II. SURVEY OF IMAGE COMPRESSION TECHNIQUES

DIGITAL image is a rectangular array of picture elements or pixels arranged in \(m\) rows and \(n\) columns. There are different types of images [1], namely, (a) Bi-level Images: - images with pixels which are either black or white in color where each pixel occupies one bit, (b) Grayscale Images: - pixels of this type of images are either 4, 8, 12, 16 or 24 bits with gray values from 0 to \(2^{n-1}\), where \(n\) is the number of bits and the grayscale image is divided into bitplanes, (c) Continuous-tone Images: - images having many similar colors with the pixel values differing by small values and hence it is difficult to differentiate the colors, and (d) Discrete-tone Images: - images with same adjacent pixels or differing by a large value.

Image compression is the technique of reducing the number of bits needed to represent a digital image. This is possible by removing the redundancy present in the digital images. Different types of redundancies can be identified in digital images. They are a) spatial redundancy (correlation between neighboring pixels), b) spectral redundancy (correlation between different color planes or spectral bands), c) temporal redundancy (correlation between different frames in sequences of images), and d) psychovisual redundancy (information in images which are less important for normal visual processing) [2].

Image compression techniques can be classified into two major categories namely, lossless compression and lossy compression techniques. Lossless compression, which is otherwise known as reversible compression, enables us to reconstruct the images which are identical to the original image on a pixel-by-pixel basis. Thus there is no loss of information. Since these techniques have to represent all the information available in the original image, they do not provide high compression ratios. These techniques are useful for storing medical images, telemedicine etc. Lossy compression also known as irreversible compression does not preserve all the information in the original image and thus the reconstructed image is not the same as the original image. They provide high compression ratios
depending on the amount of information preserved from the original image. These techniques are useful for videoconferencing, television broadcasting etc., where certain amount of degradation is acceptable.

The rest of this chapter is organized as follows. Section 2.1 describes transform based image compression techniques. Section 2.2 describes various quality metrics for performance of compression techniques. Section 2.3 describes various lossless image compression techniques. Section 2.4 gives a detailed description of some of the lossy compression techniques. Section 2.5 describes some of the image compression techniques using soft computing. Section 2.6 presents the results of lossless compression of Wireless Application Protocol Bitmap (WBMP) images and evaluates the performance of various image transforms such as wavelets, curvelets, contourlets and bandlets in terms of number of coefficients retained and Peak-Signal-to-Noise Ratio (PSNR).

2.1. TRANSFORM BASED IMAGE COMPRESSION

Various image transforms convert the spatial image pixel values to transform coefficient values. Most of the energy in the image is contained in a few large transform coefficients. Smaller coefficients can be quantized or truncated and the image can be reconstructed without much degradation.

An image representation should have the following properties [3].

1) Multiresolution: The representation should allow images to be successively approximated from coarse to fine resolutions.
2) Localization: The basis elements of the representation should be localized in both spatial and frequency domains.
3) Critical Sampling: The representation should form a basis with small redundancy which will be useful for applications like compression.
4) Directionality: The representation should contain basis elements oriented at a variety of directions, much more than that offered by separable wavelets.
5) Anisotropy: To capture smooth contours in images, the representation should contain basis elements using a variety of elongated shapes with different aspect ratios.

Wavelets have the ability to efficiently represent functions which are smooth away from point singularities. Wavelet coefficients of a point singularity behave like the wavelet coefficients of a smooth function. Thus smooth functions with point singularities can be efficiently approximated by wavelets. This fact has
enabled wavelets to be used in a number of applications, especially in image coding. The following are some of the advantages of wavelet transform a) the wavelet transform is separable b) wavelet transform can be implemented using a simple filter design and c) it is computationally less complex.

Wavelets do not efficiently approximate edges in two dimension. Wavelet basis functions are isotropic, i.e. they cannot adapt to geometrical structure. In a two-dimensional wavelet transform large wavelet coefficients are present even at fine scales along the edges. Wavelets have only the first three properties, and there comes the need for other transforms like curvelet transform, contourlet transform, wedgelet transform, bandlet transform etc. The following subsections describe popular transforms required for image compression.

2.1.1 Basics of Wavelet Transform

Multiresolution [4] provides tools to describe mathematical objects like signals and images at different levels of resolutions. It has been used in various fields like transmission of compressed data, removal of noise from signals and images, identification of important features from data etc.

Let the function \( \phi(t) \) be the scaling function. A large number of functions can be generated by taking linear combinations of the scaling function and its translates, which are given by

\[
f(t) = \sum_k a_k \phi(t - k)
\]

(1)

The function, represented by a scaling function, can also be represented by the dilated versions of the scaling function.

Let \( \phi_k(t) = \phi(t - k) \). The set of all functions represented as a linear combination of the set \( \{\phi_k(t)\} \) can be represented by

\[
f(t) = \sum_k a_k \phi_k(t)
\]

(2)

and is called the span of the set \( \{\phi_k(t)\} \) or \( \text{span}\{\phi_k(t)\} \). The sum of all the functions that are limits of sequences of functions in \( \text{span}\{\phi_k(t)\} \) is called the
II. SURVEY OF IMAGE COMPRESSION TECHNIQUES ...  

... closure of \( \text{span}\{\phi_k(t)\} \), \( \text{span}\{\phi_k(t)\} \). Let this set be represented as \( V_0 \). The scaling functions at different resolutions can be obtained using

\[
\phi_{j,k}(t) = 2^{j/2}\phi(2^jt - k)
\]  

(3)

Any function that can be represented by the translates of \( \phi(t) \) can also be represented by a linear combination of translates of \( \phi_{1,0}(t) \). If

\[
V_1 = \text{span}\{\phi_{1,k}(t)\}
\]

(4)

then \( V_0 \subset V_1 \), \( V_1 \subset V_2 \) and so on.

Scaling function has the property that any function that can be represented by an expansion at some resolution \( j \) can also be represented by dilations of the scaling function at resolution \( j + 1 \). Thus, the scaling function itself can be represented by its dilations at a higher resolution

\[
\phi(t) = \sum_k h_k \phi_{1,k}(t)
\]

(5)

By substituting \( \phi_{1,k}(t) = \sqrt{2}\phi(2t - k) \), the above equation can be written as

\[
\phi(t) = \sum_k h_k \sqrt{2}\phi(2t - k)
\]

(6)

which is called the multiresolution analysis equation.

Given a scaling function, a wavelet function \( \psi(t) \) can be defined, which along with its translates and scalings, spans the difference between any two adjacent scaling subspaces. The set \{\phi_{j,k}(t)\} of wavelets is given by

\[
\phi_{j,k}(t) = 2^{j/2}\phi(2^jt - k)
\]

(7)

for all \( k \in \mathbb{Z} \) that spans the \( W_j \) spaces, where

\[
W_j = \text{Span}_k\{\phi_{j,k}(t)\}
\]

(8)

If \( f(t) \in W_j \), then

\[
f(t) = \sum_k \alpha_k \phi_{j,k}(t)
\]

(9)

The scaling and wavelet function subspaces are related by

\[
V_{j+1} = V_j \otimes W_j
\]

(10)
where \( \otimes \) denotes the union of spaces. Any wavelet function can be expressed as a weighted sum of shifted, double-resolution scaling functions. Thus

\[
\phi(t) = \sum_k h_\phi(n) \sqrt{2} \psi(2t - n)
\]

where \( h_\phi(n) \) are the wavelet function coefficients and \( h_\phi \) is the wavelet vector.

Wavelet transform provides a time-frequency representation of the signal and it uses multiresolution technique which helps in analyzing different frequencies at different resolutions.

