Biomedical Signal Compression Using Nature Inspired Algorithms

Thesis submitted by

SUVOJIT ACHARJEE

Doctor of Philosophy (Engineering)

Department Of Electronics and Telecommunication Engineering
Faculty Council of Engineering & Technology
Jadavpur University
Kolkata, India

Year: 2023

Biomedical Signal Compression Using Nature Inspired Algorithms

Thesis submitted by

Suvojit Acharjee

(Index No: 165/15/E)

(Reference No: D-7/E/703/15)

Doctor of Philosophy (Engineering)

under the guidance of

Prof. Sheli Sinha Chaudhuri

(Professor, Dept. of ETCE, Jadavpur University, WB, India)

Department Of Electronics and Telecommunication Engineering
Faculty Council of Engineering & Technology
Jadavpur University
Kolkata, India

Year: 2023

INDEX No: 165/15/E
REFERENCE NO: D-7/E/703/15

Title of the Thesis: Biomedical Signal Compression Using Nature Inspired Algorithms.

Name, Designation & Institution of the Supervisor:

Prof. Sheli Sinha Chaudhuri
Professor, Department of ETCE
Jadavpur University

List of Publication:

Journal

(1) Suvojit Acharjee and Sheli Sinha Chaudhuri. "Modified Cuckoo Search Algorithm for Motion Vector Estimation", The International Arab Journal of Information Technology 20.3 (May 2023): 340-348.

(2) Suvojit Acharjee and Sheli Sinha Chaudhuri. "Fuzzy Inference based Decision Making Model to Control the Operational Parameters of Motion Estimation Algorithms" International Journal of Information Technology, 15.4 (May, 2023): 2197-2207

(3) Suvojit Acharjee, and Sheli Sinha Chaudhuri. "Test Zone Search Optimization Using Cuckoo Search Algorithm for VVC." International Journal of Multimedia Data Engineering and Management (IJMDEM) 13.1 (2022): 1-16.

(4) Suvojit Acharjee, and Sheli Sinha Chaudhuri. " Ant weight-lifting algorithm for motion estimation", Iran Journal of Computer Science, (2023):1-13.

(5) Suvojit Acharjee, Nilanjan Dey, Sourav Samanta, Debarati Das, Ruben Roy, Sayan Chakraborty, and Sheli Sinha Chaudhuri. "Electrocardiograph signal compression using ant weight lifting algorithm for tele-monitoring." Journal of Medical Imaging and Health Informatics 6.1 (2016): 244-251.

(6) Suvojit Acharjee, and Sheli Sinha Chaudhuri. "Fuzzy Logic Based Four Step Search Algorithm for Motion Vector Estimation." International Journal of Image, Graphics and Signal Processing 4.4 (2012).

(7) Suvojit Acharjee, and Sheli Sinha Chaudhuri. "Fuzzy logic based three step search algorithm for motion vector estimation." International Journal of Image, Graphics and Signal Processing 4.2 (2012).

List of Conferences:

(1) Suvojit Acharjee, Prof. Sheli Sinha Chaudhuri. "Genetic Algorithm Based Optimized Test Zone Search for Versatile Video Coding" Optimization, Learning Analytics in Business (OLAB 2022). (Presented)

(2) Suvojit Acharjee, Prof. Sheli Sinha Chaudhuri. "Codebook Optimization Using Jaya Algorithm for Image Compression." 6th International Conference on Electronics, Materials Engineering and Nano-Technology (IEMENTECH 2022). (Presented)

(3) Acharjee, Suvojit, et al. "An efficient motion estimation algorithm using division mechanism of low and high motion zone." 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s). IEEE, 2013.

(4) Acharjee, Suvojit, et al. "A novel Block Matching Algorithmic Approach with smaller block size for motion vector estimation in video compression." 2012 12th International Conference on Intelligent Systems Design and Applications (ISDA). IEEE, 2012.

(5) Acharjee, Suvojit, and Sheli Sinha Chaudhuri. "A new fast motion vector estimation algorithm for video compression." 2012 International Conference on Informatics, Electronics & Vision (ICIEV). IEEE, 2012.

PROFORMA – 1

"Statement of Originality"

I Suvojit Acharjee registered on Doctor of Philosophy in the department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata do hereby declare that this thesis entitled **"Biomedical Signal Compression Using Nature Inspired Algorithms"** contains literature survey and original research work done by the undersigned candidate as part of Doctoral studies. All information in this thesis have been obtained and presented in accordance with existing academic rules and ethical conduct. I declare that, as required by these rules and conduct, I have fully cited and referred all materials and results that are not original to this work. I also declare that I have checked this thesis as per the "Policy on Anti Plagiarism, Jadavpur University, 2019", and the level of similarity as checked by iThenticate software is 1%.

Suvojit Acharjee

Signature of Candidate:

Date : 30/06/2023

S. S. Chaudhuri

Certified by Supervisor:

(Signature with date, seal)

Professor
Electronics & Tele-communication
Engineering Department
Jadavpur University
Kolkata-700 032

PROFORMA - 2

CERTIFICATE FROM THE SUPERVISOR

This is to certify that the thesis entitled **"Biomedical Signal Compression Using Nature Inspired Algorithms"** submitted by Shri Suvojit Acharjee who got his name registered on 09th November, 2015 for the award of Ph. D. (Engg.) degree of Jadavpur University is absolutely based upon his own work under the supervision of Prof. Sheli Sinha Chaudhuri and that neither his thesis nor any part of the thesis has been submitted for any degree/diploma or any other academic award anywhere before.

S. S. Chaudhuri

Signature of the Supervisor

and date with Office Seal

Professor
Electronics & Tele-communication
Engineering Department
Jadavpur University
Kolkata-700 032

Acknowledgement

The dissertation writing process was extremely time-consuming. It was a very lonely experience which was not possible to complete without mental and practical support of numerous people. I owe sincere gratitude towards them who made this dissertation possible.

First and foremost, I want to express my gratitude to my advisor, Prof. Sheli Sinha Chaudhuri, for her unwavering support and advice since my MTech days. I consider myself really blessed to have her as my mentor. I am indebted to her for tolerating me and patiently listening my ideas. Her relentless pursuit of brilliance inspired me to push myself to new heights. She taught me all there is to know about research. She assisted me in developing a methodical representation of my ideas. I would like to thank her for reviewing my work. I express my deepest gratitude to her.

I'd like to thank my students, colleagues, and friends at Jadavpur University in Kolkata and the National Institute of Technology in Agartala for their help and participation. I would also want to thank all of the academic members of the Department of Electronics and Tele-Communication Engineering at Jadavpur University in Kolkata for creating a friendly research environment.

Special thanks to Kakali Mudi, Dr. Apangshu Das and Dr. Shubhro Chakraborty for their emotional assistance to overcome my creative impasse. I'd want to extend my thanks to Dr. Nilanjan Dey, who was instrumental in influencing my early research days. Mithun Bala, Biswajit Betal, Ritam Biswas, Michale and Devraj Dey Thank you very much, buddies for being there. Kiran and Mandira, thank you for the help and support.

Finally, I want to express my thanks to every single member of my family. For all the moral support and incredible opportunities, they have provided me throughout the years, I am grateful to my parents. I want to specially thank my mother for her enduring patience and understanding. I take the opportunity to thank Mampi, Papai, Anisha, Dustu, Tinni, Titir, Habu and Dubu for their unconditional love and support.

Suvojit Acharjee
30/06/2023

Suvojit Acharjee

Abstract

Biomedical signal compression is a challenging task as the information inside a biomedical signal need to be preserved during signal compression. This research work presents a few optimization processes for image and video signal compression.

The image compression technique presented in this article is based on lossy vector quantization process. The codebook of the vector quantization algorithm is optimized using Jaya algorithm. A quantized residue image has been added with the compressed signal to preserve the information inside the biomedical signal. The derived optical disk and retinal blood vessel from the original image as well as the compressed image are compared in terms of correlation, specificity and sensitivity to determine the diagnostic essence preservation. The retention of diagnostic essence is seriously improved after combining the residue image with the compressed signal.

The motion vector estimation process is the computationally expensive process in video coding. As a result, this work proposes optimizing the operational parameters of existing motion estimating algorithms in order to lower the computing complexity of the algorithms. Three different processes based on genetic algorithm, cuckoo search and fuzzy inference to modify the operational parameters of test zone search is presented in this article. The article introduces a new statistic, Motion Factor, which is an approximate assessment of motion in a coding unit and is used to optimise operating parameters of test zone search. The optimization processes significantly reduced the computation cost of test zone search.

Other meta heuristic algorithms such as the cuckoo search and ant weight lifting algorithm are modified and used in new optimised block matching algorithms. The modifications which include the nearest neighbour interpolation for fitness approximation, adaptive termination and initial population selection from deterministic distribution, are designed to lower the computation cost of these algorithms. The performance of these algorithms is compared with the state-of-the-art Jaya algorithm-based motion estimation process.

Keywords: Biomedical signal compression, Nature inspired algorithms, vector quantization, motion estimation, block matching algorithm, test zone search.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"As a matter of fact, when compression technology came along, we thought the future in 1996 was about voice. We got it wrong. It is about voice video and data and that is what we have today on these cell phones."

Compression is the heart of modern digital communication system. In a time when the data generation is increasing exponentially every year, compression helps to manage the data efficiently. Any real time communication is possible because of advancement in compression technology. Compression enabling more faster and reliable communication across network by reducing the bandwidth required to transmit the information. Compression algorithms have become essential tools for every type of signals such as audio, text documents, image as well as videos, enhancing the user experience, enabling seamless multimedia streaming, and facilitating the efficient storage and retrieval of digital content. In a nutshell, compression is the driving force behind the digital revolution, enabling us to harness the vast potential of digital signals.

Compression can be thought of as a mapping process in which the output of an information source is mapped to a series of symbols. Also, compression, on the other hand, can be considered a pattern recognition problem in which repeating patterns within a long stream of information are identified and replaced with smaller symbols. Thus, compression is essential for removing redundancy from a signal so that it may be represented more efficiently.

The compression plays an important role in digital revolution. However, one should acknowledge the limits of compression. Shannon's source coding theorem [1] highlights the basic constraint of compression. The theorem implies that the entropy of the information source is the highest amount of compression attainable for any signal without losing any information from the signal. Understanding the limitations imposed by Shannon's source coding theorem is crucial for the people working with compressed data. Ultimately, compression serves as a powerful tool for managing the vast amounts of digital data everybody encounters in today's interconnected world. While it offers remarkable benefits, it has certain drawbacks. Table 1.1 lists the advantages and drawbacks of signal compression.

Table 1.1: Advantages and drawbacks of signal compression

Advantages	Drawbacks
1) Compression is a great tool for lowering the demand for storage space.	1) The compression process is very computationally expensive.
2) Signal compression reduces the load on communication channels.	2) The compression & decompression processes can have very high space complexity at times.
3) Compressed signals have faster read and write times than uncompressed signals.	

1.1 Rationale for signal compression's potential

The correlational, statistical, and perceptual characteristics of a signal make signal compression possible. The uncompressed signal uses the same amount of memory to represent every symbol. This statistical redundancy can be avoided by allocating memory bits of varying lengths to represent various symbols according to their frequency in the signal. Signal compression techniques such as Huffman coding [2], arithmetic coding [3], and run-length coding take advantage of this statistical redundancy.

Another method for achieving signal compression is to exploit the correlation between nearby signal samples. Correlation can be classified into spatial correlation and temporal correlation. Simple one-dimensional signals like text, biomedical one-dimensional signals like the electrocardiograph (ECG) and electroencephalograph (EEG), as well as multimedia one-dimensional signals like music, speech, etc., and two-dimensional signals like images, have only spatial correlation. Three-dimensional multimedia signals, such as video, however, have both spatial and temporal correlation. Intra-video coding reduces spatial correlation within a video frame, whereas motion estimation utilizes temporal correlation to produce video compression.

A signal's perceptual redundancy refers to the quantity of irrelevant data in the signal that the human brain cannot process. The removal of perceptually redundant material does not affect the viewer's perception of the signal. Joint picture expert group (JPEG) image compression [4] schemes exploit perceptual redundancy to achieve compression.

1.2 Classification of compression algorithms

Compression techniques are classified into two groups based on the amount of information lost during the process: lossy compression and lossless compression (as shown in Figure 1). The lossy compression technique will cause some information to be lost during the compression process. The loss of detail during lossy compression is not perceptible to human senses, and therefore the loss of information is acceptable for many applications such as speech compression, image compression, etc. The most common lossy compression technique is to transform a signal from the time or spatial domain to the frequency domain and use less precise data to store the transformed coefficient. Eventually, limiting the precision of the transformed coefficient led to information loss. Discrete cosine transforms (DCT) or discrete wavelet transforms (DWT) are used to convert a signal from the time or spatial domain to the frequency domain. Other lossy compression techniques use fractal coding or vector quantization for compression.

For certain kinds of applications, every bit of information is necessary to preserve. The lossless compression technique employs entropy coding or decorrelation to reduce the size of a signal without losing any information. Shannon coding's restriction on the minimum length of a coded word is followed by entropy coding. Huffman coding, arithmetic coding, etc. are the most popular examples of

entropy coding. Another popular lossless compression technique is the reduction of autocorrelation inside a signal. Prediction-based compression techniques are a prime example of such compression techniques. The differences between lossy and lossless compression are listed in Table 1.2.

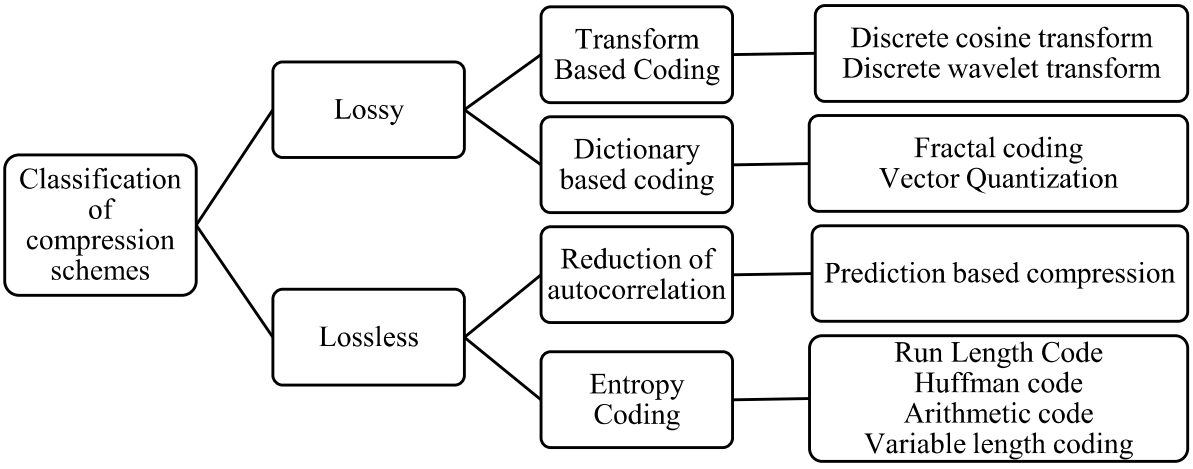


Figure 1 Classification of lossy and lossless compression

Table 1.2: Differences between lossy and lossless compression

Lossy compression	Lossless compression
1) Lossy compression will lose some information during compression.	1) Lossless compression will preserve every information during compression.
2) Lossy compression can achieve higher compression ratio.	2) Lossless compression has a higher compression ratio limit.
3) Lossy compression is an irreversible process.	3) Lossless compression is a reversible process.
4) Lossy compression includes transform-based coding such as discrete cosine transform; discrete wavelet transforms etc. and dictionary-based coding such as fractal coding, vector quantization etc.	4) Lossless compression includes entropy coding such as Huffman coding, arithmetic coding, run length code, variable length code etc. and the decorrelation coding such as prediction-based compression.

1.3 Selection of appropriate compression scheme

Compression cannot be uniform for different types of signals and different types of applications that employ those signals. Various compression schemes are suitable for various applications. The compression schemes vary based on multiple factors. The selection of an appropriate compression strategy undoubtedly improves the performance of a system. The following are the most important criteria for selecting an appropriate compression method for an application:

A) Signal dimensionality: The dimension of the signal used by the system is one of the most important aspects to consider when selecting a compression strategy. For example, systems that

use a three-dimensional signal, such as a video signal, must select a compression strategy that reduces both spatial and temporal correlation.

B) Loss of information: There are two forms of signal compression. Few compression schemes abandon unnecessary information to achieve compression. These schemes are known as “lossy compression”. A lossy compression scheme is a JPEG compression scheme that removes perceptually redundant data from the signal. Lossy compression is an irreversible technique that renders it impossible to reassemble the original signal. On the other hand, lossless compression preserves all the signal's information. For signals like biomedical signals, where every piece of information is crucial, this method is useful.

C) Operational complexity of an algorithm: The selection of an algorithm to compress a signal depends on the operational complexity of the algorithm. The choice of an algorithm to compress a signal is influenced by the algorithm's operational complexity. For instance, an operationally simple algorithm with a decent amount of compression will be given preference over a complex algorithm with a superior level of compression in an application where real-time signal compression is crucial.

D) Cost of implementation: The way the algorithm is implemented is a key selection factor for signal compression algorithms. A hardware-based application will use simple compression algorithms, which are less expensive to implement in hardware. Complex algorithms are implemented through software, as hardware implementation would be very expensive.

E) Robust or resilient: The robustness and resilience of algorithms are significant considerations when choosing a compression technique. A compression system that transmits the compressed signal across a network must be resilient to transmission errors. However, error resilience reduces coding efficiency. The technique, however, must be robust when the compressed signal is stored locally rather than transmitted across a network.

F) Symmetry between encoding and decoding: Video conferencing, for example, necessitates constant video encoding and decoding. As a result, this system has encoding and decoding symmetry. In such applications, the operational complexity is shared equally by the encoder and decoder. However, some applications, such as video databases and picture databases, feature asymmetry between encoding and decoding, where encoding is done once but continual decoding is required. As a result, the operational complexity of such systems is greatly influenced by the complexity of the decoder.

1.4 Biomedical signals

Biomedical signals are a subset of signals that are utilized to extract information about a biological entity [5]. According to Chang & Maura [6], “Biological signals are observations of physiological activities of organisms.” According to the Encyclopedia of Healthcare Information Systems [7], biomedical signals are those that “refer to signals that carry useful information for probing, exploring,

and understanding the behavior of biomedical systems under investigation." From the preceding definitions of biomedical signals, it can be concluded that any form of the electrical, mechanical, chemical, auditory, magnetic, or optical signal that describes any physiological condition of a living species is a biomedical signal. ECG, EEG, and other bioelectrical signals are examples of one-dimensional biomedical signals. Biomedical imaging techniques such as X-rays, ultrasound images, magnetic resonance imaging, and others use non-invasive imaging techniques to visualize a living being's internal physiological status. These are biomedical signals in two dimensions. A few bioimaging processes, such as echocardiography and angiography, collect image sequences or videos to investigate the state of interior organs. These signals are an example of three-dimensional biomedical signals. Even ordinary video sequences from a video conference between a patient and a doctor during tele-treatment can be considered three-dimensional biomedical signals since medical experts extract various vital health information from the patient's visual appearance and activity in the video sequence. Furthermore, several research articles used regular video sequences to detect key physiological and psychological states of the patient, such as pain [8,9], depression [10,11], etc., and assist medical practitioners throughout the telemedicine procedure. As a result, normal video sequences serve as a vital information source for physiological and psychological conditions during the telemedicine process, converting normal video sequences into a legitimate three-dimensional biomedical signal.

1.5 Motivation for biomedical signal compression in telemedicine applications

Biomedical signal compression in telemedicine applications can be classified into two groups. One will be limited to the compression of regular video sequences. Another field of investigation will be the compression method for specialized biomedical signals like ECG, EEG, x-ray images, ultrasound imaging, and so on.

The compression of regular video sequences during the video conference between patients and medical professionals is very much necessary, as the compression will decongest the communication network. Also, the fast read and write times of compressed signals are a necessity in real-time communication systems. Normal video compression also helps to maintain and organize a large video database in a telemedicine system. This enormous database, together with other clinical metrics, can be used as empirical data by several research studies to diagnose various critical physiological and psychological states of patients [8 -11].

The use of biomedical signals such as ECG, EEG, etc., and biomedical imaging systems such as computed tomography (CT), magnetic resonance imaging (MRI), etc. in modern medical diagnosis are generating the sheer amount of data every day. Also, the improvement in technology is contributing to the exponential increase in the volume of digital medical data. For example, in 1990, a CT scan consisted of only 25 slices, with each slice's resolution captured at 16 bits per sample and having 512x512 resolution. By the end of the decade, the number of slices had risen to 80. Today's CT scan is

made up of 600 slices, each with a resolution of 768x768. In thirty years, the memory required to store the output of a single CT scan went from 13MB to 700MB [12]. In another example, the size of a single-channel ECG signal with a sampling frequency of 180 Hz for 24 hours is 23MB. The size increases to 279MB when the number of channel increase to twelve. Figure 2 illustrates the annual increase in the amount of memory needed to store the output of a CT scan.

These are just a few instances of how raw biomedical signals can consume a lot of bandwidth throughout the telemedicine process. However, the public network cannot commit such large amounts of bandwidth to a single application. As a result, compression is required for such vast amounts of data. The ethical and legal reason necessitates that biomedical signal compression must be lossless as biomedical data contains critical information and lossless compression can rebuild the original signal at the receiver. But lossless compression has limitations as mentioned by the Shannon source coding theorem. Sometimes some compression schemes for biomedical signals accept some degree of information less. However, the compression process shouldn't change the diagnostic essence of the signal [13]. Also, the output of lossy compression can be utilized for study and research.

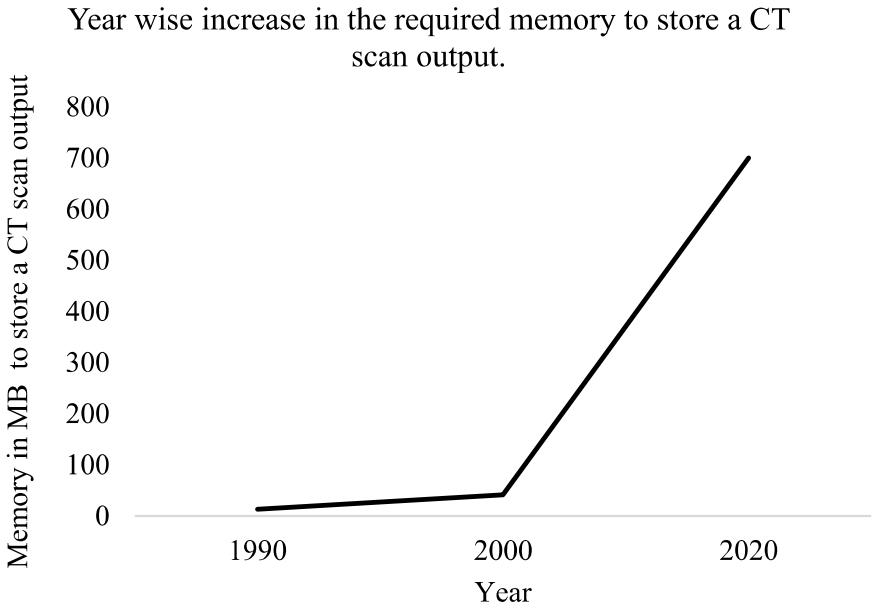


Figure 2 Year wise increase in the required memory to store a CT scan output

1.6 Problem Statement

Compression is important to store or transmit a signal. Biomedical signal compression methods strive to retain as much information as possible inside the signal. But there are many challenges to it. In the following chapters, many well-known signal, image, and video compression algorithms that handle these difficulties within an optimization framework will be thoroughly studied. Computation of rate-distortion performance caused as an effect of compression is also discussed in the chapters.

There are different types of signal and image compression algorithms namely predictive coding, transform coding, vector quantization, etc. This literature has only used the vector quantization method, where the codebook and the index map replaced the original signal. Two codebook optimization algorithms that reduced distortion between the compressed and original signals have been proposed in this literature.

Several techniques in video compression, including the block matching algorithm, optical flow, and pel recursive algorithm, can lessen the temporal correlation between subsequent frames. However, only block-matching algorithms, which estimate the movement of blocks in successive frames to reduce temporal correlation, have been considered in this literature. Several new algorithms have been proposed to optimize the block-matching process. Furthermore, new algorithms are also proposed to optimize the operational parameter of the existing block-matching process, reducing the computation complexity of the existing algorithms.

1.7 Thesis organization

The following is how the thesis is structured:

Chapter 1 discussed the need for biomedical signal compression. The fundamentals of video, image, and signal compression techniques were reviewed in Chapter 2. Here, past studies on these subjects were also explored and different statistical measures used to evaluate the performance of the algorithms were discussed. Chapter 3 will introduce a new codebook optimization algorithm for image compression. The performance of the proposed algorithm is also discussed at the end of chapter 3. Chapter 4 discuss the effect of operational parameter optimization of block matching algorithms. Chapter 5 proposed new optimized block-matching algorithms for motion estimation. The limitation and achievements of the new algorithms are also discussed at the end of this chapter. Chapter 6 provides the concluding remarks of this study. The future scope of the research article is outlined at the end.

Chapter 2:

Fundamentals of Compression and Literature Review

Everyday human being generates quintillion bytes of data [14]. Storing and transmitting such huge amounts of data is a challenging task. Therefore, compression is the most important tool for dealing with such difficulties. Compression algorithms were designed to offer the best rate-distortion performance at the fastest feasible speed.

Different dimensional signals demand a different compression strategy. For example, any video sequence features spatial redundancy produced by the correlation of pixels inside a frame, as well as temporal redundancy caused by the movement of an item in successive frames. Image, on the other hand, cannot have temporal redundancy. As a result, the compression algorithms for an image and a video differ. The upcoming sections of this chapter will go through the fundamentals of various compression algorithms for signals of various dimensions. In addition, the discussion will include a brief review of previous studies on compression and the statistical measures used in the algorithms.

2.1 Fundamentals of Video Compression

Video compression is the most important part of any video coding process. The temporal correlation between successive frames and the spatial correlation inside a frame was exploited to achieve video compression. Standards for video coding divide the entire frame into non-overlapping coding units. These coding units are used to process each frame. Each coding unit is encoded in intra-mode or inter-mode. Intra-mode encoding removes the spatial correlation and inter-coding reduces the temporal correlation. The maximum size of the coding unit varies with the video coding standards. The H.264 video coding standard [15] employed a 16x16 coding unit known as a macroblock. In the H.265 video coding standard [16], the maximum size of a coding unit was 64x64. The H.266 video coding standard [17] raises the maximum coding unit size from 64x64 to 128x128. Both the H.265 and H.266 video coding standards use a tree structure to segment the frame, allowing for additional division of the coding units. In both inter-mode and intra-mode encoding, a projected coding unit was built using samples of previously encoded frames that have the best correlation with the current coding unit. Intra-mode encoding used samples from the same frame to produce a predicted coding unit, whereas inter-mode encoding used samples from previously encoded frames to form a predicted coding unit. The difference between the predicted and actual coding units is transformed and quantized. The quantized transformed coefficients from the previous stage, coupled with the projected coding unit position, are then encoded with entropy coding and broadcast via a network. The projected unit's position is decoded by the decoder. In addition, the decoder decodes the coefficients, which are then inverse quantized, and inverse transformed to recover the difference coding unit. At the decoder, the difference coding unit is added to the predicted unit to get back the current coding unit.

In video compression, intra-mode encoding deals with a single frame at a time. As a result, it is identical to the standard image compression process, which will be covered in detail in the next sections. The most computationally challenging task in a video compression process is inter-mode encoding. Figure

3 shows the block diagram of inter mode encoding in video compression. The search for a predicted unit that has the best correlation with the current coding unit in inter-mode encoding is known as motion estimation as well as the process to add up the difference coding unit with the predicted unit in inter-mode encoding at the decoder is known as motion compensation. Together the inter-mode encoding is also termed motion estimation and motion compensation encoding. The movement of the objects in successive frames in the video will create a temporal correlation. Motion estimation reduces the correlation by estimating the movement of coding units. Motion compensation lowers the inaccuracy caused by inaccurate motion estimation.

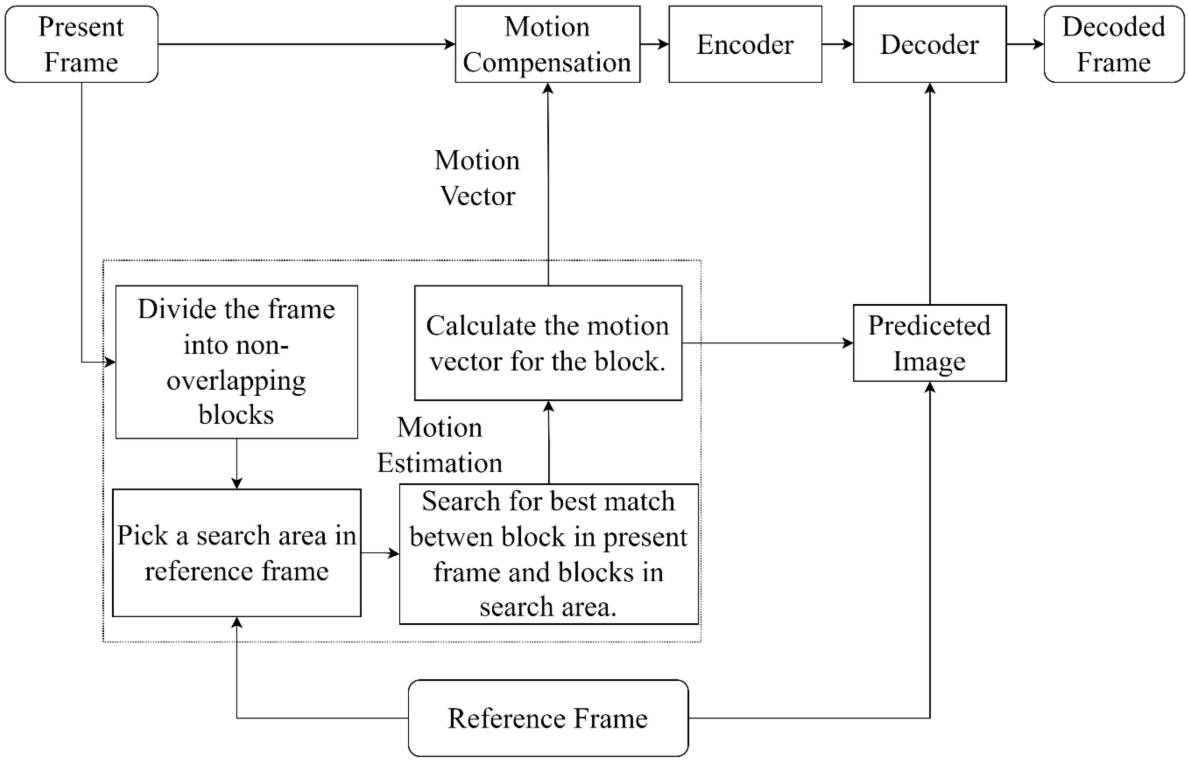


Figure 3 Block diagram of inter mode encoding in video compression

2.1.1 Fundamentals of Block Matching Algorithm

The motion estimation process is carried out through block-matching algorithm which searched the predicted unit at the search area of the reference frame. The correlation between the predicted unit and the current coding unit is measured by mean square error (MSE) [18] or mean absolute difference (MAD) [19] or sum of the absolute difference (SAD) [19]. The search region in the reference frame was centered at the same position as the present frame coding unit and the search area's border was determined at a specified distance on all sides from the center (as shown in Figure 4). The difference between the position of the predicted unit and the position of the current coding unit is known as the motion vector. The static coding units (coding units with no motion) has zero motion vector.

$$MSE = \frac{1}{N^2}\sum_{i=1}^{H}\sum_{j=1}^{W}[currentblock(i,j) - predictedblock(i,j)]^2 \qquad (Eq.1)$$

$$MAD = \frac{1}{N^2}\sum_{i=1}^{H}\sum_{j=1}^{W}|currentblock(i,j) - predictedblock(i,j)| \qquad (Eq.\ 2)$$

$$SAD = \sum_{i=1}^{H}\sum_{j=1}^{W}|currentblock(i,j) - predictedblock(i,j)| \qquad (Eq.3)$$

H and *W* are the height and width of the coding units. *currentblock(i,j)* represents the luminance value at the *(i,j)* position in the current coding unit of the present frame. *predictedblock(i,j)* represents the luminance value at the *(i,j)* position in the predicted unit of the reference frame.

