use of multimedia technology over the past decades has increased the demand for digital information. This awful demand made the current technology inefficient in handling the huge amount of data. Video compression by eradicating the redundancies present in it removes this problem [8]. The fundamental goal of video compression is to reduce the bit rate for transmission and storage of the information while maintaining the video quality. There are two types of compression techniques lossless and lossy compression [2]. In the lossless compression, the original video can be perfectly recovered from the compressed or encoded video. These are also called noiseless since they do not add noise to the original frame, while the second type is a lossy compression, in this type the recovered video does not have the same quality as the original video, this type has a small distortion, which will not affect the understanding of video generally. This type has been widely used because of its high compression rate [30, 35, 43].

Unfortunately, there are still many problems or challenges that hinder video compression from being popular [2]. This problem is how to make a trade-off between the video quality in terms of Peak Signal to Noise Ratio (PSNR) and the amount of compression. If the compression ratio is high then the quality of the reconstructed video becomes low. Also, the computational cost of the video compressing algorithm should be considered. Another challenge is the lack of understanding of image color spaces and their perceptual mechanisms that are used as bases for developing perception-aware coding techniques. Moreover, application- and context-dependent quality expectations of users have sometimes prevented researchers from reaching generally applicable perceptual compression techniques. Nevertheless, perceptual video compression has great potential as a solution to facilitate multimedia content management due to its efficiency for data rate reduction [2, 26].

Recently, several approaches have been presented that attempt to tackle the above problems. The taxonomy of these approaches can be categorized as spatial, temporal, statistical, and psycho-visual redundancies [11, 36]. Spatial redundancies (intra-coding) mean the elements are duplicated within a structure, such as pixels in a still image or frame. Exploiting spatial redundancy is how compression is performed. The sensitivity of the eye drops as spatial frequencies increase; i.e., as the spatial frequencies increase, the ability of the eye to discriminate between the changing levels decreases. Any detail that cannot be resolved is averaged. Temporal redundancies (inter-coding) mean the pixels in two video frames have the same values in the same location. Statistical redundancies mean encoded the coefficients compression. Variable length coding is used to exploit these statistical redundancies and increase compression efficiency by using binary codes in the last stage of video compression. Psycho-visual redundancy is a redundancy corresponding to different sensitivities to all frame signals by human eyes. Therefore, eliminating some less relative important information in our visual processing may be acceptable. In general, spatial redundancies can be exploited to remove or reduce higher frequencies in an effective way without affecting the perceived quality.

In video coding, quantization and entropy coding are the lossy and lossless compression procedures respectively. For quantization part, the scalar quantization strategy has dominated the hybrid video coding framework due to its low cost in computation and memory. However, this uniform scalar quantization does not conform to the characteristics of the human visual system and is not friendly to perceptual quality improvement. After quantization, the syntax elements including coding modes and transform coefficients will be fed into the entropy coding engine to further remove their statistical redundancy. Although the elaborately

designed hybrid video coding framework has achieved significant success in predominant compression performance, it becomes more and more difficult to be further improved. Moreover, it also becomes computation-intensive and inhospitable to parallel computation as well as a hardware manufacturer.

The process of video compression using Vector Quantization (VQ) can be divided into three phases: codebook generation, encoding, and decoding. In the codebook generation stage, a set of pre-computed codewords is generated based on a set of training image vectors. The goal involved in the design of each vector quantization technique is to make the technique to use more number of training vectors and less number of bits for codebook generation. The main objective is to find the most representative set of codewords that will produce the least distorted video after compression. It has been observed that as the number of bits used for codebook generation decrease the computational complexity and memory requirements decreases but the spectral distortion increases. Research efforts in codebook design have been concentrated in two directions: (1) to generate a representative codebook. (2) To reduce the computational complexity of the codebook generation [28, 30, 32].

There are two alternative approaches to codebook creation. The codebook can be pre-calculated and encoded statically at the beginning of the compression process, or it can be created adaptively and updated continually during compression. Compression systems that use static codebook creation suffer from one main problem: it is difficult to modify the codebook as pages (frames) are processed. To compress multiple pages the system must either form a codebook based on the set of pages or encode creation and deletion operators in the index stream. Adaptive codebook techniques do not require the codebook to be transmitted before the video' frames are compressed. Instead, it is created dynamically as unique frames are processed. The addition or deletion of components in the codebook is incremental and is performed after each component is processed [28, 32].

Adaptive VQ is an alternative solution that can exploit the statistical dependence between vectors while avoiding the complexity that would result from increasing the vector dimensionality. Rather than code each vector independently, we examine the context or local environment of a vector and modify the quantizer to suit our awareness of the “big picture” of how other vectors in the sequence are behaving in order to more efficiently code this particular vector. Adaptive codebook can match against any previously encoded character. One main drawback of this method is that components are not consistently matched with the same character, and the index value is not constant [32]. In the literature, the most commonly used method in VQ is the Linde-Buzo-Gary (LBG) algorithm [28]. However, LBG has the local optimal problem, and the utility of each codeword in the codebook is low. The local optimal problem is that the codebook guarantees local minimum distortion but not global minimum distortion. Since data points are represented by their index to the closest centroid, commonly occurring data have less error and rare data have a higher error. Hence VQ is suitable for lossy data compression.

In the literature, the majority of video coding algorithms utilizes neural network [23, 24] that is composed of a spatial component that encodes intra-frame visual patterns, and a reconstruction component that aggregates information to predict details. Some of these algorithms make the spatial component deep so that it can better leverage spatial redundancies for rebuilding high-frequency structures [15]. The neural video compression method based on the predictive VQ algorithm requires the correct detection of keyframes in order to improve its performance. The neural networks have the ability to learn how to do tasks based on the data given for training or initial experience (Adaptive learning). Neural networks are a “black box”

and have limited ability to explicitly identify possible causal relationships. The neural network's structure should be optimal but both these processes are time-consuming. Also, it may suffer from overfitting. Recently, evolutionary optimization techniques (e.g. genetic algorithm, and swarm intelligence) are exploited to enhance the NN learning process and build an intelligent vector quantization [12, 46]. The efficiency of the VQ relies on the appropriate codebook; hence, many research works based on optimization have developed for the generation of the global codebook.

Evolutionary algorithms such as genetic algorithm is a family of metaheuristics, population-based, meaning that a pool of solutions is used in every iteration of the solution process, that uses operators to combine/modify the solutions aiming at iteratively improving/evolving the solutions of the pool based on a fitness function. In addition, all metaheuristic algorithms use a certain trade-off of randomization and local search. Two major components of any metaheuristic algorithms are: intensification and diversification, or exploitation and exploration [12]. Diversification means to generate diverse solutions to explore the search space on a global scale, while intensification means to focus the search in a local region knowing that a current good solution is found in this region. A good balance between intensification and diversification should be found during the selection of the best solutions to improve the rate of algorithm convergence. The selection of the best ensures that solutions will converge to the optimum, while diversification via randomization allows the search to space from local optima and, at the same time, increases the diversity of solutions. A good combination of these two major components will usually ensure that global optimality is achievable. The GA repeats these processes until a fitness function satisfies a certain condition. GA has been widely used in complex optimization problems and has been shown to provide good solutions for learning NN and optimal codebook design [41]. The codebook design can be regarded as a searching problem; its goal is to search an optimal solution as the most representative codebook which could correctly be applied in the image compression. The color image becomes the source of training samples [4].

## 1.1 Contribution

This paper proposes a new codebook generation model for video compression using a combined scheme of neural network (NN) and genetic algorithm (GA). The combined scheme makes full use of the near global optimal searching ability of GA and the self-learning ability to link input and output of NN to compute the codebook. The suggested model differs from the traditional methods that rely on Learning Vector Quantization (LVQ) network for video compression by utilizing the genetic algorithm as a pre-step to construct optimal codebook for vector quantization; instead of building VQ using LVQ networks directly. This optimal codebook will act as an activation function of the NN to eliminate its sensitivity to initialization, slow convergence problems, and instabilities. The novelty of proposed video compression scheme is that it works based on removing different types of redundancies in one package. The system handles the frame's spatial redundancy by dropping the duplicate in the high-frequency coefficients of the discrete wavelet transform through adapting vector quantization based NN; whereas the redundancy inside the low-frequency (high energy) coefficients will be eliminated by using DPCM. The model controls the enter-frame temporal redundancy by utilizing background subtraction algorithm to extract motion objects within frames to generate the condensed initial codebook. Regarding statistical redundancy, the system employs run length encoding to increase the compression ratio. In general, a key issue in LVQ is the choice

of an appropriate measure of distance or similarity for training and classification (Euclidean distance in our case).

The rest of the paper is organized as follows: Section 2 describes some of the recent related works. The detailed description of the proposed system has been made in Section 3. In Section 4, the results and discussions are given. Finally, conclusions are drawn in Section 5.

## 2 Literature survey

Research in the video compression domain has attracted tremendous interest in recent years due to its challenging nature in effectively satisfying a high compression ratio without degradation of the reconstructed video. In the past decades, several traditional video compression algorithms have been introduced, such as H.264 and H.265. Most of these algorithms follow the predictive coding architecture whereby only the unpredicted elements of a signal are fed forward for further stages of information processing. The H.264 improves the video coding efficiency by performing motion compensation for the variable small block-size. HEVC stands for High-Efficiency Video Coding, also known as H.265 is able to compress video with double data compression ratio but just needs half bitrate to keep the same video quality and reduce storage by half when compared to H.264. Different from JPEG, HEVC utilizes more intra-prediction modes from neighbouring reconstructed blocks in the spatial domain. Besides intra-prediction, more coding gains of video compression come from the high efficient inter prediction, which utilizes motion estimation to find the most similar blocks as a prediction for the to-be-coded block. Nevertheless, a more efficient video coding scheme is required for higher-resolution and the newest services such as Ultra High Definition (UHD) and Virtual Reality (VR) [6, 14, 21].

After standardizing H.264/AVC and H.265/HEVC successfully, Versatile Video Coding (VVC), Scalable Video Coding (SVC) are being standardized. The basic framework of VVC is the same as HEVC, which consists of block partitioning, intra and inter prediction, transform, loop filter, and entropy coding. VVC aims for a 30 to 50% bit-rate reduction for the same perceptual quality as HEVC, but with an estimated 10 times or more encoding complexity compared to its predecessor. Meanwhile, the encoding structure of SVC includes one base layer and one or more enhancement layers. SVC is a newer form of video compression which dynamically adjusts the frame rate or resolution in real-time based on varying network conditions. SVC provides two types of quality scalability, known as coarse grain scalability and medium grain scalability. However, finding a precise correlation between consecutive frames is important to final coding performance. Block matching based motion estimation and motion compensation have been implemented in the reference software model of the previous video compression standards. The fundamental motion model of the conventional block-based motion compensation is a translational motion model. Yet, this technique suffers the limitation that it can only compensate for a pure parallel translation between frames. In actual videos, motion by a non-affine transformation appears more generally than by an affine transformation with such restriction [19–22].