A function \( f(x) \) can be written as

\[
f(x) = \sum_k c_{j_0}(k) \psi_{j_0,k}(x) + \sum_{j=j_0}^{\infty} \sum_k d_j(k) \phi_{j,k}(x)
\]

where \( \phi(x) \) is the wavelet function, \( \psi(x) \) is the scaling function, \( j_0 \) is an arbitrary scale, \( c_{j_0}(k)'s \) are approximation or scaling coefficients and \( d_k(k)'s \) are the detail or wavelet coefficients. If the expansion functions form an orthonormal basis, then the expansion coefficients can be calculated as

\[
c_{j_0}(k) = \langle f(x), \psi_{j_0,k}(x) \rangle = \int f(x) \psi_{j_0,k}(x)dx
\]

and

\[
d_j(k) = \langle f(x), \phi_{j,k}(x) \rangle = \int f(x) \phi_{j,k}(x)dx
\]

2.1.2 Discrete Wavelet Transform (DWT)

If the function being expanded is a sequence of numbers, like samples of a continuous function \( f(x) \), the resulting coefficients are called the discrete wavelet transform of \( f(x) \) [5], [6]. The approximation coefficients are calculated as

\[
W_\phi(j_0, k) = \frac{1}{\sqrt{M}} \sum_x f(x) \psi_{j_0,k}(x)
\]

and the detail coefficients are calculated as

\[
W_\phi(j, k) = \frac{1}{\sqrt{M}} \sum_x f(x) \phi_{j,k}(x)
\]
Any discrete function $f(x)$ can be written as

$$f(x) = \frac{1}{\sqrt{M}} \sum_{k} W_{\phi}(j_0, k) \psi_{j_0, k}(x) + \frac{1}{\sqrt{M}} \sum_{j=0}^{\infty} \sum_{k} W_{\phi}(j, k) \phi_{j, k}(x) \quad (17)$$

In two dimension, a two dimensional scaling function $\psi(x, y)$ and three two dimensional wavelets $\phi^H(x, y)$, $\phi^V(x, y)$ and $\phi^D(x, y)$ are used. These functions can be calculated using one dimensional scaling function and corresponding wavelet. The separable two dimensional scaling function can be calculated as

$$\psi(x, y) = \psi(x)\psi(y) \quad (18)$$

and the three separable wavelets can be calculated as

$$\phi^H(x, y) = \phi(x)\psi(y) \quad (19)$$

$$\phi^V(x, y) = \psi(x)\phi(y) \quad (20)$$

$$\phi^D(x, y) = \phi(x)\phi(y) \quad (21)$$

The three wavelets calculate the intensity variations for images along different directions. $\phi^H$ measures variations along columns, $\phi^V$ measures variations along rows and $\phi^D$ measures variations along diagonals. The block diagram of the two dimensional analysis filter bank is given in figure 1.
2.1.3 Adaptive Wavelet Transforms

Liu proposes a direction-adaptive lifting wavelet transform in [7]. Image is partitioned into non-overlapping blocks. The structure tensor is used to calculate the homogeneous/heterogeneous property of each block. If the block is heterogeneous directional lifting is performed along a selected optimal direction. If the block is homogeneous conventional horizontal/vertical lifting is performed. The direction and block mode for each block is also encoded.

Chang [8] proposes a direction adaptive discrete wavelet transform (DA-DWT) based on directional lifting. The direction is selected for each non-overlapping block based on Lagrangian cost function. Filtering in prediction and update steps, extends across block boundaries. To increase the efficiency of prediction each block is further partitioned into sub-blocks. The best block partition for each block and the best direction for each sub-block are selected based on a cost function. The direction selection is predicted from the selection of blocks in the causal neighborhood and the residual is coded using variable length coding techniques. DA-DWT provides an efficient representation of directional image features such as edges and lines by locally adapting filtering directions to the image content.

Directionlets proposed by Velisavljević [9] are critically sampled anisotropic basis functions with Directional Vanishing Moments across any two directions with rational slopes. In [10] a compression method based on space-frequency quantization (SFQ) using directionlets is proposed. In the SFQ the best hierarchical wavelet trees are chosen using Lagrangian optimization. The best quantizer step size is also found using Lagrangian optimization and the chosen quantizer is applied to all wavelet coefficients retained in the trees.

In [11] the adaptation of the directional wavelet bases is performed on the segments which describe the natural geometry of the image. In [12] the adaptive lifting scheme adapts the filtering directions according to the orientation of image features and the statistical properties of image signal.
2.1.4 Ridgelets

In ridgelet analysis, functions are represented using superpositions of ridge functions or by elements that are in some way related to ridge functions $r(a_1x_1 + \ldots + a_nx_n)$. These are functions of $n$ variables that are constant along hyperplanes i.e. $a_1x_1 + \ldots + a_nx_n = c$.

In two dimension, ridgelets help us to represent arbitrary bivariate functions by superpositions of elements of the form

$$a^{1/2}\psi((x_1\cos \theta + x_2\sin \theta - b)/a)$$

where $\psi$ is an oscillatory univariate function, $a > 0$ is a scale parameter, $\theta$ is an orientation parameter and $b$ is a location parameter [13], [14], [15]. These ridgelets are constant along lines $x_1 \cos \theta + x_2 \sin \theta$ and they are wavelets along the orthogonal direction. Using common $\theta$ & $b$ and different scales $a$, the singularities along a line can be efficiently approximated. Ridgelets occur at all scales, locations and orientations. Ridgelets have unit length.

Wavelet transform and ridgelet transform are linked through Radon transform. The Radon transform is given by

$$Rf(\theta, t) = \int f(x_1, x_2)\delta(x_1\cos \theta + x_2\sin \theta - t)dx_1dx_2$$

where $\delta$ is a Dirac function. The ridgelet coefficients $R_f(a, b, \theta)$ of $f$ are obtained by applying one-dimensional wavelet transform to the result of Radon transform which is given by

$$R_f(a, b, \theta) = \int Rf(\theta, t)a^{-1/2}\psi(t - b/a)dt$$

Ridgelet coefficients of an image $f$ are the wavelet coefficients of the Radon transform of $f$. The Radon transform converts an image into a representation in polar coordinates, where each point $(\theta, t)$ corresponds to the projection of original image along a line with angle $\theta$ and radius $t$. Thus significant ridgelet coefficients will be located on singularities in the Radon space, i.e. on parameters $(\theta, t)$ corresponding to significant lines in the original image.
Radon transform can be obtained by applying 1-D inverse Fourier transform to the 2-D Fourier transform restricted to radial lines going through the origin. Radon transform of digital data \( f(i_1, i_2), 0 \leq i_1, i_2 \leq n - 1 \) can be calculated using the following steps:

1) **Compute 2-D Fast Fourier Transform of** \( f \) **which may be represented using** \( \hat{f}(k_1, k_2), -n/2 \leq k_1, k_2 \leq n/2 - 1 \).

2) **Substitute** the sampled values of Fourier transform obtained on the square lattice with sampled values of \( \hat{f} \) on a polar lattice.

3) **Compute** 1-D inverse fast Fourier transform of each line, i.e., for each value of the angular parameter.

One of the ways to perform cartesian-to-polar conversion is given by Donoho in [16]. It considers the data \( \hat{f}(k_1, k_2) \) as samples of trigonometric polynomial \( F \) which is given by

\[
F(\omega_1, \omega_2) = \sum_{i_1=0}^{n-1} \sum_{i_2=0}^{n-1} f(i_1, i_2) \exp(-i(\omega_1 i_1 + \omega_2 i_2))
\]

(25)

on a square lattice; that is, with \( \hat{f}(k_1, k_2) = F((2\pi k_1/n), (2\pi k_2/n)) \) where \(-n/2 \leq k_1, k_2 < n/2\). Donoho [16] gives an algorithm for finding the values of \( F \) on the polar grid. The assignment of Cartesian points as nearest neighbors of rectopolar points happens in such a way that each Cartesian point is assigned as the nearest neighbor of at least one rectopolar point. Then 1-D wavelet transform is performed along the radial variable in Radon space. The block diagram of ridgelet transform is given in figure 2.