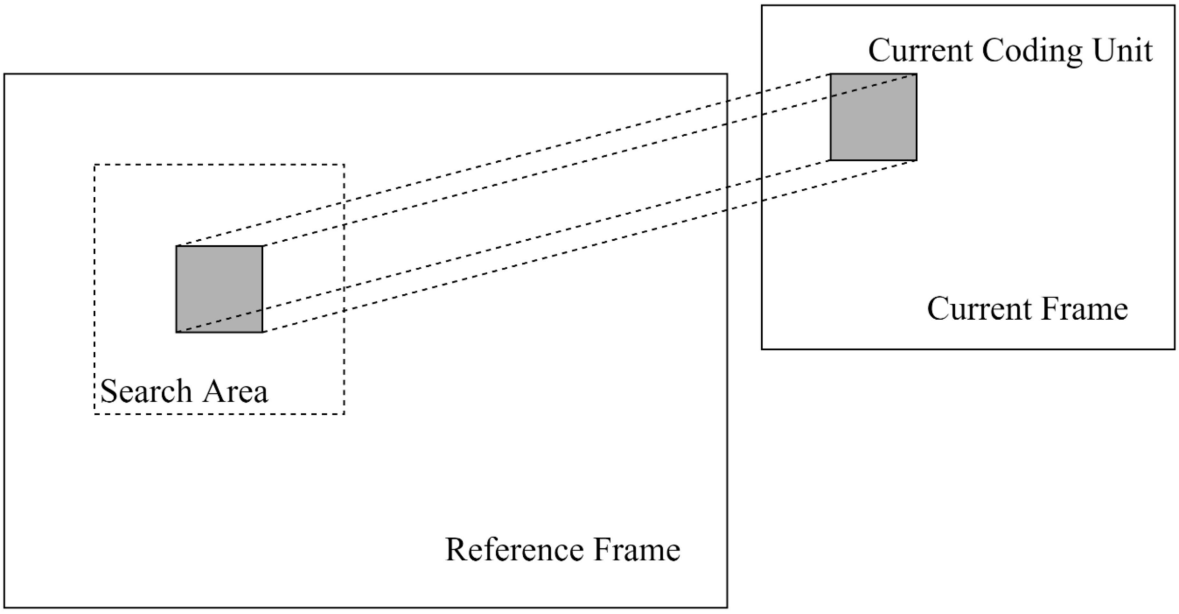


Figure 4 Position of CU in current frame and search area in reference frame for full search algorithm

The position of the frame inside a video determines the reference frame selection. A video is made up of meaningful groups of frames. Every group's initial frame marks the start of a sequence and there is no previous instance to which it can refer for motion prediction. Therefore, the first frame was always encoded exclusively in the intra-mode. This is referred to as an I-frame. Few frames solely use the preceding reference frame to estimate motion. These frames are referred to as P-frames. P-frames are placed at regular intervals throughout a group of frames. The first P-frame in the sequence uses the I-frame as a reference frame. Others refer to the prior P-frame. The remaining frames inside a group of frames are using two reference frames for motion estimation. These frames are known as B-frames and are positioned between I-frame and P-frames. Figure 5 illustrate the frame structure inside the video.

Exhaustive Search (ES) or full search [20] produces the best results of any block matching method because it compares every prediction unit in the reference frame's search region with the current coding

unit in the current frame for the best correlation. However, it is also the most computationally expensive of all block-matching techniques. Every block-matching algorithm measures the improvement in execution speed by referencing the execution speed of the ES method.

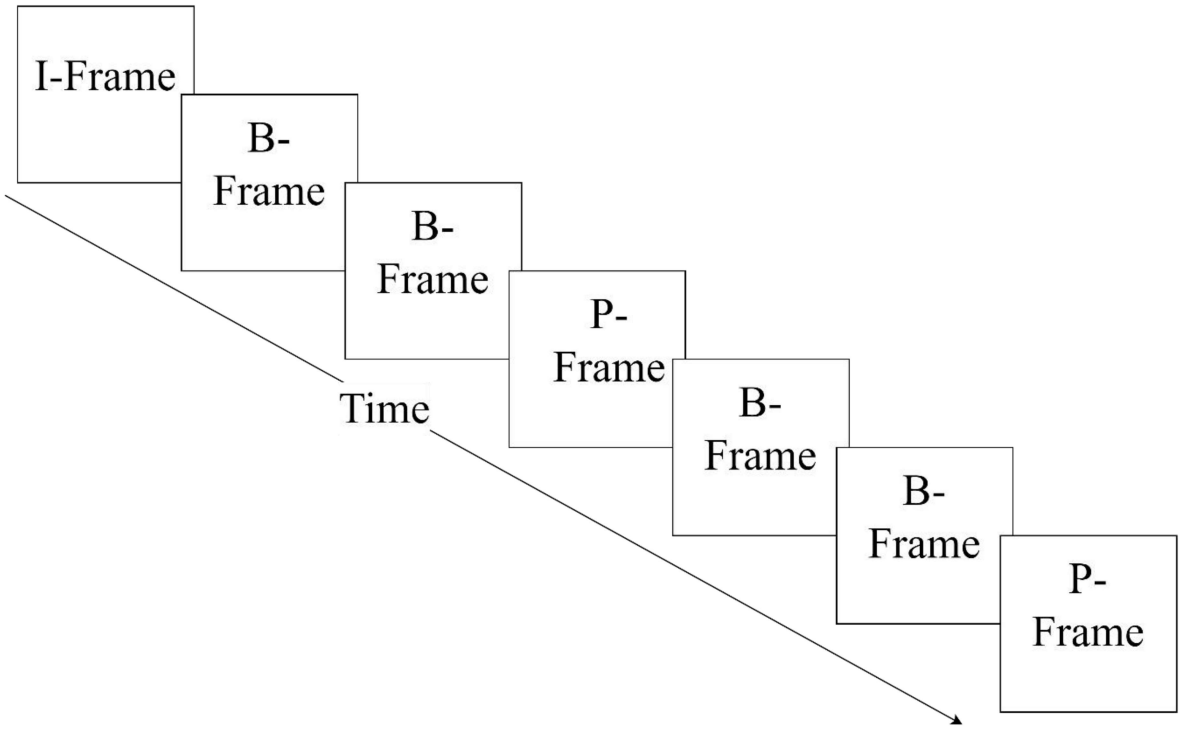


Figure 5 Frame structure inside the video

2.1.2 Literature review

This section describes a brief review of different research article on block matching algorithm.

2.1.2.1 Literature review on block matching algorithm

Several fixed pattern-based block-matching algorithms were developed such as three-step search [21], four-step search [22], diamond search [23], and so on to reduce the computation task of the ES. All the fixed pattern-based fast block matching algorithms were developed with the assumption that the prediction error minimizes as the search moves closer to the global minimum. But Chow et al [24] proved that fixed pattern-based fast block matching algorithms can converge in local minima only. Several prediction-based block-matching algorithms [25,26,27,28] were able to partially overcome the drawback of fixed pattern-based block-matching algorithms. These algorithms used different initial motion vector predictors to shift the initial search point closer toward the global minima. Median-based motion vector predictors [29] and advanced motion vector predictors [30] are quite popular motion vector predictors. But the increasing resolution of video reduced the efficiency of prediction-based block-matching algorithms. Also, the initial motion vector predictor can lead to a false initial search center sometimes. This is due to the movement of tiny elements in successive frames [31].

2.1.2.2 Literature review on optimization in block matching algorithm

Population-based evolutionary algorithms are ideal for discovering global minima. However, evolutionary algorithms have a slow convergence rate. Furthermore, in optimization problems with a small search region, the evolutionary algorithm converged prematurely. Therefore, the original evolutionary algorithms are not directly applicable to the motion vector estimation problem. To adapt the evolutionary algorithm for motion estimation, researchers recommended certain changes to the existing popular evolutionary algorithms.

In genetic motion search [24] Chow et al replace the uniform distribution of the initial population in the genetic algorithm with biased distribution towards the central region of the search space. The biased distribution increased the convergence speed of the algorithm as most of the motion vectors were centered around the search center [32]. The initial random population generation in the genetic algorithm was replaced by a deterministic population generation process by Lin et al in light genetic motion search [33]. This step reduced the use of random numbers in the genetic algorithm. The entire amount of time required by random number generation is negligible when compared to the whole runtime of the method in a broader search region. However, when the search space is small, the same random number generation becomes a computational burden. Four-step genetic motion search combined the advantages of genetic motion search and four-step search [34]. This algorithm does not use the crossover to generate new offspring. Instead, the algorithm relies on the fixed size-based mutation to generate new offspring in the future generations. Table 2.1 compares various genetic motion search.

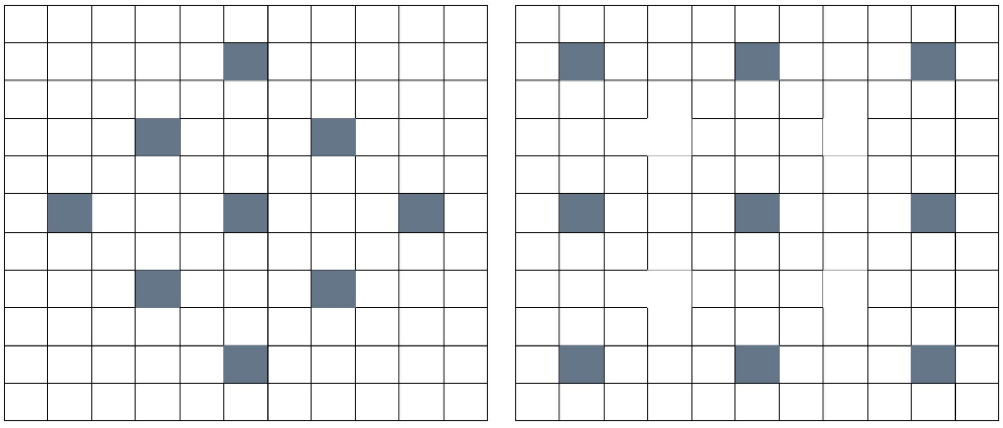
Particle swarm optimization [35,36] was also a popular algorithm for motion estimation. Several research articles explored different variants of particle swarm optimization for motion estimation [37,38]. Yuan et al [37] proposed to assign the velocity of the particles from the neighboring blocks instead of doing it randomly. Pandian et al [38] proposed two changes to improve the performance of particle swarm optimization. A deterministic pattern-based allocation was suggested by Pandian et al. to replace the random assignment of a position to the initial particles (as shown in figure 6). Additionally, Pandian et al. suggested an extra termination criterion for the particle swarm optimization-based motion estimation algorithm in addition to the two standard termination conditions of particle swarm optimization (maximum iteration and convergence of solution). When the location of the global best particle and the location of the current coding unit coincide, the algorithm will terminate. By eliminating unnecessary calculations, these two modifications speed up the algorithm's execution.

A differential evolution-based motion estimation approach was proposed by Cuevas et al. [39], where the initial population was chosen according to a predetermined pattern rather than randomly. To estimate the fitness of a new solution from the calculated fitness of earlier solutions, Cuevas et al. also presented a nearest neighbor interpolation (NNI) technique. This approach approximates the fitness of

a solution from the previously calculated fitness of other solutions. Therefore, NNI assisted in minimizing the computation load. It is a three-step approach to calculate the fitness of the solutions.

Table 2.1: Comparisons between different genetics algorithm-based block matching process.

	GMS [24]	LGMS [33]	4GMS [34]
Encoding of Chromosome	Binary	Binary	Decimal
Generation of Initial population	Randomly generated with a biasing towards the central region of search space.	Generated in a deterministic way with a biasing towards the central region of search space.	Randomly generated with a biasing towards the central region of search space.
Use of Random Number Generator	High	Low	Low
Crossover	Yes	No	No
Mutation	Probability is low.	Probability is high	Probability is High.
Maximum number of iterations	High	Small	4
Prevention against premature convergence?	Yes (Dynamic Population Control Scheme)	No	No
Elitism	No	Yes	Yes



(a) (b)

Figure 6 Deterministic search pattern for PSO based motion estimation by Pandian et al [38] (a) Diamond pattern (b) Square Pattern.

a) If the new search point is closer to the present best fit solution between all generations than a distance d then the fitness function will be calculated using normal procedure (new solution 1 in figure 7).
b) If the new search point is not closer than any of the reference search points calculated in the previous generation the fitness function will be calculated using normal procedure (new solution 2 in figure 7).
c) If the new search point is in a neighborhood by a distance d of previous any search points except the current best solution, then the fitness can be approximated for the new search point from previous reference (new solution 3 in figure 7).

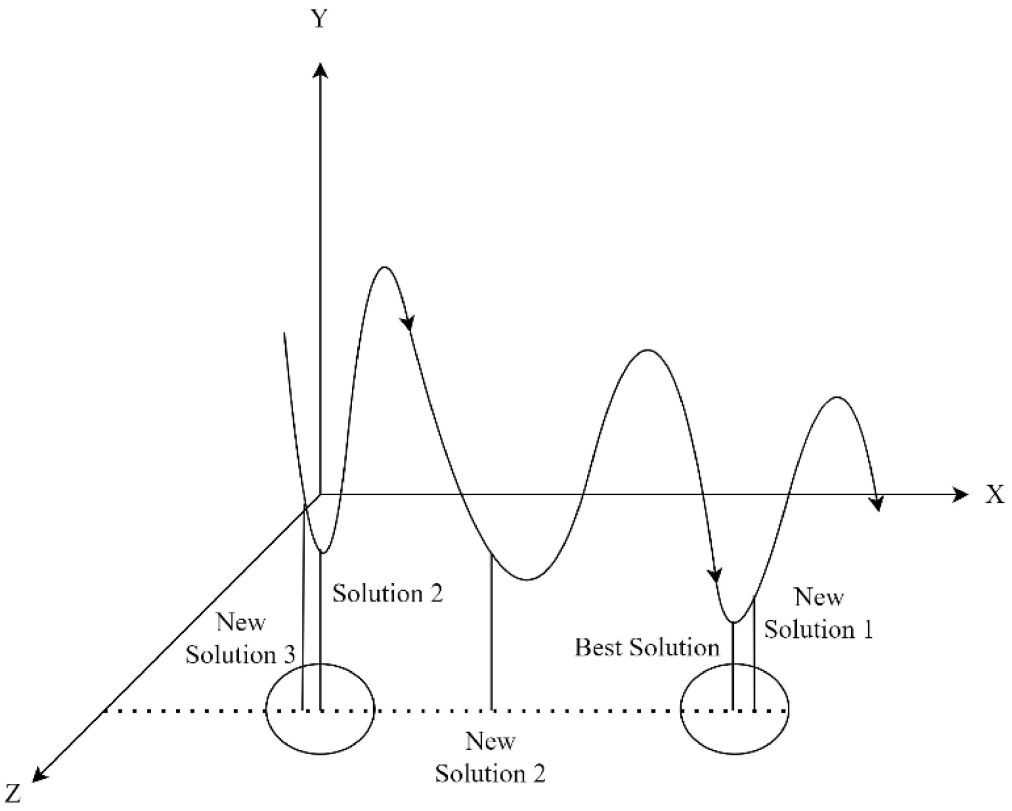


Figure 7 NNI for fitness approximation.

Dixit et al [40] used a normal differential evolution-based block-matching algorithm for motion estimation. The harmony search algorithm-based motion estimation algorithm used the same pattern as [39] to generate the position of initial harmonics. Also, the algorithm used the NNI for fitness approximation. Bhattacharjee et al [41] combined the best of harmonic search and differential evolution for motion estimation. The normal harmonic search was followed from the beginning to the end. Only the pitch adjustment of harmony search was replaced by crossover operation in differential evolution.

An artificial bee colony-based optimization algorithm was developed by Karaboga et al [42]. But the algorithm consumes a huge amount of computation to achieve the best result. This is not applicable to

motion estimation algorithms. Therefore, Cuevas et al proposed to replace the fitness calculation with NNI in the artificial bee colony optimization-based motion estimation algorithm [43]. Later, Hemanth et al [44] proposed various patterns to select the initial position of the bees in the artificial bee colony-based motion estimation process.

Jaya algorithm-based motion estimation process outperformed the test zone search (TZS) algorithm, the benchmark algorithm in the most recently developed H.266 video coding standard [45, 46]. Most evolutionary algorithms have some general parameters such as the number of initial populations, the maximum number of iterations, and so on, as well as some algorithm-specific parameters such as discovery probability in the cuckoo search algorithm, crossover probability, and mutation probability in the genetic algorithm [47], and so on. Incorrect values for algorithm-specific parameters will lead the solution to reach local extrema. Rao et al developed the Jaya algorithm to solve the problem of algorithm-specific parameters affecting the efficiency of various evolutionary algorithms [48].

Initially, the Jaya algorithm-based block matching method differentiates between static coding unit and dynamic coding unit using zero motion prejudgment, as proposed by Nie et al [25]. The algorithm does not process static coding units further, but continues motion estimation for dynamic coding units. For motion estimation, the Jaya algorithm-based block matching method chooses the initial search point from a predetermined pattern (as shown in Figure 6) to ensure speedy convergence of solutions. The fitness for the initial solutions was calculated using any function between equations 1, 2 and 3. However, algorithm employs NNI for fitness approximation of the solutions during further iterations, and preserve the fitness value from previous iterations to reduce redundant computation. The algorithm also employs early termination strategy as proposed by Pandian et al [38]. The discussion of optimization in the block matching method is summarized in Table 2.2.

2.2 Fundamentals of Image Compression

Biomedical imaging is extensively used to diagnose various physiological conditions. Computed tomography, magnetic resonance imaging, digital x-ray, and many more modalities are the source of the huge biomedical images. Effective preservation of these digital medical images is essential to monitor the progression of a patient condition. Image compression is the crucial step to effectively maintain such a biomedical image repository. Ideally, lossless compression is an ethical requirement for biomedical signals. However, minor distortions during lossy compression do not change the diagnostic essence of medical images [13]. Image compression techniques of several types are available. The most prevalent picture compression methods include predictive-based coding, vector quantization-based image compression, and transform domain-based image compression. The lossless image compression approach is predictive-based coding, whereas the other two are lossy image compression techniques.

Table 2.2: Comparisons between evolutionary algorithm-based block matching algorithms.

	Genetic Motion search [24]	Particle swarm optimization-based block matching algorithm [38]	Artificial bee colony optimization-based block matching algorithm [43]	Harmony search optimization-based block matching algorithm [41]	Differential evolution-based block matching algorithm [39]	Jaya optimization-based block matching algorithm [45, 46]
Generation of initial population	Generated from a fixed pattern. (in LGMS and 4GMS)	Generated from a fixed pattern (in PB-PSO)	Generated from a fixed pattern	Generated from a fixed pattern	Generated from a fixed pattern	Generated from a fixed pattern
Use of random number generator	High (in GMS), Low (in LGMS and 4GMS)	Low (in normal PSO) Nil (in PB-PSO)	High	High	High	High
Diversification of search space	Yes (in GMS) No (in LGMS, 4GMS)	Yes	Yes	Yes	Yes	Yes
Minimisation of search space	No (in GMS) Yes (in LGMS, 4GMS)	Yes	Yes	Yes	Yes	Yes
Early termination strategy	No	No (in PSO-BMA), Yes (in PB-PSO-BMA)	No	No	No	Yes
Reduction in computation (history protection, NNI)	No	No	Yes	Yes	Yes	Yes

These popular image compression techniques are block-based compression techniques. The image is divided into non-overlapping blocks and then processed the image using these blocks. The strong correlation between nearby blocks is used in predictive image coding to forecast the current block from prior samples. Transform domain-based image compression transforms the spatial domain signal to the frequency domain signal using discrete wavelet transform or discrete cosine transform. The coefficient of high-frequency components in the blocks is considered noise and therefore discarded. Vector quantization separates the entire picture into numerous tow dimensional non-overlapping blocks, which are ultimately represented as one-dimensional image vector. The vector encoder maps these image vectors to the nearest codeword in the codebook. Instead of the image vectors, the index of the codeword in the codebook along with the codebook are transferred to the decoder. The vector decoder reconstructs the image vector using the index and codebook. However, certain information loss was there in the reconstructed image due to the quantization of image vectors into codewords. The codeword size is the sole parameter that matters for the vector quantization algorithm to provide the optimum rate-distortion performance. As a result, it is superior to other image compression techniques mentioned [49]. So, the scope of this research is limited to different vector quantization approaches in an optimization framework.

2.2.1 Literature review

This section describes a brief review of different research article on vector quantization algorithms.

2.2.1.1 Literature review on vector quantization

The vector quantization method that has gained the most popularity was proposed by Linde, Buzo, and Gray [50]. The algorithm was given the acronym LBG algorithm as a result. The LBG algorithm can be considered a generalized Lloyd scalar algorithm [51]. Consequently, it is also known as the generalized Lloyd algorithm (GLA). This algorithm is composed of four phases.

Step 1, Initially, N_C image vectors were chosen at random to serve as codewords in a codebook of size N_C. Both the codewords and the image vectors are N_b in size.

Step 2, Based on the Euclidian distance between the image vector and the codewords in the codebook, map the image vectors $\{I_1, I_2, \ldots, I_M\}$. The codeword $\{C_1, C_2, \ldots, C_{Nc}\}$ with the lowest Euclidian distance is the one to which the image vector belongs.

$$FlagCluster_{mi} = \begin{matrix} TRUE & if\ difference(C_i, I_m) < difference\left(C_j, I_m\right)_{j \in 1 \ldots N_c | j \neq i} \\ FALSE & otherwise \end{matrix} \quad (Eq.4)$$

The m^{th} row and i^{th} column of the ($M \times N_C$) Boolean matrix *FlagCluster* are denoted as $FlagCluster_{mi}$. This position will be true when the m^{th} image vector belongs to the i^{th} cluster as the Euclidian distance between the m^{th} image vector and the i^{th} codeword is the minimum among all other codewords.

Step 3, The codewords are updated by computing the centroid of the new clusters.

$$C_i' = \frac{\sum_{m=1}^{M} I_m \times FlagCluster_{mi}}{Count\ of\ image\ vector\ in\ i^{th} cluster.} \qquad (Eq\ 5)$$

Step 4, The algorithm will terminate when the overall mean square error between the image vector and their respective codeword of the present iteration is smaller than the mean square error of the next iteration. Otherwise, the algorithm repeats the process from step 2.

There are two major drawbacks to the LBG algorithm. The LBG algorithm creates a locally optimized codebook. Additionally, the LBG algorithm may create an unequal distribution of the codeword and the number of image vectors it represents. Sometimes the LBG algorithm assigns more codewords for a smaller cluster which results in a better approximation of the cluster. As a consequence, the large clusters have a smaller number of codewords which results in a poor approximation of the cluster. The improved LBG suggested a utility index, which is nothing more than the cluster's normalized distortion. [52]. This utility index minimized the distortion inequality between various clusters. But the proposed algorithm cannot completely optimize the codebook. The pattern-based masking LBG algorithm recognizes the repetitive patterns inside an image using the local histogram [53]. The predefined patterns are added up with the peak of the local histogram to generate initial codewords. These codewords are then used in two-step LBG algorithm for the final codebook. But, the pattern-based masking approach will have severe complications when the selection of the local histogram peak is incorrect.

2.2.1.2 Literature review on optimization in vector quantization

Several researchers proposed evolutionary algorithms to optimize the codebook of vector quantization. This section provides a concise overview of the methodology employed by a select few of these algorithms.

2.2.1.2.1 Particle swarm optimization based LBG algorithm

The particle swarm optimization (PSO) algorithm was employed to optimize the codebook generated by the LBG algorithm [54]. The PSO algorithm considered a codebook as a particle in the swarm. Initially, the codebook generated by the LBG algorithm was assigned as the globally best particle in the PSO. Other particles in the swarm are generated randomly. The velocity of the particles is also assigned randomly during the initial phase. The fitness of the particles is calculated based on equation 6.

$$F = \frac{N_b}{\sum_{i=1}^{N_c} \sum_{m=1}^{M} \sum_{j=1}^{N_b} ||I_{mj} - C_{ij}|| * FlagCluster_{mi}} \qquad (Eq.6)$$

The fitness will help to identify the particle best and global best solution during the present iteration. The velocity and the position of the particles are modified based on the updated best solution using equation 7 and 8. The operation will continue until the PSO output converges or the maximum iteration is reached.

$$V'_{ijk} = V_{ijk} + K1.r1.\left(pbest_{ij} - C_{ijk}\right) + K2.r2.\left(gbest_j - C_{ijk}\right) \qquad (Eq.7)$$

$$C'_{ijk} = C_{ijk} + V'_{ijk} \qquad (Eq.8)$$

In equation 7 and 8, *i* represents the size of the codebook and varies between *1* to N_C, *j* represents the dimension of the codeword and varies between *1* to N_b and *k* is the number of particles and varies between *1* to *n*. V'_{ik} and V_{ik} represent the velocity of the i^{th} codeword in the k^{th} particle. *K1* and *K2* are learning rates whereas *r1* and *r2* are the random numbers vary between *0* and *1*. $pbest_i$ is the best position of the i^{th} particle whereas *gbest* is the global best particle. C_{ik} and C'_{ik} are the position of the particle in the present and next iterations. The PSO algorithm-based vector quantization cannot handle a high-velocity particle in the swarm.

2.2.1.2.2 Firefly optimization based LBG algorithm

The firefly optimization algorithm (FOA) [55] works on the mating principle of fireflies. Fireflies use their brightness to attract partners during mating and the dark firefly will move towards the brighter firefly. Chiranjeevi et al [56] proposed a vector quantization algorithm based on FOA. In the approach, the codebook is modelled as a firefly, and its fitness is modeled as the firefly's brightness. All but the brightest firefly is assigned randomly. The output from the LBG algorithm is assigned as the brightest firefly in initial iteration. Fitness of each solution is calculated based on equation 6. Using equation 9, the codebook with the lowest fitness will get closer to the best codebook.

$$r_{ijk} = \left|\left|C_{ijk} - C_{ij1}\right|\right| \qquad (Eq.9)$$

$$\beta = \beta_0.e^{\gamma.r_{ijk}} \qquad (Eq.10)$$

$$C'_{ijk} = (1-\beta).C_{ijk} + \beta.C_{ijk} + u \qquad (Eq.11)$$

C_{ij1} is the best particle in i^{th} iteration. α, β_0, and γ are three random variables that FOA assigns a value at the start. *u* is a random variable between *0* and *1*. The performance of FOA deteriorates when the brighter firefly is not available in the solution space.

2.2.1.2.3 Bat algorithm based LBG algorithm

The Bat algorithm (BA) mimics the echolocation of bats [57]. The bats use echo signals to map their surroundings. The loudness, frequency, and pulse rate of the echo signals help the bats to identify the desired objects in the environment. Karri et al. proposed a fast vector quantization process based on BA [58]. This algorithm treats the codebook as a bat. The output codebook from the LBG algorithm was used as one of the bats among n number of initial bats. The other bats are generated randomly. The velocity, frequency, loudness, and pulse rate for each bat are assigned randomly during the initial phase. The fitness of the bats is calculated based on equation 6. The position of the bats is sorted based on the

calculated fitness. The bat at the top of the list is the best-fit bat. The frequency, velocity, and the position of all the bats except the best bat are updated using equations 12 to 14.

$$f_k' = f_{max} + (f_{min} - f_{max}) * P \qquad (Eq. 12)$$

$$V_{ijk}' = V_{ijk} + \left(C_{ijk} - sbest_{ij}\right) * f_k' \qquad (Eq. 13)$$

$$C_{ijk}' = C_{ijk} + V_{ijk}' \qquad (Eq. 14)$$

The highest and minimum frequencies at which the bat may create an echo signal are given by f_{max} and f_{min}. P is the pulse rate of the echo signal and l is the loudness of the signal. The best fit will have a random walk around the best solution.

$$C_{ij1}' = C_{ij1} + r * P \quad (Eq. 15)$$

r is a random number between *0* and *1*. The new bats are selected if the loudness l is larger than a randomly generated value. The whole process will repeat itself until the BA reaches termination criteria. The BA has several algorithmic-specific operational parameters. Improper selection of the values for these parameters can result a locally optimized codebook.

2.2.1.2.4 Cuckoo Search based LBG algorithm

The cuckoo search (CS) algorithm works on the breeding principle of cuckoo [59]. Cuckoo has a parasitic breeding strategy where the cuckoo lays egg on the nest of other birds. The eggs of the cuckoo and the host bird have an uncanny resemblance. Therefore, the host bird cannot differentiate between its own egg and the cuckoo's egg. Cuckoo eggs hatch faster, and the young hatchlings toss out the host bird's egg from the nest to avoid competition. This will provide the young cuckoo hatchlings access to all of the host bird's resources which is a key to their survival. Sometimes, the host bird may find cuckoo eggs and abandon the nest. Xin she Yang et al proposed the CS based on this behavior of cuckoo. CS has only one algorithm-specific operational parameter, discovery probability (P_d). It is a measurement that how often the host bird will be able to identify the cuckoo egg. As the nest will be abandoned by the host bird after the discovery of the cuckoo egg, a new nest will replace the old nest in the solution pool. Therefore, the discovery probability can be considered as the balance between the intensification and maximization of the search.

Chiranjeevi et al. proposed a vector quantization algorithm based on this CS [60]. The algorithm treats each codebook as a nest of the cuckoo. The fitness of the solutions is calculated based on equation 6. The best solution was selected based on the calculated fitness. A levy flight was performed around the best solution to find better solutions. The algorithm considers P_d% of the worst fit net as the abandoned nests and therefore replaces them with new ones. The process will continue till the CS satisfy the termination condition.

2.2.1.2.5 Whale Optimization based LBG algorithm

Whale optimization algorithm employs local and global search at the same time. This gives whale optimization algorithm a distinct advantage over other metaheuristic algorithms. This algorithm was used to optimize the codebook of vector quantization process [61]. Initial solution in the algorithm was generated randomly. Peak Signal to Noise Ratio was used as cost function by the whale optimization algorithm to evaluate the fitness of each codebook. The algorithm utilizes a circular motion to look for a better solution in the vicinity whereas it also employs a spiral motion for global search. Equation 16 describes the circular as well as spiral motion.

$$C'_{ijk} = \begin{matrix} C_{ij1} - A.dist & when\ r \leq 0.5 \\ dist + e^{bl}.\cos(2.\pi.l) + C_{ij1} & when\ r > 0.5 \end{matrix} \quad (Eq.\ 16)$$

Where,

$$dist = \left|K\ .C_{ij1} - C_{ijk}\right|\ , \qquad A = 2.a.r - a\ , \qquad K = 2.r$$

In the Equation 16, C'_{ijk} is the codebook of next iteration, C_{ijk} is the codebook of current generation, C_{ij1} is the optimal codebook in present iteration. r, l are random numbers between 0 and 1, a is a variable number between 2 and 0, and b is a fixed number.

2.2.1.2.6 Lion Optimization based LBG algorithm

The lion optimization (LO) algorithm simulates the roaming and mating procedure of the lion. LO algorithm-based vector quantization use opposition-based learning [62]. The LO treats each codebook as a lion. The codebook from the LBG algorithm is one of the initial solutions. Other solutions are initialized randomly. Based on the fitness of the solution the best lion is selected. The new position of the lions is generated through mating. The mating process is explained using equation 17. The LO-based vector quantization stops when the algorithm reaches its termination condition.

$$C'_{ijk} = \begin{matrix} C_{ijk} & if\ rand() > pr_i \\ RAND() & otherwise \end{matrix} \quad (Eq.\ 17)$$

where

$$pr_i = 0.1 + \min\left(0.5, \frac{C_{ijk} - C_{ij1}}{C_{ij1}}\right)$$

The result of the LBG algorithm is one of the initial solutions in every evolutionary algorithm-based vector quantization technique. This was due to the fact that including a locally optimized solution as an initial population will assist the evolutionary algorithm in quicker convergence [63]. However, the computation time of these evolutionary algorithm-based vector quantization processes is significantly longer than that of the LBG algorithm. Table 2.3 summarizes the study on vector quantization in optimization frameworks.

2.3 Fundamentals of Signal Compression

Biomedical signal compression is one of the important challenges in telemedicine applications as this signal accounts for the majority of data during continuous telemonitoring. Signal compression, like image compression, may be accomplished through the use of transform coding or vector quantization. In transform coding, the signal is transformed to the frequency domain using Fourier or wavelet transform, and the process uses less precise data to store the transform coefficients. As a result, the compression is achieved at the expense of information loss. Vector quantization is the dictionary-based compression process that quantize the signal vector to the nearest codeword. This is also another lossy compression which is already discussed in detail. Entropy encoder is the lossless signal compression process that follow the limit of Shannon's source coding theorem. The most common entropy encoders are Huffman coding and arithmetic coding.

2.3.1 Literature review on Signal Compression

Huffman coding encodes a symbol based on its probability of occurrence in the signal. The encoded bits that are assigned to a symbol, are decided by the Huffman tree. The symbol with the highest likelihood will have the fewest bits, whereas the symbol with the lowest probability will have the most bits. However, in many cases, the number of independent symbols in a symbol set is enormous. The majority of these symbols have an extremely low probability of occurrence. Nonetheless, this symbol will appear in the Huffman tree and the dictionary. This increases the size of the codewords as well as the process's complexity. Modified implementation of Huffman encoding [64] resolves the issue by combining all the less probable symbols into one symbol.