Recently, neural networks have achieved significant success in many fields including video compression [5, 31]. Neural networks can either be used to augment a standard video compression technique, or they can be used as part of a neural compression technique. The primary way in which neural networks have been used to augment traditional video compression techniques is in the finding of motion vectors. The traditional approach involves an exhaustive search of a region of the frame in which the block under examination may have

come from. Alternatively, one could use the logarithmic block matching technique suggested in the MPEG standard. This is a heuristic that produces good results, but not necessarily the optimal results produced by the exhaustive search. The alternative to neural assisted video compression is to create a video compression scheme based entirely on neural networks. One of the major problems with the neural network, in general, is that some neurons are under-utilized. In order to handle this problem within the context of generating VQ codebooks, some authors suggested an adaptation of the neural network; begins by designing a small codebook then the authors search the training set to find the block clusters corresponding to each index in the codebook [38].

Recently, deep neural network (DNN) based autoencoder for video compression has achieved comparable or even better performance than the traditional video codecs. One possible explanation is that the DNN based video compression methods can exploit large scale end-to-end training and highly non-linear transform, which are not used in the traditional approaches. These methods are used to improve the performance of one particular module of the traditional video compression algorithms instead of building an end-to-end compression scheme [5, 31, 38]. The authors in [5] suggested a fully end-to-end deep learning framework for video compression. Their framework inherits the advantages of both classic predictive coding scheme in the traditional video compression standards and the powerful non-linear representation ability from DNNs. However, DNN requires very large amount of data in order to perform better than other techniques. Furthermore, there is no standard theory to guide for selecting the right deep learning tools as it requires knowledge of topology, training method, and other parameters. As a result, it is difficult to be adopted by less skilled people [31].

An insight into the potential of using vector quantization for real-time neural video codec is provided in [24]. This technique utilizes Predictive Vector Quantization (PVQ) that combines vector quantization and differential pulse code modulation. The neural video compression method based on the PVQ algorithm requires the correct detection of keyframes in order to improve its performance. For keyframe detection, their method uses techniques based on the Restricted Boltzmann Machine method (RBM). Unfortunately, training RBMs with this approach is known to be difficult, as learning easily diverges after initial convergence. This difficulty has been reported recently by many researchers. Another work involving hybrid transformation-based video compression may be seen in [2]. The hybrid DWT- DCT transforms exploits the properties of both the DWT and DCT techniques and provides better compression. The hybrid compressed frame is quantized and entropy coded with Huffman coding. This method utilized the motion vectors, found from estimation using adaptive rood pattern search, and is compensated globally. Their system was more complex because the hybrid transforms with quantization needs a lot of time to compress the video.

With this same objective, in 2015 A. Elmolla et al. [8] introduced run-length and Huffman coding as a means of packaging hybrid coding. Initially, the video is converted to numbers of frames and then applying fast curvelet transform to get the coefficient of the frame, this coefficient will be the input for the Run-length Encoding (RLE) then applying Huffman Coding to the output of RLE the final output image will be compressed. This type of compression has the ability to overcome the drawbacks of wavelet analysis, but there are some of the limitations, they are not optimal for the sparse approximation of curve features beyond-singularities. More formal description, as well as a review of video compression based on Huffman coding, can be found in [1]. Yet, Huffman coding requires two passes one to build a statistical model of the data and a second to encode it so is a relatively slow process. This, in turn, means Huffman coding is slower than other techniques when reading or writing files.



Recently a lot of research interest is being shown in optimization techniques that can obtain the temporal redundancy that deals with motion estimation and compensation based on edge matching which can alleviate the problem of local minima and at the same time reduces computational complexity [13, 42]. The ant colony edge detector is used to create edges for motion compensation. The frame is divided into non-overlapping rectangular blocks. The best match to the current block is the search for in the previous frame to the search area and on the basis of mutual information match is found. In order to remove the spatial redundancy, the modified fast Haar wavelet transformation is used. The main disadvantages of block matching are the heavy computation involved and the motion averaging effect of the blocks. If the chosen blocks are too large then many different moving objects may be enclosed by one block and the chosen motion vector is unlikely to match the motion of any of the objects. Furthermore, the main problem of the ant colony algorithm is that convergence is guaranteed, but time to convergence uncertain and problem representation (coding) is not straightforward [42].

Another approach was introduced by Rubina in 2015 [37], defining a technique to provide temporal-based video compression based on fast three-dimensional cosine transform. It has been shown that the efficiency of removing temporal redundancy algorithms can be improved by increasing the efficiency of inter prediction algorithms by the way of increasing the efficiency of transformation coding. One of the main problems of this transform is represented by video sequences mixing inside the group of frames. Recently, motion compensation has become a cutting-edge and promising approach to video compression. For instance, Zhang and others [47] employed a Motion Compensation Temporal Filtering (MCTF) technique to eliminate temporal redundancy in the frames of video. MCTF is based on inter-frame wavelet transform and lifting construction to overcome the disadvantage of block motion-compensation. However, using the advantage of a curvelet in signal analysis can improve the function of video compression. The disadvantage of block motion compensation is that it introduces discontinuities at the block borders (blocking artifacts).

To minimize the influence caused by the hybrid transformation in terms of compression quality and increase the compression ratio; Esakkirajan et al. [11] incorporated the advantages of multiwavelet coefficients, possesses more than one scaling function, and adaptive vector quantization scheme, the design of the codebook is based on the dynamic range of the input data. In this work, the spatial redundancy is minimized using a multiwavelet transform, temporal redundancy is minimized using a motion estimation algorithm, and the psycho-visual redundancy is minimized using the adaptive vector quantization technique. The objective of the paper is to develop a low bit rate video coder with acceptable visual quality. The performance of their scheme is compared with a wavelet-based video coder. Simulation results showed that multiwavelet based adaptive vector quantization gives better coding performance than wavelet-based adaptive vector quantization scheme. However, reducing the three types of redundancies make the system more complex.

Another approach was suggested by Nithin et al. in 2016 [34], defining a technique to provide component-level information to support spatial redundancy reduction based on properties of fast curvelet transform, Burrows-Wheeler Transform (BWT) and Huffman coding. BWT coding is used to encode the fast curvelet transform coefficient to compress many back to back data elements and keep in the form of the isolated data value and count. The remaining data of the BWT are encoded by using Huffman coding and it is deposited in a book and it may be created for each video frame. For reconstruction, the system enabled decoding by using combined codebook data and encoded data and it is transmitted. However, bigger blocks

might slow things down considerably. In general, the majority of the video compression algorithms rely on hand-crafted modules, e.g., block-based motion estimation, to reduce the redundancies in the video sequences. Although each module is well designed, the whole compression system is not end-to-end optimized. It is desirable to further improve video compression performance by jointly optimizing the whole compression system. Although they provide highly efficient compression performance, they are manually designed and cannot be jointly optimized in an end-to-end way.

## 2.1 The need to extend the related work

Although video compression has been studied for nearly many decades, there is still room to make it more efficient and practical in the real application. According to the aforementioned review. It can be found that past studies were primarily devoted to (1) devising different types of redundancies that employ the video information (e.g., intra-frame, and inter-frame), (2) not addressing the issues associated with the building of codebook for vector quantization compression algorithms (most often built randomly), and (3) most neuro-coding techniques rely on weights matrix adjustment to achieve compression which on the other hand requires extensive training. However, to the best of our knowledge, little attention has been paid to devising a new optimal codebook and improving its efficiency for vector quantization as well.

While there are many standard compression models that have proven to be very successful, but these models are mainly rely on motion estimation module to achieve high compression with acceptable quality. Yet, this module faces many difficulties such as unwanted camera motion, occlusion, noise, lack of image texture, illumination changes and the aperture problem. To reduce the impact of these difficulties on the efficiency of video compression systems; the suggested model concentrates on building context-based vector quantization that relies on the adaptive codebook design with the aim to tune the expected error from applying the simple background subtraction algorithm instead of using the complex motion estimation module to rise the concept of optimizing the whole compression system in in an end-to-end way.

## 3 Methodology

### 3.1 Mathematical analysis

An area that has close affinity with clustering is that of VQ. Vector quantization techniques are used mainly for [data compression](#), which is a prerequisite for achieving better computer storage utilization and better [bandwidth utilization](#) (in communications). Given  $T$  be the set of all possible vectors for the problem at hand. A vector quantizer  $Q$  of dimension  $I$  and size  $m$  is a mapping of  $T$  to a finite set  $C$ , which is called the reproduction set and contains  $m$  output reproduction points, the code vectors or codewords. Thus  $Q: T \rightarrow C$  where  $C = \{\theta_1, \theta_2, \dots, \theta_m\}$  with  $\theta_1 \in T$ . Each code vector  $\theta_i$  represents a specific region  $R_i$  of the vector space. Herein, the question is how one can select the code vectors  $\theta_i$  in such a way as to achieve the least possible distortion. A usual approach is to optimize an appropriate criterion function (distortion function), with respect to  $\theta_i$ 's. A commonly used distortion criterion is the average expected [quantization error](#). For a finite number of samples  $x_1, x_2, \dots, x_N$  of  $x$ , this error is defined as:



$$D(\mathcal{Q}) = \sum_{j=1}^m D_j(\mathcal{Q}), D(\mathcal{Q}) = \sum_{i=1}^N d(x_i, \mathcal{Q}(x_i))p(x_i) \quad (1)$$

where  $x$  is a random vector that models  $T$  and  $p(x)$  its corresponding pdf.  $D_j(\mathcal{Q})$  is known as the average quantization error for region  $R_j$ . The quantity  $d$  is a distance measure, for example, Euclidean (distortion measure).  $\mathcal{Q}(x_i) \in C$  is the code vector that represents  $x_i$  and  $p(x_i) > 0, i = 1, 2, \dots, N$ , the respective probabilities. To achieve optimal quantizer (optimal codebook), two conditions are necessary:

- Nearest neighbor condition: For fixed  $C$ ,

$$\mathcal{Q}(x) = \theta_j \text{..only if } d(x, \theta_j) \leq d(x, \theta_k), k \neq j \quad (2)$$

- Centroid Condition:

$$\sum_{i=1}^N d(x_i, \mathcal{Q}(x_i))p(x_i) = \min_y \sum_{i=1}^N d(x, y)p(x) \quad (3)$$

One way to compute the code vectors of the set  $C$  is to start with an arbitrary initial estimate of the code vectors and to iteratively apply the nearest neighbor condition and the centroid condition, interchangeably, until a termination criterion is satisfied. In the suggested model, a new codebook  $\mathcal{Q}(x_i)$  generation algorithm for video compression is presented. This new algorithm combines the LBG algorithm, genetic algorithm, and neural network in a unified framework in order to efficiently search for an optimal codebook based on the training video.