### 2.1.5 Curvelets

Candes and Donoho introduced curvelets which would efficiently represent objects with discontinuities along curves. Curvelets also occur at all scales, locations and orientation. Curvelets occur at all dyadic lengths and exhibit an anisotropy increasing with decreasing scale like a power law. Curvelets obey a scaling relation in which the width of a curvelet element is about the square of its length, i.e. \( \text{width} \approx \text{length}^2 \). Curvelet basis functions can be considered as local grouping of wavele basis functions so that they capture smooth discontinuity curve more efficiently.
Curvelet representation is more effective for representing objects with edges than wavelets. Candes and Donoho in [17] prove that curvelets provide optimally sparse representations of $C^2$ objects with $C^2$ edges. Curvelet transform is based on multiscale ridgelets combined with a spatial bandpass filtering. Curvelet transform can be considered as a multiscale pyramid with many directions and positions at each length scale and needle-shaped elements at fine scales. Candes and Donoho [17], [15] have developed tight frames of curvelets $(\gamma_\mu)$ obeying

$$f = \sum_\mu \langle f, \gamma_\mu \rangle \gamma_\mu$$  \hspace{1cm} (26)$$

and

$$\|f\|_{L_2(R^2)}^2 = \sum_\mu |\langle f, \gamma_\mu \rangle|^2$$  \hspace{1cm} (27)$$

Curvelets can be obtained by applying dilations, rotations and translations to a specifically shaped function $\psi$. Curvelets partition the frequency plane into dyadic coronae and unlike wavelets subpartition them into angular wedges which again display the parabolic aspect ratio. Thus curvelet transform refines scale/space representation by the orientation component.

The curvelet decomposition [17], [15] consists of the following steps:
1) **Subband decomposition:** Let $P_0, (\Delta_s, s \geq 0)$ be a set of subband filters. The image $f$ is filtered into subbands

$$f \mapsto (P_0 f, \Delta_1 f, \Delta_2 f, ...).$$

2) **Smooth partitioning:** A collection of smooth windows $w_Q(x_1, x_2)$ localized around dyadic squares

$$Q = [k_1/2^s, (k_1 + 1)/2^s) \times [k_2/2^s, (k_2 + 1)/2^s)$$

Multiplying $f$ with $w_Q$ for all $Q$ produces a smooth dissection of the image into squares. In this step windowing dissection is applied to each of the subbands isolated in the previous step of the algorithm.

$$\Delta_s \mapsto (w_Q \Delta_s f)_{Q \in Q_s},$$

where $Q_s$ is $Q(s, k_1, k_2)$ with $k_1$ and $k_2$ varying and fixed $s$.

3) **Renormalization:** For a dyadic square $Q$, let

$$(T_Q f)(x_1, x_2) = 2^s(2^s x_1 - k_1, 2^s x_2 - k_2)$$

be the operator which transports and renormalizes $f$ so that part of the input supported near $Q$ becomes the part of the output supported near $[0, 1]^2$. In this step each square of the previous stage is normalized to unit scale

$$g_Q = (T_Q)^{-1}(w_Q \Delta_s f), \quad Q \in Q_s$$

4) **Ridgelet analysis:** Orthonormal ridgelet transform is applied to each square. The orthonormal ridgelet system is a system of basis element $p_\lambda$ making an orthobasis for $L^2(R^2)$

$$\alpha_\mu = \langle g_Q, p_\lambda \rangle, \quad \mu = (Q, \lambda)$$

**2.1.6 Contourlets**

There are some problems associated with curvelet transform also. It is a block based transform, so the approximated images have blocking effects or overlapping windows. Ridgelet transform is used in curvelet transform. Ridgelet transform uses polar coordinates and thus implementing curvelet transform for digital images on a rectangular grid becomes challenging. Many interpolation techniques have been proposed [18], [19]. But the problem with these interpolation techniques is that they also introduce redundancy.

Do and Vetterli [20] proposed pyramidal Directional Filter Bank (PDFB). It overcomes the block based approach of curvelet transform by a directional filter bank. It is also known as contourlet transform. Contourlet transform is a multiscale and directional decomposition of a signal using a combination of modified Laplacian Pyramid (LP) [21], [22] and Directional Filter Bank (DFB) [20].
PDFB allows different directions at each scale. The DFB is designed to capture high frequency components. Also the LP allows subband decomposition so that low frequencies are avoided in several directional subbands. These two features help in capturing directional information.

Due to the redundancy of the LP, the contourlet transform has a redundancy factor of 4/3. So it may not be optimal for image coding. Do and Vetterli [3] show that with parabolic scaling and sufficient directional vanishing moments, contourlets achieve the optimal approximation rate for a 2D piecewise smooth functions which have twice continuously differentiable contours. Wavelets use dyadic scaling and have sufficient vanishing moments and they can approximate any twice continuously differentiable 2D functions but the efficiency of wavelets is less than that of contourlets. Contourlet transform allows for different number of directions at each scale/resolution to achieve a critical sampling.

LP is used to capture point discontinuities and DFB is used to link point discontinuities into linear structures [23]. The combination of this double filter bank is called PDFB. The result is a representation of an image using basis images as contour segments. Contourlet transform can perform any number of DFB decomposition levels to be applied at each LP level. To satisfy this property, the number of directions is doubled at every other finer scale of pyramid.

Other interesting works are (a) nonsubsampled pyramid structure and nonsubsampled directional filter banks to develop a nonsubsampled contourlet transform [24], and (b) a coding technique based on contourlet transform and multistage vector quantization [25].

2.1.7 Wedgelets

Wavelets detect isolated edge points rather than edges. Thus, the fact that edge is smooth, is not reflected adequately by wavelets. Wedgelets were proposed by Donoho [13] to approximate piecewise constant images with smooth boundaries. The images are considered as elements of function space $R^I$, where $I = \{0, ..., 2^J - 1\} \times \{0, ..., 2^J - 1\}$. 
The wedgelet approach is a two step procedure. The first step decomposes the image domain $I$ into a disjoint union of wedge-shaped sets, $I = \bigcup_{w \in P} w$. In the second step on each set $w \in P$, the image is approximated by a constant. $w$ are the elements of a fixed set $W$ of wedges. A wedgelet approximation of an image $f$ is a minimizer of the functional

$$H_{\gamma, f}(P, f_P) = \gamma |P| + ||f - f_P||_2^2$$

(34)

where $\gamma$ is the regularization parameter, $f_P$ is a function which is constant on each wedge $w \in P$ and $||.||_2$ is the $l_2$ norm on $R^I$,

$$||g||_2^2 = \sum_{x \in I} |g(x)|^2$$

(35)

The regularization parameter $\gamma$ can be considered as a scale which ranges from 0 to $\infty$. If $\gamma$ is 0, the minimizer $\hat{f}_\gamma$ is the data $f$ and if $\gamma$ is $\infty$, the minimizer $\hat{f}_\gamma$ is a constant image. The properties of the wedgelet scheme depend on the definition of set $W$ of admissible wedges. One possible choice is to take the set $Q$ of dyadic squares contained in $I$.

$$Q = \{ [2^j k, 2^j (k + 1)] \times [2^j m, 2^j (m + 1)) : 0 \leq j \leq J, 0 \leq k, m < 2^{J-j} \}$$

(36)

In [26] the authors describe a method for multiscale wedgelet representation of images and an efficient algorithm for multiscale wedgelet decomposition. A wedgelet $w$ is a function on a square $S$ that is piecewise constant on either side of a line $l$ through $S$. Let $(v_1, v_2)$ be the points where $l$ intersects the perimeter of $S$. Let $c_a$ and $c_b$ be the values of $w$ above and below $l$. Thus the wedgelet can be represented as $w(v_1, v_2, c_a, c_b)$.