Another limitation of Huffman encoding is that it only employs codewords of integer length, resulting in suboptimal coding. Arithmetic coding successfully overcomes this issue by encoding the signal with a real number between 0 and 1. The interval shrank as new symbols were encoded. The intervals are separated depending on the probability of occurrence of the symbols, and the interval where the new symbol fits in was zoomed in for the following symbol. The operation will continue until the signal expires. However, the procedure is also terminated when the machine's precision is exceeded. This is the most significant constraint of arithmetic coding, limiting the number of symbols that may be encoded in a codeword. The decoding process of arithmetic coding cannot begin without receiving the whole codeword. This is another drawback of arithmetic coding.

Table 2.3: Comparisons between evolutionary algorithm-based vector quantization algorithms.

	PSO based LBG [54]	FOA based LBG [56]	BA based LBG [58]	CS based LBG [60]	WOA based LBG [61]	LO based LBG [62]
Include the output from LBG as initial solution	Yes	Yes	Yes	Yes	Yes	Yes
Algorithm specific initial parameter	Two	Three	Three	One	Two	One
Use of random number	High	High	High	Low	High	High

2.4 Statistical Measures

Peak signal-to-noise ratio (PSNR): PSNR is described as the logarithmic ratio between the achievable maximum power of a signal and the distortion present in the signal [65]. The PSNR is used to measure the quality of the output signal reconstructed from the compressed symbols. PSNR is high when the distortion in the output is low and vice versa.

$$PSNR = 10\log_{10}\frac{(maximum\ achievable\ power\ of\ a\ signal)^2}{MSE} \qquad (Eq.18)$$

Structure Similarity Index Metric (SSIM): SSIM measures the structural similarity between two signals. It is used to identify the deviation of the output signals from the actual signal [66]. SSIM is 1 when the signals are identical. However, the value of SSIM moves toward 0 as the output signal deviates from the original.

$$SSIM = \frac{(2 * Av1 * Av2 + C1)(2 * Var_{12} + C2)}{(Av1^2 + Av2^2 + C1)(Var_1^2 + Var_2^2 + C2)} \qquad (Eq.19)$$

Av1 and *Av2* are the average amplitude of the two signals. Var_1 and Var_2 are the variances of two signals. Var_{12} is the joint variance between two signals. *C1* and *C2* are the two constants that provides stability to equation 19 when any of the denominators reaches zero.

Speed Improvement Ratio (SIR): SIR quantifies an algorithm's improvement in execution speed when compared to the reference algorithm [67]. It is the ratio of the execution time difference between the algorithm and the reference algorithm to the execution time of the reference algorithm. The SIR will be zero when the speed of the two algorithms is identical. The SIR will increase when the speed of the algorithm is higher than the reference algorithm. But the SIR will be negative when the algorithm is slower than the reference algorithm.

$$SIR = \frac{\left(Time_{ReferenceAlgo} - Time_{Algorithm}\right)}{Time_{ReferenceAlgo}} \quad (Eq.20)$$

Bjontegaard metrics: Bjontegaard metrics include Bjontegaard PSNR (BDSNR) and Bjontegaard rate (BDR). G. Bjontegaard [68] introduced this metric to compare the two video coding settings. BDR calculates the bitrate difference between two video coding settings having the same output distortion. The difference in distortion between two video coding settings with the same bitrate is determined by BDSNR. A negative BDSNR implies that the initial video coding settings perform better in terms of distortion than the second at the same bitrate. A positive BDSNR suggests the inverse. At an identical amount of output distortion, the first video coding settings have a smaller bitrate than the second one when the BDR is positive. Negative BDR implies the contrary.

Bits per pixel (bpp): Bits per pixel [69] evaluates the number of memory bits required in a compressed signal to represent one pixel of a raw signal. It was mostly used to evaluate the performance of image compression. Low bpp is desirable for any image compression algorithm.

$$bpp = \frac{number\ of\ bits\ required\ for\ compression\ signal}{Total\ number\ of\ pixles\ in\ the\ raw\ image} \quad (Eq.21)$$

Compression ratio (C_r) and Compression Factor (C_F): The compression factor (C_F) is a ratio between the size of the uncompressed and compressed signal [70]. The Compression ratio (C_r) is just the opposite. The High compression factor and low compression ratio indicate that the size of the compressed signal is significantly reduced by the algorithm.

$$C_F = \frac{Size\ before\ compreession}{Size\ after\ compression} \quad (Eq.22)$$

$$C_r = \frac{1}{C_F} \quad (Eq.23)$$

Pearson Correlation Coefficient (P_C): Pearson correlation coefficient measures the degree of association between two signals [71]. The correlation is 1 when both signals are identical. The coefficient will move toward 0 when the signals do not correlate.

$$P_C = \frac{covariance(signal_1, signal_2)}{\sigma_{signal_1}\sigma_{signal_2}} \quad (Eq.24)$$

The $\sigma_{signal1}$ and $\sigma_{signal2}$ are the standard deviation of two signals $signal_1$ and $signal_2$.

Sensitivity and Specificity: Sensitivity and specificity describe the correctness of a test [72]. Sensitivity measures the true positive rate by the ratio between true positive events and the sum of true positive and false negative events. Specificity calculates the real negative rate as the ratio of truly identified negative events to the sum of true negative events and false positive events.

$$Sensitivity = \frac{True\ Positive}{True\ Positive + False\ Negative}$$
$$Specificity = \frac{True\ Negative}{True\ Negative + False\ Positive} \qquad (Eq. 25)$$

Chapter 3:

Effect of codebook optimization: Diagnostic preservation

Image compression in medical and health services aids in the efficient use of the network and storage capacity of a hospital. However, attaining significant compression without losing the diagnostic essence of a medical image is a significant challenge. This chapter presents a thorough investigation of the preservation of diagnostic essence in Jaya algorithm-based image compression techniques. The Jaya algorithm is used to optimize the codebook of vector quantization. The performance of the proposed algorithm is evaluated based on PSNR and SSIM between the original image and the compressed image. In addition to the typical assessment metrics, the preservation of diagnostic essence was assessed by comparing the diagnostic region of interest in the original and compressed images in terms of correlation, sensitivity, and specificity.

3.1 Jaya Algorithm

Rao et al. presented the Jaya algorithm to alleviate the difficulties associated with algorithm-specific parameters in nature-inspired optimization algorithms [48]. It only depends on general parameters such as the size of the initial population and maximum iteration number. The Jaya algorithm operates on the premise of pushing the present solution towards the global best solution and away from the global worst solution. The steps of the Jaya algorithm are stated below and the algorithm flow chart of the algorithm is presented at figure 4:

Step 1, The value of the initial population size and maximum iteration is assigned.

Step 2, The initial populations are placed randomly in the search space. Random initialization ensures that solutions are distributed uniformly throughout the search space.

Step 3, The fitness of the solutions is calculated based on the fitness function.

Step 4, Arrange the solutions based on their fitness. The best solution will be at the top of the list. The solution with the worst fit will be at the bottom of the list.

Step 5, Identify the best and worst solution as $Solution_{best}$ and $Solution_{worst}$.

Step 6, The solutions are moved towards the best and away from the worst using equation 26.

$$Solution_i' = Solution_i + r_1 \times (|Solution_i - Solution_{best}|) - r_2 \times (|Solution_i - Solution_{worst}|) \quad (Eq.26)$$

In equation 26, r_1 and r_2 are two randomly generated numbers that lies between 0 and 1.

Step 7, The new position of the solution will be accepted if the new position has better fitness than the present solution.

Step 8, Look for the termination conditions of the algorithm. If maximum iteration has not been reached or the solution has not yet converged, repeat the process from step 3.

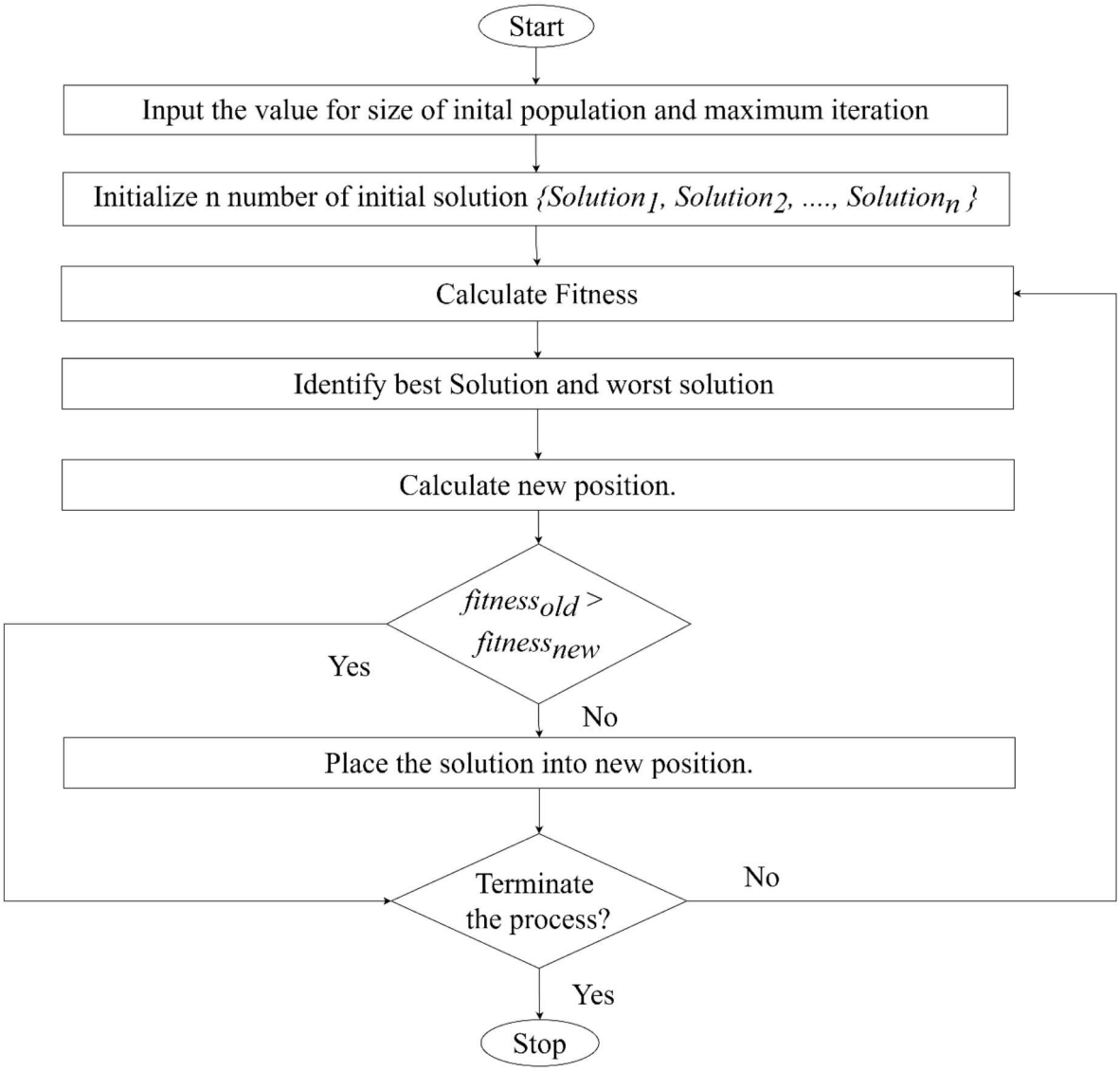


Figure 8 Flowchart of the Jaya algorithm.

3.2 Jaya algorithm-based vector quantization

Image is a two-dimensional signal of size $(M \times N)$. The vector quantization process breaks down the image into small non-overlapping $(m \times m)$ equal-sized blocks. Each block was then converted to a $(1 \times N_b)$ image vector where $N_b=m^2$. The vector quantization process generates a codebook that consist of N_C codewords. The size of each codeword is similar to the size of an image vector. The image vector in the compressed signal was replaced by the index of the codeword in the codebook with the shortest distance to that vector. The N_b number of pixels has been replaced with a value that takes $log_2(N_C)$ bits to represent. Therefore, the computing method in equation 21 for the number of bits required to represent a pixel (*bpp*) may be updated to equation 27 for the vector quantization process.

$$bpp = \frac{\log_2 N_C}{N_b} \qquad (Eq.27)$$

Each codebook $Codebook_i$ is treated as a distinct solution in the Jaya algorithm-based vector quantization process, which is a $(N_C \times N_b)$ size matrix. There are n initial solutions. The solution set $\{C\}$ has a total size of $(n \times N_C \times N_b)$. Equation 6 is used to assess the fitness of each solution or codebook. The Jaya algorithm will try to obtain a solution with maximum fitness. The details of the algorithm are as follows and the algorithm is presented as flowchart in figure 10:

Step 1, The vector quantization process divides the image into M number of image vectors.

Step 2, As stated by the Jaya algorithm, n number of initial solutions must be generated randomly. However, Merwe et al [63] established that using a locally optimized codebook as an initial solution in an evolutionary algorithm-based clustering makes the solution converge quicker. As a result, the locally optimized codebook from the LBG algorithm is included as one of the initial solutions in the Jaya algorithm-based-vector quantization process. Other $(n-1)$ initial codebooks or solutions are generated randomly. Figure 9 depicts a visual depiction of the whole solution matrix.

$$C = \{Codebook_1, Codebook_2, \ldots\ldots, Codebook_n\} \qquad (Eq.28)$$

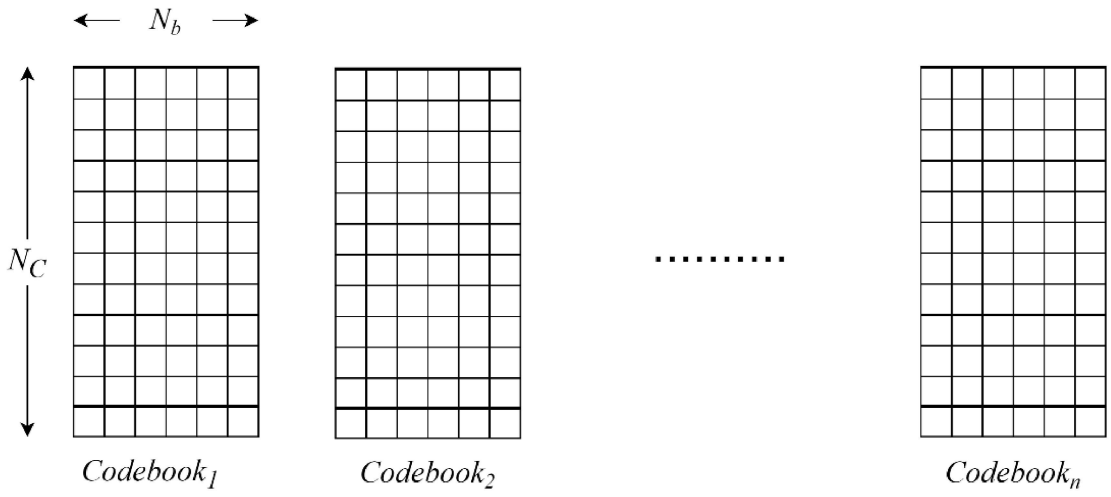


Figure 9 Initial population of Jaya algorithm-based codebook optimization.

Step 3, The fitness of each solution was computed based on equation 6. Arrange the solution based on their fitness. The best and worst fit solution will have the first and last solution in set {C}.

Step 4, The position of the solution has been modified based on equation 26.

Step 5, The solution will move to the new solution if it has better fitness than the present position of the solution.

Step 6, The algorithm will terminate when the solution has converged or the iteration has reached its maximum limit. Otherwise, the process will repeat from step 3.

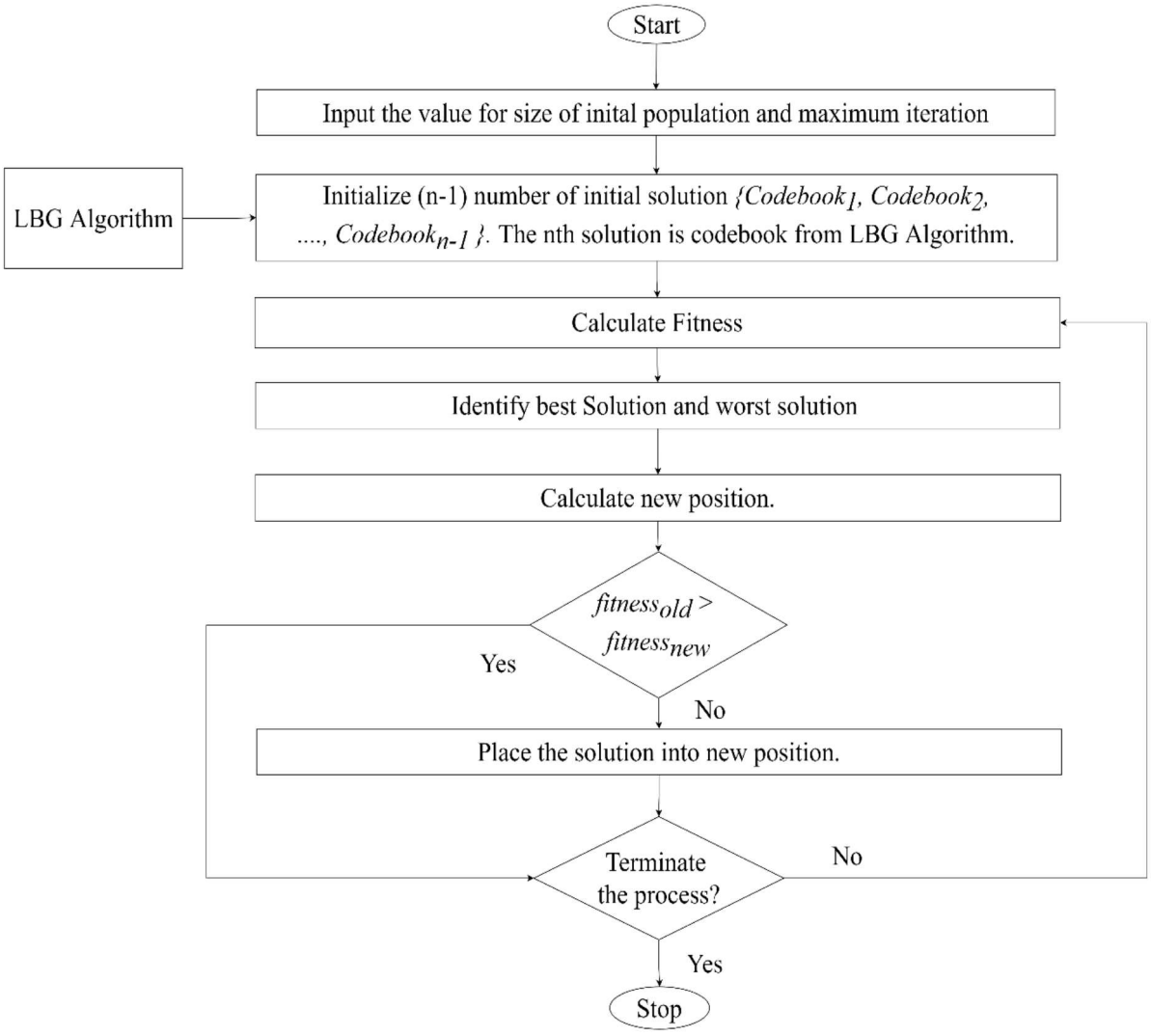


Figure 10 Flowchart of the Jaya algorithm-based vector quantization.

3.3 Performance evaluation of Jaya algorithm-based vector quantization with normal test image

Four standard test images of 512×512 resolution ("Lenna.jpg", "baboon.png", "peppers.png", and "goldhill.png") are used to evaluate the performance of the algorithm. These images are standard test image and used by various algorithms to evaluate performance. The images are available at hlevkin open database (https://www.hlevkin.com/hlevkin/06testimages.htm (last accessed on 21st June, 2023)). The algorithm's initial population and its maximum iteration are set to 30. In the Figures 11 and 12, bpp vs PSNR performance and the speed of the discussed algorithm were compared with several other vector quantization algorithms listed below. The methodologies of the following algorithms have been previously discussed in the literature review (Chapter 2).

(a) LBG algorithm
(b) PSO based LBG algorithm (PSOLBG).
(c) Firefly optimization algorithm based LBG algorithm (FOALBG).

(d) Bat algorithm based LBG algorithm (BALBG).

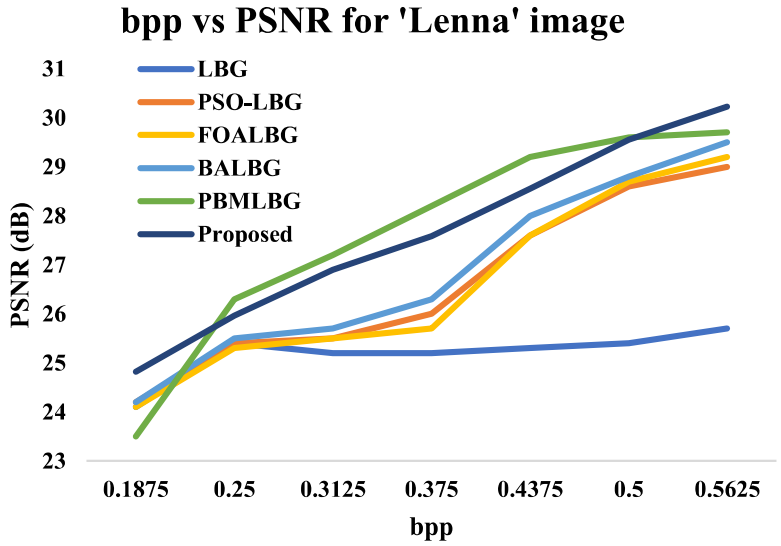
(e) Pattern based masking LBG algorithm (PBMLBG).


(a)

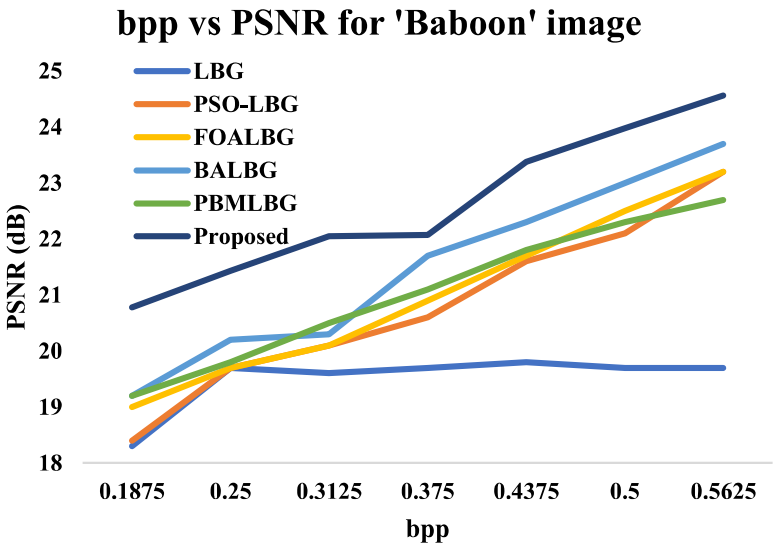

(b)


(c)

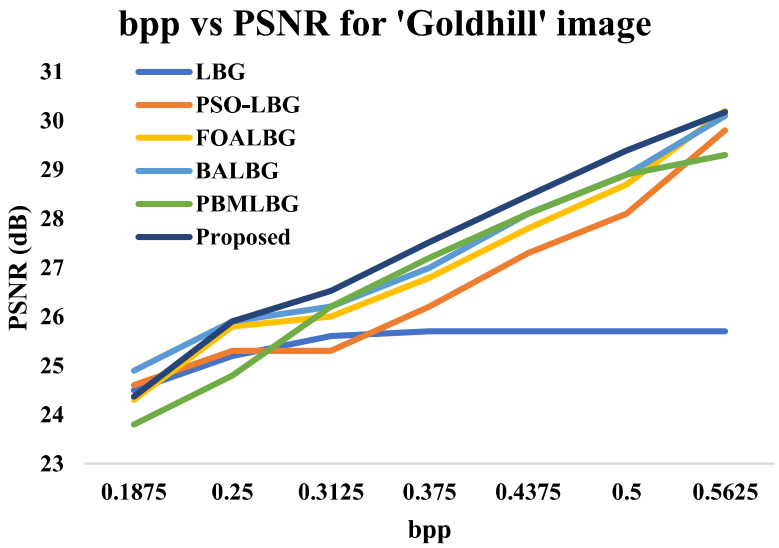

(d)



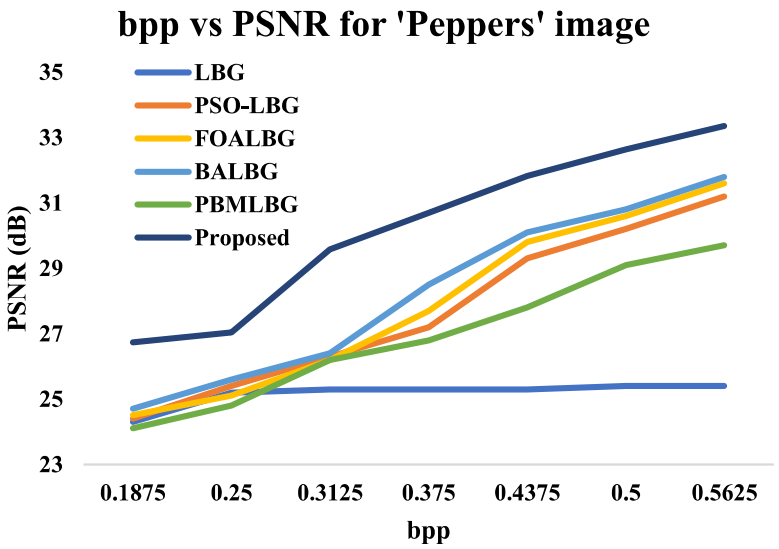
(e)



(f)

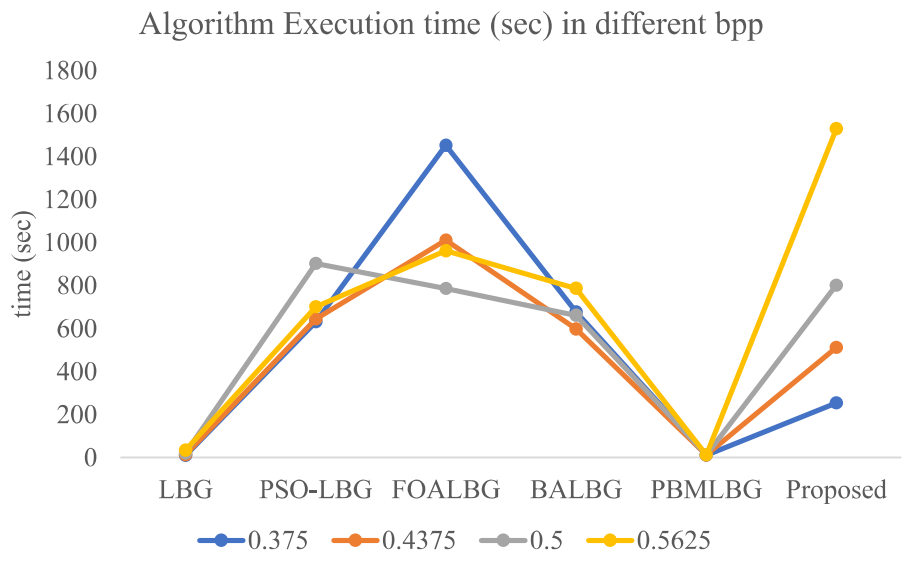


(g)

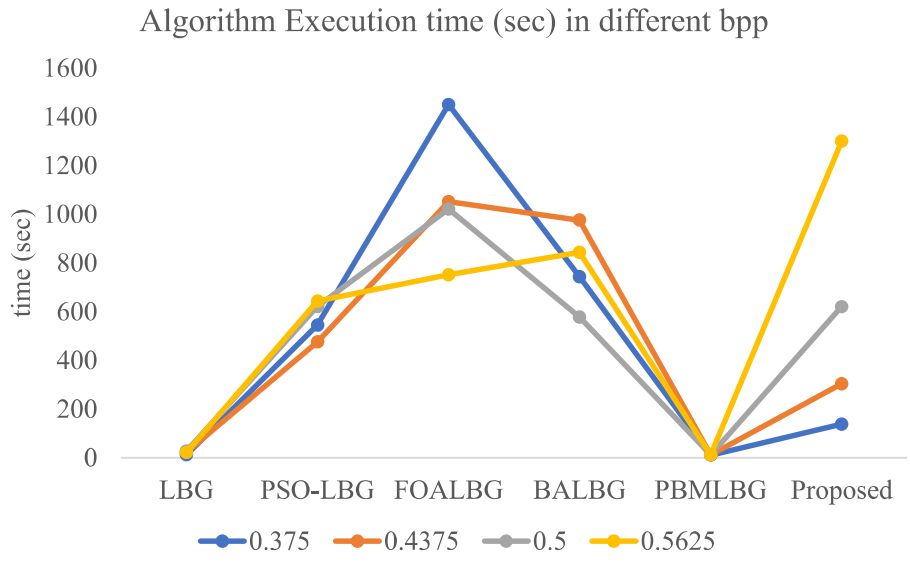


(h)

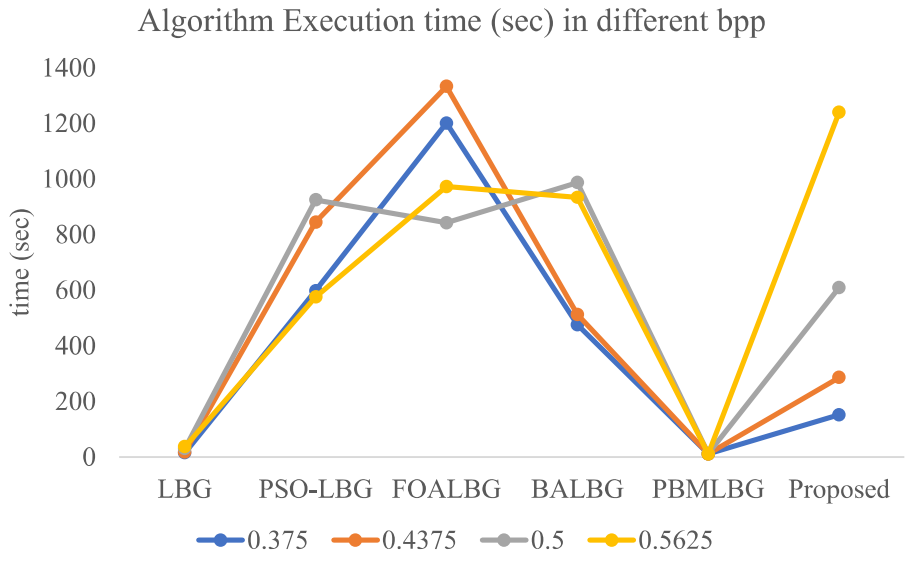
Figure 11 The test images and the comparison of bpp vs PSNR performance (a) Test image Lenna (b) Test image Baboon (c) Test image Goldhill (d) Test image Peppers (e) bpp vs PSNR for test image Lenna (f) bpp vs PSNR for test image Baboon (g) bpp vs PSNR for test image Goldhill (h) bpp vs PSNR for test image Peppers


Algorithm Execution time (sec) in different bpp
time (sec)
1800
1600
1400
1200
1000
800
600
400
200
0
LBG
PSO-LBG
FOALBG
BALBG
PBMLBG
Proposed
0.375
0.4375
0.5
0.5625

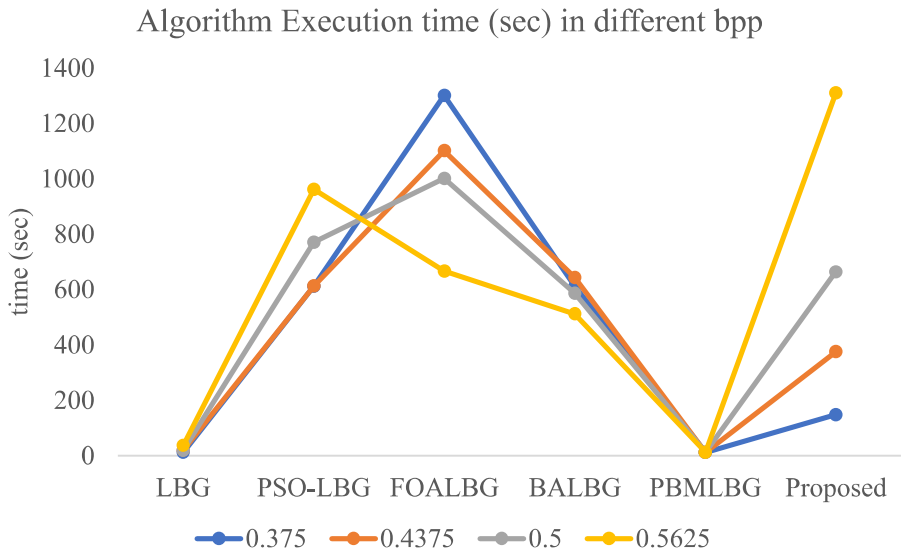
(a)


Algorithm Execution time (sec) in different bpp
time (sec)
1600
1400
1200
1000
800
600
400
200
0
LBG
PSO-LBG
FOALBG
BALBG
PBMLBG
Proposed
0.375
0.4375
0.5
0.5625

(b)


Algorithm Execution time (sec) in different bpp
time (sec)
1400
1200
1000
800
600
400
200
0
LBG
PSO-LBG
FOALBG
BALBG
PBMLBG
Proposed
0.375
0.4375
0.5
0.5625

(c)



(d)

Figure 12 Execution time of various algorithms execution time for different test images with varying bpp (a) Lenna (b) Baboon (c) Goldhill (d) Peppers

In terms of PSNR, the Jaya algorithm-based vector quantization process outperforms all benchmark algorithms except PBMLBG. For the test image "Lenna", Figure 11e demonstrates that the proposed algorithm and LBG algorithm almost maintain the same output image quality. But the performance of the proposed algorithm is superior to the PBMLBG for other test images. The speed of the Jaya algorithm-based vector quantization is faster than every evolutionary algorithm-based LBG process when the size of the codebook is small. However, the algorithm performs slower than others when bpp reached 0.5625. The iteration of the proposed algorithm is set to 30 whereas the iteration of PBMLBG was limited to 2. Therefore, PBMLBG executes faster than Jaya algorithm-based vector quantization.