### 3.2 Proposed model

This paper proposes a new model that combines the two types of video coding: intra-frame and inter-frame coding in a unified framework with the aim of removing different types of redundancies (spatial, temporal, and statistical). The intra-frame coding is achieved by fusing the information come from both of wavelet transform that decorrelates the pixels of the input frame, converting them into a set of coefficients that can be coded more efficiently than the original pixel values themselves and quantization information originates from differential pulse code modulation that forms the core of **lossless compression** algorithms. The vector quantization technique is adapted for inter-frame coding based on the background subtraction algorithm to condense the codebook length. Finally, the run-length encoding algorithm is used to merge information for the two encoded data to achieve high compression by removing the statistical redundancy. Figures 1 and 2 show the main model's components for both compression and decompression phase respectively and how they are linked to each other.

- Step 1. **Generate Initial Codebook:** In this step, a codebook for each video is built offline that relies on extracting the moving parts of the frames (foreground) beside the background; each of them is represented as a codeword. The separating of moving objects is performed based on the background subtraction technique. Background subtraction technique is a widely used approach for detecting moving objects in videos from static cameras that is formally defined as [3]:

$$|f_c - f_c| < \delta \quad (4)$$

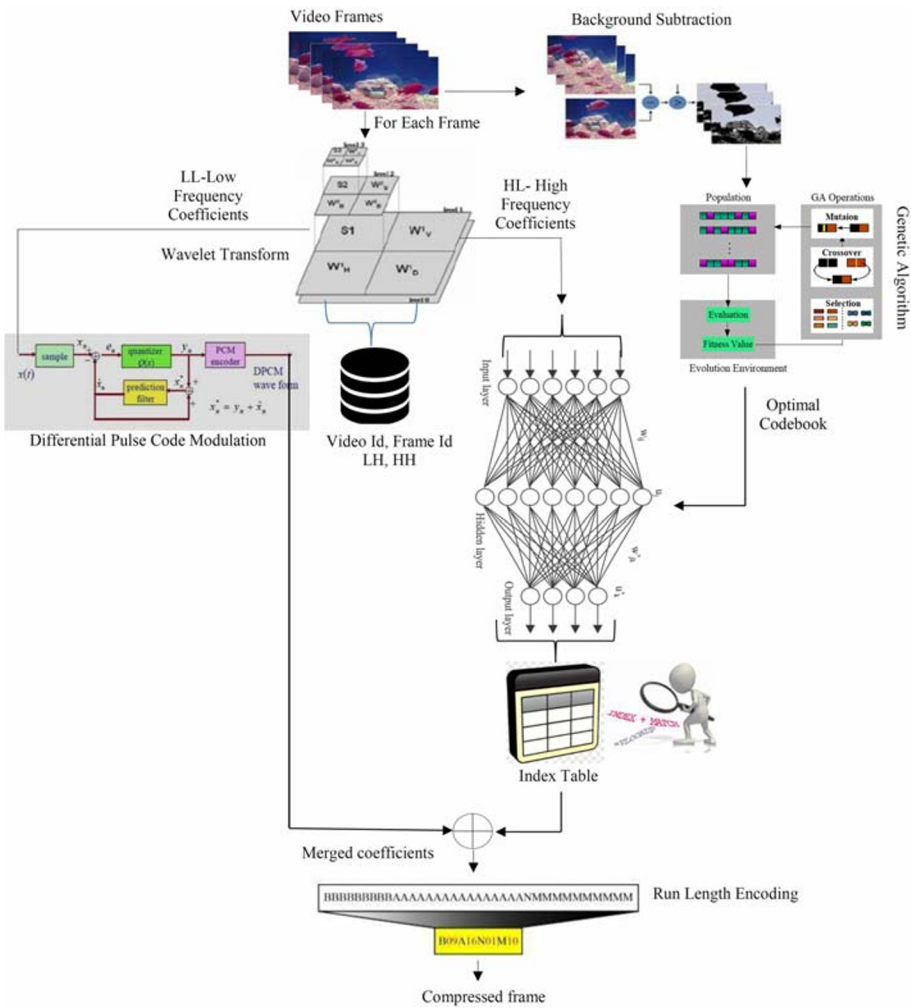


Fig. 1 Flow diagram of proposed model: Compression phase

where  $f_c$  is a current frame,  $f_p$  is a previous frame,  $\delta$  is the threshold. The accuracy of this approach is dependent on the speed of movement in the scene. Faster movements may require a higher threshold. Among the available algorithms for optimal codebook design, the suggested model uses LBG algorithm that is the most popular one and is always used by researchers in VQ as the “standard” codebook design algorithm for benchmarking to build the initial codebook. The steps of this algorithm are described as follows [32]:

- $P$  vectors will be picked up from the data set  $P = \{p_i | i = 1, 2, \dots, N_p\}$  as initial codewords with  $N_p$  is the total number of moving objects plus background. The codewords are denoted as.  $c_1 c_2 \dots c_k$
- The closest codeword based on square of the Euclidean distance, will be found for each vector in the data set  $P$ , and then the vector is added into the corresponding cluster of the closest codeword found using:

Video Id, Frame Id, LH, HH

Compressed frame data

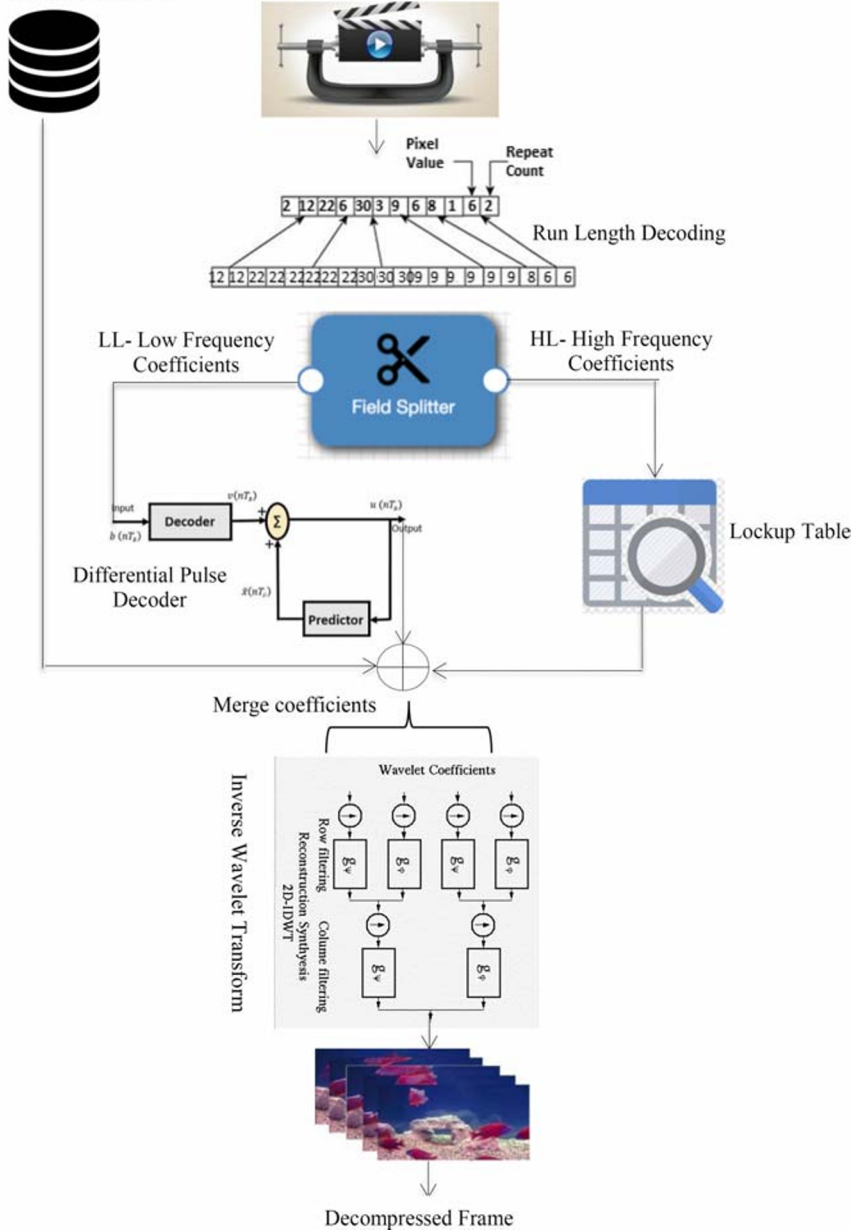


Fig. 2 Flow diagram of proposed model: Decompression phase

$$\begin{aligned}
 p_i \in Z_j, i = 1, \dots, N_p, j = 1, \dots, k \text{ if and only if,} \\
 \|p_i - c_j\| < \|p_i - c_q\| \quad q = 1, 2, \dots, k, j \neq q
 \end{aligned}
 \tag{5}$$

where  $\|\cdot\|$  denote Euclidean distance and  $Z_i$  is  $i^{th}$  cluster.

- The new codewords  $\bar{c}_1 \bar{c}_2 \dots \bar{c}_k$  are calculated as

$$\bar{c}_i = \frac{1}{S_i} \sum_{p_j \in Z_i} p_j \quad (6)$$

$S_i$  denotes the number of vectors that belongs to cluster of  $Z_i$ .

- The iteration will be stopped, if the discrepancy between codewords in two successive iterations is smaller than a predefined threshold or a specific number of iterations has been reached. However, the result is affected by the selection of the initial codebook, which often leads to the generation of a suboptimal final codebook.