Let $I$ be an image supported on $[0, 1]^2$ and $S_k \subset [0, 1]^2$, where $k = \{k_1, k_2\}$ is a dyadic block at scale $j \in Z$, $S = [k_1/2^j, (k_1 + 1)/2^j] \times [k_2/2^j, (k_2 + 1)/2^j]$, for $0 \leq k_1, k_2 < 2^j$. The wedgelet decomposition $W(I(S_j,k))$ of an image $I$ over $S_{j,k}$ is a collection of projections of $I(S_{j,k})$ onto wedgelets at a finite set of orientations $V$. For each orientation $(v_1, v_2) \in V$, $S_{j,k}$ is divided into region $R_a$, which is the
II. SURVEY OF IMAGE COMPRESSION TECHNIQUES ...

region above \( l \) and region \( R_b \), which is the region below \( l \). The profile of the wedgelet at orientation \((v_1, v_2)\) is given by averaging \( I(S_{j,k}) \) in these regions:

\[
\hat{c}_a = \text{Average}(I(S_{j,k})/R_a)
\]

\[
\hat{c}_b = \text{Average}(I(S_{j,k})/R_b)
\]

The collection of wedgelets is given by

\[
W(I(S_{j,k})) = \{w_{(v_1,v_2,c_a,c_b)} : (v_1,v_2) \in V\}
\]

The multiscale wedgelet decomposition \( W^J(I) \) of an image \( I \) is the collection of wedgelet decompositions \( W(I(S_{j,k})) \) for all dyadic squares \( S_{j,k} \),

\[
W^J(I) = \{W(I(S_{j,k})) : j = 0, \ldots, J, \text{ and } k_1, k_2 = 0, \ldots, 2^J\}
\]

A multiscale wedgelet representation \( W \) consists of a set of dyadic squares that partition \([0,1]^2\), each supporting one wedgelet. Given an image, the choice of multiscale wedgelet representation can be optimized for approximation or parsimony or geometry. The authors also describe ways to incorporate geometry in the selection of multiscale wedgelet representation. They use the quadtree structure of Markov model to capture the orientation of the best wedgelet fit in the corresponding dyadic square. They find an exact solution to

\[
\min_W D(W) + \lambda[-\log_2 P(W) + |W|]
\]

where \( D(W) \) is the mean square error between image \( I \) and the multiscale wedgelet representation \( W \), \(|W|\) is the number of terms in \( W \) and \( P(W) \) is the likelihood of \( W \) under the geometry model.

2.1.8 Bandlets

The bandlet decomposition [27], [28], [29], [30] is computed with a geometric orthogonal transform that is applied on orthogonal wavelet coefficients. Wavelet transform, when applied to an image of \( N \) pixels, computes the set of \( N \) dot products

\[
<f, \psi_{jn}^s> \text{ for } 2^{-J} \leq 2^{-j} < \sqrt{N} \text{ and } 0 \leq n_1, n_2 < 2^{-j},
\]
where the projection on $\phi_{j,n}^m$ functions produces a coarse approximation at scale $2^j$. The scale $2^j$ represents the level at which the wavelet transform is stopped. Those values can be conveniently stored in an array of $N$ pixels.

A dyadic square is a square obtained by recursively splitting the original wavelet transformed image $f_j$ into four sub-squares of equal size. Let the width of the squares be $L$ pixels with $4 \leq L \leq 2^{-j/2}$. For each dyadic square $S$ at a given scale $2^j$ and orientation $s$ of the wavelet transform, 1D reordering of the grid points is performed. The possible number of 1D reordering may be equal to the number of directions $d$ joining pairs of points in square $S$ of width $L$. 1D reordering is done by projecting the sampling location along $d$ and sorting the resulting 1D points from left to right.

To the resulting 1D discrete signal, $f_d$, 1D discrete wavelet transform is performed. For a given threshold $T$, the direction $d$, which generated the less approximation error, is selected. Let $b_k$ denote the coefficients of 1D wavelet transform of $f_d$, and $R_B$ be the number of bits needed to code the quantized coefficients $QT(b_k)$. To select the best geometry, the direction $d$ that minimizes the Lagrangian

$$\xi(f_d, R) = \|f_d - f_d R\|^2 - \lambda T_2(R_G - R_B),$$

where $f_d R$ is the signal recovered from the quantized coefficients and $R_G$ is the number of bits needed to code the geometric parameter $d$ with an entropy coder. $\lambda$ is taken as $3/28$ [31].

2.2. QUALITY METRICS FOR COMPRESSION PERFORMANCE

A quality metric which is used for both lossy and lossless compression methods is compression ratio. Quality metrics are important for lossy compression methods since the reconstructed image differs from the original one and it is needed to measure the quality of the reconstructed image.
The quality metrics are divided into two categories: subjective and objective. Subjective measures are based on the Human Visual System (HVS). Most widely used subjective image quality metric is Mean Observer Score (MOS). The objective quality measures are based on the statistics of the original and reconstructed image. Some of them consider the features of HVS. Some objective quality metrics are Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Signal to Noise Ratio (SNR), Peak Signal to Noise Ratio (PSNR), Universal image Quality Index (UQI), Multidimensional Quality Measure using SVD and Structural SIMilarity index (SSIM).

Apart from these metrics some metrics like coding delay and coding complexity are used to determine the quality of the compression method. Let the original and reconstructed images be represented by \( P \) and \( Q \) respectively, the pixels of original image be represented by \( P_i \) and the pixels of the reconstructed image be represented by \( Q_i \) respectively where \( 1 \leq i \leq N \) and \( N \) is the number of pixels.

### A. Compression Ratio

The compression ratio \([1]\) is defined as

\[
\text{Compression ratio} = \frac{\text{size of input stream}}{\text{size of compressed stream}}
\]

A greater value of compression ratio indicates better compression. Other terms associated with compression ratio are bits-per-pixel or bit-rate which is the number of bits used to represent a pixel.

### B. Mean Observer Score (MOS)

It is also called as Mean Opinion Score. One of the ways to compute it is as follows. A good number of observers are chosen to evaluate the visual quality of the reconstructed image. Each observer assigns a number to the reconstructed image to describe the quality of the image, say 1 being worst and 10 being best. The average of the scores assigned by all the observers to the reconstructed image is called the MOS \([32]\).
C. Mean Squared Error (MSE)

It is the average of the square of the pixel differences of the original and the reconstructed image. MSE [33] is given by

\[
MSE = \frac{1}{N} \sum_{i=1}^{N} (P_i - Q_i)^2
\]  
(46)

D. Root Mean Squared Error (RMSE)

It is defined as the square root of MSE which is given by

\[
RMSE = \left( \frac{1}{N} \sum_{i=1}^{N} (P_i - Q_i)^2 \right)^{\frac{1}{2}}
\]  
(47)

E. Signal to Noise Ratio (SNR)

SNR [4] is given by

\[
SNR = 20 \log_{10} \sqrt{\frac{\frac{1}{N} \sum_{i=1}^{N} P_i^2}{RMSE}}
\]  
(48)

F. Peak Signal to Noise Ratio (PSNR)

PSNR is defined as

\[
PSNR = 20 \log_{10} \frac{\max_{i} |P_i|}{RMSE}
\]  
(49)

For grayscale image with eight bits per pixel, \( \max_{i} |P_i| \) is 255. A larger value of PSNR implies greater resemblance between original and reconstructed image.