3.4 Preservation of diagnostic essence in Jaya algorithm-based vector quantization

The preservation of diagnostic essence in Jaya algorithm-based vector quantization was demonstrated using two test images from the diabetic retinopathy dataset (available at https://www.kaggle.com/c/diabetic-retinopathy-detection (Last accessed on 23rd December 2022)). This dataset is an open dataset and free to use for non-commercial research purposes. Table 3.1 lists the PSNR and SSIM between the actual image and the compressed image. Besides these traditional similarity metrics, The article also compare the diagnostic region of interest in the original retinal images and compressed retinal images.

There are several diagnostic regions of interest in retinal imaging. The optical disc and blood vessels in retinal images are two important examples of such regions. The few anomalies associated with retinal blood vessels are hemorrhages, microaneurysms, neovascularization, and so on. On the other hand, pigment epithelium is the anomaly in the optical disc. As a result, we chose these two important areas

of interest to assess the algorithm's ability to preserve diagnostic essence. To detect blood vessels in retinal images, spatial filtering-based image segmentation is performed. Active contour-based image segmentation is utilized in retinal imaging to detect optic discs. Figures 13,14,15 and 16 show the identified blood vessels and optic discs in retinal imaging. The correlation, sensitivity, and specificity between the blood vessels extracted from the actual image are the blood vessels extracted from the compressed images are presented in Table 3.2. Also, Table 3.3 presents the correlation, sensitivity, and specificity between the extracted optical disc from the original image and the compressed image.

Table 3.1: PSNR and SSIM between actual image and the compressed image.

Size of the codebook	13_right		13_left	
	PSNR	SSIM	PSNR	SSIM
8	18.57	0.5156	19.0502	0.5039
16	18.8843	0.5352	32.1804	0.7148
32	30.3243	0.5846	32.349	0.5758
64	36.9035	0.7329	31.0367	0.5938
128	38.5508	0.832	38.1879	0.8156
256	39.7243	0.9649	39.65	0.9417
512	41.0718	0.9712	41.0391	0.952

Table 3.2: Correlation, sensitivity and specificity between the blood vessels extracted from the actual image and the compressed image

Size of the codebook	13_right			13_left		
	Correlation	Sensitivity	Specificity	Correlation	Sensitivity	Specificity
8	0.6038	0.6135	0.9733	0.5856	0.6041	0.979
16	0.6821	0.7268	0.9824	0.7005	0.6944	0.9852
32	0.7755	0.8901	0.9801	0.7414	0.8106	0.9864
64	0.8128	0.9012	0.985	0.7795	0.8397	0.9908
128	0.7787	0.9387	0.9713	0.8029	0.8857	0.9862
256	0.8361	0.9344	0.9849	0.8249	0.8986	0.9859
512	0.8431	0.9378	0.9851	0.8475	0.9094	0.9915

Table 3.3: Correlation, sensitivity and specificity between the optical disc extracted from the actual image and the compressed image

Size of the codebook	13_right			13_left		
	Correlation	Sensitivity	Specificity	Correlation	Sensitivity	Specificity
8	-0.0113	0	0.9863	-0.0025	0	0.9992
16	-0.0057	0	0.9963	-0.0086	0	0.9906
32	0.9379	0.8796	0.9998	0.0498	0.0185	0.9997
64	0.9321	0.0115	0.9997	0.9488	0.9676	0.999
128	0.9545	0.9761	0.9999	0.9662	0.9832	0.9993
256	0.9588	0.981	0.9999	0.9568	0.9931	0.9991
512	0.9601	0.9802	0.999	0.9761	0.9448	1

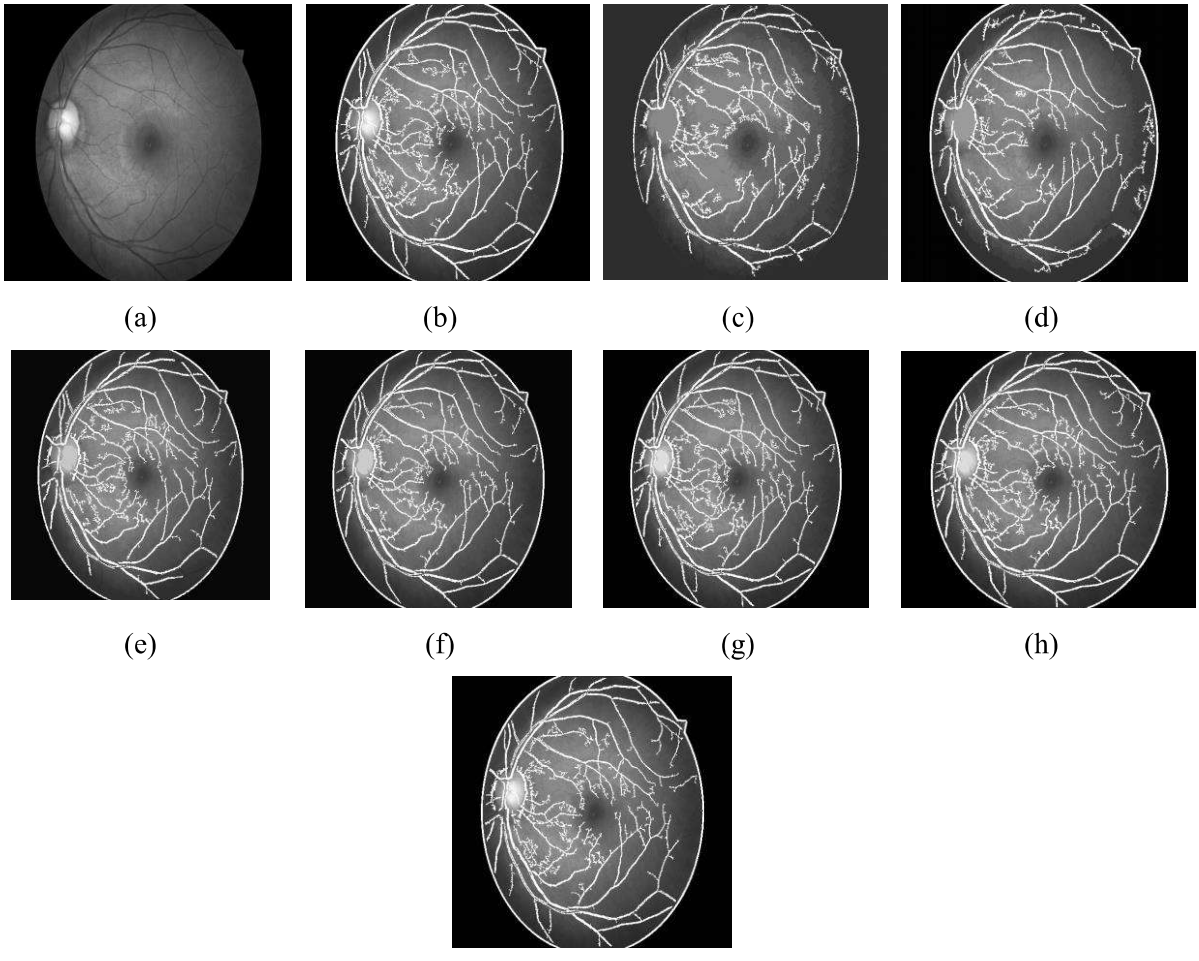
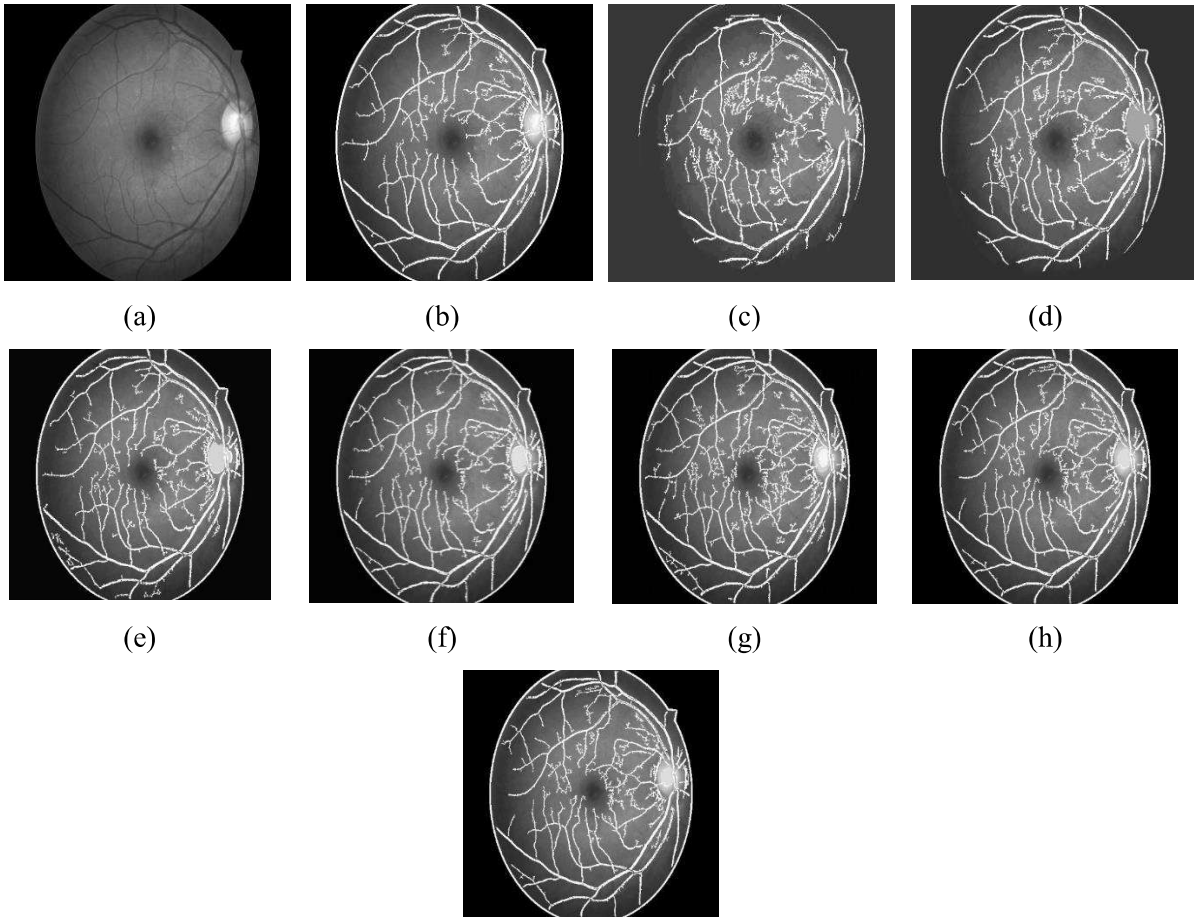


Figure 13 The retinal test image (a) “13_left” (b) Extracted blood vessel on “13_left”. (c-i) Extracted blood vessel on “13_left” compressed by Jaya algorithm-based vector quantization with codebook size (c) 8 (d) 16 (e) 32 (f) 64 (g) 128 (h) 256 (i) 512

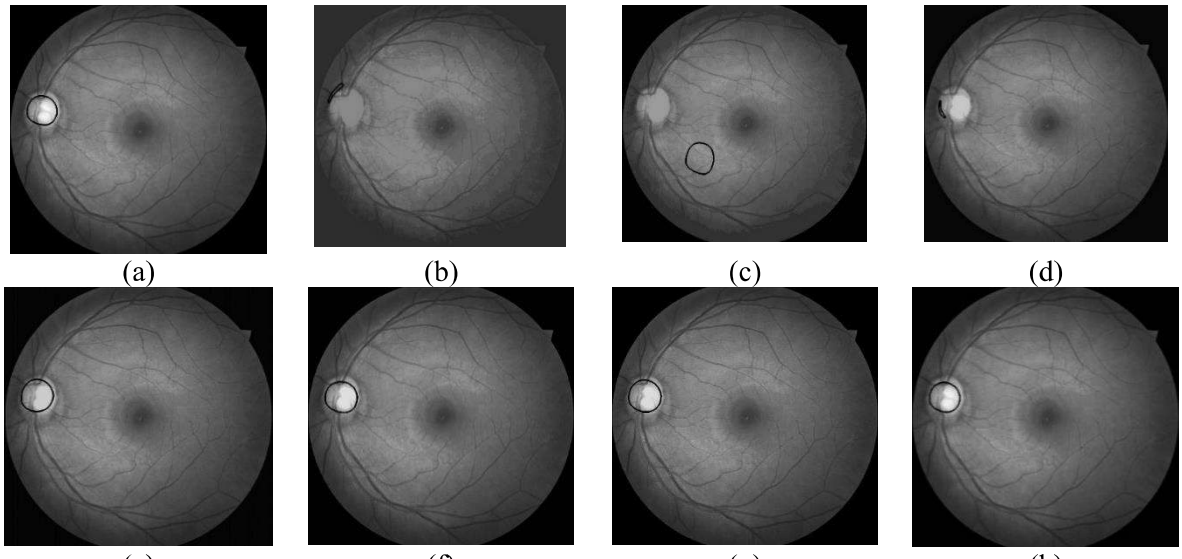


(a) (b) (c) (d)

(e) (f) (g) (h)

(i)

Figure 14 The retinal test image (a) “13_right” (b) Extracted blood vessel on “13_right”. (c-i) Extracted blood vessel on “13_right” compressed by Jaya algorithm-based vector quantization with codebook size (c) 8 (d) 16 (e) 32 (f) 64 (g) 128 (h) 256 (i) 512



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 15 (a) Extracted optical disc on “13_left”. (b-h) Extracted optical disc on “13_left” compressed by Jaya algorithm-based vector quantization with codebook size (b) 8 (c) 16 (d) 32 (e) 64 (f) 128 (g) 256 (h) 512.

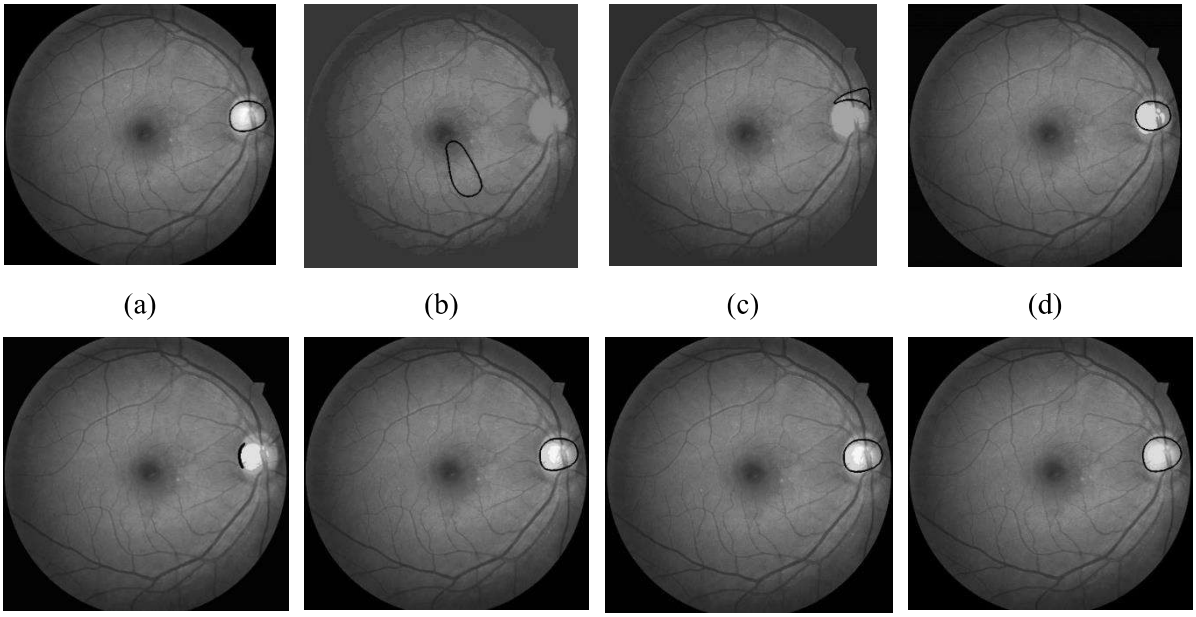


Figure 16 (a) Extracted optical disc on "13_right". (b-h) Extracted optical disc on "13_right" compressed by Jaya algorithm-based vector quantization with codebook size (b) 8 (c) 16 (d) 32 (e) 64 (f) 128 (g) 256 (h) 512.

From Table 3.1, it is evident that the quality of the compressed image increased with the size of the codebook. The diagnostic essence is also improved with the codebook size. After the size of the codebook is 64 or larger than 64, the correlation between the retrieved optical disc from the original and the compressed image improves significantly. The same is true for the correlation between the retrieved retinal blood vessel from the original and compressed images. Image compressed with codebook having codeword number equal to or below 64 often detect false blood vessels and optic disc in retinal images.

3.5 Jaya algorithm-based vector quantization with added residue

Vector quantization is a lossy compression. It is evident from Tables 3.1, 3.2, and 3.3, that the Jaya algorithm-based vector quantization is losing information. But biomedical signal compression should preserve information. As a result, rather than only transferring the codebook and index map to the decoder, a quantized residue matrix produced by subtracting the original and compressed images is also transmitted to the decoder. This residue matrix will compensate for the information loss at the decoder. However, adding an extra residue matrix will reduce the compression efficiency. The process to add and encode the residue matrix is described below:

Step 1, Calculate the codebook and the index map using Jaya algorithm-based vector quantization algorithm.

Step 2, Reassemble the image using the codebook and index map.

Step 3, Calculate the difference between the original image and the reassembled image. The gray image has an intensity between 0 and 255. Therefore, the difference matrix will have a value between *-255* and *255*.

$$D = Reassembled\ Image - Original\ Image \quad (Eq.29)$$

Step 4, The quantized difference matrix will have 256 quantization levels. The first half of these quantization levels are for the positive numbers in the difference matrix which lies between 0 and 255. Another half of the quantization levels are for the negative numbers. As a result, the positive values in the difference matrix will be assigned to a level between 0 and 128. Negative values in the difference matrix will be placed above the 128th quantization level in the quantized difference matrix using equation 30.

$$Q_D(x,y) = \begin{cases} \dfrac{D(x,y)}{2} & D(x,y) \geq 0 \\ 127 + \left|\dfrac{D(x,y)}{2}\right| & D(x,y) < 0 \end{cases} \quad (Eq.30)$$

In Equation 30, $Q_D(x, y)$ is the quantized difference matrix. x and y are position in the matrix.

Step 5, The quantized residue matrix is encoded using the lossless Lempel-Ziv-Markov chain compression algorithm (LZMA) [73]. The LZMA technique is also used to encode the index map.

Step 6, The decoder receives the codebook, as well as the encoded index map and quantized residue matrix.

In the decoder, the quantized difference matrix will do the reverse steps to reconstruct the difference matrix. However, due to quantization error, the reconstructed difference matrix at the decoder will not be the same as the difference matrix in the encoder. The decoder will follow the below steps:

Step 1, Decode the encoded index map and quantized difference matrix.

Step 2, Perform the opposite steps of equation 30.

$$D'(x,y) = \begin{cases} 2 \times Q_d(x,y) & Q_d(x,y) < 128 \\ \big(127 - Q_D(x,y)\big) \times 2 & Q_d(x,y) \geq 128 \end{cases} \quad (Eq.31)$$

The difference matrix in the encoder and decoder are a little different because of the quantization error. Therefore, the difference matrix is represented using $D'(x,y)$.

Step 3, Reassemble the image using the codeword and the index map.

Step 4, Add the difference matrix with the reassembled image to get back the original image.

Table 3.4: PSNR and SSIM between actual image and image compressed with Jaya algorithm-based vector quantization with added residue.

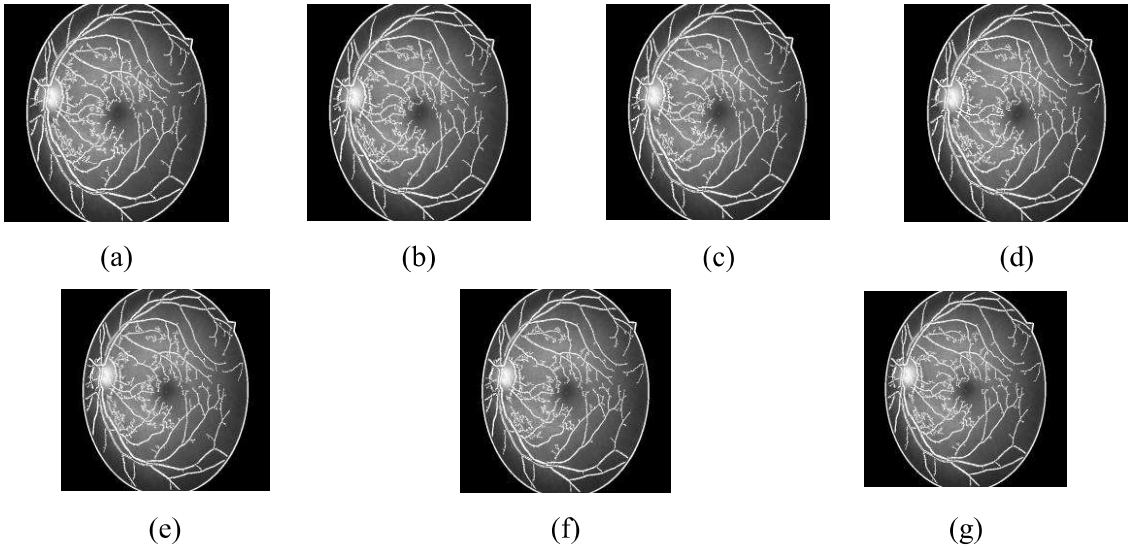
Size of the codebook	13_right		13_left	
	PSNR	SSIM	PSNR	SSIM
8	53.0742	0.9965	53.0762	0.9965
16	53.0767	0.9962	53.0772	0.9965
32	53.0692	0.9961	53.0834	0.9962
64	53.0796	0.9961	53.0698	0.9962
128	53.07	0.996	53.0645	0.9962
256	53.0668	0.9959	53.0763	0.9962
512	53.0613	0.9958	53.0901	0.9962

Table 3.5: Correlation, sensitivity and specificity between the blood vessels extracted from the actual image and the image compressed with Jaya algorithm-based vector quantization with added residue

Size of the codebook	13_right			13_left		
	Correlation	Sensitivity	Specificity	Correlation	Sensitivity	Specificity
8	0.9229	0.9676	0.9937	0.9302	0.965	0.994
16	0.8649	0.9284	0.9892	0.939	0.9686	0.9951
32	0.8488	0.9263	0.989	0.931	0.945	0.9956
64	0.8588	0.9795	0.9949	0.9318	0.9643	0.9955
128	0.8458	0.9742	0.9954	0.9394	0.9726	0.9962
256	0.8531	0.9816	0.9936	0.9335	0.9706	0.9955
512	0.8543	0.9855	0.9938	0.921	0.9727	0.9955

Table 3.6: Correlation, sensitivity and specificity between the optical disc extracted from the actual image and the image compressed with Jaya algorithm-based vector quantization with added residue

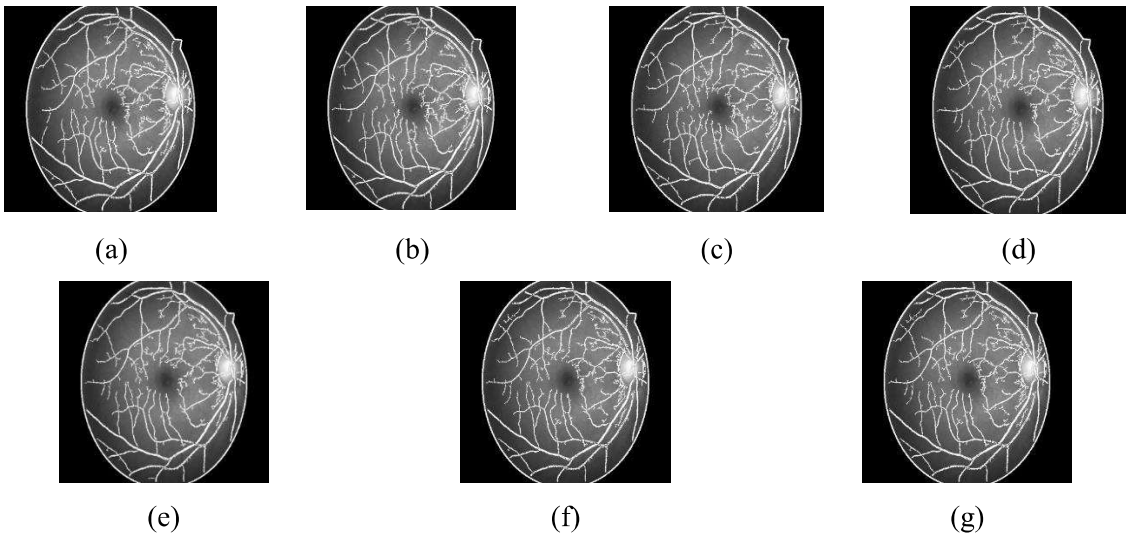
Size of the codebook	13_right			13_left		
	Correlation	Sensitivity	Specificity	Correlation	Sensitivity	Specificity
8	0.9969	0.9958	0.9999	0.9958	0.9963	0.9999
16	0.9963	0.9984	0.9999	0.996	0.9964	0.9999
32	0.9951	0.9922	0.9999	0.9971	0.9949	1
64	0.996	0.9917	1	0.9968	0.9945	1
128	0.9964	0.9936	1	0.9972	0.996	1
256	0.9959	0.9921	1	0.9965	0.9926	1
512	0.9961	0.9924	1	0.9967	0.9964	1



(a) (b) (c) (d)

(e) (f) (g)

Figure 17 Extracted blood vessel on "13_left" compressed by Jaya algorithm-based vector quantization with added residue and codebook size (a) 8 (b) 16 (c) 32 (d) 64 (e) 128 (f) 256 (g) 512.



(a) (b) (c) (d)

(e) (f) (g)

Figure 18 Extracted blood vessel on "13_right" compressed by Jaya algorithm-based vector quantization with added residue and codebook size (a) 8 (b) 16 (c) 32 (d) 64 (e) 128 (f) 256 (g) 512

3.6 Diagnostic essence preservation in Jaya algorithm-based vector quantization with added residue

The two biomedical test images "13_left" and "13_right" are used to evaluate the changes in performance after adding the residue matrix in Jaya algorithm-based vector quantization. Table 3.4 illustrates the PSNR and SSIM of the original image with the image compressed using the new algorithm modification. In addition, Tables 3.5 and 3.6 assess the retention of diagnostic essence by contrasting the optic disc and blood vessels retrieved from the original image with the image

compressed using the algorithm with added residue. The comparison is made in terms of correlation, sensitivity, and specificity.

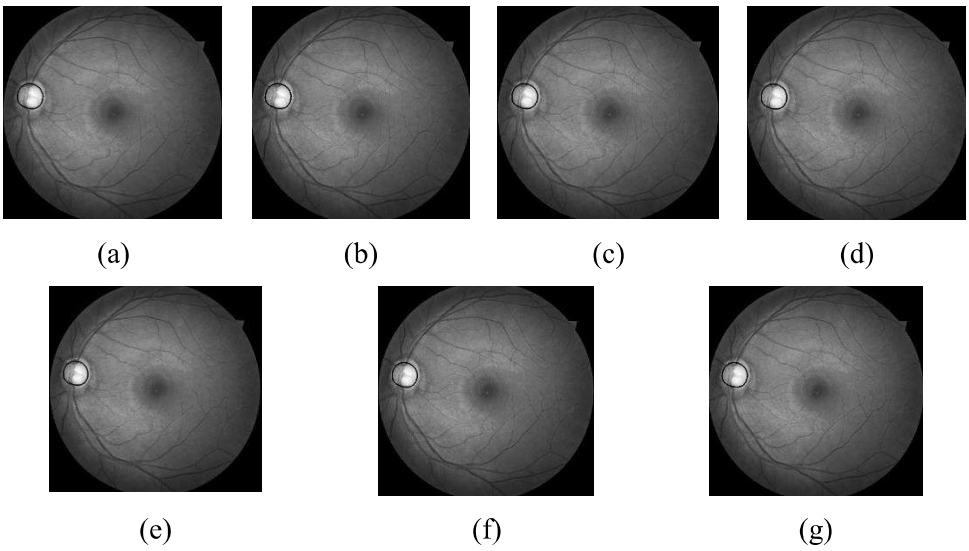


Figure 19 Extracted optic disc on "13_left" compressed by Jaya algorithm-based vector quantization with added residue and codebook size (a) 8. (b) 16. (c) 32. (d) 64. (e) 128. (f) 256. (g) 512.

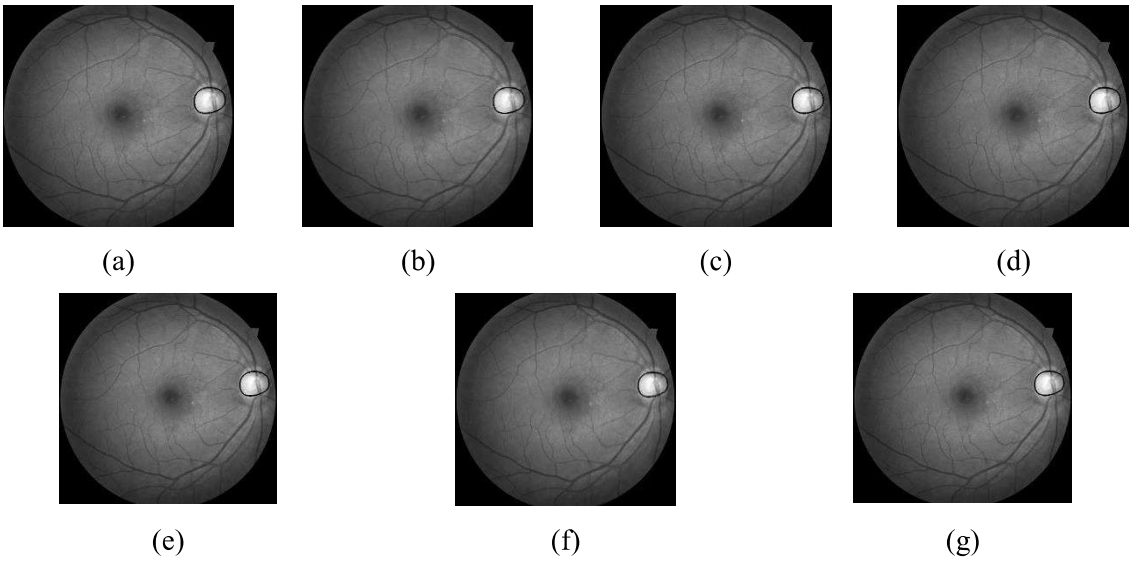
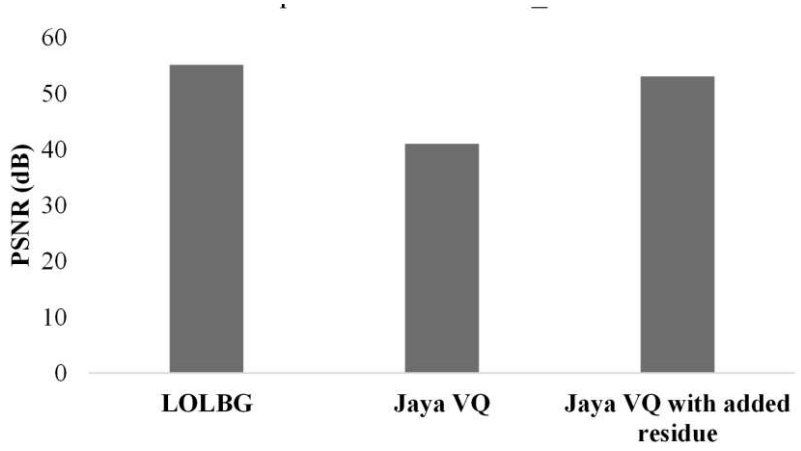


Figure 20 Extracted optic disc on "13_right" compressed by Jaya algorithm-based vector quantization with added residue and codebook size (a) 8. (b) 16. (c) 32. (d) 64. (e) 128. (f) 256. (g) 512.