**Step 2. Codebook Optimization:** given the initial codebook, the next step is to tune the codewords inside the codebook by a specific objective function. The genetic algorithm is adopted here to realize this step. In this case, an instance of a GA-based codebook optimization problem can be described in a formal way as a tuple  $(Y, Q, T, f)$  where [4, 12, 40, 41]:

- $Y$  is the solution space (initial population – a combination of different codewords). Each codeword is signified as a gene and every codebook is represented as a chromosome.
- $Q$  is the feasibility predicate (different operators- selection, crossover, and mutation). The crossover is the process of exchanging the parents' genes to produce one or two offspring that carry inherent genes from both parents. Herein, the crossover is a single point crossover to increase the diversity of mutated individuals and it is the most popular crossover and it is widely used. The purpose of mutation (uniform) is to prevent falling into a locally optimal solution of the solved problem. Tournament selection is probably the most popular selection method in the genetic algorithm due to its efficiency and simple implementation.
- $T$  is the set of feasible solutions (new generation populations). With these new generations, the nearest neighbour of the source vector gives the least dissimilarity (distortion) among all the codevectors in the codebook. The performance of VQ is influenced by the size of the codebook and dimension of the code vector, which is a couple of contradictions between distortion and complexity. GA is used to search for the best centroid of each partition. VQ is basically a clustering method, grouping similar vectors (blocks) into one class. The vectors are obtained from image data (frame) by extracting non-overlapping square blocks. The pixels in each block are arranged in a line-by-line order. VQ can be considered as a mapping of features. It maps input vectors into a set of codewords. Similar vectors are mapped to the same class or codeword in the codebook.
- $f$  is the objective function (fitness function). The individual that has higher fitness will win to be added to the predicate operators' mate. Herein, the fitness function is computed based on the Euclidean distance between the initial codebook and each frame that is converted to a vector for matching purpose [45]:

$$d(X, Y) = \sqrt{\sum_{i=1}^N \sum_{l=1}^k (X_{il} - Y_{il})^2}, f = \frac{1}{d(X, Y)} \quad (7)$$

Where  $N$  is the total number of frames,  $k$  represents the total number of the codeword in the initial codebook,  $X$  is the frame vector and  $Y$  is the initial codebook. In this case,  $X$  is partitioned to many sub-vectors each of which equals  $Y$  in length. This approach fully exploits the robustness of the genetic algorithm to improve the performance of video compression and to simplify the computation complexity of the VQ coding through generating optimal codebook that supports to represent the majority of frame data in a correct index. This codebook table will be stored in a database for use later in the decompression phase.

### 3.2.1 Compression phase

Data compression is subject to a space-time complexity trade-off. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed, and the option to decompress the video in full before watching it may be inconvenient or require additional storage. The design of data compression schemes involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (when using lossy data compression), and the computational resources required to compress and decompress the data [33]. The compression phase consists of two main stages, lossless compression based on DPCM and lossy compression based on an enhanced LVQ neural network. Both stages operate on the wavelet domain of each frame.

Wavelet transform decomposes a signal into a **weighted linear combination** of a set of scaling functions and wavelet functions (mother wavelet). Scaling and wavelet functions are orthogonal functions that divide the function space into a series of orthogonal high and low-frequency spaces. Coefficients (weights) associated with the scaling function, called approximation coefficients, capture **low frequency information**, while coefficients associated with wavelet function, called detail coefficients, capture high-frequency information. Unlike **Fourier transform**, wavelet transform provides local information (in both time and frequency) of a given signal, which makes this transform very useful for extracting disturbance information from power signals. Wavelet transform can be performed using different types of wavelet functions (e.g. Haar). The number of levels for **multiresolution analysis** is another parameter for this analysis. After a certain number of levels, detail coefficients are susceptible to noise. Features extracted from higher-level coefficients often are correlated to features extracted from lower-level coefficients and do not add much value to the classification system [4, 12, 18, 41]. We can approximate a discrete signal as:

$$f[n] = \frac{1}{\sqrt{M}} \sum_k W_{\varnothing}[j_0, k] \varnothing_{j_0, k}[n] + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_k W_{\psi}[j, k] \psi_{j, k}[n] \tag{8}$$

Here  $f[n]$ ,  $\varnothing_{j_0, k}[n]$ , and  $\psi_{j, k}[n]$  are discrete functions defined in  $[0, M - 1]$ , totally  $M$  points. Because the sets  $\{\varnothing_{j_0, k}[n]\}_{k \in \mathbb{Z}}$  and  $\{\psi_{j, k}[n]\}_{(j, k) \in \mathbb{Z}^2, j \geq j_0}$  are orthogonal to each other. The inner product can be taken simply to obtain the wavelet coefficients.

$$W_{\varnothing}[j_0, k] = \frac{1}{\sqrt{M}} \sum_n f[n] \varnothing_{j_0, k}[n] \tag{9}$$

$$W_{\psi}[j, k] = \frac{1}{\sqrt{M}} \sum_n f[n] \psi_{j, k}[n] \quad j \geq j_0 \tag{10}$$

$W_{\varnothing}[j_0, k]$  are called approximation coefficients while  $W_{\psi}[j, k]$  are called detailed coefficients.

$$\varnothing(t) = \sum_n h_{\varnothing}[n] \sqrt{2} \varnothing(2t-n) \quad (11)$$

It is evident to show that  $h_{\varnothing}[n] = \{1/\sqrt{2}, 1/\sqrt{2}\}$  for Haar scaling functions. It is designed properly such that we apply a discrete low pass filter  $h_{\varnothing}[n]$  to have  $\varnothing(t)$ . Similar relationship exists for the wavelet functions. It is [18]

$$\psi(t) = \sum_n h_{\psi}[n] \sqrt{2} \varnothing(2t-n) \quad (12)$$

Again, for Haar wavelets,  $h_{\psi}[n] = \{1/\sqrt{2}, -1/\sqrt{2}\}$ . These two filters are related by:

$$h_{\psi}[n] = (-1)^n h_{\varnothing}[1-n] \quad (13)$$

The computation time can be reduced.

$$\varnothing_{j,k}[n] = 2^{j/2} \varnothing(2^j n - k) = \sum_n h_{\varnothing}[n] \sqrt{2} \varnothing(2^j n - k) - n \quad (14)$$

Let  $n = m - 2k$ , we have

$$\varnothing_{j,k}[n] = \sum_m h_{\varnothing}[m-2k] \sqrt{2} \varnothing(2^{j+1} n - m) \quad (15)$$

Combine Eq. 6 with Eq. 12, it becomes

$$\begin{aligned} W_{\varnothing}[j, k] &= \frac{1}{\sqrt{M}} \sum_n f[n] 2^{j/2} \varnothing[2^j n - k] \\ &= \frac{1}{\sqrt{M}} \sum_n f[n] 2^{j/2} \sum_m h_{\varnothing}[m-2k] \sqrt{2} \varnothing(2^{j+1} n - m) \\ &= \sum_m h_{\varnothing}[m-2k] \left( \frac{1}{\sqrt{M}} \sum_n f[n] 2^{(j+2)/2} \varnothing(2^{j+1} n - m) \right) \\ &= \sum_m h_{\varnothing}[m-2k] W_{\varnothing}[j+1, m], = h_{\varnothing}[-n] * W_{\varnothing}[j+1, n] \Big|_{n=2k, k \geq 0} \end{aligned} \quad (16)$$

Similarly, for the detail coefficients, it is

$$W_{\psi}[j, k] = h_{\psi}[-n] * W_{\varnothing}[j+1, n] \Big|_{n=2k, k \geq 0} \quad (17)$$

For the commonly used discrete signal, say, a video frame, the original data can be viewed as approximation coefficients with order  $J$ . That is,  $f[n] = w_{\varnothing}[j, n]$ . By Eq. 13, next level of approximation and detail can be obtained. This algorithm is “fast” because one can find the coefficients level by level rather than directly using Eqs. 8 and 9 to find the coefficients.

### 3.2.2 Stage 1: Lossless compression

For each frame, the low wavelet frequency coefficients that have a large amount of energy are losslessly compressed to preserve the most important features from loss. In this, DPCM is employed as a signal encoder that uses the baseline of [Pulse-Code Modulation](#) (PCM) but adds some functionality based on the prediction of the samples of the signal [39]. DPCM takes the



values of two consecutive samples; if they are analog samples, quantize them; calculates the difference between the first one and the next; the output is the difference, and it can be further entropy-coded. Applying this process, the short-term redundancy (positive correlation of nearby values) of the signal is eliminated; compression ratios on the order of 2 to 4 can be achieved if differences are subsequently entropy coded; because the entropy of the difference signal is much smaller than that of the original discrete signal treated as independent samples.

### 3.2.3 Stage 2: Lossy compression

For each frame, the high wavelet frequency coefficients that have a small amount of energy are lossy compressed to achieve a high compression ratio. In this LVQ neural network are adapted to compress these coefficients; LVQ neural network utilizes the optimized codebook for each video as a dynamic vector quantization to be embedded into the hidden layer as an activation function. The generalized net model of a neural network algorithm is consists of a set of neuron; each neuron or a group of neurons are represented by a token of  $\alpha$  – type (learning rate). When alpha is close to 0, the neural net will engage in more conservative weight modifications, and when it is close to 1, it will make more radical weight modifications. These tokens enter the net through the place  $X_1$  and have the following initial characteristics [7]:

$$y(\alpha_{i(l)}) = \langle l, i, f_{i(l)} \rangle, i = 1, 2, \dots, N_l, l = 0, 1, \dots, L \quad (18)$$

In this,  $i$  represents the number of the token (neuron),  $f_{i(l)}$  is an activation function of the  $i^{\text{th}}$  neuron associated with the  $l^{\text{th}}$  layer of the neural network. The basic generalized net model of the neural network algorithm contains three transitions.

- Every token  $\alpha_{i(l)}$  is transferred from the place  $X_1$  to the place  $X_2$  as well as  $X_3$  via the transition  $Z_l$ .
- The tokens are transferred sequentially according to increasing indexes for given in order to be aggregated with other tokens of the same level  $l$  into one new token  $\alpha_{i(l)}$ .
- Representing the whole layer  $l$ , according the following conditions of transition  $Z_l$ .

$$Z_l = \langle \{X_1, X_2\}, \{X_2, X_3\}, r, \vee(X_1, X_2) \rangle \quad (19)$$

where  $r$  is the transition's condition determining which tokens will pass the transition. Herein, the whole neural network is represented by three transitions  $Z_{l=1,2,3}$  with three places  $X_1, X_2$ , and  $X_3$ . In practical both  $Z_1$  and  $Z_3$  is represented by a sigmoid function, whereas  $Z_2$  for the hidden layer is represented by the optimized codebook.

Unlike the current compression methods that employed the neural network as a black-box for lossy compression, the suggested model adapts the optimized codebook derived from step 2 as an activation function embedded in each hidden layer's neurons. In our case, the vector quantization neural network is employed as a compression algorithm combining vector quantization and supervised learning. In general, Learning Vector Quantization (LVQ) neural network does better than the back-propagation one in the field of supervised video compression as it has a superior performance in the sense of minimizing errors while maintaining rapid convergence [9].