G. Universal image Quality Index (UQI)

It is proposed in [34]. It gives the distortion in the reconstructed image as a combination of loss of correlation, mean distortion and variance distortion. It is given by

\[
UQI = \frac{\sigma_{PQ}}{\sigma_P \sigma_Q} \cdot \frac{2\overline{PQ}}{(\overline{P})^2 + (\overline{Q})^2} \cdot \frac{2\sigma_P \sigma_Q}{\sigma_P^2 + \sigma_Q^2}
\]  
(50)

where

\[\overline{P} = \frac{1}{N} \sum_{i=1}^{N} P_i\] is the mean value of \( P \),
\[ \bar{Q} = \frac{1}{N} \sum_{i=1}^{N} Q_i \] is the mean value of \( Q \),

\[ \sigma_P^2 = \frac{1}{N-1} \sum_{i=1}^{N} (P_i - \bar{P})^2 \] is the variance of \( P \),

\[ \sigma_Q^2 = \frac{1}{N-1} \sum_{i=1}^{N} (Q_i - \bar{Q})^2 \] is the variance of \( Q \),

\[ \sigma_{PQ} = \frac{1}{N-1} \sum_{i=1}^{N} (P_i - \bar{P})(Q_i - \bar{Q}) \] is the covariance between \( P \) and \( Q \).

The first term is the linear correlation coefficient between \( P \) and \( Q \) which is a measure of loss of correlation. The second term is the mean distortion, i.e., how close the mean values of \( P \) and \( Q \) are. The third term is the variance distortion.

**H. Multidimensional Quality Measure using Singular Value Decomposition (SVD)**

It is proposed in [35]. Let a matrix \( A \) be decomposed into 3 matrices \( A = U S V^T \) where \( U^T U = I \), \( V^TV = I \) and \( S = \text{diag}(s_1, s_2, ...) \). The diagonal values of \( S \) are called as singular values of \( A \), the columns of \( U \) are called left singular vectors of \( A \) and the columns of \( V \) are called the right singular vectors of \( A \). If the SVD is applied to full images, a global measure is obtained. The measure is the distance between the singular values of original image block and singular values of the reconstructed image block. It is given as

\[ D_i = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \]  

(51)

where \( p_i \) are the singular values of original block, \( q_i \) are the singular values of reconstructed block and \( n \) is the block size.

**I. Structural SIMilarity index (SSIM)**

It is proposed in [36] and is a generalized form of UQI. It is defined as

\[ SSIM = [l(P, Q)]^\alpha \cdot [c(P, Q)]^\beta \cdot [s(P, Q)]^\gamma \]  

(52)

where

\[ l(P, Q) = \frac{2\mu_P\mu_Q+C_1}{\mu_P^2+\mu_Q^2+C_1}, \quad c(P, Q) = \frac{2\sigma_P\sigma_Q+C_2}{\sigma_P^2+\sigma_Q^2+C_2}, \quad s(P, Q) = \frac{\mu_P\sigma_Q+C_3}{\mu_Q\sigma_P+C_3}, \quad C_1 = (K_1 L)^2, \quad C_2 = (K_2 L)^2, \quad C_3 = C_2. \]

Here \( L \) is the dynamic range of pixel values, \( K_1 \leq 1, K_2 \leq 1 \). \( \alpha, \beta \) and \( \gamma \) determine the importance of the three components \( l, c, s \).
J. Coding Delay

It is a measure of the time required for the encoding or decoding algorithms to execute. This measure of compression algorithms is used where interactive encoding and decoding is required like interactive video teleconferencing, online image browsing, real time video communication etc. Algorithms must be designed in such a manner that they provide higher compression ratios along with lesser coding delay.

K. Coding Complexity

It is a useful performance measure where the computational requirement to implement the coder is an important criteria. The computational requirement is measured in terms of number of arithmetic operations and memory requirements.

2.3. LOSSLESS COMPRESSION

2.3.1 Run-Length Encoding

Run-length coding is based on Capon model [37]. The image is scanned row by row and the runs of pixels of same color are found. Each run of pixels of same gray level is encoded as two values, the run length and the pixel value. So instead of coding the color of each pixel individually, the length of the runs of each color is coded. Lossy run-length encoding can also be performed by ignoring the short runs. The maximum length of the run which can be ignored can be obtained by the user. One of the image file formats, BMP, uses run-length encoding to compress images with 4 or 8 bitplanes.

2.3.2 Huffman Coding

Huffman coding technique is a very popular technique developed by David Huffman [38]. A list of all the symbols in descending order of their probabilities is generated. A tree is then constructed from bottom up with a symbol at every leaf. At each step two symbols with smallest probabilities are selected and added to
II. SURVEY OF IMAGE COMPRESSION TECHNIQUES

Symbol Probability

<table>
<thead>
<tr>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
</tr>
<tr>
<td>$a_2$</td>
</tr>
<tr>
<td>$a_3$</td>
</tr>
<tr>
<td>$a_4$</td>
</tr>
<tr>
<td>$a_5$</td>
</tr>
<tr>
<td>0.5</td>
</tr>
<tr>
<td>0.2</td>
</tr>
<tr>
<td>0.2</td>
</tr>
<tr>
<td>0.05</td>
</tr>
<tr>
<td>0.05</td>
</tr>
</tbody>
</table>

Fig. 3. Huffman source reduction

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Probability</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>$a_2$</td>
<td>0.2</td>
<td>10</td>
</tr>
<tr>
<td>$a_3$</td>
<td>0.2</td>
<td>100</td>
</tr>
<tr>
<td>$a_4$</td>
<td>0.05</td>
<td>1010</td>
</tr>
<tr>
<td>$a_5$</td>
<td>0.05</td>
<td>1011</td>
</tr>
</tbody>
</table>

Fig. 4. Procedure for Huffman code assignment

the top of the partial tree with an auxiliary symbol representing both of them. The selected symbols are deleted from the list. The construction of the tree is complete when the list is reduced to one auxiliary symbol. The tree is then traversed to find the codes of the symbols.

Let us consider a five letter alphabet $A = \{a_1, a_2, a_3, a_4, a_5\}$ with probabilities $P(a_1) = 0.5, P(a_2) = 0.2, P(a_3) = 0.2, P(a_4) = 0.05, P(a_5) = 0.05$. The first step in Huffman coding is arranging the symbols in the descending order of probabilities and combining the lowest probability symbols to a single symbol that replaces them in the next source reduction. Figure 3 illustrates this process. The second step is to code each reduced source starting with the smallest source and proceeding towards the original source to obtain the code for each symbol. This step is illustrated in figure 4.

Huffman coding assumes that the probabilities of the symbols are known. If the probabilities are not known, in the first pass the probabilities are computed and in the second pass encoding is performed. Such a process is practically slow. To improve this, Faller [39] and Gallagher [40] developed the adaptive Huffman coding procedure which are based on the probabilities of the symbols already
encountered. These algorithms were later improved by Knuth [41] and Vitter [42]. In adaptive Huffman coding the compression starts with an empty Huffman tree. The first symbol encountered is written to the output stream in its original form. The symbol is added to the tree and a code is assigned. If the symbol is encountered again the code is written and the frequency is incremented by one. The tree is updated to ensure that it is still a Huffman tree with best codes.

2.3.3 Facsimile Compression

The modified Huffman (MH) coding is a compression technique specified in the Recommendation T.4 [43]. Separate tables containing code words for black and white runs are used. The run-length $r$ is expressed as $64 \times m + t$, where $m$ ranges from 1 to 27 and $t$ ranges from 0 to 63. $m$ is called the make-up code and $t$ is called the terminating code. Terminating code is used if the run-length is less than 63. Both terminating and make-up codes are used for runs from 64 to 1728. The first run of every line is considered to be white. If first pixel is black, a run-length 0 for white is assumed [43], [44], [45], [4].

The modified Relative Element Address Designate (MR) coding scheme, is a two-dimensional scheme for coding binary images [43]. In this technique instead of coding the run-lengths, the location of changing pixel on the coding line is coded. The position of the changing pixel is considered either with respect to previous line or with respect to the preceding changing pixel on the same line [43], [44], [45], [46].