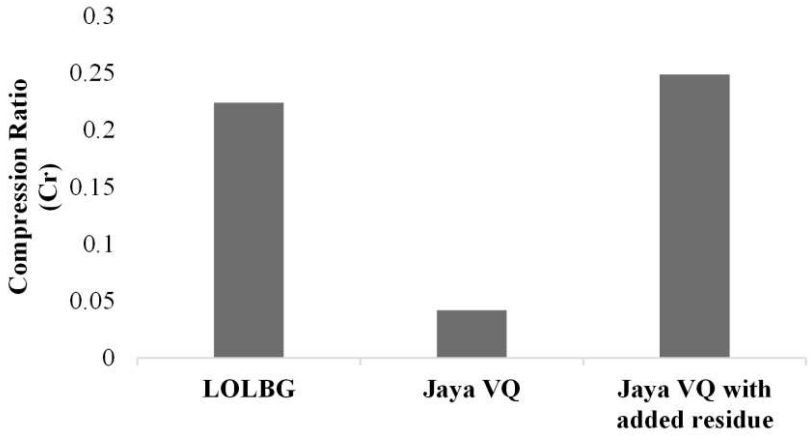
The PSNR and SSIM performance in Table 3.4 indicates that the added residue reduces the loss of information to a negligible level. The retrieved blood vessel and optical disc from the compressed retinal pictures have high sensitivity and specificity which indicates a significant improvement in the diagnostic essence after the inclusion of the residue matrix.

3.7 Comparison with Lion optimization-based vector quantization process

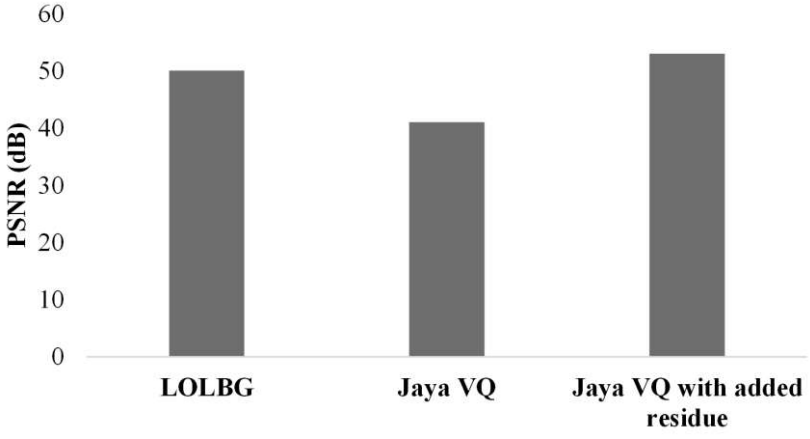
The lion optimization-based LBG algorithm is the state-of-the-art bio-medical image compression algorithm. The Lion optimization-based LBG [62] outperformed other image compression algorithms in terms of PSNR and compression ratio. The output from Jaya algorithm-based vector quantization process is also compared with the Lion optimization-based LBG algorithm in Figure 21.



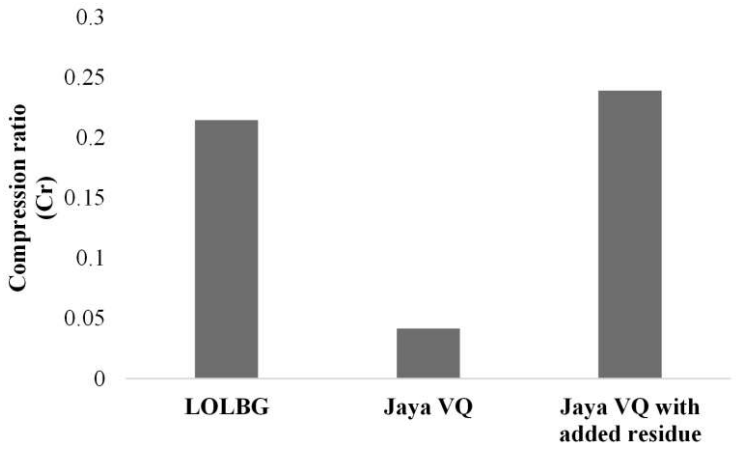
(a)



(b)



(c)



(d)

Figure 21 Comparison of performance of various algorithms (a) PSNR for "13_left" (b) Compression ratio (C_r) for "13_left" (c) PSNR for "13_right" (d) Compression ratio (C_r) for "13_right".

The PSNR from the lion algorithm-based LBG and Jaya algorithm-based vector quantization with added residue is almost equal for "13_left". However, PSNR from Jaya algorithm-based vector quantization is slightly better than lion algorithm-based LBG for "13_right". However, the lion optimization-based LBG always achieved a better compression ratio than Jaya algorithm-based vector quantization with added residue. The Jaya algorithm-based vector quantization without the residue will always achieve a far better compression ratio. But, the output distortion in Jaya algorithm-based vector quantization without the residue is very high compared to Lion optimization-based vector quantization algorithm and Jaya algorithm-based vector quantization with the residue.

Appendix

The extraction of retinal blood vessel from fundus retina images

The following steps are used to extract retinal blood vessels from retinal fundus images.

Step 1, The intensity profile was extracted from the smoothened fundus retinal images.

Step 2, The contrast of the intensity profile was enhanced and a special average filter was used to identify the high frequency regions.

Step 3, The high frequency regions are the blood vessels in the retinal images which are then extracted using an adaptive threshold-based image binarization process.

The extraction of optical disc from fundus retina images

The extraction process of optical disc from the retinal fundus images are listed below.

Step 1, The arteries inside the retinal images were cleared using image morphological operations.

Step 2, Filter was applied to remove the noise from the image.

Step 3, The brightest area in the filtered image is selected as the optic disc. The disc area was further segmented using an active contour model.

Chapter 4:

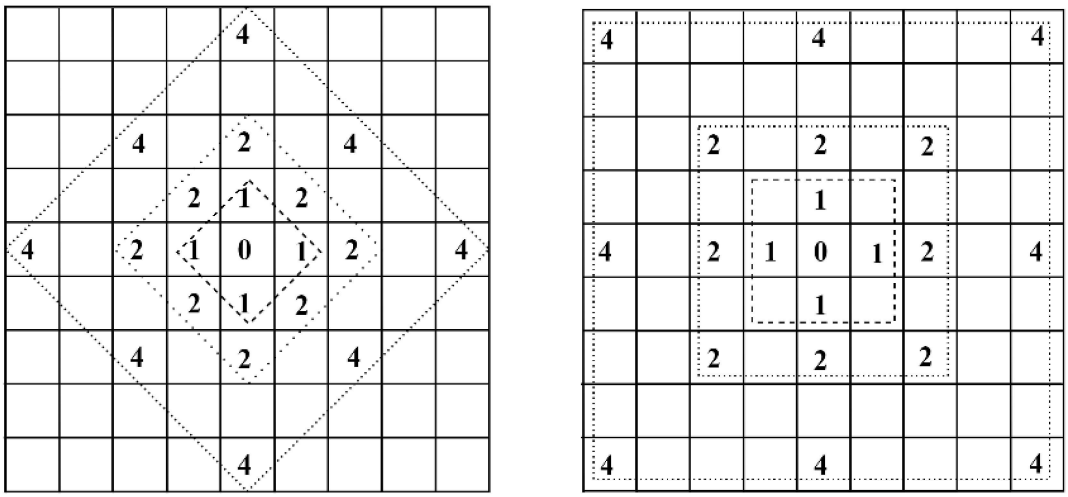
Operational parameter optimizations in block matching algorithm

Motion estimation is an integral part of any video coding standard as well as the most computationally expensive process. Block matching algorithms are the most common technique to estimate the motion inside a frame. These algorithms divide the whole frame into non-overlapping coding units and the position of each coding unit is searched in the search area of the previous frame. The difference in coding unit position in the reference frame and current frame is the motion of the coding units. The full search algorithm compares every coding unit in the search area for the best estimation. However, that is very computationally expensive. Therefore, several other block-matching motion estimation algorithms were developed to reduce the computation expense of a full search. Test zone search is one among them which is able to retain the motion estimation quality of the full search algorithm at reduced computation cost. The test zone search is also the benchmark algorithm in versatile video coding.

4.1 The test zone search algorithm

The test zone search algorithm is a combination of zonal and sampled extensive exploration of the search area. As a result, the algorithm maintains the balance between local search and expensive brute-force search. The test zone search algorithm initially performed a zonal search around the search center. The search center was decided by the motion vector predictor. The advanced motion vector predictor [30] or the history-based motion vector predictor [74] are commonly used to predict the initial search center. The zonal search was accomplished by a variable-size square or diamond pattern search window. The size of the search window will vary between 0 and the size of the search area. The test zone search terminates when the best match is found at the search center after the initial zonal search. The test zone search continues with a two-point search to investigate the remaining undiscovered coding units for a further better result when the distance of the best match is merely one from the search center. The search process will perform a refinement search when the best match was at a distance greater than one but smaller than the sub-sampling frequency of the search window during raster search (I_{raster}). The raster search is performed when the best match of the zonal search is having a distance greater than the I_{raster} from the search center. The raster search samples the whole search window uniformly at a rate of I_{raster} and each sampled coding unit in the search area is compared with the present coding unit. The best match among the sampled coding unit is used as the new search center for the refinement search.

The test zone search is faster than the full search. However, there are some areas for improvement. Parmar et al. [75] proposed a pentagon search pattern for the initial zonal search to further improve the search speed. The pentagon search pattern was reduced 32% search points in the zonal search. Kibyea et al. [76] replaced the raster search with a small and large diamond pattern search, which increased the speed of the test zone search up to 49%. Goncalves et al. proposed an octagonal axis pattern-based raster search. This increased the test zone search algorithm speed by 60% [77].



(a) (b)

Figure 22 Zonal search with variable sized window (a) Diamond pattern (b) Square pattern

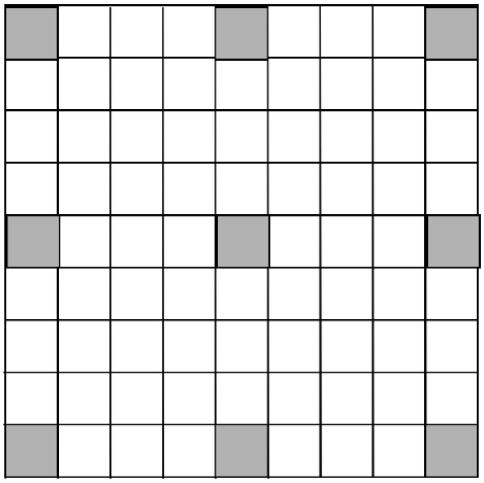


Figure 23 Sampling of search area during raster search with I_{raster}=4.

4.2 Operational parameters in the test zone search algorithm

The pace of the test zone search is influenced by the size of the search area since the total number of search points explored during the initial zonal search is directly proportional to the size of the search area.

$$N_{ZonalSearchpoints} = \{(\log_2(size\ of\ search\ area) - 1) \times 9 + 5\} \qquad (Eq.32)$$

$N_{ZonalSearchpoints}$ in Equation 32 denotes the number of investigated search points in the initial zonal exploration of the test zone search method. Sant'Anna et al [78] documented that the most of coding units have observed very minor motion from their primary search center. Furthermore, a survey in Table 4.1 with test sequences having various motion characteristics establishes that 50%-60% of the coding units observe no motion and 10% observe very small motion in the test sequences with complex motion. But, the value of $N_{ZonalSearchpoints}$ for the coding unit with complicated motion and the coding unit with minimal motion is the same as the search area of the test zone search is constant for every coding unit.

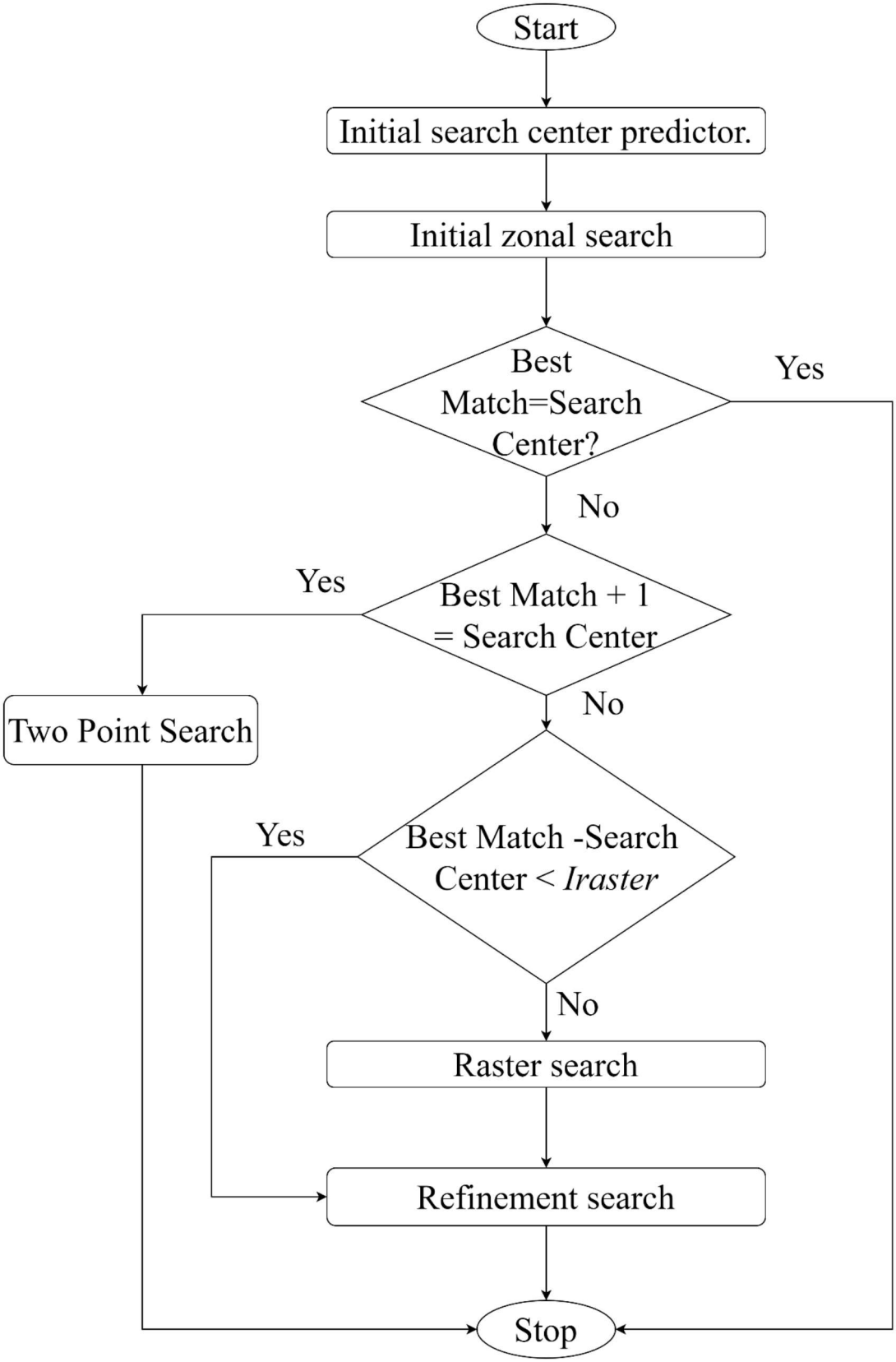


Figure 24 Flowchart of the test zone search algorithm.

The speed of the test zone search is also influenced by the sampling frequency of the search area during the raster search, I_{raster}, as the number of search points explored during the raster search is inversely

proportional to I_{raster}. A high value of I_{raster} will have inadequate sample points in the search area whereas a low value of I_{raster} will eliminate the difference in search points between a complete full search and sampled full search. Therefore, the value of I_{raster} should balance between the adequate search points in the raster search as well as the speed of the search.

Every coding unit has the same operating parameters in the test zone search algorithm. However, a coding unit with complicated motion requires a substantially different set of values for these parameters than a coding unit with a simple motion. As a result, the value of the parameters may be optimized to increase the performance of the test zone search method.

Table 4.1: Percentage of coding units with zero, low and high motion vectors in various test sequences

Video name	Video characteristics	Coding unit with no motion or zero motion (%)	Coding unit with smaller than 7-pixel movement or low motion (%)	Coding unit with greater than 7-pixel movement or high motion (%)
Akiyo	Low motion	99.95	0.03	0.02
Bowing	Low motion	100	0	0
Carphone	Moderate motion	92	7	1
Coastguard	Moderate motion	82	13	5
Football	Moderate motion	86	12	2
In to Tree	Low motion	95	5	0
Old Town Cross	Moderate motion	89	11	0
Crowd Run	High motion	60	28	12
Park Joy	High motion	54	16	30
Rush Field Cuts	Moderate motion	86	6	8

4.3 Motion Factor

The motion factor (M_f) is a crude approximation of motion inside a coding unit. To compute the motion factor, it employs the motion information matrix proposed by Tang et al [79]. The binarized difference matrix created by subtracting the reference frame from the current frame is the motion information matrix. The motion information matrix's true pixels represent motion, whereas the false pixels reflect no motion. The P-frames have one reference frame. Therefore, it has only one difference matrix and one motion information matrix. However, the B-frames have two reference matrices and two difference matrices. As a result, the motion information matrix is formed by combining two binarized difference matrices using logical AND operation.

$$DifferenceMatrix_1 = |\, ReferenceFrame_{\,1} - CurrentFrame \,| \quad (Eq.33)$$

$$DifferenceMatrix_2 = |\, ReferenceFrame_{\,2} - CurrentFrame \,| \quad (Eq.34)$$

$$BinarizedDifferenceMatrix_1 = \begin{matrix} 1 \text{ when } & DifferenceMatrix_1 > Th \\ 0 \text{ when } & DifferenceMatrix_1 < Th \end{matrix} \quad (Eq.35)$$

$$BinarizedDifferenceMatrix_2 = \begin{matrix} 1 \text{ when } & DifferenceMatrix_2 > Th \\ 0 \text{ when } & DifferenceMatrix_2 < Th \end{matrix} \quad (Eq.36)$$

Motion Information Matrix

$$= BinarizedDifferenceMatrix_1 \cap BinarizedDifferenceMatrix_2 \quad (Eq.37)$$

The motion factor is the ratio of the number of true pixels contained within a coding unit of the motion information matrix to the total number of pixels contained within the coding unit. The motion factor is highly correlated with the motion vector. In Figure 25, a comparison of the motion factor calculated by Equations 33-38 and the motion vector generated by the full search process for the various test sequences listed in Table 4.1 demonstrates a strong positive relationship between these two variables. Therefore, a coding unit with a high motion factor has the highest motion. This coding unit requires a big search window for better motion estimation. Furthermore, the value of I_{raster} should be small for such coding units as a small I_{raster} will sample adequate search points in the search area during the raster search. The coding unit with a small motion factor has minor motion. Hence, coding units with such motion need a small search window and a large value for I_{raster}. For the majority of the coding units, the motion vector for such coding was centered on the primary search center. As a result, a raster search will be unnecessary in the majority of such scenarios.

$$M_f = \frac{number\ of\ true\ pixles\ inside\ the\ coding\ unit\ of\ motion\ information\ matrix}{number\ of\ total\ pixels\ inside\ the\ coding\ unit} \quad (Eq.38)$$

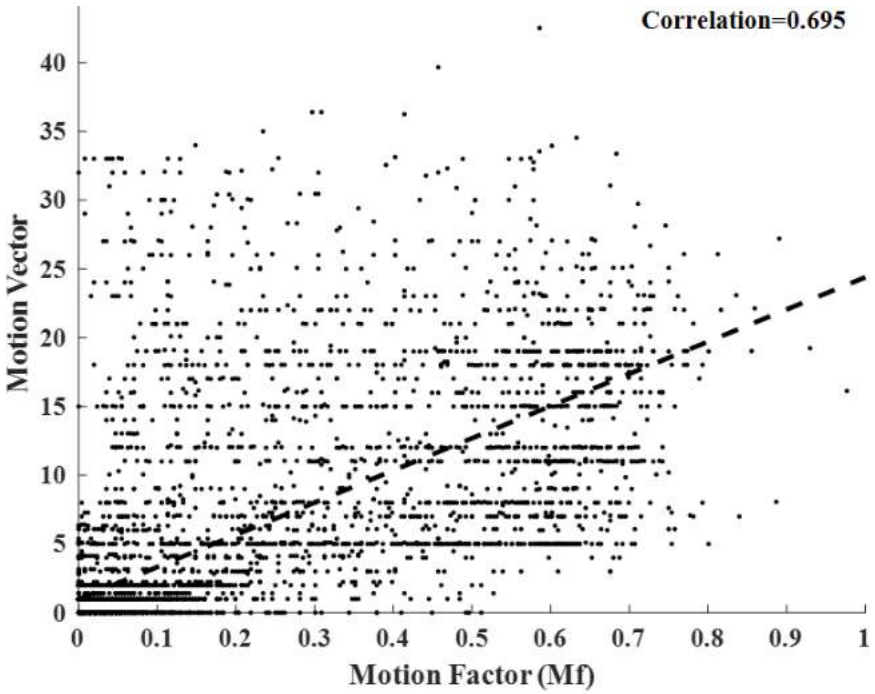


Figure 25 Relationship between motion factor (M_f) and motion vector.

4.4 Cuckoo Search

Inspired by the parasitic breeding behavior of cuckoos, Xin-She Yang developed the Cuckoo search algorithm to solve the optimization problem in numerous engineering problems. Cuckoo never create their nests and depend on other birds to nurture the cuckoo offspring. The mother cuckoo laid eggs in the nest of another bird. The cuckoo egg bears a resemblance to the egg of the host bird. Therefore, the host bird accepts the egg of the cuckoo and raises the offspring of the cuckoo. These observations are mimicked using three rules in the cuckoo search algorithm.

(a) One cuckoo will lay only one egg in a randomly chosen nest of a host bird.

(b) All host birds will not be able to hatch every egg laid by the cuckoo. Therefore, some eggs will be destroyed and replaced by new ones in future generations.

(c) The host bird occasionally discovers the egg of the cuckoo and leaves the nest completely. Such nests are replaced by new nests accompanied by the probability of discovery in the algorithm representing the host bird's odds of discovering the cuckoo egg. cuckoo egg. The chances of discovery of the cuckoo egg by the host bird are represented by the probability of discovery in the algorithm.

Based on these three rules, the steps of the cuckoo search algorithm are described below.

Step 1, The n number of initial solutions are randomly placed of the search space.

Step 2, The algorithm evaluates the fitness of the n solutions and arranges the solution based on fitness.

Step 3, One new solution is generated using Levy distribution as Levy distribution explores the search space better than the uniform distribution [80].

Step 4, Select any solution randomly from the list of solutions. Compare the fitness of the selected solution with the recently generated solution.

Step 5, When the new solution is more fit, the selected solution from the list will be replaced.

Step 6, Compare the probability of discovery to a random number between 0 and 1. When the randomly generated number is greater, the old unfit solutions are replaced with new ones in the following generation.

Step 7, Examine the termination criteria. Continue the algorithm from step 2 if the algorithm has not met the termination conditions.

4.5 Optimization of operational parameter using cuckoo search

Step 1, Generate n number of solutions randomly. Every solution consists of a pair of randomly generated values of search area size and I_{raster}, the sampling frequency of the search area during raster search for all coding units. The size of the search area will have any value between 16 and 64. The I_{raster} will vary between 3 and 16.

The versatile video coding divides a frame into multiple coding units. Figure 27 depicts an example of a partitioned frame. Equation 39 is an example of a k^{th} solution for the frame shown in figure 27. The k^{th} solution is made up of a search area size and I_{raster} value for all the coding units in the

example. In equation 39, SW^k_X and $I^k_{rasterX}$ denote the size of the search area and the sampling frequency of the search area during the raster search of the X^{th} coding unit.

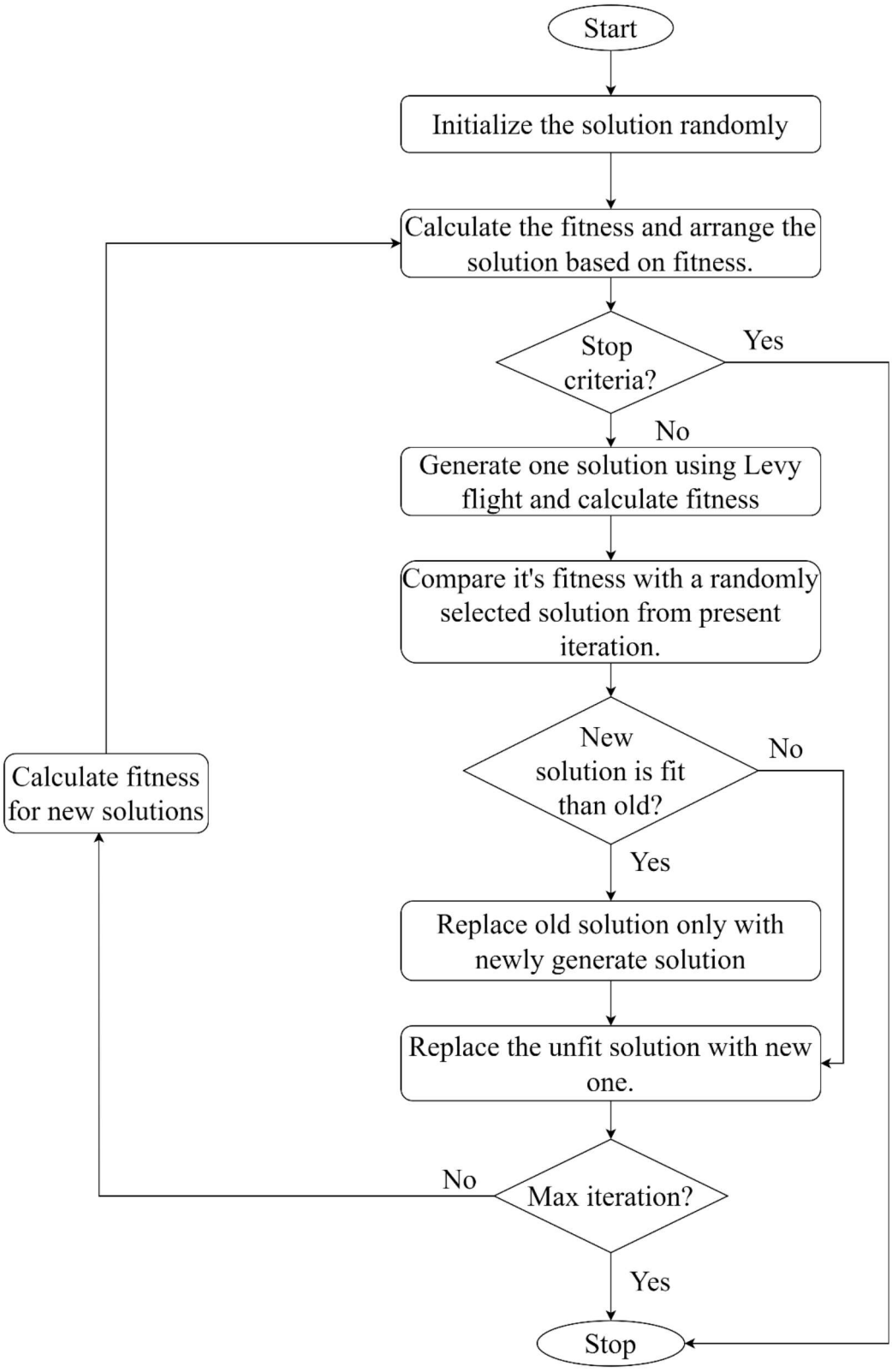


Figure 26 The flowchart for cuckoo search algorithm

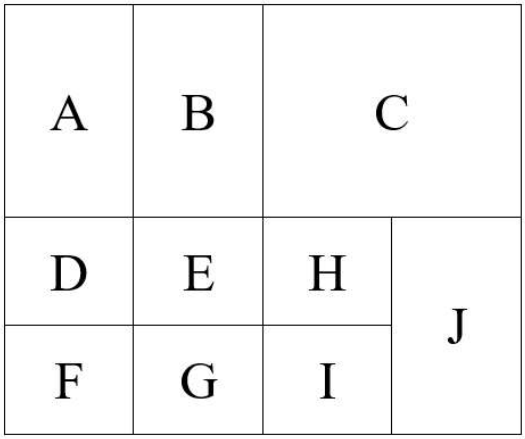


Figure 27 Example of a partitioned frame.

$$Solution_k = \left\{ \begin{matrix} SW_A^k,\ I_{rasterA}^k, SW_B^k,\ I_{rasterB}^k, SW_C^k,\ I_{rasterC}^k, SW_D^k, I_{rasterD}^k, SW_E^k, I_{rasterE}^k, \\ SW_F^k, I_{rasterF}^k, SW_G^k, I_{rasterG}^k, SW_H^k, I_{rasterH}^k,\ SW_I^k, I_{rasterI}^k, SW_J^k, I_{rasterJ}^k \end{matrix} \right\} \quad (Eq.39)$$

Step 2, The motion vector and the motion factor are highly correlated. Therefore, this relation is exploited to reduce the computation of fitness. The motion factor has a value between 0 and 1. Therefore the value of I_{raster} and size of the search area are also normalized between 0 and 1.

$$NormalizedSearchAreaSize_{k,X} = \frac{Size\ of\ Search\ Area}{Upper\ limit\ of\ Search\ Area\ Size} \quad (Eq.40)$$

$$NormalizedIraster_{k,X} = \frac{I_{raster}}{Upper\ limit\ of\ I_{raster}} \quad (Eq.41)$$

$NormalizedSearchAreaSize_{k,X}$ and $NormalizedIraster_{k,X}$ represents the normalized search area size and normalized I_{raster} of the X^{th} coding unit at k^{th} iteration. The motion factor has an inversely proportional relation with I_{raster} and a directly proportional relation with the size of the search area. Therefore, a proportion constant multiplied by the ratio between the normalized search area size and normalized I_{raster} must be nearer to the motion factor. Hence, the difference between the ratio and the motion factor will be near zero when the motion factor is high, the search area is large and I_{raster} is low or the motion factor is low, the search area is small and I_{raster} is large. The cuckoo search minimizes the differences between the ratio and the motion factor of all the coding units to get the optimal values of search area size and I_{raster} for all coding units.

$$Ratio_k^X = \frac{NormalizedSearchAreaSize_{k,X}}{NormalizedIraster_{k,X}} \quad (Eq.42)$$

$$difference_k^X = \left| M_{f_X} - \left(C * Ratio_k^X\right) \right| \quad (Eq.43)$$

$$Error_k = \sum_{X=1}^{Last\ Coding\ Unit} difference_k^X \quad (Eq.44)$$

The M_{f_X} is the motion factor of the X^{th} coding unit. C is the proportion constant and has a value of 1. The $Error_k$ sums up all the absolute differences between the ratio and motion factor for all coding units. The value of $Error_k$ is minimized to achieve optimization.

Step 3, The solutions are arranged based on the fitness evaluated using Equations 41 to 44.

Step 4, A solution is randomly generated using levy flight. The fitness of the newly generated solution was evaluated using Equations 41 to 44. A randomly selected solution from the solution list of the present iteration is compared with the new solution based on fitness. The new solution will replace the randomly selected solution from the solution list if the new solution has better fitness.

Step 5, The worst solutions are replaced by randomly generated new solutions. The probability of discovery decides the number of solutions to be replaced.

Step 6, Look for the termination condition of the algorithm. The algorithm terminates when the termination criteria are fulfilled. Else, the process continues from the calculation of fitness.

Step 7, After the termination of the cuckoo search, the solution at the top of the last generation is the optimal solution. This solution has the optimal values of the search area and I_{raster} for every coding unit.

4.6 Genetic Algorithm

Inspired by natural selection, researchers proposed a genetic algorithm to solve optimization problems. Genetic algorithm uses genetic operators such as mutation and crossover to reach the optimal solution. The genetic algorithm is performed based on the below steps.

Step 1, Initialize the solution randomly. The random solution will uniformly distribute the solution to all over the search space.

Step 2, Calculate the fitness of the solution and arrange the solution based on the fitness.