The first step in the design of the LVQ neural network is to configure the parameters of both competitive and linear layers. The HL wavelet band (selected according to the

experimental results) for each frame is employed as an input vector to the input layer. The hidden layer contains the optimized codebook; one vector for each hidden layer's node; while the output is the index that will be used in the encoding phase. In general, LVQ is a supervised learning algorithm that can be used to modify the codebook. It combines clustering and classification processes based on the feed-forward neural network [9].

Inside LVQ [9, 25], the first step has been accomplished using a competitive layer of the network that works similarly to the Self-Organizing Map (SOM). The layer clusters the input data vectors using a table of vectors (codebook). The number of codebook vectors is much less than the number of input data vectors, however, it has to be predefined by the user. In clustering operation, every data vector is assigned to the closest codebook vector according to a predefined distortion measure. The second step of the LVQ is accomplished using the linear layer of the network that maps each codebook vector to the target class (index). The learning procedure works as follows: (1) Codebook initialization: The number of codebook vectors for each target class has to be comparative to the number of occurrence of that class and these vectors are initialized to the centre of the input ranges, (2) Winner determination: The Euclidean distance has to be calculated between training data vector and each codebook vector [25]. The neuron  $m_c$  with a codebook vector that has the least Euclidean distance to a data vector  $X_i$  is considered a winner. Formally the codeword  $W_i$  from the optimized codebook, is modified as follows [25]:

- If the input vector agrees with the codeword assignment, i.e.  $i=j$  then

$$W_i(t+1) = W_i(t) + \alpha(t)[X - W_i(t)] \quad (20)$$

- If the input vector doesn't agree with the codeword assignment, i.e.  $i \neq j$

$$W_i(t+1) = W_i(t) - \alpha(t)[X - W_i(t)] \quad (21)$$

where  $\alpha$  is the learning rate, a parameter in the range  $0 < \alpha < 1$ . Typically, the learning-rate parameter is initialized to, say, 0.1 and then decreases monotonically with each iteration. After a suitable number of iterations, the codebook typically converges and the training is terminated. In general, the proposed system does not need much time to learn by adjusting the neural network weights because the vector quantization inside the hidden layer is optimal and thus the process of matching the VQ's index and input vector will be done fast.

### 3.2.4 Stage 3: Run length coding

Given both of the quantized coefficients vector obtained from the DPCM lossless compression stage and VQ index vector obtained from the LVQ neural network lossy compression stage; the two vectors are merged into a unified vector with specific delimitation between them for decoding. In this case, there exists one unified vector for each frame. To increase the compression ratio, RLE is utilized to handle statistical redundancy among unified vector elements. Formally the RLE is defined as a tuple  $(R, L, S)$  for (run-flag, run-length, and run symbol) respectively, where  $S$  is a member of the alphabet of the symbols [16]. In general, Run-length algorithms are very effective if the source contains many runs of consecutive symbols. In the final, each video is represented as a matrix; each row contains the RLE of a frame; the number of rows determined by the number of video frames.

### 3.2.5 Decompression phase

Decompression process is done in a reverse way to the compression process as illustrated in Fig. 2 that includes the following steps:

- First, apply run-length decoding to each row of the matrix  $V_r$  that contains the compressed video to retrieve the merged coefficients vector  $f_r$ . This vector comprises the quantized coefficients  $LL_C$  and VQ index vector  $id_x$  for each frame.
- For the quantized coefficients  $LL_C$ , apply inverse DPCM to obtain the uncompressed coefficients (low frequencies)  $LL$ .
- For the given VQ index vector  $id_x$ , by utilizing the stored codebook table, this index value is converted to the equivalent vector to retrieve the high-frequency coefficients (each frame has one vector that contains  $HL$  coefficients).
- Given  $LL$ , and  $HL$  from the previous steps, these bands are combined with their other two unaltered bands ( $LH$ ,  $HH$ ) that given from the stored database and utilizing inverse DWT to get the decompressed frame [27]:

$$f_d[k] = \sum_{-\infty}^{\infty} HL[K]\varnothing[-n + 2k] + LL[k]\psi[-n + 2k] \quad (22)$$

where  $f_d$  represent the decompressed frame.

- Repeat the previous steps for each row in the compressed matrix  $V_r$ ; collect the frames to retrieve the original video. One important feature of VQ is the ease of control of the compression ratio and the amount of loss through the variation of the number of bits used for quantization. Another important advantage of VQ image compression is its fast decompression by table lookup technologies

## 4 Experimental results

In this section, the efficiency of the proposed model is analysed. Experiments were conducted to determine the variation in the robustness of the proposed model. Experiments were conducted on a benchmark video dataset [44] ([https://www.cs.utexas.edu/~chaoyeh/web\\_action\\_data/dataset\\_list.html](https://www.cs.utexas.edu/~chaoyeh/web_action_data/dataset_list.html)) (available at <http://www.nada.kth.se/cvap/actions/> and <https://media.xiph.org/video/derf/>). The testbed is a set of videos with different resolutions, different numbers of frames, and different extensions like AVI and MPEG. Herein, the background for all these videos is unmovable, while their foreground is varying from near stability like Miss America to movement like Aquarium. Furthermore, to verify the reliability of the proposed model in dealing with Full HD (1920 × 1080) and Ultra HD ((3840 × 2160) video sequences, eight video sequences with different spatial and temporal information were downloaded in the uncompressed format \*.YUV from the SJTU Media Lab (SJTU 4 K Video sequences, online: <http://medialab.sjtu.edu.cn/web4k/index.html>). Samples of the testbed videos are shown in Fig. 3 and Table 1 list the description of this dataset. The suggested model has been implemented in MATLAB (R2015a). The model has been implemented using the laptop computer with the following specifications: Processor: Intel (R), Core (TM) i3 CPU, Q720 @ 1.60GHz. RAM: 4 GB. System type: 64-bit operating system. Microsoft Windows 7 Single language as running operating system, and Hard Disk: 500 GB. To compare the



Fig. 3 Benchmark dataset

efficiency of the proposed model with other existing techniques, the popular measure, compression ratio (CR), and Peak Signal-to-Noise Ratio (PSNR) were employed for performance evaluation. Logically, a higher value of *PSNR* is good because it means that the ratio of

**Table 1** Dataset description

Video	Frame resolution	Number of Frames	Time (Sec)	Extension
Aquarium	720 × 480	155	5	MPEG
Man running	160 × 120	154	5	AVI
Traffic road	512 × 512	48	2	MPEG
Akiyo	720 × 480	89	3	AVI
Boxing	160 × 120	154	5	AVI
Miss America	640 × 480	89	3	AVI
Tennis	320 × 240	30	1	MPEG
Suzie	192 × 144	30	1	MPEG
Runners	3840 × 2160	30	10	YUV
Fountains	1920 × 1080	50	8	YUV
Construction Field	3840 × 2160	50	10	YUV
Wood	1920 × 1080	60	8	YUV

signal to noise is higher. Here, the signal is the original frame, and the noise is the error in reconstruction [26, 28, 32].

In this paper, the suggested intelligent vector quantization scheme that relies on a genetic algorithm has been tested with several benchmark videos. The suggested model has been tested with different GA parameters to perceive the influence of these parameters on the model's accuracy. Table 2 shows GA parameters that were taken for optimizing the codebook indexing table. In general, the performance of GAs largely depends on the parameters such as chromosome size, recombination operator probability value, and the selection mechanism. Identifying suitable values for these parameters is not very easy. It varies according to the complexity of the problem and dimension. Any advanced knowledge of these parameters makes GA more effective to reach the global optimum in a time-effective manner. GA always suffers from the problem of premature convergence and the appropriate value of crossover and mutation probability helps GA not to get trapped at local optima. The good GA balances between wide exploration and deep exploitation. The Local search method can be applied to the best-obtained solution at each iteration in order to accelerate the convergence of the algorithm.

Crossover is made in the hope that new chromosomes will have good parts of old chromosomes and maybe the new chromosomes will be better. However, it is good to leave some parts of the population to survive to the next generation. The Mutation is made to prevent falling GA into local extreme, but it should not occur very often, because then GA will in fact change to random search. Regarding chromosome size, if there are too few chromosomes, GA has a few possibilities to perform crossover and only a small part of the search space is

**Table 2** Genetic algorithm parameters

Encoding style	Decimal encoding
Population size	30
Generations number	100
Crossover method	Single point
Mutation rate	Uniform mutation
Selection type	Tournament selection
Probability of crossover	0.3
Probability of mutation	0.2

explored. On the other hand, if there are too many chromosomes, GA slows down. Research shows that after some limit (which depends mainly on encoding and the problem) it is not useful to increase population size, because it does not make solving the problem faster [10]. In our case, some of these parameters were determined experimentally as they depend mainly on the problem such as crossover and mutation probabilities; whereas the other parameters were specified according to the values stated in the previous studies as they represent default values that not affected by the size and type of the problem.

**Experiment no. 1: The efficiency of the suggested model regarding with the different types of high frequency subbands** The first set of experiments was performed to show how the quality of the proposed model in terms of PSNR and CR affected by the utilized wavelet details bands. As shown in Table 3, CR is not affected by the band type. One possible explanation of these results is that the suggested system encodes a small number of detail coefficients lossy as compared to approximation coefficients that contain a large number of coefficients. These small coefficients are encoded based on the lookup table in the form of one index. As the coefficients in the three bands are related; so the representation of these coefficients as an index is also similar. Therefore there is no difference between the compression ratios with the difference of these bands. The results in the Table also reveal that the HL band gives the highest PSNR as this band contains the most salient features compare to the other bands particularly in the frame with the textures / smooth area. It must be taken into account that each video has features that differ from the other and therefore these results may change slightly, but it remains in the overall that the HL subband is the one that achieves the best results in terms of PSNR.