In the two-dimensional scheme encoding a line is done based on the previous line, so error in one line can be propagated to all other lines. To restrict this propagation of errors, T.4 recommends that after coding each line based on one-dimensional coding, $K-1$ remaining lines from a block of $K$ lines must be coded based on the previous line. This requirement reduces compression. To maximize compression this requirement was removed in the Recommendation T.6 [47]. The modified version of MR is the modified modified READ (MMR) coding scheme which does not have the one-dimensional coding scheme [47], [44], [45], [4].
2.3.4 Arithmetic Coding

Arithmetic coding was first proposed by Peter Elias and was described by Abramson in his book on Information Theory [48]. In arithmetic coding the entire input file is assigned with one code. The method starts with a certain interval. The input file is read symbol by symbol and the probability of symbols is used to narrow the interval. A narrow interval requires more bits to represent it. A high probability symbol narrows the interval less than a low probability symbol. Arithmetic coding is more efficient than Huffman coding for alphabets with highly skewed probabilities [4].

Let us consider a three letter alphabet $A = a_1, a_2, a_3$ with probabilities $P(a_1) = 0.4, P(a_2) = 0.3, P(a_3) = 0.3$. The initial subintervals are $[0.0, 0.4), [0.4, 0.7), [0.7, 1.0)$. If the first symbol encountered is $a_2$ then the subinterval $[0.4, 0.7)$ is considered. If the next two symbols are $a_1$ and $a_3$, the intervals get narrowed. This process is illustrated in figure 5. Any number within the final range $[0.484, 0.52)$ can be used to represent the sequence.

The arithmetic coder of Joint Photographic Experts Group (JPEG) is called the QM-coder [49], [50]. The input symbols are single bits. It uses a statistical model to classify the bits as more probable symbol (MPS) and less probable symbol.
(LPS). If the statistical model predicts that 0 is more probable, but the actual bit input is 1 then it is classified as LPS. The decoder also uses the same statistical model to determine the relation between 0/1. The probabilities for LPS ($Q_e$) and MPS ($1 - Q_e$) are computed. The interval $[0, A)$ is divided by placing LPS interval whose size is $A * Q_e$ above MPS interval whose size is $A(1 - Q_e)$. Let $C$ be the output string. After MPS, $C$ is unchanged and $A$ is set as $A(1 - Q_e)$. After LPS, $C$ is set to $C + A(1 - Q_e)$ and $A$ is set as $A * Q_e$.

2.3.5 Limpel-Zev-Welch (LZW) Coding

LZW was developed by Terry Welch [51], [52]. For 8-bit symbols the first 256 entries of the dictionary are 0-255. The encoder reads the first symbol $x$. The symbol is found in the dictionary. The next symbol, say $S$, is input and encoder searches for $xS$ and is not found in the dictionary. The index of $x$ is sent and $xS$ is added as the last entry. $I$ is initialized to $S$. The next symbol $y$ is input. $xS$ is present in the dictionary, so index of $xS$ is sent and $I$ is set to $y$. This process continues till the entire string is scanned.

Application which uses LZW are Graphics Interchange Format (GIF) and v.42 bis. GIF was developed by CompuServe Information Service in 1987. It is a graphics file format which uses a variant of LZW [53], [54]. The International Telegraph and Telephone Consultative Committee (CCITT) Recommendation v.42 bis is a compression standard used in modems connecting computers in a network [55].

2.3.6 Prediction with Partial Match (PPM)

It was developed by J. Cleary and I. Witten [56] which was implemented and extended by A. Moffat [57]. It is one of the context based methods. If the next symbol is $S$ and $N$ previous symbols are considered, the context of $S$ is called as order-$N$ context $C$. The PPM starts with an order-$N$ context $C$. It searches for $C$ followed by next symbol $S$. It looks at $C$ and based on the input data previously seen, it determines the probability $P$ that $S$ will appear for a context
C. The encoder then invokes an adaptive arithmetic coding algorithm to encode $S$ with probability $P$. If there is no occurrence it switches to order $N - 1$ context and repeats the same procedure.

2.3.7 Context based Adaptive Lossless Image Compression (CALIC)

CALIC [58], [59] uses both context and prediction of pixel values. The original image $I$ has $M$ rows and $N$ columns of pixels. The first pass computes $M/2 \times N/2$ values

$$a[i, j] = (I[2i, 2j] + I[2i + 1, 2j + 1])/2$$

where $0 \leq i < M/2$ and $0 \leq j < N/2$. Each of the pixel in the $M/2 \times N/2$ image is predicted from four of its neighbors. The predicted value is given by

$$a' = \frac{1}{2}a[i - 1, j] + \frac{1}{2}a[i, j - 1] + \frac{1}{4}a[i - 1, j + 1] - \frac{1}{4}a[i - 1, j - 1]$$

The error value $a - a'$ is encoded. In the second pass each of the pixels in $M/2 \times N/2$ image is predicted using five known neighbors, one above it, one to its left and three averages $a$ which are below and to its right from the first pass. So if $a = I[2i, 2j]$ represents the current pixel, its prediction is given by

$$a' = 0.9a[i, j] + \frac{1}{6}(I[2i + 1, 2j - 1] + I[2i - 1, 2j - 1] +$$

$$I[2i - 1, 2j + 1]) - 0.05(I[2i, 2j - 2] + I[2i - 2, 2j])$$

$$-0.15(a[i, j + 1] + a[i + 1, j])$$

The encoder then encodes $a - a'$ for every pixel. The third pass predicts the pixels $I[2i, 2j + 1]$ and $I[2i + 1, 2j]$ where $0 \leq i < M/2$ and $0 \leq j < N/2$. Each pixel is predicted from six of its neighbors by

$$a' = \frac{3}{8}(I[i, j - 1] + I[i - 1, j] + I[i, j + 1] + I[i + 1, j])$$

$$-\frac{1}{4}(I[i - 1, j - 1] + I[i - 1, j + 1])$$

2.3.8 Joint Photographic Experts Group - Lossless (JPEG-LS)

JPEG-LS [60] is also based on context based predictive coding. A simplified block diagram for JPEG-LS encoder and the causal template used for modeling and prediction is shown in figure 6.
JPEG-LS consists of two distinct parts: modeling and coding. In the causal template $x$ is the current pixel and $a, b, c$ and $d$ are neighboring pixels. The context modeling in JPEG-LS is based on two-sided geometric distribution model. The context determines whether $x$ should be encoded in regular mode (when the neighboring pixels are not identical) or in run mode (when the neighboring pixels are identical). In the regular mode the predictor guesses the value of current pixel $x'$ as

$$x' = \begin{cases} 
\min(a, b) & \text{if } c \geq \max(a, b) \\
\max(a, b) & \text{if } c \leq \min(a, b) \\
a + b - c & \text{otherwise}
\end{cases}$$

The current pixel is predicted to be 'b' if a vertical edge exists left of the current location, 'a' if a horizontal edge is above the current location or $a + b - c$ if no edge is detected. The prediction error is the difference between actual sample value and its predicted value.

Finally, the prediction errors are encoded using adaptively selected codes based on Golomb codes. In the run mode the encoder looks, starting from $x$, for a sequence of consecutive pixels with values identical to the reconstructed value at $a$. The length information is then encoded.
2.3.9 Differential Pulse Code Modulation (DPCM)

Differential encoding techniques capture the correlation between neighboring pixels. They calculate the differences \( d_i = a_i - a_{i-1} \), between pixels \( a_i \) and encode them. The first pixel \( a_0 \) is either encoded or written as such. In lossy compression rather than encoding \( d_i \) the quantized value \( d'_i \) is encoded. This leads to accumulation of errors. To overcome this problem the encoder calculates the differences \( d_i = a_i - a'_{i-1} \) and encodes them. The data items are correlated. Thus prediction of \( a_i \) is done based on \( N \) previously seen neighbors. DPCM is a technique which uses prediction of pixel \( S \). The most commonly used predictor is a linear predictor given by

\[
P_i = \sum_{j=1}^{N} w_j a_{i-j}
\]  

(60)

where \( w_j \) are the weights and \( a_i \) is the value of current pixel which is predicted. Figure 7 illustrates a typical DPCM encoder and decoder.
2.4. LOSSY COMPRESSION

2.4.1 Scalar Quantization

Scalar quantization converts large numbers to small numbers with the help of operations like rounding off or truncation etc. These small numbers occupy less space. The space between these output values is called bin width ($\Delta$). If the subband coefficients have to be scalar quantized, a quantizer with bin width $\Delta$ is used to quantize all the coefficients [62]. A coefficient lying in a particular interval is represented by its midpoint. The coefficient is encoded as an index which represents the interval. As the size of $\Delta$ increases, the distortion increases and entropy decreases so that the compression is higher. As $\Delta$ decreases, distortion decreases and the compression is less. Scalar quantization is not suitable for image compression as it may create annoying artifacts.