Step 3, The fit solution will have a higher possibility of survival. Therefore, top-fit solutions are retained for the future generations. Others are replaced by new offspring in the next generation which is generated using the crossover operation. The tournament selection [81] process will be used to select the solution from the present generation for crossover operation.

Step 4, Mutation is a sudden change in the solution. The mutation was applied to a randomly selected solution.

Step 5, The algorithm will look for the termination criteria. If the solutions are not converged, the process will repeat from the fitness evaluation.

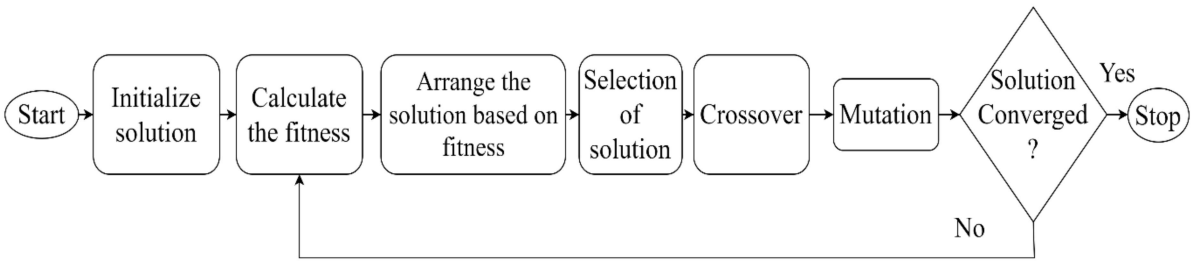


Figure 28 The flowchart of genetic algorithm

4.7 Optimization of operational parameter using genetic algorithm

Step 1, Generate n number of solutions randomly. Each solution contains a pair of search area sizes and I_{raster} for every coding unit in the frame. Equation 39 is an example of such a solution.

Step 2, Calculate the fitness using equations 41 to 44. The solution with minimum error has the highest fitness. The solutions are arranged based on their fitness. T

Step 3, The crossover probability for the problem is taken as 0.5. Half of the solutions in the present generation will survive in the next generation. The unfit solutions are replaced by generating new solutions using genetic operations. Uniform crossover is applied to generate new solutions.

Step 4, The mutation probability is 0.1. One out of ten solutions is randomly chosen from the solution list for mutation. The mutation will replace a part of the solution with a new value.

Step 5, The algorithm will terminate when the output is converged or the maximum iteration has been reached. Otherwise, the process repeats from step 2.

Step 6, The best solution after the termination of the algorithm contains the optimized value of the search area and I_{raster} for every coding unit in the frame.

4.8 Fuzzy set, fuzzy rule and fuzzy inference

The uncertainty in real-world challenges cannot have a proper representation in the binary crisp logic set. Zadeh et al [82] proposed the Fuzzy set to represent the uncertainty in a more accurate way. In this set, every object in a universe is a member of every set in that universe. The association between the object and the universe is depicted by the continuous grad membership value. In Equation 45, $\mu_F(A)$ is the membership value of an object A in a fuzzy set F. $\mu_F(A)$ has a value between 0 and 1.

$$F = A, \mu_F(A) \quad where\ A \in U \qquad (Eq.45)$$

U in Equation 45 represents a universal set. Fuzzy inference is the mapping between two fuzzy sets. The relations between two fuzzy sets are described using *if-then* rules. These rules have the advantage of describing complex mathematical relation in simple language. A binary crisp set infers the subsequent portion as true only when the antecedent component is completely true. The partial truth of antecedence can be inferred as a partial truth of consequence via fuzzy inference. Out of many fuzzy inference methodologies, the Mamdani inference [83] system is used in this process and the inference may be calculated using equation 46.

$$Rule\ R_i : If\ k\ is\ P\ then\ m\ is\ O$$
$$R_i(k;m) = \min\big(\mu_P(k), \mu_O(m)\big) \quad mamdani\ inference \qquad (Eq.46)$$

$R_i(k;m)$ is the relational matrix obtained by inferring the rule R_i. Multiple relation matrix $R_i(k;m)$ are combined using union operation to form a final relational matrix $R(k;m)$.

$$R(k;m) = max_{i=1}^{n} R_i(k;m) \qquad (Eq.47)$$

The membership distribution of m is O' may now be determined using the final relational matrix and the observed membership distribution for k is P'.

$$\mu_{O'}(m) = \max\big(\min\big(\mu_{P'}(k), R(k;m)\big)\big) \quad (Eq.48)$$

4.9 Fuzzy inference-based decision-making module for adaptive operational parameter

The fuzzy inference-based decision-making module is a pre-process module that computes the value of operational parameters of test zone search using fuzzy rules and Mamdani inference of the rules. The frameworks consist of a rule base, fuzzification, Mamdani inference engine, and defuzzification blocks.

The rule base comprises the relationship between the framework's input parameter, the motion factor, and the framework's output parameter, the size of the search area, and I_{raster}. Fuzzification converts the normal crisp set into a fuzzy set and it was performed on every input and output variable using conventional fuzzy membership functions. Defuzzification converts the output of a fuzzy set into a crisp set and it was performed using the center of gravity method [84].

4.9.1 Fuzzy rule base

The rule base, which contains the relation between input and output parameters, is the foundation of the fuzzy decision-making framework. However, the relations between the parameters are not a mathematically deterministic one. Hence, the relationship was defined using *if-then* rules by observing the relation between the motion factor and size of the search area as well as the value of I_{raster}. A small study between the motion factor and the mean square error from the test zone search motion estimation process with various sizes of the search area is conducted. The value of I_{raster} is constant throughout the study. The research offers three findings.

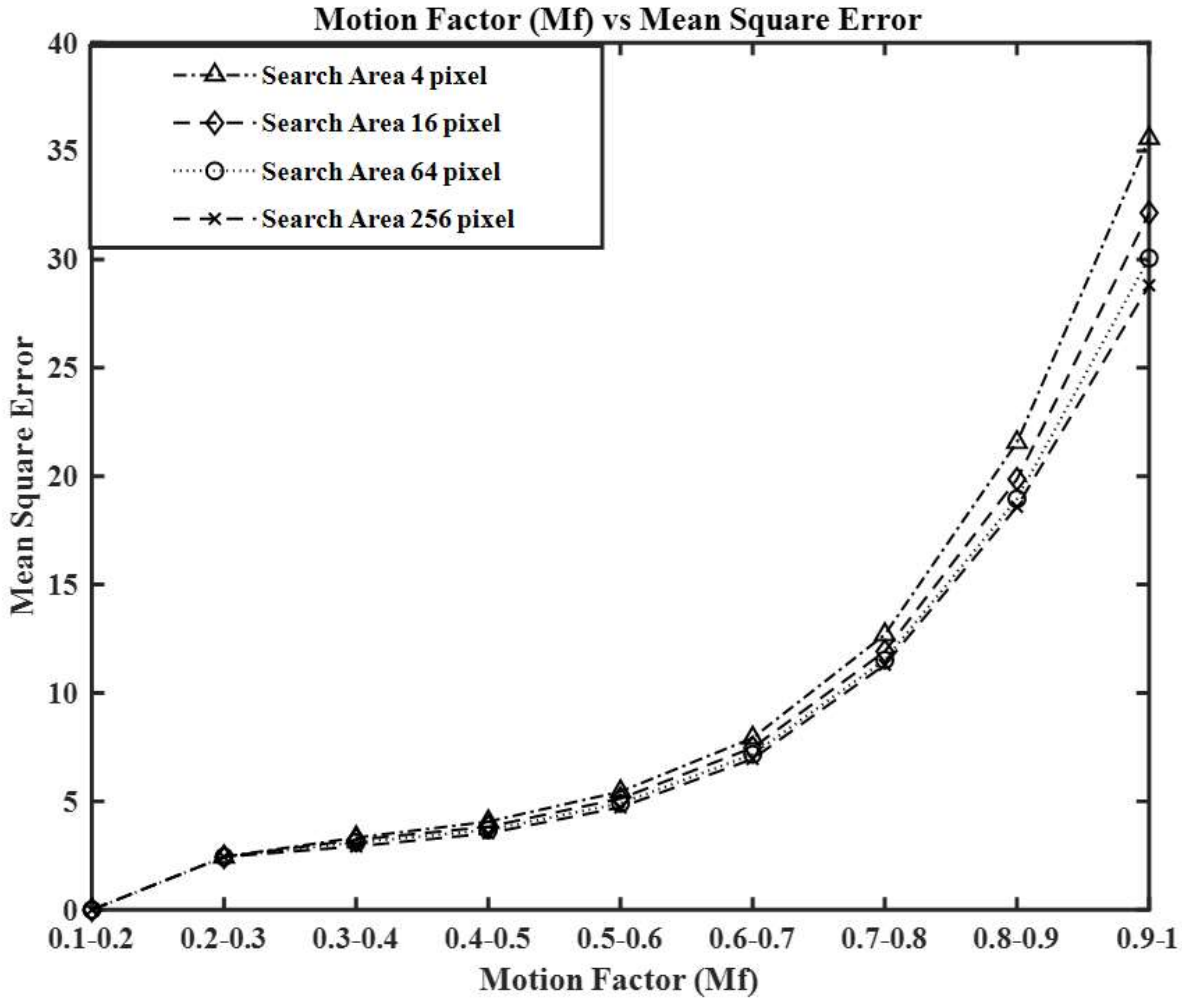


Figure 29 The Motion factor vs the mean square error from the test zone search motion estimation process with various size of search area

(a) Coding units with small motion (M_f <0.25) can have a smaller search area (size is less than 16 pixels). The smaller search areas will have simple operational complexity.

(b) Coding units with moderate motion ($0.25<M_f \leq 0.6$) will have a medium search area (size is greater than 16 pixels but smaller than 64 pixels).

(c) Coding units with high motion ($M_f>0.6$) will have complex motion, therefore the large search area (size is greater than 64 pixels).

Three laws may be derived from these three findings.

$$Rule\ 1: If\ M_f\ is\ High\ then\ Size\ of\ search\ area\ is\ Large$$
$$Rule\ 2: If\ M_f\ is\ Moderate\ then\ Size\ of\ search\ area\ is\ Medium$$
$$Rule\ 3: If\ M_f is\ Low\ then\ Size\ of\ search\ area\ is\ small.$$

In the test zone search algorithm, a zonal search is sufficient to estimate the minor motions of the coding unit. Hence, the value of I_{raster} has zero effect on the motion estimation. However, the complex motion necessitates a detailed exploration of the search area. Therefore, the value of I_{raster} has an effect on the coding units with complex motion. The study between the motion factor and the difference in motion vectors derived from full search and test zone search with various values of I_{raster} is depicted in figure 30. Full search motion vectors are ideal, and every motion estimation approach is aimed to achieve those identical motion vectors. As a consequence, the error is calculated using the difference in motion vectors acquired by full search and test zone search. The value of the search area is constant throughout the study. The study offers two findings.

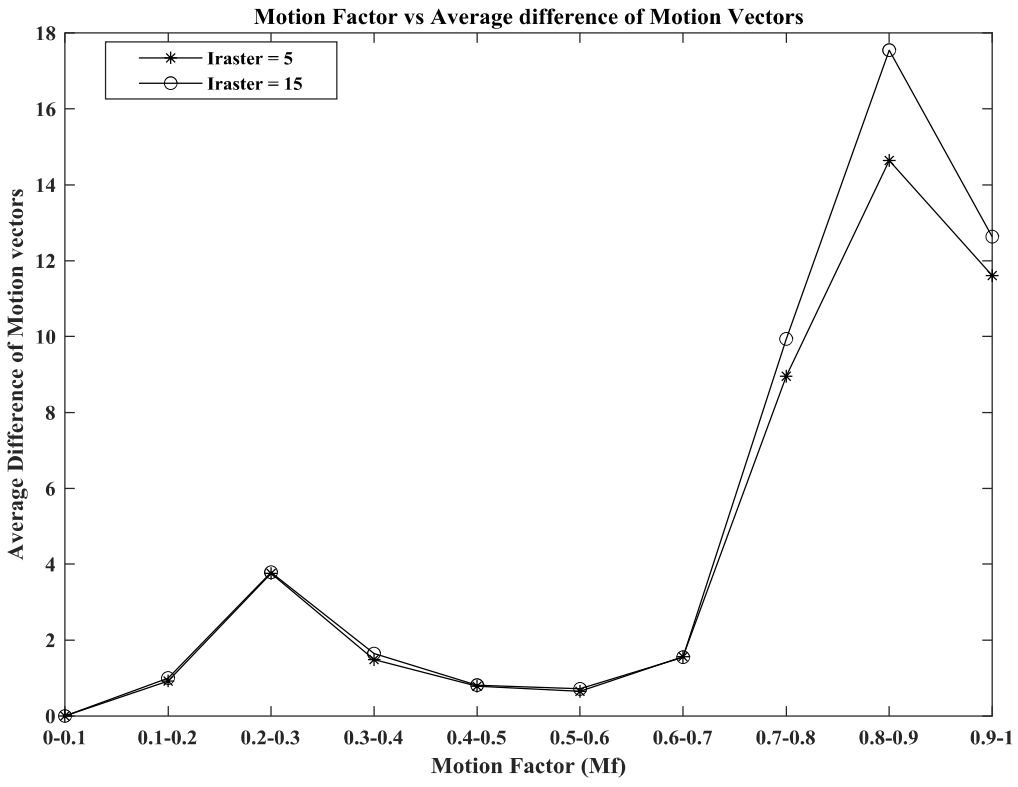


Figure 30 The study between the motion factor and the difference in motion vectors derived from full search and test zone search with various value of I_{raster}

(a) The value of I_{raster} has no effect on the output motion vector when the motion factor is less than 0.65 ($M_f<0.65$). Therefore, the coding units with low and moderate motion can have a higher value for I_{raster}.

(b) The complex motion ($M_f>0.65$) requires a smaller value of I_{raster}.

Two laws can be derived from the findings.

$$Rule\ 1: If\ M_f\ is\ Low\ or\ Moderate\ then\ I_{raster} is\ High$$
$$Rule\ 2: If\ M_f\ is\ High\ then\ I_{raster} is\ Low$$

4.9.2 Fuzzification

Fuzzification is the process to convert the normal crisp set into the fuzzy set. The motion factor (M_f) quantifies the motion profile of a coding unit. The relationship illustrated in Figures 29 and 30 classified the coding units into three motion profile sets: low, moderate, and high. The motion profile of a coding unit with a motion factor less than 0.25 is low. The motion profile of the coding unit with a motion factor between 0.25 and 0.7 can be inferred as medium motion. Other coding units with a motion factor greater than 0.7 have a high motion profile. Equations 49, 50, and 51 are used to compute the membership value, and figure 31 illustrates the membership distribution function.

$$\mu_{low}(M_f) = \begin{cases} = 1 & for\ M_f \leq 0.25 \\ = \dfrac{m_f - 0.35}{(-0.1)} & for\ 0.25 < M_f \leq 0.35 \\ = 0 & for\ M_f > 0.35 \end{cases} \quad (Eq.49)$$

$$\mu_{moderate}(M_f) \begin{cases} = 0 & for\ M_f \leq 0.25 \\ = \dfrac{M_f - 0.25}{(0.1)} & for\ 0.25 < M_f \leq 0.35 \\ = 1 & for\ 0.35 < M_f < 0.65 \\ = \dfrac{M_f - 0.65}{(-0.1)} & for\ 0.65 \leq M_f < 0.75 \\ = 0 & for\ M_f > 0.75 \end{cases} \quad (Eq.50)$$

$$\mu_{high}(M_f) = \begin{cases} = 0 & for\ M_f < 0.65 \\ = \dfrac{M_f - 0.65}{(0.1)} & for\ 0.65 < M_f \leq 0.75 \\ = 1 & for\ M_f > 0.75 \end{cases} \quad (Eq.51)$$

From figure 29, the size of the search area is classified into three different groups: {small, medium, and large}. A Small search area has a size smaller than 16 pixels. Medium search areas are those that are greater than 16 pixels but smaller than 64 pixels. The large search area has a search area that is greater than 64 pixels. Equations 52, 53, and 54 are derived from this observation to fuzzify the search area size.

$$\mu_{small}(SearchArea) = \begin{cases} = 1 & for\ SearchArea \leq 8 \\ = \dfrac{SearchArea - 16}{(-8)} & for\ SearchArea \leq 16 \\ = 0 & for\ SearchArea > 16 \end{cases} \quad (Eq.\ 52)$$

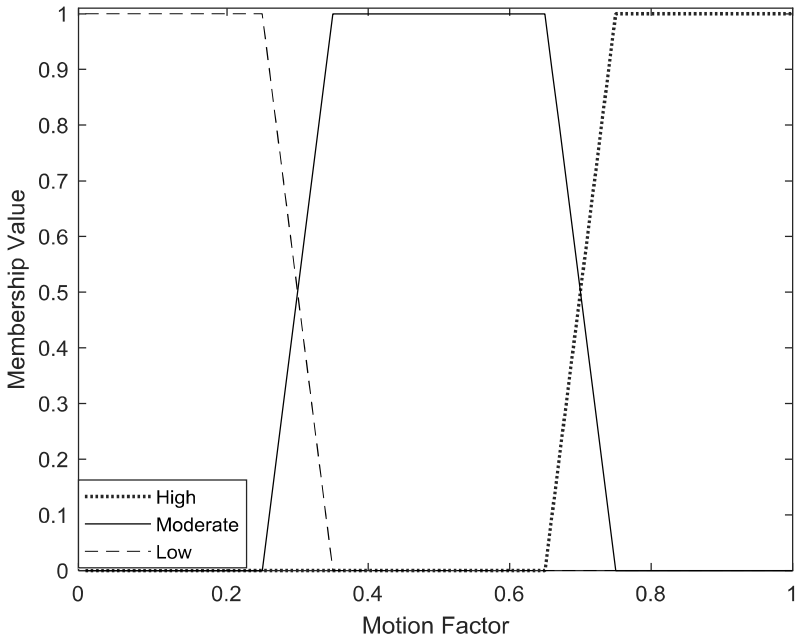


Figure 31 Membership distribution for fuzzy set with {high, moderate, and low} motion

$$\mu_{medium}(SearchArea) \begin{cases} = 0 & for\ SearchArea \leq 8 \\ = \dfrac{SearchArea - 8}{(8)} & for\ 8 < SearchArea \leq 16 \\ = 1 & for\ 16 < SearchArea \leq 32 \\ = \dfrac{SearchArea - 64}{(-32)} & for\ 32 < SearchArea \leq 64 \\ = 0 & SearchArea > 64 \end{cases} \quad (Eq.53)$$

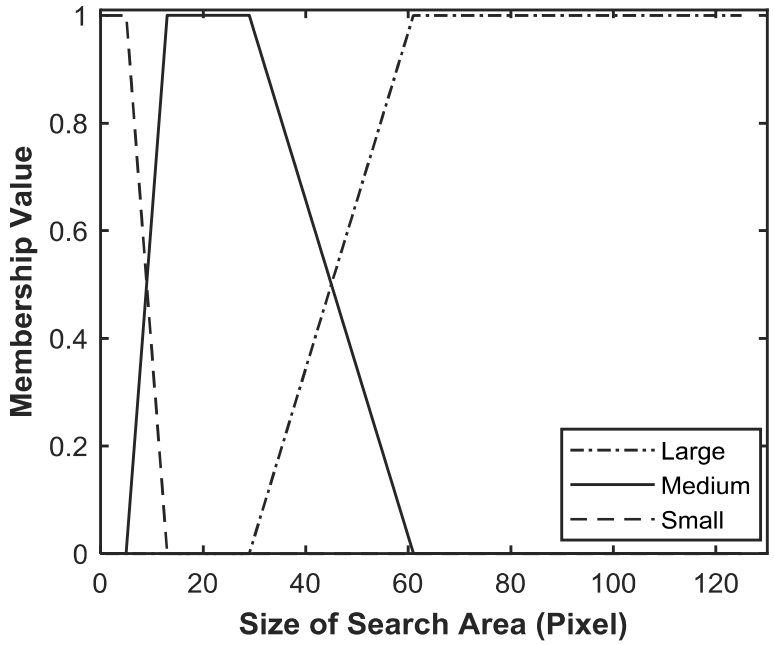


Figure 32 Membership distribution for fuzzy set with {large, medium, small} search area

$$\mu_{large}(SearchArea) \begin{cases} = 0 & for\ SearchArea < 32 \\ = \dfrac{SearchArea - 32}{(32)} & for\ 32 \leq SearchArea < 64 \\ = 1 & for\ SearchArea > 64 \end{cases} \quad (Eq.54)$$

The relation between M_f and I_{raster} identifies two groups for I_{raster} {large, small}. The membership value for the two sets of I_{raster} can be calculated based on Equations 55 and 56.

$$\mu_{small}(I_{raster}) = \begin{cases} = 1 & for\ I_{raster} \leq 7 \\ = \dfrac{I_{raster} - 11}{(-4)} & for\ 7 < I_{raster} \leq 11 \\ = 0 & for\ I_{raster} > 11 \end{cases} \quad (Eq.55)$$

$$\mu_{large}(I_{raster}) = \begin{cases} = 0 & for\ I_{raster} \leq 7 \\ = \dfrac{I_{raster} - 7}{(4)} & for\ 7 < I_{raster} \leq 11 \\ = 1 & for\ I_{raster} > 11 \end{cases} \quad (Eq.56)$$

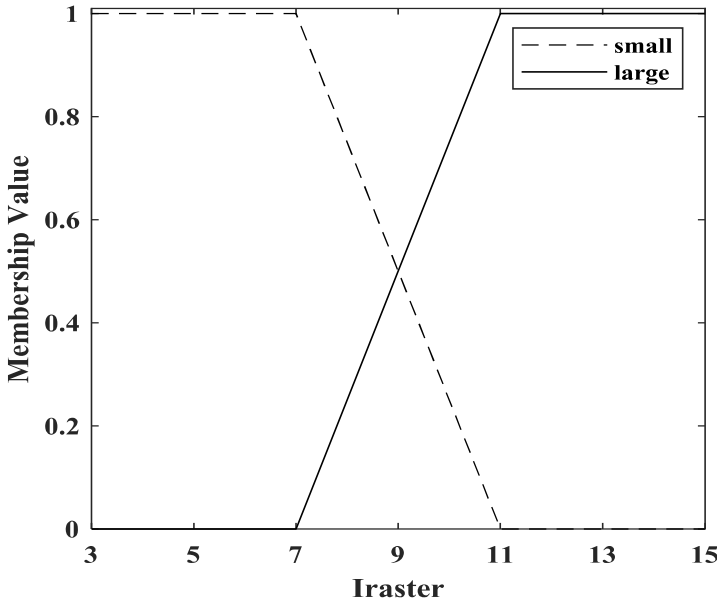


Figure 33 Membership distribution for fuzzy set with {large, small} I_{raster}.

4.9.3 Mamdani inference engine

Mamdani inference requires the fuzzy set as input and output parameters. Therefore, the fuzzification is immediately followed by the Mamdani inference engine. Using Equations 46 and 47, this inference process generates the relational matrix which records the intensity of every rule in the rule base for every conceivable input and output combination. Then using equation 48, a union operation was performed to combine all the relational matrices into a final relational matrix.

$$R_{low-small}(M_f, SearchArea) = \min\left(\mu_{low}(M_f), \mu_{small}(SearchArea)\right) \quad (Eq.57)$$

$$R_{moderate-medium}(M_f, SearchArea) = \min\left(\mu_{moderate}(M_f), \mu_{medium}(SearchArea)\right) \quad (Eq.58)$$

$$R_{large-high}(M_f, SearchArea) = \min\left(\mu_{large}(M_f), \mu_{high}(SearchArea)\right) \quad (Eq.59)$$

$$R(M_f, SearchArea) = \max(R_{low-small}(M_f, SearchArea), R_{moderate-medium}(M_f, SearchArea), R_{large-high}(M_f, SearchArea)) \quad (Eq.60)$$

$$R_{low-mod-large}(M_f, I_{raster}) = \min\left(\max\left(\mu_{low}(M_f), \mu_{moderate}(M_f)\right), \mu_{large}(I_{raster})\right) \quad (Eq.61)$$

$$R_{high-small}(M_f, I_{raster}) = \min\left(\mu_{high}(M_f), \mu_{small}(I_{raster})\right) \quad (Eq.62)$$

$$R(M_f, I_{raster}) = \max\left(R_{low-mod-large}(M_f, I_{raster}), R_{high-small}(M_f, I_{raster})\right) \quad (Eq.63)$$

4.9.4 Defuzzification

Output from the Mamdani inference engine was a fuzzy set. However, real-world problems require a crisp value. As a result, the defuzzification procedure converts the output fuzzy sets to crisp values. The center of gravity method is the most popular process for defuzzification. This calculates the centroid of the fuzzy set and divided the whole fuzzy set into two equal parts.

Figure 34 Sample frame from each test sequences (a) Akiyo (b) Bowing (c) Carphone (d) Coastguard (e) football (f) Crowd Run (g) In to tree (h) Old town cross (i) Park Joy (j) Rush Field Cut

With the help of the motion factor of a new coding unit and the relational matrix in Equation 60, the Mamdani inference engine will produce a new fuzzy set for the search area size of the new coding unit. Similarly, with the help of the relational matrix in Equation 63 and the motion factor, the Mamdani inference engine will generate the fuzzy set for the I_{raster}. Both the output fuzzy set was converted to a crisp value using Equation 64 and 65.

$$COG_{SearchArea} = \frac{\sum_{All}(\mu(Search\ Area) \times Search\ Area)}{\sum_{All} \mu(Search\ Area)} \qquad (Eq.64)$$

$$COG_{I_{raster}} = \frac{\sum_{All}(\mu(I_{raster}) \times I_{raster})}{\sum_{All} \mu(I_{raster})} \qquad (Eq.65)$$

4.10 Performance evaluation

The test sequences mentioned in Table 4.1 is used to compare the output of traditional test zone search as well as the optimized test zone searches. PSNR, SSIM, and the speed of the search are used to compare the performances. Figure 34 provides the sample frame from each test sequence. Table 4.2 and Table 4.3 compare the performance of PSNR and SSIM among all the algorithms.

The size of the search area and the value of I_{raster} are 128 and 7 for classical test zone search as well as enhanced test zone search. The search area size varied between 16 and 128 and the value of I_{raster} lies between 5 and 15 for the other three processes. From Table 4.2 and 4.3, it is evident that the optimization of the operational parameters does not impact the output quality of the test zone search algorithm. However, the optimization of the parameters reduced the computational load of the test zone search. Table 4.4 summarizes the improvement in execution speed of the test zone search algorithms with optimized operational parameters relative to the conventional test zone search methods. The speed of the test zone search is improved by every optimization process. However, genetic algorithm-based optimized operational parameters is able to boost the speed of the test zone search better than others.

Table 4.2: Comparison of the PSNR performance among all algorithms.

Video name	Classical test zone search	Enhanced test zone search	Genetic algorithm based optimized parameter-test zone search	Cuckoo search algorithm based optimized parameter-test zone search	Fuzzy inference-based framework for adaptive parameter- test zone search
Akiyo	30.77	30.71	30.77	30.77	30.77
Bowing	30.08	29.87	30.23	30.05	30.35
Carphone	24.09	24.06	24.39	24.40	24.50
Coastguard	21.23	21.01	21.32	21.27	21.28
Football	17.31	17.21	16.95	17.04	17.24
CrowdRun	19.47	18.60	19.29	19.63	19.52
Old Town Cross	23.44	23.19	23.36	23.43	23.45
In To Tree	24.31	23.90	24.17	24.29	24.30
Park Joy	18.64	17.61	18.43	17.35	18.59
Rush Field Cut	22.97	22.77	22.91	22.97	22.98
Average	23.23	22.89	23.18	23.12	23.30

Table 4.3: Comparison of the SSIM performance among all algorithms.

Video name	Classical test zone search	Enhanced test zone search	Genetic algorithm based optimized parameter-test zone search	Cuckoo search algorithm based optimized parameter-test zone search	Fuzzy inference-based framework for optimized parameter- Test zone search
Akiyo	0.75	0.74	0.75	0.75	0.75
Bowing	0.73	0.72	0.73	0.73	0.73
Carphone	0.72	0.72	0.72	0.72	0.72
Coastguard	0.65	0.65	0.65	0.65	0.65
Football	0.52	0.51	0.51	0.51	0.52
CrowdRun	0.61	0.58	0.60	0.61	0.61
Old Town Cross	0.60	0.60	0.60	0.60	0.60
In To Tree	0.59	0.58	0.59	0.59	0.59
Park Joy	0.61	0.57	0.61	0.59	0.61
Rush Field Cut	0.67	0.67	0.67	0.67	0.67
Average	0.64	0.63	0.64	0.64	0.64

Table 4.4: Comparison of the SIR between traditional test zone search and other algorithms.

Video name	Enhanced test zone search	Genetic algorithm based optimized parameter-test zone search	Cuckoo search algorithm based optimized parameter-test zone search	Fuzzy inference-based framework for optimized parameter- Test zone search
Akiyo	0.28%	6.22%	6.05%	29.90%
Bowing	3.62%	31.98%	25.36%	24.20%
Carphone	8.02%	13.72%	18.53%	41.64%
Coastguard	3.23%	-2.96%	11.63%	1.85%
Football	6.87%	54.78%	44.77%	37.61%
CrowdRun	1.55%	76.85%	24.84%	35.54%
Old Town Cross	11.72%	79.26%	50.21%	65.08%
In To Tree	0.20%	50.49%	2.85%	46.62%
Park Joy	0.66%	1.52%	81.13%	22.41%
Rush Field Cut	6.03%	64.07%	2.30%	40.38%
Average	4.22%	37.59%	26.77%	34.52%

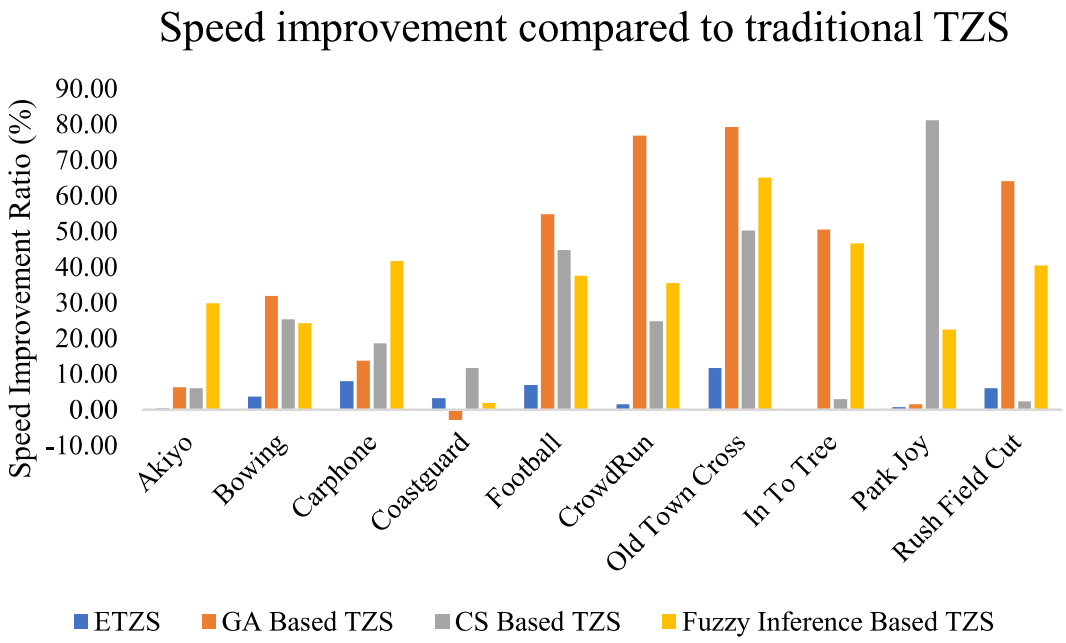


Figure 35 The speed improvement ratio between traditional test zone search and other algorithms.

Chapter 5:

Optimized block matching algorithms

The optimization of the operational parameters significantly improved the speed of the test zone search algorithm. However, the block-matching algorithm itself is an optimization problem where the goal is to minimize the SAD between the template coding unit in the current frame and the coding unit within the search area of the reference frame. The genetic motion search, PSO-based block matching process, and other evolutionary algorithm-based block matching processes specifically address the minimization problem in block matching algorithm. But the speed of such algorithms is slow due to the repeated calculation of SAD. Also, the high use of random numbers makes the algorithm slower. Cuevas et al developed several block-matching procedures based on different evolutionary algorithms. These algorithms used the evolutionary control approach [85] where the computationally expensive fitness evaluation is replaced by fitness approximation to reduce the computation. The local fitness approximation model has several advantages over the global approximation model such as the local approximations are fast, and reliable and it considers the closest neighbor during approximation. Cuevas et al. proposed using nearest-neighbor interpolation to approximate the fitness of the solutions. The state-of-the-art Jaya algorithm-based block matching process used fitness approximation, zero motion prejudgment, and early termination criteria to reduce the computation cost. This algorithm selects the initial population from a predefined position for quick convergence and uses Equation 26 to change the position of the solutions. However, Equation 26 has two random numbers which reduce the speed of the Jaya algorithm-based motion estimation process

5.1 Modified cuckoo search algorithm

The traditional cuckoo search algorithm was employed by Bhattacharjee et al. [86] for the block matching process. In traditional cuckoo search, the solutions are using Levy flight [80] to move towards a better position. Furthermore, the worst solutions in the current iteration are replaced by new solutions generated randomly in the next iteration. All of these reduced the speed of the traditional cuckoo search-based block-matching process. Traditional cuckoo search algorithm also has the algorithm-specific parameter, probability of discovery. The value of the probability of discovery has a great impact on the problem output. An incorrect value of this parameter can lead to false optima.