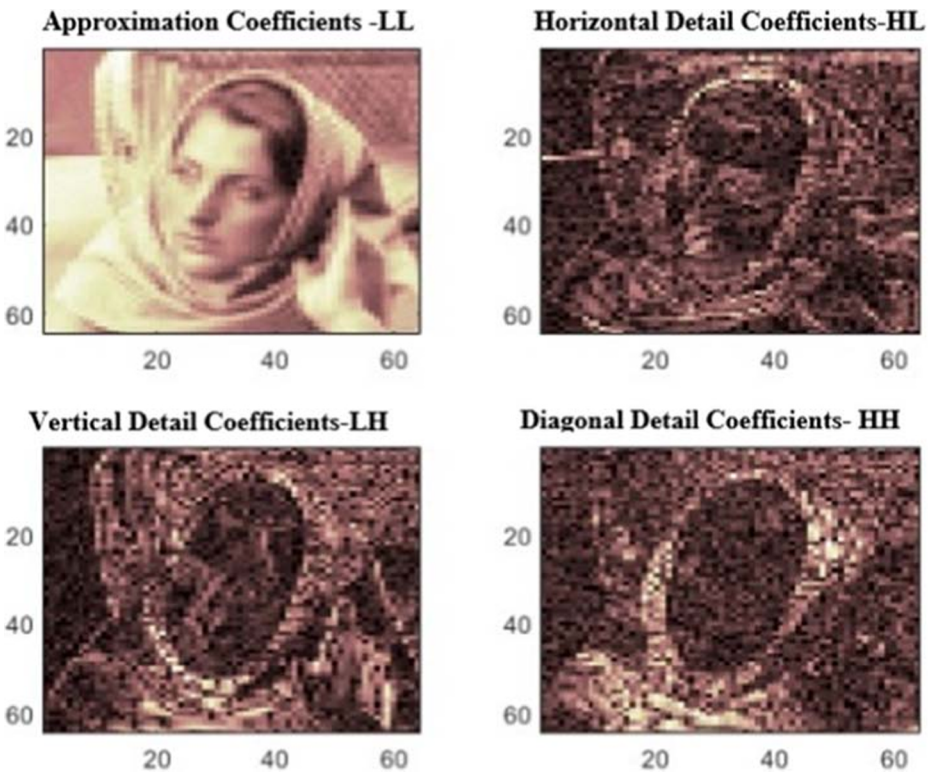
Figure 4 shows the concepts of wavelet bands that include the approximation output (coarse bands), which is the low-frequency content of the input signal component, and the multidimensional output (fine bands), which gives the details, or the high-frequency components of the input signal at various levels for vertical, horizontal and diagonal orientations. In this case, the frequency distribution of the LL-subband coefficients approximates the frequency distribution of the original image, while the wavelet coefficients in every other subband have a generalized Gaussian distribution with zero means. This property remains valid at all decomposition depth. Moreover, the furthest away a non-LL coefficient is from the mean in that subband, the more probable the corresponding position(s) in the original image have a significant feature [4, 12, 41]. In general, thresholding (quantizing) wavelet coefficients at very coarse scales usually increases the compression ratio but may result in the elimination of important features, whereas thresholding only at very fine scales may not eliminate enough noise. Therefore, the depth of [wavelet decomposition](#) needs to be selected to optimize the quality of the filtered signal, while maintaining a high compression ratio as possible [10].

**Experiment no. 2: The efficiency of the suggested model regarding with the different codebook sizes** The second set of experiments was performed to show how the proposed

**Table 3** Model performance evaluation regarding the wavelet details bands for man running video

Wavelets Bands	CR	PSNR
HH	30.512	37.126
HL	30.512	44.235
LH	30.512	35.581





**Fig. 4** Wavelet bands and salient features

model's performance depends on the size of the codebook. As shown in Table 4, the higher the size of the codebook, the higher the CR and the lower the PSNR ratio. In many cases, CR stabilizes or increases slightly when the codebook size is 256 or greater. The justification of these outputs lies in the fact that the more the size of the codebook vector, the more data it has stored. These data are represented as one Index; thus increasing the codebook size leads to increase the compression ratio. In the case of decompression, the search is done inside the lookup table; if there is a mistake in the index representation, the representation of the mistake for a large amount of information leads to reduce the PSNR and hence reduce the video quality.

**Experiment no. 3: The role of genetic algorithm and neural network for improving the model performance** The third set of experiments was performed to compare between the proposed model that utilizes both of genetic algorithm and neural network to build optimal codebook for vector quantization and the approach in [41] that utilizes a combined scheme of principal component analysis (PCA) and genetic algorithm for codebook construction. The combined scheme makes full use of the near-global optimal searching ability of GA and the computation complexity reduction of PCA to compute the codebook. Also, the comparison is made with a traditional LBG based video compression technique (without GA) that relies on the randomness to build the codebook. Herein, the module combining GA and NN in the suggested model is replaced once by the module integrating PCA and GA, and the second time with the traditional LBG module; and every time the model performance is measured in terms of CR and PSNR. For the LBG algorithm, the distortion threshold  $\varepsilon$  was set to 0.001.

**Table 4** Effect of codebook size of vector quantization on video coding in terms of CR and PSNR

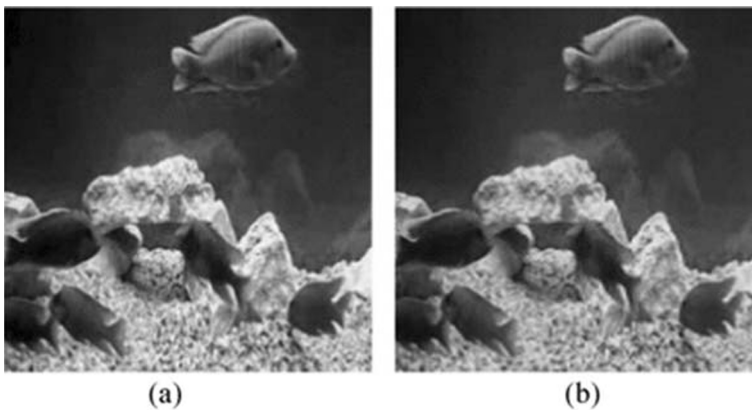
Codebook size	Video	CR	PSNR
16	<b>Man Running</b>	27.925	44.235
	<b>Traffic Road</b>	24.904	30.553
	<b>Aquarium</b>	23.245	33.224
	<b>Akiyo</b>	25.512	33.985
	<b>Miss America</b>	27.970	45.325
32	<b>Boxing</b>	25.194	43.653
	<b>Man Running</b>	29.656	40.589
	<b>Traffic Road</b>	25.724	30.423
	<b>Aquarium</b>	26.561	32.403
	<b>Akiyo</b>	27.512	32.598
128	<b>Miss America</b>	29.365	45.305
	<b>Boxing</b>	27.652	41.359
	<b>Man Running</b>	30.512	40.343
	<b>Traffic Road</b>	30.512	30.348
	<b>Aquarium</b>	30.551	32.257
256	<b>Akiyo</b>	30.512	30.141
	<b>Miss America</b>	30.512	44.254
	<b>Boxing</b>	30.512	41.325
	<b>Man Running</b>	32.512	40.271
	<b>Traffic Road</b>	30.665	30.048
	<b>Aquarium</b>	30.552	31.810
	<b>Akiyo</b>	30.512	30.141
	<b>Miss America</b>	30.512	44.254
	<b>Boxing</b>	30.512	40.936

As shown in Table 5, using GA + NN achieves an improvement of about 8.16% in compression ratio and 5.51% in PSNR compared with the PCA + GA based video coding technique. One possible explanation for this result is that feature of inter-pixel and inter-block correlation is effectively exploited by our model to compress the codebook and the indices of the representative codewords. For CPU time comparisons, the time for generating codebooks needed by LBG increases with the codebook size (it needs on average 45 s for 7 bit/codewords), while the time needed by our algorithm with GA + NN or PCA + GA remains fairly consistent regardless of the codebook size (it needs about 10 s with GA + NN module and 8 s with PCA + GA on average). Figure 5 shows the visual difference between the original and the reconstructed video's frame.

**Experiment no. 4: Model's self-assessment** The fourth set of experiments investigates the effect that different parameter settings have on the proposed model, which includes the

**Table 5** System performance evaluation with and without GA

Video	LBG		PCA+ GA		GA + NN	
	CR	PSNR	CR	PSNR	CR	PSNR
Man Running	30.365	37.547	30.983	38.167	33.512	40.271
Traffic Road	29.354	27.241	29.844	28.095	32.665	30.048
Aquarium	28.968	28.248	29.786	28.848	32.552	31.810
Akiyo	28.785	27.954	29.365	28.495	32.512	30.141
Miss America	28.417	40.696	28.971	42.292	33.876	44.254
Boxing	29.657	37.857	29.743	39.146	33.234	40.936









**Fig. 5** a Original frame b Reconstructed frame (PSNR =31.810)

background subtraction's threshold  $\delta$ , crossover ratio  $p_c$ , and mutation ratio  $P_m$ . To keep the number of parameter setting combinations small, the model will only vary the setting for one parameter ( $\delta$ ) at a time while keeping the setting for other parameters ( $p_c, p_m$ ) to its default value. In Table 6,  $\delta$  value changes from 0.06 to 0.1, and the PSNR achieved by the model is observed in the near stability video like Suzie. In general, faster movements require higher thresholds, while if the video has a little movement then the threshold must be small as much as possible to achieve a reasonable quality for video compression algorithms. As shown in Table 6, increasing the threshold according to the nature of the video leads to an increase in PSNR. This increase stop after a specific value (in the case of Suzie video after  $\delta=0.08$  there is no increase in PSNR. The reason for these results is that the large threshold leads to a decrease in the number of dissimilar pixels between the frames; therefore each codeword in the codebook contains a small amount of data. After decompression, these small data if there exists a distortion will cause a small PSNR as compared with the codebook with the large size.

An issue when applying GAs is to determine a set of control parameters that balance the exploration and the exploitation capabilities of the given algorithm. There is always a trade-off between the efficient exploration of the search space and its effective exploitation. In extreme cases, an inadequate choice of the parameter values can hinder the algorithm's ability to locate the optimum [10]. For example, if the mutation rate is too high, much of the space will be explored, but there is a high probability of losing promising solutions; the algorithm has difficulty to converge to an optimum due to insufficient exploitation. More specifically, some of the GA mutation operators favor the exploration of the search space, while some other operators favor its fast exploitation. The explorative mutation operators have a greater possibility of locating the minima of the objective function, but generally, need more iteration (generations). On the other hand, the exploitive mutation operators rapidly converge to a minimum of the objective function [17]. Table 7 shows the PSNR values for Boxing video resulting from the change in  $p_c$  values and clarified that the best value was achieved when  $p_c$  equals 0.7. Similarly, Table 8 shows the PSNR values for Boxing video resulting from the change in  $p_m$  values and clarified that the best value was achieved when  $p_m$  equals 0.3.