2.4.2 Vector Quantization

Instead of quantizing individual pixels, if groups of pixels are quantized the distortion is much less [64]. Thus vector quantization is preferred over scalar quantization. The image is divided into equal size blocks (vectors) of pixels. A list of same size blocks (codebook) is maintained by the encoder. Each image block $A$, is compared with all the blocks in the codebook and the one which is closest to $A$ is found. If the closest block is $B$ then the pointer to $B$ is written to the compressed stream. If the pointer is smaller than the block, compression is achieved. When the decoder obtains the pointer, it fetches the corresponding block from the codebook and puts them together to obtain the reconstructed image. If a pixel has $k$ bits, then each block is $4k$ bits and the number of different blocks are $2^{4k}$. If a block of $4k$ bits is replaced with a $4k$ bit pointer, no compression is achieved.

To overcome this problem a slightly modified version of vector quantization is developed. The method starts with an empty codebook. The image is scanned block by block, the codebook is searched for each block. If the block is available in the codebook the encoder writes the pointer, otherwise the block is added to
Lossy vector quantization can be achieved by analyzing a large number of different images and finding the $B$ most common blocks. Measures have to be used to match image blocks to codebook entries. Let $B = (b_1, b_2, ..., b_n)$ and $C = (c_1, c_2, ..., c_n)$ be a block of image pixels and a codebook entry respectively. The distance between them $d(B, c)$ can be measured in three different ways:

\[ d_1(B, C) = \sum_{i=0}^{n} |b_i - c_i| \]  
\[ d_2(B, C) = \sum_{i=0}^{n} (|b_i - c_i|)^2 \]  
\[ d_3(B, C) = \max_{i=0}^{n} (|b_i - c_i|) \]

The measure $d_1(B, C)$ gives the distance between $B$ and $C$. The measure $d_2(B, C)$ gives the Euclidean distance between them. The measure $d_3(B, C)$ finds the component where $B$ and $C$ differ most and returns this difference. If this codebook is used for any general image then the distortion would be more. To solve this problem the codebook entries must be adapted to the image being compressed. LBG [63] is an example for such an algorithm. It iteratively improves a codebook by using a minimum distortion or nearest neighbor mapping.

2.4.3 Entropy-Constrained Vector Quantization

If the outputs of a vector quantizer are entropy coded then a quantizer which minimizes the average distortion subject to a constraint on the quantizer output entropy must be used. One such algorithm is described in [65]. The Entropy-Constrained VQ can improve rate distortion performance when a VQ is followed by an entropy coder [66].

2.4.4 Tree-Structured Vector Quantization

A binary decision tree is built with label of each node representing the binary minimum distortion decision that is compared with the input vector. The binary sequence from the root node of the tree through the terminal node while comparing for an input vector represents its codeword. Based on the indexes of the codewords
the tree can be balanced leading to fixed length code or unbalanced leading to variable length code [67]. Certain output points can be pruned from the codebook. This reduces the size of the codebook and the rate. Pruning also takes care that minimum distortion occurs for a certain rate.

Chou, Lookabaugh and Gray [68] developed the generalized Breiman, Friedman, Olshen and Stone (generalized BFOS) algorithm which is an optimal pruning algorithm. Such vector quantizers are called as pruned tree-structured vector quantizers.

2.4.5 Lattice Vector Quantization

The arrangements of output points are called lattices. Let $a_1, a_2, \ldots, a_L$ be $L$ independent $L$-dimensional vectors. Then the set

$$L = x : x = \sum_{i=1}^{L} u_i a_i$$

is a lattice if $u_i$ are all integers. A vector quantizer is called lattice vector quantizer if a subset of lattice points is used as the output points.

Conway and Sloane [69], [70] have described the best lattices for various dimensions. As the dimension $L$ of the vector increases, almost all the vectors lie on a contour of constant probability given by $-h(X)$, where $X$ is a random variable and $h(X)$ is the differential entropy. To encode such a source Fischer developed pyramid vector quantizer [71]. The quantizer consists of points of rectangular quantizer that are on the hyperpyramid given by

$$\sum_{i=1}^{L} |x_i| = C$$

where $C$ is a constant depending on the variance of the input. This value is quantized and sent to the receiver. The input is normalized by this term and quantized using a single hyperpyramid. The output point on the hyperpyramid closest to the scaled input is found and the binary codeword of this output point is found.
2.4.6 Trellis-Coded Quantization (TCQ)

The Trellis-Coded Quantizer can be considered to have a super codebook containing $2^{R+1}$ reconstruction levels and a set of sub codebooks having $2^R$ reconstruction levels where $R$ is the number of bits per sample used by TCQ. The encoder has a collection of possible state transition between them, where a transition corresponds to one of the sub codebooks. For a given input vector the encoder looks at all of the available sub codebooks and finds the closest one. The information is encoded using a two part binary vector, where the first part is the index of transition and the second part is the index of the possible codeword in the codebook chosen [72], [73].

2.5. IMAGE COMPRESSION USING SOFT COMPUTING

2.5.1 Neural Network Techniques

The adaptable nature of Artificial Neural Networks makes them suitable for compression of images [74]. Parallel architecture, ability to self organize, and a high degree of connectivity of Neural Networks make them suitable for Image Compression, especially when the data is large or when the data is incomplete [75].

A. Back-Propagation Neural Network

One of the Neural Networks most popularly used for image compression is the Back-Propagation Neural Network [75]. The Back-Propagation Neural Network is designed to have one input layer, one hidden layer and one output layer with the number of neurons in the hidden layer less than the number of neurons in the input layer or the output layer. The image is divided into blocks and the pixels in one block are given to the input layer. The Back-Propagation Neural Network is trained to scale the inputs through the hidden layer and to generate an optimum value at the output layer which is nearest to the inputs. A typical Back-propagation neural network is shown in figure 8. Input vector has $N$ pixels from each block. The coupling weights between input layer and hidden layer are represented by $w_{ij}$.
and the weights between hidden layer and output layer are represented by $w'_{ij}$. In case of linear network,

$$h_j = \sum_{i=1}^{N} w_{ji} x_i$$

is used for encoding where $1 \leq j \leq N$ and

$$\bar{x}_i = \sum_{j=1}^{K} w'_{ij} h_j$$

is used for decoding where $1 \leq i \leq N$.

In the Hierarchical Back-Propagation Neural Network there are three hidden layers [76]. The inner hidden layer is trained to exploit the correlation between the pixels in one block and the outer hidden layers are trained to exploit the correlation between the blocks of the image. Adaptive Back-Propagation Neural Network is designed to compress image blocks with different measures of complexity with different Neural Networks [77].

**B. Image Compression based on Hebbian Learning**

Neural Networks based on Hebbian Learning rules are developed to directly extract the principal components from the image blocks for compression [75], [78]. Hebbian Learning rule based Neural Networks have one input layer and
one output layer with less number of neurons than the Back-Propagation Neural Network.