The modified cuckoo search algorithm replaces the algorithm-specific parameter with a deterministic parameter, the evolution factor. The modification in the cuckoo search algorithm was inspired by the breeding behavior of certain species of cuckoo. Generally, cuckoo lays an egg on the nest of other bird and depends on the host bird to raise their offspring. Therefore, the egg of the cuckoo and the host bird should look similar. The host bird quickly develops certain changes to reduce the parasitic incident. Some species of cuckoo specialize in a single host bird and continue an evolutional race with the host bird [87]. Hence, the egg of the cuckoo and the egg of the host bird looks completely identical with time. The host bird has zero probability to recognize the egg of the cuckoo. These observations are simulated in a modified cuckoo search algorithm with four rules.

(a) The number of cuckoos and host nests are equal.

(b) Instead of laying eggs in any available nest, each cuckoo will try to lay eggs in only one specific nest in every iteration. The fittest egg between the host bird and the cuckoo will survive the iteration.

(c) As one individual cuckoo specializes only in one nest, the resemblance between the egg of the host bird and the cuckoo is magical. It is impossible for the host bird to recognize and destroy the egg of the cuckoo.

(d) An evolutionary race continues between the cuckoo and the host bird. This race is represented by the evolution factor α in the modified cuckoo search algorithm. The α can be calculated by dividing the fitness of a solution with the worst fitness of that iteration. The solution with the best fitness has a small evolution factor and the worst fit solution will have the highest evolution.

$$\alpha_i = \frac{fitness_i}{fitness_{worst}} \qquad (Eq.66)$$

Based on these four rules, the modified cuckoo search algorithm can be described as follows.

Step 1, Initialize the first *n* number of solutions randomly for uniform distribution.

Step 2, Calculate the fitness of the solution.

Step 3, Arrange the solutions based on their fitness.

Step 4, Update the position of the solutions for next iteration *t+1*. The solution S_i^t in present iteration *t* will be pushed away from the worst solution S^t_{worst} of the iteration *t* by a factor of α_i. Also, a random element using levy flight was added to the final equation to avoid a local optima trap.

$$S_i^{t+1} = S_i^{\,t} - \alpha_i(S_i^{\,t} - S^t{}_{worst}) + Levy\ Flight \qquad (Eq.67)$$

Step 5, Compare the fitness of S_i^{t+1} with S_i^t. If the fitness is better than the new solution will replace the old solution in next iteration. The same process from step 4 and step 5 will continue for solution 1 to *n*.

Step 6, Check for the termination criteria. If the conditions are not satisfied, then continue the process from step 2.

```
begin
  Generate n number of initial solution using random/biased distribution.
  Calculate the fitness of the solutions.
  Arrange the solution based on their fitness.
  while (iteration< maxIteration or stopping criteria)
        Get n number of cuckoos (V_i^{t+1}) using equation 6.
        Evaluate the fitness of n cuckoos
        for i=1: n
            Compare i_th cuckoo with i_th nest
            if (V_i^{t+1} < V_i^t)
                  Replace V_i^t with V_i^{t+1}.
              else
                  Discard V_i^{t+1}
            end if
        end for
        Rank the solution and find the current best and worst.
        Increase iteration by one
   end while
  Publish the last best solution
end
```

Algorithm 1: The pseudocode for modified cuckoo search algorithm.

Table 5.1: Differences between cuckoo search and modified cuckoo search algorithm.

	Cuckoo Search	Modified Cuckoo Search
New solution generation	In cuckoo search, the new solution generation is a levy flight around the solution.	The present solution will be pushed away from the worst solution of the present iteration. Also, a random factor is added using levy flight.
Evolution factor	No concept of scaling factor. However, one scaling factor was added up with the output from Levy Flight. For most of the applications, the scaling factor is 1.	Represent the evolutionary race between cuckoo and the host bird. It can be calculated using the ratio between the fitness of present solution and the fitness of the worst solution.
Algorithm specific parameter	Probability of discovery.	No such parameter.

5.2 Block matching algorithm using modified cuckoo search

The modified cuckoo search algorithm-based block matching process employs several strategies to reduce the computation cost. Zero motion prejudgment, deterministic initial search patter, fitness approximation, history preservation and adaptive termination strategy of the algorithm are few of them.

Step 1, The zero-motion prejudgment was proposed by Nie et al in adaptive rood pattern search. It was estimated by Nie et al that the template coding unit has no motion when the SAD between the template coding unit in current frame and the coding unit at same position in reference frame is below or equal to 512. Therefore, the motion vector for such coding unit is zero and requires no calculation. In first step of block matching algorithm using cuckoo search, zero motion prejudgment differentiate between static and dynamic coding units.

Step 2, The random initialization of solution is performed to distribute the solution uniformly across the search space. But, incorporating the domain knowledge in initial solution distribution help to converge the solution faster [88,89]. Therefore, the initial solutions for block matching algorithms are generated from a deterministic pattern. The pattern can have the square, diamond or hexagonal geometrical shape around the search center as shown in figure 36.

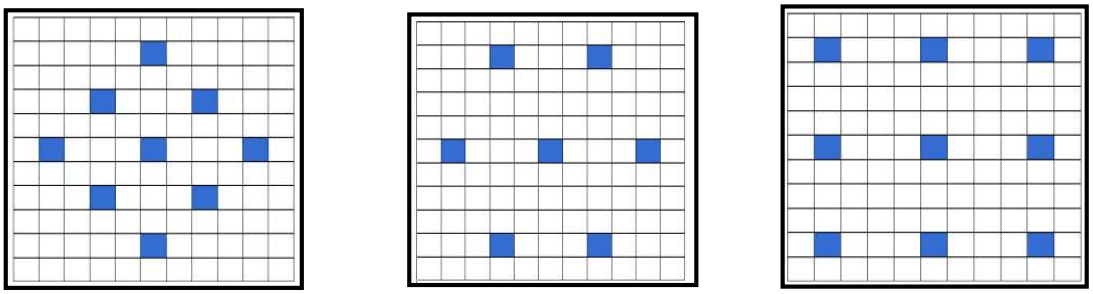
Step 3, The fitness calculation is performed using NNI. The fitness of initial solutions is calculated normally. But the solutions from next iteration follow the rules of NNI.

Step 4, The generation of new solution should follow Equation 67. However, the motion vector has vertical and horizontal components and two random number are going to be used by Equation 67 to calculate the new solutions. But the algorithm used a randomly selected solution from the preserved search history of previous generations to replace the random levy flight operation.

Therefore, equation 67 are updated to equation 68 which minimize the computation of random numbers. Here, only one random number requires to select one solution from previous search history. The value of the random number lies between 1 and size of preserved search history.

$$S_n = randomly\ selected\ from\ \{S\}\ and\ S_n \neq {S_{worst}}^t and\ S_n, S_{i,} S_{worst} \in \{S\}$$

$${S_i}^{t+1} = S_n - \alpha_i\left({S_i}^t - {S^t}_{worst}\right) \quad \text{(Eq. 68)}$$



(a) (b) (c)

Figure 36 Deterministic initial search patterns (a) Diamond (b) Hexagonal (c) Square

Step 5, The algorithm terminates when

(a) The algorithm reached maximum iteration or the solutions are already converged.

(b) The best solution in present iteration has the same position of the search center.

(c) The SAD between template coding unit and the best coding unit in present iteration is below the terminating threshold T_S. The terminating threshold is first time proposed by this algorithm and the value of terminating threshold is one third of zero motion prejudgment threshold.

When the terminating conditions are not satisfied the algorithm continue from fitness calculation.

5.3 Ant weight lifting algorithm

Ant has some interesting features. They maintain a hierarchy in their society. The queen ant is responsible for reproduction where the worker ants are responsible for building the infrastructures. These worker ants have a wonderful weight carrying capacity compared to own weight. Such surprising capability comes from the small surface area of their body. The worker ants do not grow any reproduction organ which further aids their ability to lift weight. Some species of ant are able to lift five thousand times more weight than their body weight. Researchers are studying this ability to create strong robotic arm. Three rules are idealized to mimic the behavior of ant weight lifting capability.

(a) The ant that have the worst performance will move to new location in search of food.

(b) Every ant will avoid the area where the food is not available.

(c) The ant which has reached its weight lifting capacity, will move away from the search space. A new ant will replace the ant with full carrying capacity.

Using the aforementioned rules, the ant weight lifting algorithm is established.

Step 1, The *n* number of ants are randomly placed across the search space.

$${s_i}^k = random\ position\ of\ ants$$

Here $k=1$ as this is the first iteration, ${s_i}^k$= the position of i^{th} number of ants, $i=1$ *to* n and n is size of the initial population size.

Step 2, The fitness of the solutions is evaluated using the fitness function. The collection of weight by an ant in present iteration is the normalized fitness value of the ant in present iteration. The normalization process divided the fitness of the ant with the sum of fitness of all ant in that iteration.

$$W_i^k = fitness_i / \sum_{j=1}^{n} fitness_j \qquad (Eq.69)$$

Step 3, Rank the performance of the ant based on their collection in current iteration. Identify the best ant of the iteration and worst ant of all the iterations.

Step 4, Update the position of the ant using Equation 70 and 71. The best ant will look for more better solution around their present position. Other ants will move away from the worst ant position.

$$s_{best}{}^{k+1} = s_{best}{}^{k} + r_i.d \qquad (Eq.70)$$

$$s_i{}^{k+1} = s_i{}^{k} - r_i\left(s_i{}^{k} - s_{worst}{}^{all}\right) \quad (Eq.71)$$

In Equation 70 r_i is a random variable lies in between -1 and 1. d is a small distance defined based on the problem. The positions of the ant will only change if the new position has better fitness than previous position.

Step 5, The cumulative weight collected by ant is the sum of all collected weight by that particular ant till present iteration. The cumulative weight is reset to zero when the ant changes its position.

Step 6, The ant whose cumulative weight has reached to maximum capacity will leave the search space and move to central repository. The ant will be replaced by a new ant in random position. The cumulative weight of the new ant is set to zero.

Step 7, When n number of ant in the repository is within a small enclosed circle with diameter d, then perform one iteration based local search with n ants randomly placed inside the circle. Best from this local search is the final output.

Step 8, Check for the termination criteria. The process will repeat from fitness calculation when the fitness condition is not satisfied.

5.4 Ant weight lifting based block matching algorithm

Step 1, The zero-motion prejudgment will evaluate the current coding unit. If the current coding unit has no motion, then motion vector will be set as 0 and the process moves to next coding unit. Otherwise, the algorithm continues the motion estimation process.

Step 2, Place the initial solution from a predefined pattern. The ant weight lifting algorithm use a square geometrical pattern. However, the pattern can have diamond or hexagonal geometrical shape.

Step 3, Calculate the SAD between the template and coding unit and initial solutions. Arrange the solutions based on their fitness and identify the best and worst solution.

Step 4, Check for early termination. The early termination condition is same as modified cuckoo search algorithm-based block matching process. The algorithm terminates if the condition satisfied.

Step 5, Generate new position for the ant using Equation 70 and 71. Move the ant to new position if the new position has better fitness.

Step 6, Replace the ant which have reached their maximum weight lifting ability with a new ant in new position. The old ant are placed into the central repository.

Step 7, Terminate the algorithm when the maximum iteration has reached or the solution has converged. The best solution among the present generation as well as the solution in the repository is final motion vector.

5.5 Performance evolution

The performance of these algorithms is compared with the full search algorithm and Jaya algorithm-based block matching process. The whole frame is divided into 16x16 coding units. Each coding unit has a search area of 16 pixels around it. The maximum iteration size is set as 50. The algorithms are compared in terms of PSNR, SSIM, and speed improvement ratio. The full search algorithm is used as a reference to measure the improvement in speed of other algorithms. Ten standard-definition and five high-definition test sequences are used for the performance evaluation. The test sequences have different resolutions and motion characteristics. The test sequences with such diverse characteristics will test the robustness of the algorithms.

Tables 5.2 and 5.3 lists the PSNR and SSIM performance of the algorithms. The modified cuckoo search-based block matching algorithm and ant weight lifting algorithm-based block matching algorithm outperformed the Jaya algorithm-based block matching process. The selection of initial solutions from different patterns has a very serious impact on the output. The ant weight-lifting algorithm performs better than the modified cuckoo search-based block-matching algorithm. However, the difference in performance is very minimal.

Table 5.4 lists the speed improvement ratio for various block-matching algorithms. The speed of the full search algorithm is the reference speed. Every algorithm observed a significant speed improvement compared to the full search. However, the ant weight-lifting algorithm-based block matching is relatively slower than the modified cuckoo search-based block-matching process. The modified cuckoo search-based block matching process converges quicker when the initial solutions are selected from a square pattern.

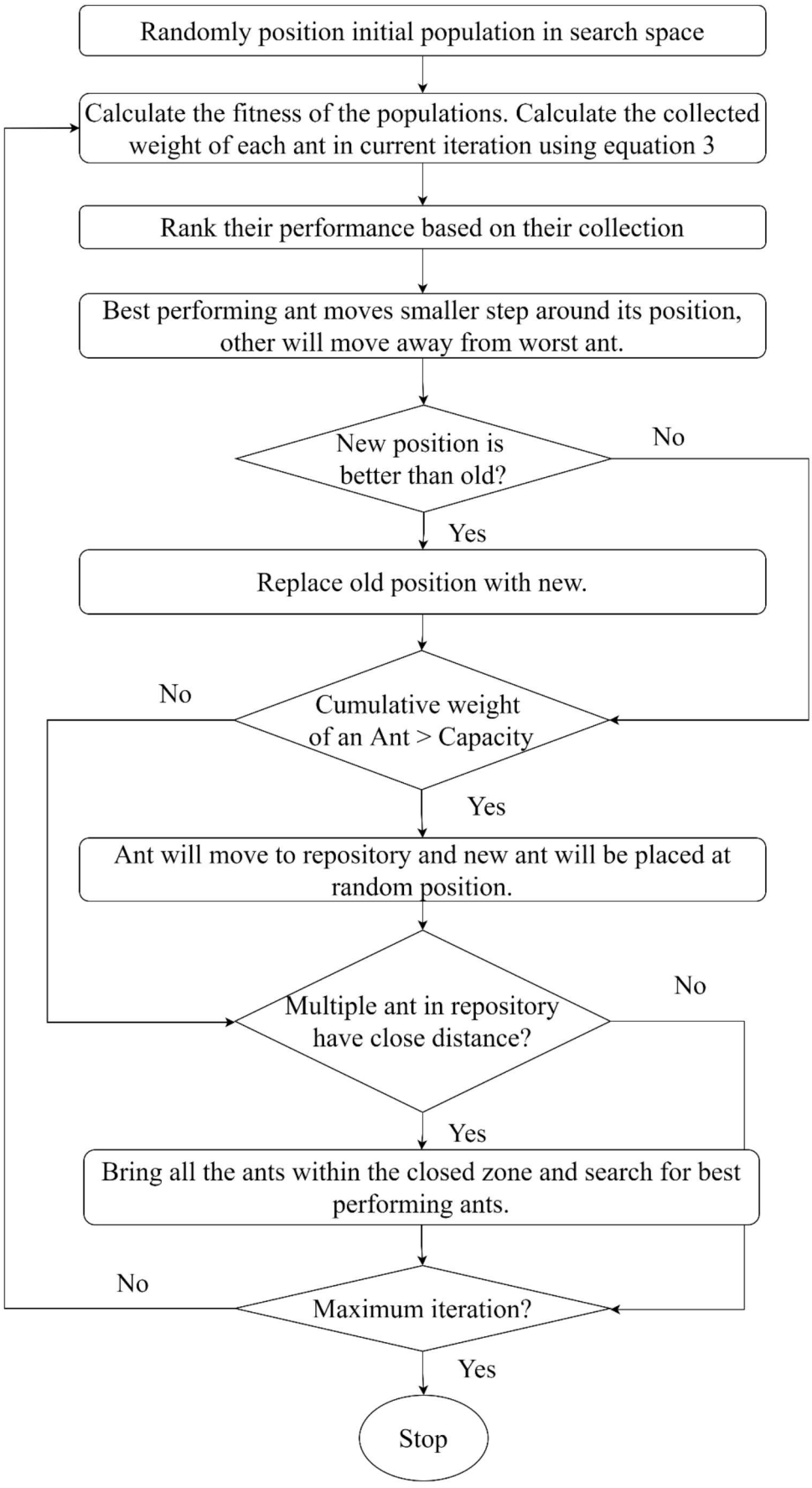
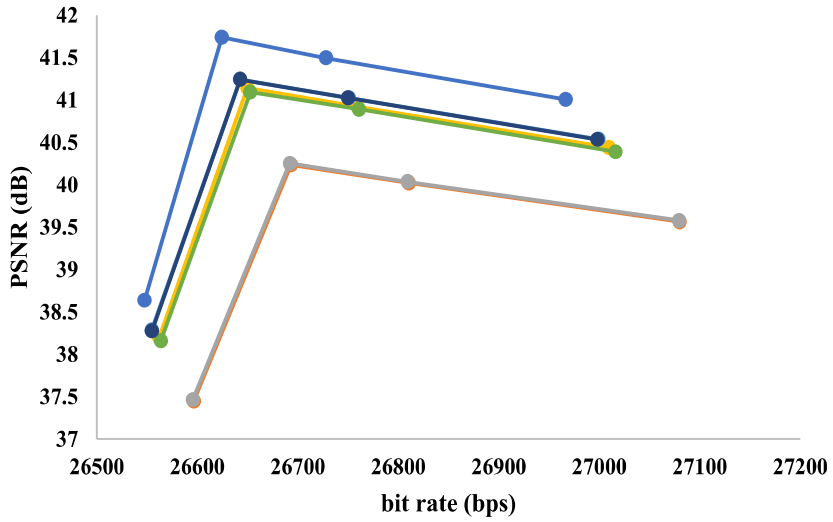


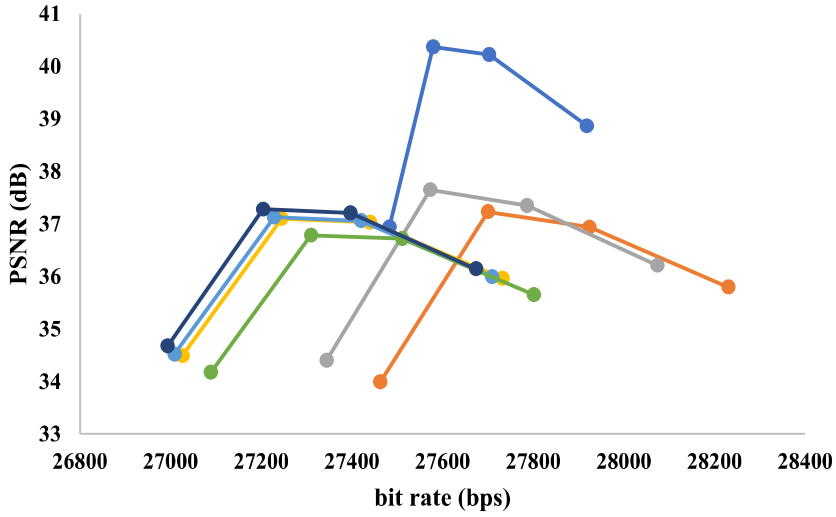
Figure 37 Ant weight lifting algorithm flowchart

5.6 The rate-distortion analysis

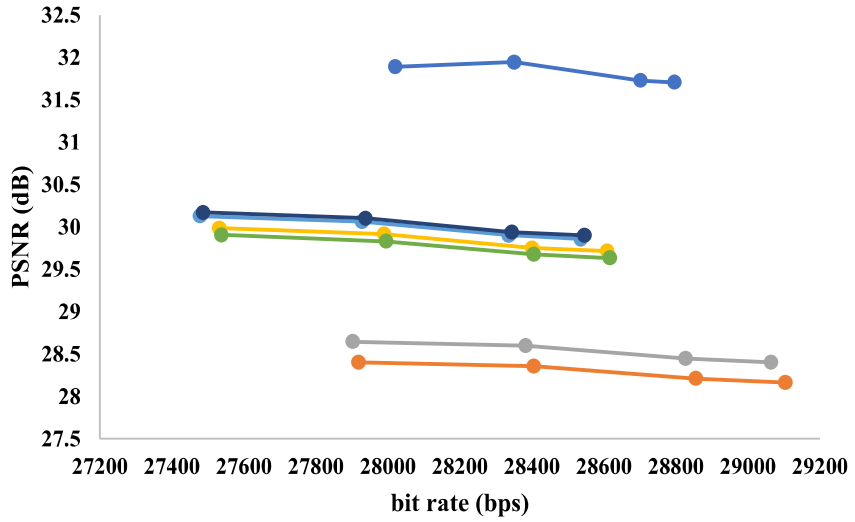
The rate-distortion analysis compares the block-matching algorithms in terms of the required bitrate to transmit the video data and the distortion in the output of the video sequence. The video coding settings used four different quantization parameters {22, 27, 32, 37} to calculate the different bitrate and PSNR values. From the rate-distortion plots, it is evident that the full search algorithm has the best performance. The ant weight lifting algorithm-based block matching process performed best among other algorithms. The difference in performance between the modified cuckoo search algorithm-based block matching process and the ant weight lifting algorithm-based block matching process is very minimum. However, both of the proposed algorithms have better rate-distortion performance compared to the Jaya algorithm-based block matching method.



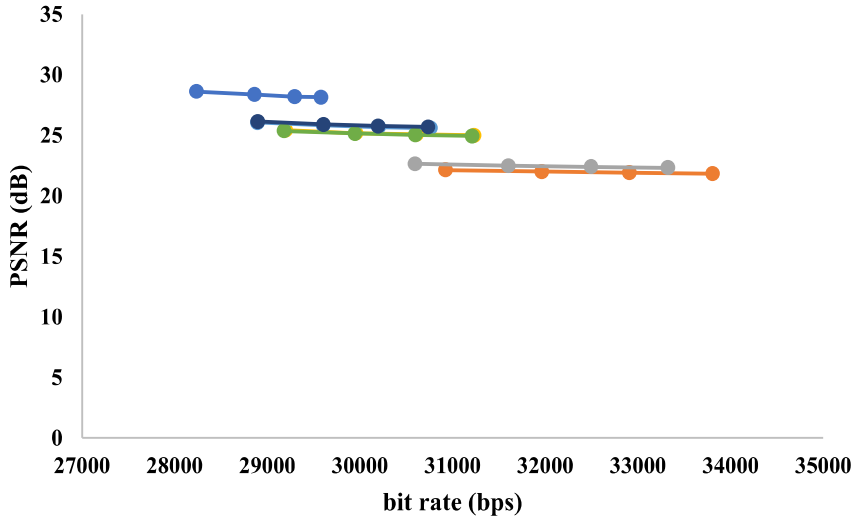
(a)



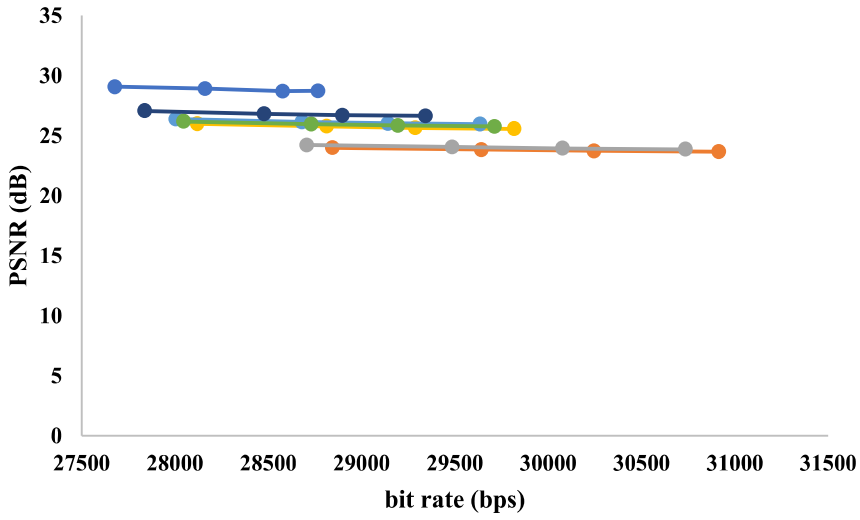
(b)


32.5
32
31.5
31
30.5
30
29.5
29
28.5
28
27.5
PSNR (dB)
27200 27400 27600 27800 28000 28200 28400 28600 28800 29000 29200
bit rate (bps)

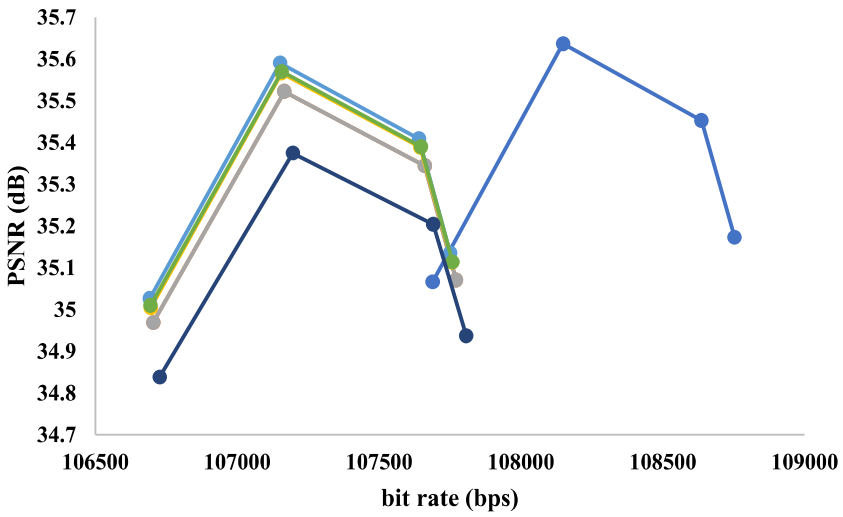
(c)


35
30
25
20
15
10
5
0
PSNR (dB)
27000 28000 29000 30000 31000 32000 33000 34000 35000
bit rate (bps)

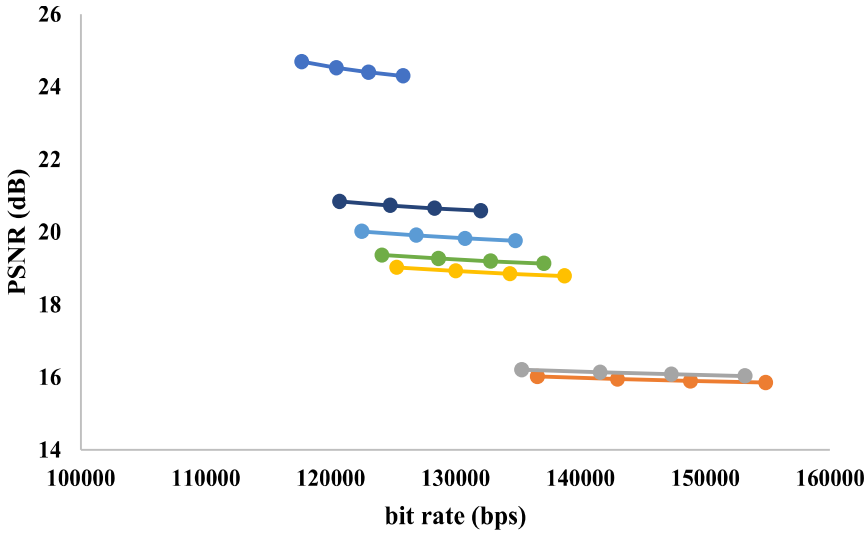
(d)


35
30
25
20
15
10
5
0
PSNR (dB)
27500 28000 28500 29000 29500 30000 30500 31000 31500
bit rate (bps)

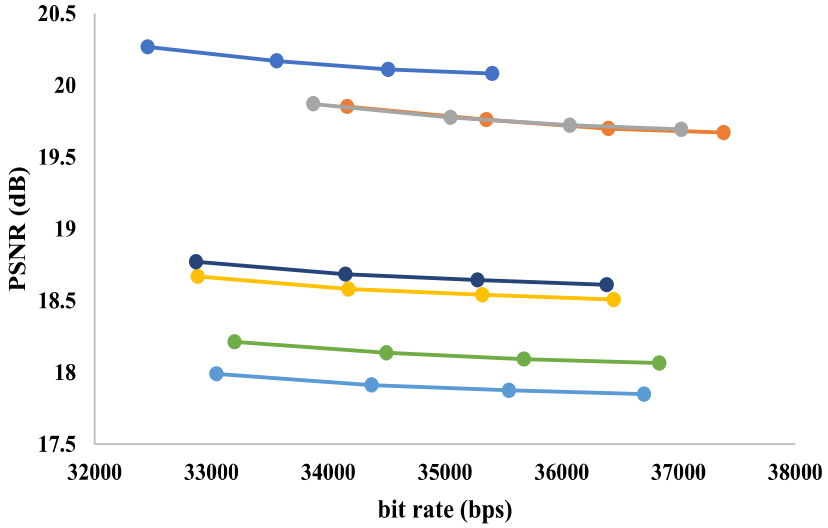
(e)


35.7
35.6
35.5
35.4
35.3
35.2
35.1
35
34.9
34.8
34.7
PSNR (dB)
106500
107000
107500
108000
108500
109000
bit rate (bps)

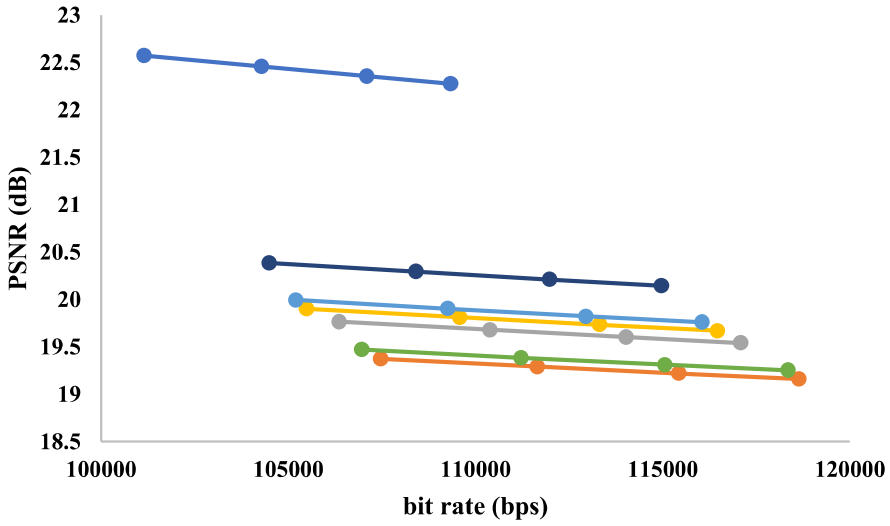
(f)


26
24
22
20
18
16
14
PSNR (dB)
100000
110000
120000
130000
140000
150000
160000
bit rate (bps)

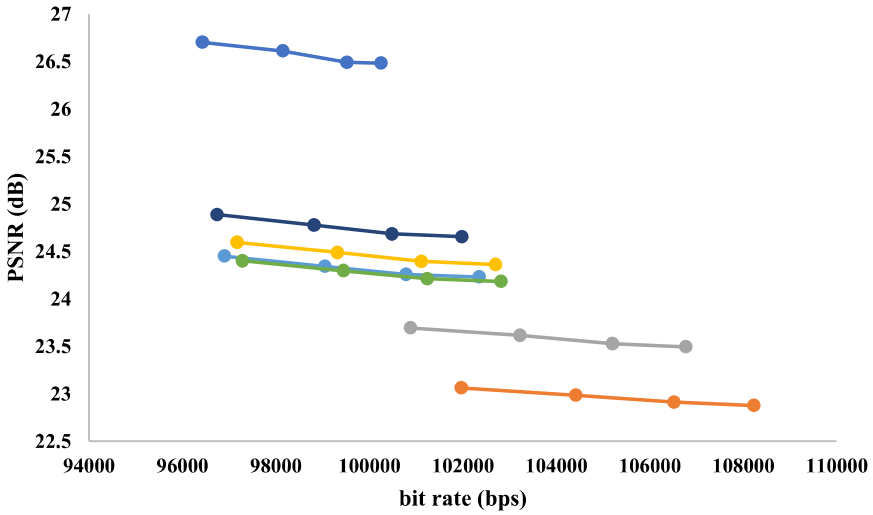
(g)