One possible explanation for these results is that the balance between exploration and exploitation is achieved for the aforementioned two values in order to obtain the optimal solution. In GA, mutation operators are mostly used to provide exploration and crossover operators are widely used to lead the population to converge on one the good solutions find so

**Table 6** The effect of background subtraction threshold for reconstructed suzie video in terms of PSNR

Video Frame	<i>Threshold Value</i>	<i>PSNR</i>	
		0.06	41.487
		0.07	42.008
		0.075	42.054
		0.08	42.308
		0.1	42.308

far (exploitation) [10]. Consequently, while crossover tries to converge to a specific point in the landscape, the mutation does its best to avoid convergence and explore more areas. Obviously, we prefer to explore much more at the beginning of the search process (to ensure the population coverage and diversity). On the other hand, we prefer more exploitations at the end of the search process to ensure the convergence of the population to the global optimum. There is just an exception; when population converges to a local optimum, we should (if we can) increase the population diversity to explore other areas.

**Table 7** The effect of crossover ratio on the reconstructed boxing video in terms of PSNR. (Population Size = 20, Generation Number = 100,  $P_m = 0.3$ )

$P_c$	<i>PSNR</i>
0.01	40.115
0.05	40.054
0.1	41.028
0.3	41.154
0.7	<b>41.336</b>
0.9	41.102

**Table 8** The effect of mutation ratio on the reconstructed boxing video in terms of PSNR. (Population Size = 20, Generation Number = 100,  $P_c = 0.7$ )

$P_m$	PSNR
0.01	41.115
0.05	41.054
0.1	41.028
0.2	41.154
0.3	<b>41.336</b>
0.5	41.102

The mutation operator is aimed to produce a little modification to an individual (chromosome) to produce a new offspring stochastically. So the aim of the mutation is to produce a limited random unbiased change in the population to exploit a certain promising region. This is why the mutation rate is usually small because the aim is to exploit the promising regions found by the crossover operator. It is also small so that the information accumulated in the chromosome over the past generations is not wasted by introducing a high variation in the population [17]. Regarding population size, too small population can converge early and it will not provide adequate initial surface exploration. The too-large population takes an excessive number of function evaluations to converge. In general, population size depends on the optimization problem. In evolutionary algorithms, the enormous population size usually does not improve performance. The best population size depends on the encoding and size of the encoded string.

An advantageous point of a GA is its ability to find a global optimal solution in multidimensional space, and this ability is also useful for constructing an optimal codebook of VQ for video compression. This means that we can obtain a better quality of a representative codebook. In previous studies to design a codebook using GA, genes were coded by binary values, zero and one. The suggested system utilized the real-coded GA to design a codebook. Its variable space is continuous, while the binary-coded GA is not. The real-coded GA is therefore thought to outperform the binary-coded GA for codebook design. The reason for the low compression ratio between the two systems is that the proposed system utilizes the lossless compression to compress a large number of important coefficients.

**Experiment no. 5: Comparative analysis** The following set experiments were conducted to validate the efficiency of the suggested model in comparison with related optimization-based compression techniques for both low resolutions, FHD, and UHD video sequences. The comparative algorithm [42] utilizes a motion estimation technique based on the ant colony and modified fast Haar wavelet transform to remove the temporal redundancy. On the contrary, the proposed model removes both temporal redundancy by utilizing optimal vector quantization, spatial redundancy by employing DPCM, and finally statistical redundancy by implementing run-length encoding. The results in Table 9 confirm the superiority of the suggested model as compared with the video codec method in [42] in terms of CR and PSNR with an average improvement of 7–10% for CR and 21–23% for PSNR respectively for all video sequences. Both methods use optimization algorithms in the video coding process, yet the comparative one uses it to estimate residual frames. The main disadvantage of this method is that it introduces discontinuities at the block borders (blocking artifacts). These artifacts appear in the form of sharp horizontal and vertical edges that are easily spotted by the human eye and produce false edges and ringing effects (large coefficients in high-frequency subbands)

**Table 9** Comparison between our model and the optimization based video codec in [42]

Video	Ant colony-based codec [42]		Proposed Model	
	PSNR	CR	PSNR	CR
Tennis	30.433	32.654	37.547	34.970
Suzie	34.574	33.452	42.308	36.874
Runners	35.146	32.654	42.247	33.309
Fountains	35.987	33.452	42.321	34.356
Construction Field	34.932	32.654	42.905	33.732
Wood	35.716	33.452	42.295	33.242

due to quantization of coefficients of the **Wavelet-related transform** used for **transform coding** of the **residual frames**. In opposite, the suggested model uses the optimization module in the form of GA in conjunction with NN to extract the optimal codebook in which the feature of inter-pixel and inter-block correlation is effectively exploited.

Another set of experiments was conducted with the aim of comparing the proposed model with similar state-of-the-art systems includes the method in [8], which employs fast curvelet transform with run-length encoding and Huffman coding to remove spatial redundancy; the method in [29] that utilizes end-to-end deep learning framework for video compression. This framework inherits the advantages of both classic predictive coding scheme in the traditional video compression standards and the powerful non-linear representation ability from DNNs. Besides, a comparison is made with the standards methods such as H.264 as well as H.265/HEVC using online software such as  $\times 265$ , the free H.265/HEVC encode, and  $\times 264$ , the best H.264/AVC encoder (<https://www.videolan.org/developers/x265.html>). The comparative methods were used in the testbed as a BlackBox with their default parameters; see [6, 14, 21, 29].

The results in Table 10 confirm the superiority of the proposed model over the codec system in [8] with improvement 20–23% for PSNR and 13–15% for CR. This result is expected as the method in [8] removes spatial and correlation redundancies only, whereas, the suggest model removes spatial, temporal, and statistical, i.e. increasing CR. Furthermore, utilizing the optimal codebook increases the PSNR in the decompression phase. We note the convergence of the proposed model results with the standard methods and the method in [29]; however, they are achieving a slight increase in PSNR in the case of equal compression ratios. The higher bit rate (high resolutions) means more data is stored, which means higher quality compression. The convergence in the results confirms the validity of the proposed model in

**Table 10** Comparison between our model and state-of-the art video codecs in [8, 29], H.264, and H.265

Video	Ashraf et al. method [8]		G. Lu et al. method [29]		H.264		H.265		Proposed Model	
	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR
Traffic road	25.11	31.08	32.56	37.65	33.11	38.12	33.67	38.73	33.71	37.50
Aquarium	24.93	30.64	32.10	37.13	33.87	38.88	33.86	38.85	33.54	38.34
Runners	29.62	34.17	32.87	42.24	32.82	42.75	32.91	43.06	33.23	42.36
Fountains	28.69	35.06	33.30	42.58	32.97	42.88	32.98	43.65	33.18	42.43
Construction	28.34	34.32	34.11	42.22	33.26	43.25	33.65	43.87	33.96	42.32
Wood	29.63	35.72	33.15	42.20	32.98	42.72	33.14	43.14	33.20	42.16



**Table 11** Average Time Consumed for the Whole mode in Seconds

Video	No. Frames	Compression Time	Decompression Time
Suzie	30	64.774742	1.828832
Miss America	89	109.879026	2.998986
Aquarium	155	267.751918	5.2201428

terms of methodology and application. Herein, the focusing is on building an optimal codebook for quantization-based compression instead of focusing on motion compensation that faces many difficulties especially, in high- resolution videos.

In general, the codebook design can be regarded as a searching problem; its goal is to search an optimal solution as the most representative codebook which could correctly be applied in the video compression [4]. It is assumed that the GA-based vectors are mapped to their nearest representative in the codebook with respect to a distortion function i.e. more PSNR. GA is applied by different natural selection to find the most representative codebook that has better fitness value in video compression. To expedite evolution and prevent the solution from getting out of searching space, tuning crossover and mutation ratio are firstly determined specifically.

**Time complexity analysis** The last set of experiments was conducted to evaluate the complexity of the suggested model. Time complexity analysis is a part of computational complexity theory that is used to describe an algorithm's use of computational resources; in most cases, the worst-case running time expressed as a function of its input using big- $O$  notation. As the proposed model was built using Matlab, which in turn depends on calling many built-in functions, therefore, it is difficult to extract the big- $O$ , herein; time was used as a measurement to evaluate the complexity of the system. As shown in Table 11, more processing time is needed for compression and decompression as the number of frames increase. It is also clear to us that the compression phase needs about 97% of the total time to execute the program as this step runs many subprograms that include wavelet transform, neural network, DPCM, RLE. However, this step can be implemented once before video sending (offline) and store its output in a database for later use. Of course, both compression and decompression time increases in the case of FHD and UHD video sequences. This time depends mainly on the machine to which the proposed model has been applied and this time can be reduced using a machine with high-configuration or using a parallel processing concept.

## 5 Conclusion

The video compression has evolved into a mature technology in the last decade; however, several issues remain to be solved until large-scale deployment is to be expected. Most of the current video methods were based on traditional vector quantization that built based on the partition-based variant in which the clusters are practically random and thus the resulting code vectors are concentrated near the centroid of the training set. In recent times, the merging of optimization techniques with the video compression scheme to improve its performance and effectiveness in different areas has received considerable attention among researchers working in this field. Inspired by the challenges that faced the video compression and in order to deal with it, in this paper, an enhanced model has been established to produce an intelligent vector

quantization scheme for video compression to facilitate the transfer and storage of videos. Herein, the genetic algorithm is utilized to optimize codebook in vector quantization to achieve high compression ratio with an acceptable quality of the reconstructed video. The enhanced model fuses different techniques for redundancies (spatial, temporal, and statistical redundancy) removing to increase video compression ratio with acceptable quality. The differential pulse code modulation is utilized to remove the spatial redundancy, the optimal vector quantization and neural network are exploited to remove the temporal redundancy; finally the statistical redundancy will be removed by implementing run length encoding technique.

In general, the application of the proposed Model faces some constraints that include (1) using static background videos (not a movable camera) to efficiently apply the background subtraction algorithm; (2) the model is applied with gray scale video. Working with color video requires applying the same steps to each color channel; finally (3) extra memory space is needed to store some information (wavelet HH and LH bands) that are used in the decoding process. The proposed model has been evaluated using benchmark videos. The results confirmed its efficiency in terms of compression ratio and peak signal to noise ratio. Utilizing genetic algorithm achieves an improvement of about 5.54% in compression ratio and 9.14% in PSNR compared with the traditional video coding techniques. In general, the efficiency of the suggested model can be enhanced with tuning variables like codebook size, background subtraction's threshold, and GA parameters. To set a plan for future works, the model can be upgraded to work with videos with movable camera instead of the static camera. Furthermore, adaptive DPCM can be employed instead of DPCM to increase compression ratio. More transformations can be investigated for precise representation. Finally, the GA can be replaced with another appropriate optimization method to find optimal codebook for vector quantization.