20.5
20
19.5
19
18.5
18
17.5
PSNR (dB)
32000
33000
34000
35000
36000
37000
38000
bit rate (bps)

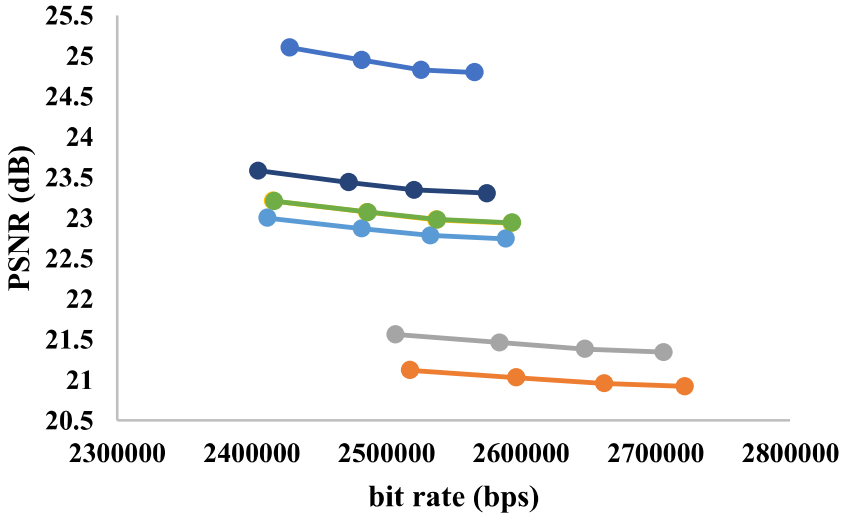
(h)


23
22.5
22
21.5
21
20.5
20
19.5
19
18.5
PSNR (dB)
100000
105000
110000
115000
120000
bit rate (bps)

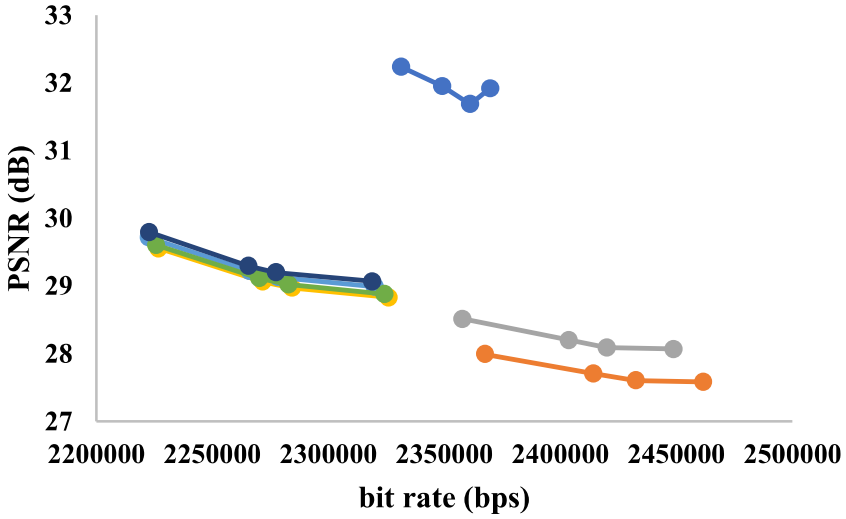
(i)


27
26.5
26
25.5
25
24.5
24
23.5
23
22.5
PSNR (dB)
94000
96000
98000
100000
102000
104000
106000
108000
110000
bit rate (bps)

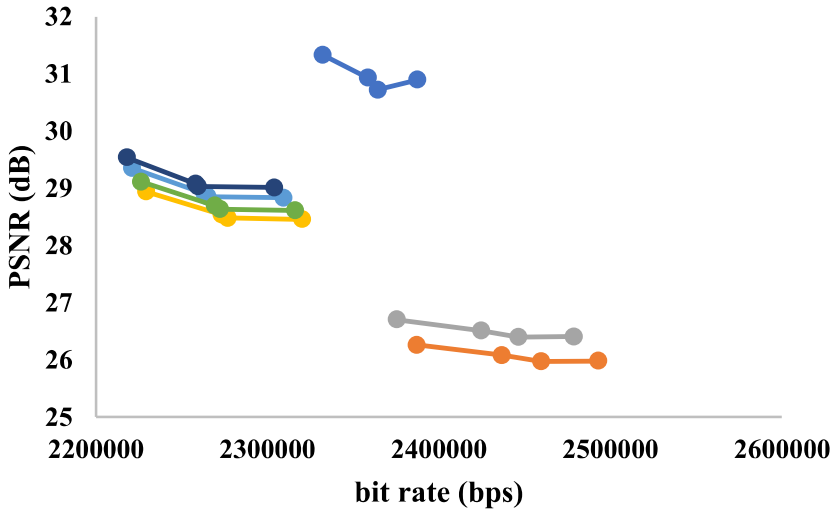
(j)


25.5
25
24.5
24
23.5
23
22.5
22
21.5
21
20.5
PSNR (dB)
2300000
2400000
2500000
2600000
2700000
2800000
bit rate (bps)

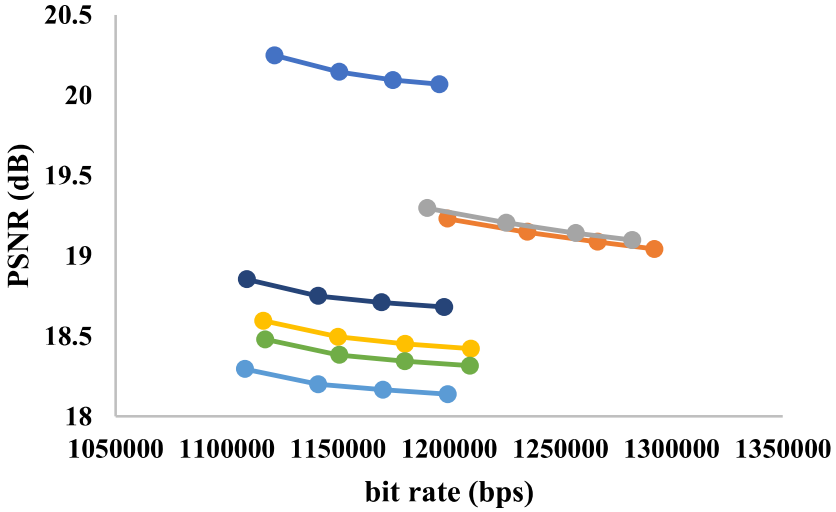
(k)


33
32
31
30
29
28
27
PSNR (dB)
2200000
2250000
2300000
2350000
2400000
2450000
2500000
bit rate (bps)

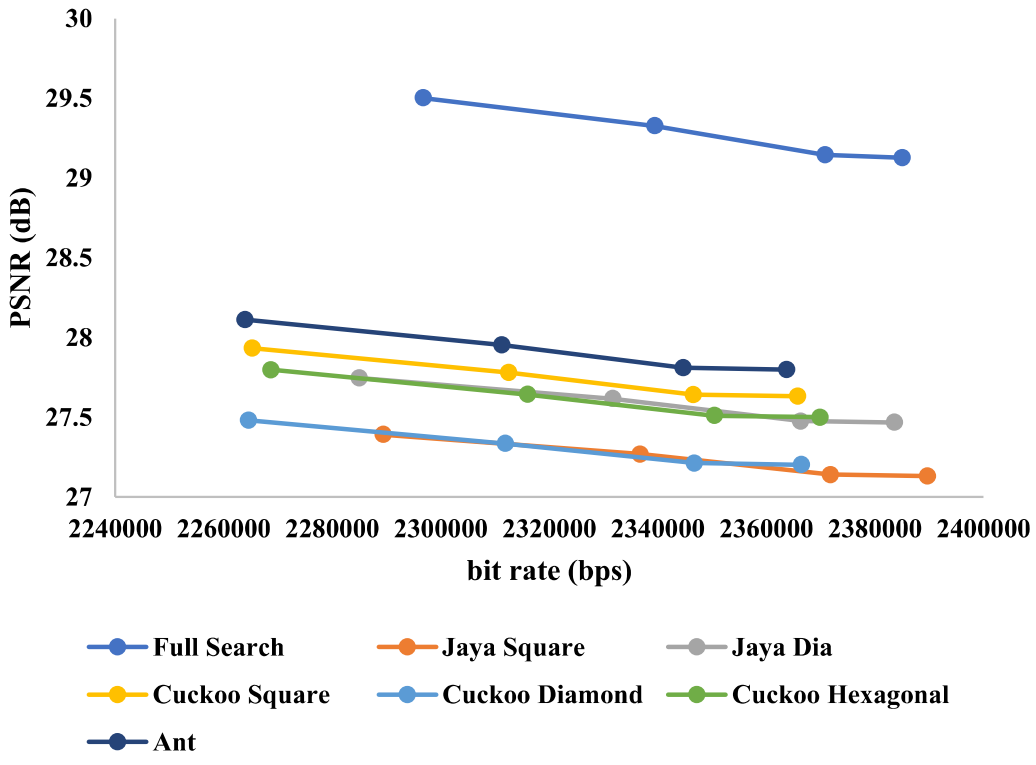
(l)


32
31
30
29
28
27
26
25
PSNR (dB)
2200000
2300000
2400000
2500000
2600000
bit rate (bps)

(m)


20.5
20
19.5
19
18.5
18
PSNR (dB)
1050000
1100000
1150000
1200000
1250000
1300000
1350000
bit rate (bps)

(n)



(o)

Figure 38 The rate distortion curve for various test sequences (a) Akiyo (b) bowing (c) Car Phone (d) City (e) Coastguard (f) Container (g) Flower (h) Football (i) Stefan (j) Tennis (k) Crowd Run (l) In to Tree (m) Old town cross (n) Park Joy (o) Rush field cut

Table 5.2: Comparison of PSNR performance between block matching algorithms

	Full search	Jaya algorithm-based block matching process-Initial solution selection from square pattern	Jaya algorithm-based block matching process-Initial solution selection from diamond pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from square pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from diamond pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from hexagon pattern	Ant weight lifting based block matching process-Initial solution selection from square pattern
Akiyo	42.80	42.05	42.05	42.36	42.57	42.49	42.57
Bowing	40.58	38.09	38.42	39.17	39.23	38.73	39.30
Car Phone	33.13	30.65	30.80	32.17	32.46	32.19	32.40
City	28.90	23.66	24.22	26.80	27.32	26.53	27.10
Coastguard	29.66	26.02	26.10	27.67	28.19	27.87	28.58
Container	37.20	37.20	37.20	37.19	37.20	37.20	36.66
Flower	25.26	17.29	17.39	20.28	21.26	20.59	21.92
Football	20.89	20.84	20.89	19.90	19.46	19.53	19.98
Stefan	23.21	20.80	21.25	21.14	21.27	20.49	21.59
Tennis	26.33	23.70	24.33	25.02	24.97	24.70	25.17
Crowd Run	27.62	23.93	24.34	25.91	25.78	25.90	26.21
In to Tree	33.20	30.02	30.40	30.82	30.94	30.88	31.03
Old Town Cross	32.34	28.69	29.03	30.45	30.74	30.57	30.88
Park Joy	22.61	21.39	21.42	21.05	20.86	20.97	21.32
Rush Field Cut	31.73	29.98	30.29	30.47	30.14	30.35	30.57
Average	30.36	27.62	27.88	28.69	28.83	28.60	29.02

Table 5.3: Comparison of SSIM performance between block matching algorithms

	Full search	Jaya algorithm-based block matching process-Initial solution selection from square pattern	Jaya algorithm-based block matching process-Initial solution selection from diamond pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from square pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from diamond pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from hexagon pattern	Ant weight lifting based block matching process-Initial solution selection from square pattern
Akiyo	0.995	0.994	0.994	0.995	0.995	0.995	0.995
Bowing	0.978	0.949	0.955	0.966	0.967	0.963	0.968
Car Phone	0.962	0.935	0.937	0.954	0.956	0.954	0.956
City	0.867	0.611	0.652	0.795	0.817	0.785	0.809
Coastguard	0.921	0.841	0.846	0.881	0.893	0.883	0.902
Container	0.975	0.975	0.975	0.975	0.975	0.975	0.975
Flower	0.943	0.700	0.706	0.839	0.870	0.849	0.888
Football	0.623	0.609	0.617	0.575	0.562	0.562	0.582
Stefan	0.857	0.783	0.803	0.795	0.798	0.769	0.812
Tennis	0.832	0.730	0.747	0.790	0.800	0.790	0.804
Crowd Run	0.853	0.756	0.769	0.797	0.796	0.797	0.805
In to Tree	0.813	0.724	0.731	0.716	0.719	0.718	0.721
Old Town Cross	0.824	0.756	0.762	0.766	0.770	0.767	0.772
Park Joy	0.798	0.697	0.703	0.719	0.727	0.723	0.737
Rush Field Cut	0.904	0.888	0.891	0.886	0.884	0.885	0.887
Average	0.876	0.796	0.806	0.830	0.835	0.828	0.841

Table 5.4: Comparison of SIR (%) performance between block matching algorithms

	Jaya algorithm-based block matching process-Initial solution selection from square pattern	Jaya algorithm-based block matching process-Initial solution selection from diamond pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from square pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from diamond pattern	Modified cuckoo search algorithm-based block matching process-Initial solution selection from hexagon pattern	Ant weight lifting based block matching process-Initial solution selection from square pattern
Akiyo	98.80	99.51	99.61	99.52	99.45	98.80
Bowing	96.65	98.23	98.61	98.32	97.94	96.53
Car Phone	96.46	96.10	96.90	96.26	95.52	96.42
City	96.07	93.58	94.82	94.00	92.43	96.09
Coastguard	96.46	94.15	95.23	94.36	93.27	96.35
Container	96.53	98.93	99.14	98.93	98.81	96.51
Flower	97.27	95.92	96.70	96.11	95.08	97.15
Football	93.20	93.74	95.07	94.24	92.89	92.53
Stefan	95.40	95.02	95.97	95.09	94.24	94.98
Tennis	94.85	93.98	95.23	94.24	93.07	94.34
Crowd Run	95.27	96.65	97.36	96.75	95.99	94.93
In to Tree	94.72	98.74	99.00	98.78	98.49	94.20
Old Town Cross	94.92	98.53	98.82	98.57	98.22	94.46
Park Joy	92.13	96.13	96.92	96.91	95.39	91.38
Rush Field Cut	96.69	98.66	98.94	98.70	98.43	96.54
Average	95.69	96.52	97.22	96.72	95.95	95.41

Chapter 6:

Conclusion and Future work

The image compression method based on Jaya algorithm-based codebook optimization presented in this study outperforms existing vector quantization-based image compression processes. The output of the proposed algorithm with an LZMA encoded residue successfully preserves the diagnostic essence of the biomedical images.

The operational parameters optimization of test zone search algorithm successfully improves the speed of the test zone search without degrading the quality of the output. The fuzzy inference-based decision-making module outperforms the other two proposed processes for modifying the operational parameters of test zone search.

The modified cuckoo search as well as the ant weight lifting algorithm successfully optimized the block-matching process. The optimization enhances both the speed and the quality of the output compared to the state-of-the art Jaya algorithm-based motion estimation processes. The ant weight lifting algorithm-based motion estimation process surpasses the modified cuckoo search-based block-matching algorithm.

The codebook optimization approaches may be extended to the one-dimensional signal compression. There is a high possibility that the codebook optimization method will perform similarly well for one-dimensional signals. The metric motion factor (M_f), which is employed in the operational parameter optimization of test zone search, may be applied in other applications such as frame partitioning, initial motion vector predictor, and many more. Overall, the compression process of different biomedical signals is successfully optimized using the proposed methods.

Bibliography

[1] C. Shannon, "A mathematical theory of communication (1948)," in Ideas That Created the Future, The MIT Press, 2021, pp. 121–134.

[2] D. Huffman, "A method for the construction of minimum-redundancy codes," Proc. IRE, vol. 40, no. 9, pp. 1098–1101, 1952.

[3] G. G. Langdon, "An introduction to arithmetic coding," IBM J. Res. Dev., vol. 28, no. 2, pp. 135–149, 1984.

[4] R. A.m, K. W.m, E.-D. M. A, and W. Ahmed, "Jpeg image compression using discrete cosine transform - A survey," Int. j. comput. sci. eng. surv., vol. 5, no. 2, pp. 39–47, 2014.

[5] J. Mulindi, "Sources of biomedical signals," Biomedical Instrumentation Systems, 09-Apr-2020. [Online]. Available: https://www.biomedicalinstrumentationsystems.com/sources-of-biomedical-signals/. [Accessed: 21-Oct-2022].

[6] C. H.-H. A. J. Moura, "Biomedical signal processing," in Biomedical engineering and design handbook 2, M. Kutz, Ed. The McGraw-Hill Companies, Inc., 2010, pp. 559–579.

[7] N. Wickramasinghe and E. Geisler, Eds., Encyclopedia of Healthcare Information Systems. Hershey, PA: Medical Information Science Reference, 2008.

[8] X. Xu, "Automated Pain Detection in Facial Videos Using Transfer Learning," UC San Diego, 2021.

[9] P. Werner, A. Al-Hamadi, R. Niese, S. Walter, S. Gruss, and H. C. Traue, "Automatic pain recognition from video and biomedical signals," in 2014 22nd International Conference on Pattern Recognition, 2014.

[10] M. Mohan, A. K. Abhinav, A. Ashok, A. V. Akhil, and P. R. Achinth, "Depression detection using facial expression and sentiment analysis," in 2021 Asian Conference on Innovation in Technology (ASIANCON), 2021.

[11] Q. Wang, H. Yang, and Y. Yu, "Facial expression video analysis for depression detection in Chinese patients," J. Vis. Commun. Image Represent., vol. 57, pp. 228–233, 2018.

[12] A. Nait-Ali and C. Cavaro-Menard, Eds., Compression of Biomedical Images and Signals, 1st ed. London, England: ISTE Ltd and John Wiley & Sons, 2013.

[13] Foran, D. J., et al. "Compression Guidelines for Diagnostic Telepathology." IEEE Transactions on Information Technology in Biomedicine: A Publication of the IEEE Engineering in Medicine and Biology Society, vol. 1, no. 1, 1997, pp. 55–60, doi:10.1109/4233.594046.

[14] "Data Never Sleeps 5.0." Domo.com, https://www.domo.com/learn/infographic/data-never-sleeps-5. Accessed 21 Dec. 2022.

[15] H. Kalva, "The H.264 Video Coding Standard," IEEE multimed., vol. 13, no. 4, pp. 86–90, 2006.

[16] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, “Overview of the high efficiency video coding (HEVC) standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, pp. 1649–1668, 2012.

[17] B. Bross, J. Chen, J.-R. Ohm, G. J. Sullivan, and Y.-K. Wang, “Developments in international video coding standardization after AVC, with an overview of versatile video coding (VVC),” Proc. IEEE Inst. Electr. Electron. Eng., vol. 109, no. 9, pp. 1463–1493, 2021.

[18] S. Acharjee and S. S. Chaudhuri, “Fuzzy logic based three step search algorithm for motion vector estimation,” Int. J. Image Graph. Signal Process., vol. 4, no. 2, pp. 37–43, 2012.

[19] S. Acharjee, S. Chakraborty, W. B. A. Karaa, A. T. Azar, and N. Dey, “Performance evaluation of different cost functions in motion vector estimation,” Int. J. Serv. Sci. Manag. Eng. Technol., vol. 5, no. 1, pp. 45–65, 2014.

[20] A. Barjatya, “Block matching algorithms for motion estimation,” IEEE Transactions Evolution Computation, vol. 8, no. 3, pp. 225–239, 2004.

[21] S. Acharjee and S. S. Chaudhuri, “Fuzzy logic based three step search algorithm for motion vector estimation,” Int. J. Image Graph. Signal Process., vol. 4, no. 2, pp. 37–43, 2012.

[22] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” IEEE Trans. Circuits Syst. Video Technol., vol. 6, no. 3, pp. 313–317, 1996.

[23] J.-X. Yang, Y.-B. Zhang, L.-H. Huang, D.-C. Guo, and Y.-K. Yang, “A novel diamond search algorithm for fast block motion estimation,” in International Conference on Image Processing and Pattern Recognition in Industrial Engineering, 2010.

[24] K. Hung-Kei Chow and M. L. Liou, “Genetic motion search algorithm for video compression,” IEEE Trans. Circuits Syst. Video Technol., vol. 3, no. 6, pp. 440–445, 1993

[25] Y. Nie and K.-K. Ma, “Adaptive rood pattern search for fast block-matching motion estimation,” IEEE Trans. Image Process., vol. 11, no. 12, pp. 1442–1449, 2002.

[26] P. I. Hosur and K.-Y. Ma, Motion vector field adaptive fast motion estimation. 1999.

[27] A. M. Tourapis, O. C. L. Au, and M. L. Liou, “Predictive motion vector field adaptive search technique (PMVFAST): enhancing block-based motion estimation,” in Visual Communications and Image Processing 2001, 2000.

[28] A. M. Tourapis, “Enhanced predictive zonal search for single and multiple frame motion estimation,” in Visual Communications and Image Processing 2002, 2002.

[29] H.-M. Wong, O. C. Au, C.-W. Ho, and S.-K. Yip, “Enhanced predictive motion vector field adaptive search technique (E-PMVFAST)-based on future mv prediction,” in 2005 IEEE International Conference on Multimedia and Expo, IEEE, 2005.

[30] C.-C. Wang and G.-L. Li, “Hardware-friendly advanced motion vector prediction method and its architecture design for high efficiency video coding,” Multimed. Tools Appl., vol. 76, no. 23, pp. 25285–25296, 2017.

[31] Nisar, Humaira, Aamir Saeed Malik, and Tae-Sun Choi. "Content adaptive fast motion estimation based on spatio-temporal homogeneity analysis and motion classification." Pattern Recognition Letters 33.1 (2012): 52-61.

[32] H. Jia and L. Zhang, “Directional cross Diamond search algorithm for fast block motion estimation,” arXiv [cs.CV], 2008.

[33] C.-I. Lin and J.-L. Wu, “A lightweight genetic block-matching algorithm for video coding,” IEEE Trans. Circuits Syst. Video Technol., vol. 8, no. 4, pp. 386–392, 1998.

[34] M. F. So and A. Wu, “Four-step genetic search for block motion estimation,” in Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP ’98 (Cat. No.98CH36181), 2002.

[35] R. Poli, J. Kennedy, and T. Blackwell, “Particle swarm optimization: An overview,” Swarm Intell., vol. 1, no. 1, pp. 33–57, 2007.

[36] Shi, Y. and Eberhart, R., 1998, May. A modified particle swarm optimizer. In 1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360)(pp. 69-73). IEEE.

[37] X. Yuan and X. Shen, “Block matching algorithm based on particle swarm optimization for motion estimation,” in 2008 International Conference on Embedded Software and Systems, 2008.

[38] S. I. A. Pandian, G. J. Bala, and J. Anitha, “A pattern based PSO approach for block matching in motion estimation,” Engineering Applications of Artificial Intelligence, vol. 26, no. 8, pp. 1811–1817, 2013.

[39] Cuevas, Erik, Daniel Zaldivar, Marco Pérez-Cisneros, and Diego Oliva. "Block-matching algorithm based on differential evolution for motion estimation." Engineering Applications of Artificial Intelligence 26, no. 1 (2013): 488-498.

[40] Dixit, Abhishek, Ashish Mani, and Rohit Bansal. "DEPSOSVM: variant of differential evolution based on PSO for image and text data classification." International Journal of Intelligent Computing and Cybernetics (2020).

[41] Bhattacharjee, Kamanasish, Arti Tiwari, and Sushil Kumar. "Block matching algorithm based on hybridization of harmony search and differential evolution for motion estimation in video compression." In Soft Computing: Theories and Applications, pp. 625-635. Springer, Singapore, 2018.

[42] Karaboga, Dervis. "Artificial bee colony algorithm." scholarpedia 5, no. 3 (2010): 6915.

[43] Cuevas, Erik. "Artificial Bee Colony (ABC) algorithm and its use in digital image processing." Inteligencia Artificial 18, no. 55 (2015): 50-68.

[44] Hemanth, D. Jude, and J. Anitha. "A pattern-based artificial bee colony algorithm for motion estimation in video compression techniques." Circuits, Systems, and Signal Processing 37, no. 4 (2018): 1609-1624.

[45] Dash, Bodhisattva, Suvendu Rup, Figlu Mohanty, and M. N. S. Swamy. "A hybrid block-based motion estimation algorithm using JAYA for video coding techniques." Digital Signal Processing 88 (2019): 160-171.

[46] Praveena, Manne, N. Balaji, and C. D. Naidu. "FPGA design of area efficient and superfast motion estimation using JAYA optimization-based block matching algorithm." Communication, Software and Networks: Proceedings of INDIA 2022. Singapore: Springer Nature Singapore, 2022. 137-148..

[47] Goldberg, David E., and Manohar P. Samtani. "Engineering optimization via genetic algorithm." In Electronic computation, pp. 471-482. ASCE, 1986.

[48] Rao, R. "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems." International Journal of Industrial Engineering Computations 7, no. 1 (2016): 19-34.

[49] Gersho, Allen. "On the structure of vector quantizers." IEEE Transactions on Information Theory 28.2 (1982): 157-166.

[50] Linde, Yoseph, Andres Buzo, and Robert Gray. "An algorithm for vector quantizer design." IEEE Transactions on communications 28.1 (1980): 84-95.

[51] Lloyd, Stuart. "Least squares quantization in PCM." IEEE transactions on information theory 28.2 (1982): 129-137.

[52] Patanè, Giuseppe, and Marco Russo. "The enhanced LBG algorithm." Neural networks 14.9 (2001): 1219-1237.

[53] Bilal, Muhammad, Zahid Ullah, and Ihtesham Ul Islam. "Fast codebook generation using pattern based masking algorithm for image compression." IEEE Access 9 (2021): 98904-98915.

[54] Feng, Hsuan-Ming, Ching-Yi Chen, and Fun Ye. "Evolutionary fuzzy particle swarm optimization vector quantization learning scheme in image compression." Expert Systems with Applications 32.1 (2007): 213-222.

[55] Yang, Xin-She, and Adam Slowik. "Firefly algorithm." Swarm Intelligence Algorithms. CRC Press, 2020. 163-174.

[56] Chiranjeevi, Karri, et al. "Modified firefly algorithm (MFA) based vector quantization for image compression." Computational Intelligence in Data Mining—Volume 2. Springer, New Delhi, 2016. 373-382.

[57] Yang, Xin-She, and Xingshi He. "Bat algorithm: literature review and applications." International Journal of Bio-inspired computation 5.3 (2013): 141-149.

[58] Karri, Chiranjeevi, and Umaranjan Jena. "Fast vector quantization using a Bat algorithm for image compression." Engineering Science and Technology, an International Journal 19.2 (2016): 769-781.

[59] Yang, Xin-She, and Suash Deb. "Cuckoo search: recent advances and applications." Neural Computing and applications 24.1 (2014): 169-174.

[60] Chiranjeevi, Karri, and Uma Ranjan Jena. "Image compression based on vector quantization using cuckoo search optimization technique." Ain Shams Engineering Journal 9.4 (2018): 1417-1431.

[61] Rahebi, Javad. "Vector quantization using whale optimization algorithm for digital image compression." Multimedia Tools and Applications 81.14 (2022): 20077-20103.

[62] Geetha, Karuppaiah, et al. "An evolutionary lion optimization algorithm- based image compression technique for biomedical applications." Expert Systems 38.1 (2021): e12508.

[63] Van der Merwe, D. W., and Andries P. Engelbrecht. "Data clustering using particle swarm optimization." The 2003 Congress on Evolutionary Computation, 2003. CEC'03.. Vol. 1. IEEE, 2003.

[64] Hankamer, Michael. "A modified Huffman procedure with reduced memory requirement." IEEE Transactions on Communications 27.6 (1979): 930-932.

[65] Huynh-Thu, Quan, and Mohammed Ghanbari. "Scope of validity of PSNR in image/video quality assessment." Electronics letters 44.13 (2008): 800-801.

[66] Hore, Alain, and Djemel Ziou. "Image quality metrics: PSNR vs. SSIM." 2010 20th international conference on pattern recognition. IEEE, 2010.

[67] Dikbas, Salih, Tarik Arici, and Yucel Altunbasak. "Fast motion estimation with interpolation-free sub-sample accuracy." IEEE Transactions on Circuits and Systems for Video Technology 20.7 (2010): 1047-1051.

[68] Bjontegaard, Gisle. "Calculation of average PSNR differences between RD-curves." VCEG-M33 (2001).

[69] Cai, Jianrui, Zisheng Cao, and Lei Zhang. "Learning a single tucker decomposition network for lossy image compression with multiple bits-per-pixel rates." IEEE Transactions on Image Processing 29 (2020): 3612-3625.

[70] Setia, Amit, and Priyanka Ahlawat. "Enhanced LZW algorithm with less compression ratio." Proceedings of International Conference on Advances in Computing. Springer, New Delhi, 2013.

[71] Benesty, Jacob, et al. "Pearson correlation coefficient." Noise reduction in speech processing. Springer, Berlin, Heidelberg, 2009. 1-4.

[72] Loong, Tze-Wey. "Understanding sensitivity and specificity with the right side of the brain." Bmj 327.7417 (2003): 716-719.

[73] Leavline, E. Jebamalar, and D. A. A. G. Singh. "Hardware implementation of LZMA data compression algorithm." International Journal of Applied Information Systems (IJAIS) 5.4 (2013): 51-56.

[74] Zhang, Li, et al. "History-based motion vector prediction in versatile video coding." 2019 Data Compression Conference (DCC). IEEE, 2019.

[75] Parmar, Nidhi, and Myung Hoon Sunwoo. "Enhanced test zone search motion estimation algorithm for HEVC." 2014 International SoC Design Conference (ISOCC). IEEE, 2014.

[76] Kibeya, Hassan, et al. "TZSearch pattern search improvement for HEVC motion estimation modules." 2014 1st International Conference on Advanced Technologies for Signal and Image Processing (ATSIP). IEEE, 2014.

[77] Goncalves, Paulo, et al. "Octagonal-axis raster pattern for improved test zone search motion estimation." 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018.

[78] Sant'Anna, Gabriel B., et al. "Relying on a rate constraint to reduce motion estimation complexity." ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021.

[79] Tang, Na, et al. "Fast CTU partition decision algorithm for VVC intra and inter coding." 2019 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS). IEEE, 2019.

[80] Tsallis, Constantino, et al. "Statistical-mechanical foundation of the ubiquity of Lévy distributions in nature." Physical Review Letters 75.20 (1995): 3589.

[81] Blickle, Tobias. "Tournament selection." Evolutionary computation 1 (2000): 181-186.

[82] Zadeh, Lotfi Asker. "Fuzzy sets as a basis for a theory of possibility." Fuzzy sets and systems 1.1 (1978): 3-28.

[83] Chai, Yuanyuan, Limin Jia, and Zundong Zhang. "Mamdani model based adaptive neural fuzzy inference system and its application." International Journal of Computer and Information Engineering 3.3 (2009): 663-670.

[84] Van Broekhoven, Ester, and Bernard De Baets. "A comparison of three methods for computing the center of gravity defuzzification." 2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No. 04CH37542). Vol. 3. IEEE, 2004.

[85] Karafotias, Giorgos, Mark Hoogendoorn, and Ágoston E. Eiben. "Parameter control in evolutionary algorithms: Trends and challenges." IEEE Transactions on Evolutionary Computation 19.2 (2014): 167-187.

[86] Bhattacharjee, Kamanasish, and Sushil Kumar. "A novel block matching algorithm based on Cuckoo search." 2017 2nd International Conference on Telecommunication and Networks (TEL-NET). IEEE, 2017.

[87] Fossøy, Frode, et al. "Ancient origin and maternal inheritance of blue cuckoo eggs." Nature communications 7.1 (2016): 1-6.

[88] Li, Xiang, et al. "Initialization strategies to enhancing the performance of genetic algorithms for the p-median problem." Computers & Industrial Engineering 61.4 (2011): 1024-1034.

[89] Xiao, Ningchuan. "A unified conceptual framework for geographical optimization using evolutionary algorithms." Annals of the Association of American Geographers 98.4 (2008): 795-817.

Databases

[1] [Database] Test Image Collection available at https://www.hlevkin.com/hlevkin/06testimages.htm (Last accessed on 23rd December, 2022)

[2] [Database] Diabetic Retinopathy Detection available at https://www.kaggle.com/c/diabetic-retinopathy-detection (Last accessed on 23rd December, 2022)

[3] [Database] Xiph.org Video Test Media [derf's collection] available at https://media.xiph.org/video/derf/ (Last accessed on 23rd December, 2022)