## References

1. Atheeshwar M, Mahesh K (2014) Efficient and robust video compression using Huffman coding. *Int J Adv Res Eng Technol* 2(8):5–8
2. Bernatin T, Sundari G (2014) Video Compression based on Hybrid Transform and Quantization with Huffman Coding for Video Codec. In: *Proceedings of the IEEE International Conference on Control, Instrumentation, Communication and Computational Technologies*, IEEE, India, pp452–456 Jul 2014
3. Boufares O, Aloui N, Cherif A (2016) Adaptive threshold for background subtraction in moving object detection using stationary wavelet transforms 2D. *Int J Adv Comput Sci Appl* 7(8):29–36
4. Chavan PU, Chavan PP, Dandawate YH (2009) Codebook Optimization in Vector Quantization using Genetic Algorithm. In: *Proceedings of the 2nd IEEE International Conference on Computer and Electrical Engineering*, vol 1. IEEE, Dubai, pp 280–283 Dec 2009
5. Chen T, Liu H, Shen Q, Yue T, Cao X, Ma Z (2017) Deepcoder: A Deep Neural Network Based Video Compression. In: *Proceedings of the IEEE Visual Communications and Image Processing (VCIP)*, IEEE pp1–4 Dec 2017
6. Choi Y, Jun D, Cheong W, Kim B (2019) Design of Efficient Perspective Affine Motion Estimation/ Compensation for Versatile Video Coding (VVC) Standard. *Electronics* 8(9):1–15
7. Duch W, Kacprzyk J, Zadrozny S (2005) *Artificial Neural Networks: Formal Models and Their Applications*. In: *Proceedings of the 15th International Conference on Science & Business Media*. Springer, Poland, pp 11–15 Aug 2005
8. Elmolla AM, Salama GI, Elbayoumy AD (2015) A novel video compression scheme based on fast Curvelet transform. *Int J Comput Sci Telecommun* 6(3):7–10
9. AM Elsayad (2016). "Classification of breast Cancer database using learning vector quantization neural networks", Technical Report, Saudi Association of Health Informatics, Saudi Arabia, Jul 2016

10. Epitropakis M, Plagianakos V, Vrahatis M (2008) Balancing the Exploration and Exploitation Capabilities of the Differential Evolution Algorithm. In: Proceedings of the IEEE International Congress on Evolutionary Computation, IEEE, Hong Kong, pp 2686–2693 Jun 2008
11. Esakkirajan S, Veerakumar T, Navaneethan P (2009) Adaptive Vector Quantization based Video Compression Scheme. In: Proceedings of the IEEE International Conference on Signal Processing and Communication Technologies, IEEE, India, pp 40–43 Mar 2009
12. Feng H, Tang M, Qi J (2011) A Back-propagation neural network based on a hybrid genetic algorithm and particle swarm optimization for image compression. *Int Congress Image Sig Process* 3:1315–1331
13. George NP, Anitha J (2015) Motion Estimation in Video Compression based on Artificial Bee Colony. In: Proceedings of the 2nd IEEE International Conference on Electronics and Communication Systems, IEEE, India, pp 730–733 Feb 2015
14. Goswami K, Lee D, Kim J, Jeong S, Kim H, Kim B (2017) Two-Step Rate Distortion Optimization Algorithm for High Efficiency Video Coding. *J Multimed Inf Syst* 4(4):311–316
15. Guo J, Chao H (2017) Building an End-to-End Spatial-Temporal Convolutional Network for Video Super-Resolution. In: Proceedings of the International Conference on Artificial Intelligence Thirty-First, China, pp 4053–4060 Feb 2017
16. Ida MP (2006) Fundamental data compression, 1st edn. Elsevier chapter 3, pp.49–65, Nov 2006
17. Inoue K, Hasegawa T, Mori N, Matsumoto K (2015) Analyzing Exploration Exploitation Trade-off by Means of P-I Similarity Index and Dictyostelium based Genetic Algorithm. In: Proceedings of the IEEE International Congress on Evolutionary Computation (CEC), IEEE, Japan pp 2548–2555 May 2015
18. Kanithi A (2011) Study of spatial and transform domain filters for efficient noise reduction, Master Thesis, National Institute of Technology, Rourkela, India
19. Kim B (2008) Fast Selective Intra-Mode Search Algorithm Based on Adaptive Thresholding Scheme For H. 264/AVC Encoding. *IEEE Trans Circ Syst Video Technol* 18(1):127–133
20. Kim B (2008) Novel Inter-Mode Decision Algorithm Based on Macro block (MB) Tracking for The P-Slice In H. 264/AVC Video Coding. *IEEE Trans Circ Syst Video Technol* 18(2):273–281
21. Kim B (2017) Fast coding unit (CU) determination algorithm for high-efficiency video coding (HEVC) in smart surveillance application. *J Supercomput* 73(3):1063–1084
22. Kim H, Lee J, Kim C, Kim B (2012) Zoom Motion Estimation Using Block-Based Fast Local Area Scaling. *IEEE Trans Circ Syst Video Technol* 22(9):1280–1291
23. Knop M, Cierniak R, Shah N (2014) Video Compression Algorithm based on Neural Network Structures. In: Proceedings of the International Conference on Artificial Intelligence and Soft Computing, Springer, Cham, pp 715–724 India
24. Knop M, Kapuściński T, Mleczo WK, Angryk R (2016) Neural Video Compression based on RBM Scene Change Detection Algorithm. In: Proceedings of the International Conference on Artificial Intelligence and Soft Computing, pp 660–669 Cambodia
25. Kumar G, Sharma S, Malik H (2016) Learning Vector Quantization Neural Network based External Fault Diagnosis Model for Three Phase Induction Motor using Current Signature Analysis. In: Proceedings of 6th the International Conference on Advances in Computing & Communications, India, *Procedia Computer Science*. 93(1):1010–1016
26. Lee J, Ebrahimi T (2012) Perceptual video compression: a survey. *IEEE Trans Sel Top Sig Process* 6(6): 684–697
27. Lin H, Lianga S (2014) Discrete wavelet transform based noise removal and feature extraction for ECG signals. *Int J Innov Res Bio-Med Eng* 35(6):351–361
28. Lu T, Chang Y (July 2010) A survey of VQ codebook generation. *Int J Inf Hiding Multimed Signal Process* 1(3):190–203
29. Lu G, Ouyang W, Xu D, Zhang X, Cai C, Gao Z (2019) Dvc: An End-To-End Deep Video Compression Framework. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 11006–11015 Jun 2019
30. Ma X, Pan Z, Hu S, Wang L (2014) Enhanced side match vector quantization based on constructing complementary state codebook. *IET Image Process* 9(4):290–299
31. Ma S, Zhang X, Jia C, Zhao Z, Wang S, Wang S (June 2020) Image And Video Compression With Neural Networks: A Review. *IEEE Trans Circ Syst Video Technol* 30(Issue 6):1683–1698
32. Mirzaei B, Nezamabadi H, Abbasi-Moghada D (2014) An Effective Codebook Initialization Technique for LBG Algorithm using Subtractive Clustering. In: Proceedings of the IEEE International Conference on Intelligent Systems, IEEE, Iran pp 1–5 Feb 2014
33. Mittal S, Vetter JS (2015) A survey of architectural approaches for data compression in cache and Main memory systems. *IEEE Trans Parallel Distrib Syst* 27(5):1524–1536

34. Nithin S, Suresh LP (2016) Video Coding on Fast Curvelet Transform and Burrows Wheeler Transform (BCH). In: Proceedings of the IEEE International Conference on Circuit, Power and Computing Technologies, IEEE, India pp 1–5 Mar 2016
35. Patel BK, Agrawal S (October 2013) Image compression techniques using artificial neural network. *Int J Adv Res Comput Eng Technol* 2(10):1–5
36. Ponlatha S, Sabeenian R (December 2013) Comparison of video compression standards. *Int J Comput Electron Eng* 5(6):549–549
37. Rubina I (2015) Novel Method for Fast 3D DCT for Video Compression. In: Proceedings of the International Conference on Creativity in Intelligent Technologies and Data Science, vol 535. Communications, Russia, pp 674–685
38. Singh MP, Arya KV, Sharma K (2009) Video Compression using Self Organizing Map and Pattern Storage using Hopfield Neural Network. In: Proceedings of the International Conference on Industrial and Information Systems, IEEE, Sri Lanka, pp 272–278 Dec 2009
39. Singh AV, Murthy KS (2013) Neuro-Curvelet Model for Efficient Image Compression using Vector Quantization. In: Proceedings of the International Conference on VLSI Communication Advanced Devices Signals and Systems and Networking, Lecture Notes in Electrical Engineering, vol 258. Springer, India, pp 179–185
40. Sivanandam SN, Deepa SN (2007) Introduction to genetic algorithms. Springer Science & Business Media Chapter 2, pp 15–37
41. Sun H, Lam K-Y, Chung S-L, Dong W, Gu M, Sun J (2005) Efficient vector quantization using genetic algorithm. *Neural Comput Applic* 14(3):203–211
42. Suri A, Goraya A (2014) Hybrid approach for video compression using ant Colony optimization and modified fast Haar wavelet transform. *Int J Comput Appl* 97(17):26–30
43. Tomar R, Jain K (2015) Lossless Image Compression using Differential Pulse Code Modulation and its Application. In: Proceedings of the IEEE International Conference on Computational Intelligence and Communication Networks, IEEE, India, pp 397–400 Dec 2015
44. Montgomery C, Rosedale T. Video test media (n.d.) <https://media.xiph.org/video/derf/>. (Accessed 03 Oct. 2016).
45. Wang W, Yang S, Tung C (2005) Codebook Design for Vector Quantization using genetic algorithm. *Int J Electron Bus* 3(2):83–89
46. Wei J (2015) Application of Hybrid Back Propagation Neural Network in Image Compression. In: Proceedings of the IEEE International Conference on Intelligent Computation Technology and Automation, Nanchang, IEEE, China, pp 209–212 Jun 2015
47. Zhang L, Wang SQ, Fang ZJ, Shu ZH, Zhang WR (2008) MCTF-Based Curvelet Video Compression Algorithm. In: International Conference on Natural Computation, vol 5. IEEE, China, pp 559–562 Oct 2008

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Affiliations

Saad M. Darwish<sup>1</sup> • Ahmed A. J. Almajtomi<sup>2</sup>

<sup>1</sup> Department of Information Technology, Institute of Graduate Studies and Research, Alexandria University, 163 Horreya Avenue, El-Shatby 21526, P.O. Box 832, Alexandria, Egypt

<sup>2</sup> Department of Computer Science, College of Science, Al-Nahrain University, Baghdad, Iraq