C. Vector Quantization Neural Network

In Vector Quantization, an image is divided into blocks or vectors and these vectors are represented by closest code-vector selected from a set of code-vectors called the codebook. A number of learning algorithms using Neural Networks have been developed for Vector Quantization [79]. The Vector Quantization Neural Networks are designed to have one input layer and one output layer with each neuron in the output layer trained to one code-vector in the codebook. A typical Vector Quantization neural network is shown in figure 9. The $M$ neurons compute the vector quantization codebook in which each neuron is a codeword based on its coupling weights. The coupling weight $w_{ij}$ associated with $i^{th}$ neuron is trained to represent the codeword $c_i$.

Kohonen-Self Organizing Feature Networks are designed to implement Vector Quantization which overcome the under-utilization problem by updating the weights of the winning neuron as well as the weights of the neurons in the neighborhood of the winning neuron [80].

Neural Networks have been designed for non-linear prediction in the Dif-
ferential Pulse Code Modulation Techniques [81]. These networks remove redundant information existing between neighboring pixels. Neural Networks have been designed to implement Predictive Vector Quantization which combines the advantages of Vector Quantization and Differential Pulse Code Modulation [82]. Various open loop and closed loop methods for the design of Predictive Vector Quantization scheme have been proposed and the design with modified closed loop with parametric predictor or with neural predictor works well [83]. Asymptotic closed loop method for Predictive Vector Quantizer design has also been proposed which combines the advantages of both closed loop and open loop designs [84].

2.5.2 Fuzzy Logic Techniques

The fuzzy nature of image features allows us to use Fuzzy Logic to quantify the image data appropriately [85]. Thus Fuzzy techniques have been used for Image Compression, especially during the design of codebook in Vector Quantization [86]. Fuzzy Vector Quantization Neural Networks are designed to utilize the fuzziness of the vectors during early stages of training [87]. Various algorithms have been developed to increase the efficiency of Vector Quantization [86], [88], [89].

Fuzzy competitive algorithms have been developed to assign vectors to clusters during the training phase of the codebook design in Vector Quantization [90]. Fuzzy algorithms have been developed for Learning Vector Quantization [91]. By minimizing the objective function in the Entropy Constrained Fuzzy Clustering Algorithm, the Entropy Constrained Learning Vector Quantization algorithms were developed [92].

2.6. EXPERIMENTS AND RESULTS

2.6.1 Lossless Compression of WBMP Images

Wireless application protocol BitMaP (WBMP) image file format which contains only graphical information is used by a variety of Wireless Application Protocol devices especially mobile phones [61].
This subsection deals with the compression of WBMP images using lossless compression algorithms like RLE, Huffman coding, LZW coding, MH coding, MR coding and MMR coding. In this study 12 WBMP images as shown in Figure 10 are used. Table I shows the compression ratios of the above said images for various algorithms. The plot of compression ratios of various compression algorithms for the WBMP images are shown in Figure 11. This clearly indicates that MMR coding and MR coding techniques perform much better than RLE, Huffman, LZW and MH coding techniques for WBMP images.

2.6.2 Performance of Image Transforms for Compression

In this subsection, the performance of image transforms such as wavelet transform, curvelet transform, contourlet transform and bandlet transform for a set of three benchmarking grayscale images, each of dimension 512*512, is evaluated.

For the wavelet transform Daubechies wavelet is used. The curvelet transform is based on the wrapping implementation. The contourlet transform uses PDFB with ‘9-7’ filter for the pyramidal decomposition and ‘pkva’ filter for the direction
II. SURVEY OF IMAGE COMPRESSION TECHNIQUES

TABLE I

<table>
<thead>
<tr>
<th>Image</th>
<th>Compression Ratio</th>
<th>Original Size(Bytes)</th>
<th>RLE</th>
<th>Huffman</th>
<th>LZW</th>
<th>MH</th>
<th>MR</th>
<th>MMR</th>
</tr>
</thead>
<tbody>
<tr>
<td>medcom1</td>
<td>406</td>
<td>1.49</td>
<td>0.92</td>
<td>1.60</td>
<td>1.23</td>
<td>1.86</td>
<td>2.22</td>
<td></td>
</tr>
<tr>
<td>elm26</td>
<td>361</td>
<td>1.25</td>
<td>0.96</td>
<td>1.61</td>
<td>1.03</td>
<td>2.14</td>
<td>3.17</td>
<td></td>
</tr>
<tr>
<td>cartpark38</td>
<td>361</td>
<td>1.98</td>
<td>0.90</td>
<td>1.64</td>
<td>1.58</td>
<td>2.96</td>
<td>3.80</td>
<td></td>
</tr>
<tr>
<td>cartwik158</td>
<td>316</td>
<td>1.02</td>
<td>0.98</td>
<td>1.36</td>
<td>0.92</td>
<td>1.38</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>cartwik16</td>
<td>225</td>
<td>1.11</td>
<td>0.99</td>
<td>1.48</td>
<td>0.98</td>
<td>1.50</td>
<td>1.91</td>
<td></td>
</tr>
<tr>
<td>cartobj42</td>
<td>496</td>
<td>1.82</td>
<td>0.93</td>
<td>1.82</td>
<td>1.54</td>
<td>2.12</td>
<td>2.41</td>
<td></td>
</tr>
<tr>
<td>birds8</td>
<td>271</td>
<td>1.16</td>
<td>0.98</td>
<td>1.30</td>
<td>0.96</td>
<td>1.46</td>
<td>1.79</td>
<td></td>
</tr>
<tr>
<td>eyes</td>
<td>126</td>
<td>2.17</td>
<td>1.00</td>
<td>1.52</td>
<td>1.27</td>
<td>2.10</td>
<td>2.68</td>
<td></td>
</tr>
<tr>
<td>finger</td>
<td>93</td>
<td>1.19</td>
<td>0.89</td>
<td>1.24</td>
<td>0.91</td>
<td>1.26</td>
<td>1.41</td>
<td></td>
</tr>
<tr>
<td>grump</td>
<td>139</td>
<td>0.58</td>
<td>0.82</td>
<td>0.82</td>
<td>0.46</td>
<td>0.87</td>
<td>1.19</td>
<td></td>
</tr>
<tr>
<td>cartobj3</td>
<td>271</td>
<td>0.97</td>
<td>0.94</td>
<td>1.17</td>
<td>0.82</td>
<td>1.18</td>
<td>1.36</td>
<td></td>
</tr>
<tr>
<td>birds25</td>
<td>181</td>
<td>0.56</td>
<td>0.88</td>
<td>0.90</td>
<td>0.51</td>
<td>0.69</td>
<td>0.77</td>
<td></td>
</tr>
</tbody>
</table>

decomposition stage with 5 levels of pyramidal decomposition. The bandlet transform is based on the ‘CDF’ biorthogonal filter.

For all the transforms only a certain percentage of largest or significant coefficients were retained and the images were reconstructed. The experimental results are given in table II. The results for lena, boat, barbara and fingerprint are plotted in figures 12, 13, 14 and 15 respectively. It can be seen from the table and figures that, bandlet and contourlet transforms perform better than others.
Fig. 11. Compression Ratios for images (medcom1, elm26, cartpop38, cartwild58, cartwild6, cartobj42, birds8, eyes, finger, grump, cartobj3, birds25) for various lossless compression techniques

TABLE II
MSE AND PSNR FOR WAVELET, CURVELET, CONTOURLET AND BANDLET TRANSFORMS

<table>
<thead>
<tr>
<th>Image</th>
<th>% of Coefficients Retained</th>
<th>Wavelet Transform</th>
<th>Curvelet Transform</th>
<th>Contourlet Transform</th>
<th>Bandlet Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>90</td>
<td>MSE 0.0109</td>
<td>0.0119</td>
<td>0.0069</td>
<td>0.0053</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 67.76</td>
<td>67.38</td>
<td>69.73</td>
<td>71.13</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>MSE 0.2554</td>
<td>0.2152</td>
<td>0.1934</td>
<td>0.1516</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 54.06</td>
<td>54.80</td>
<td>55.27</td>
<td>56.33</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>MSE 1.5162</td>
<td>1.1240</td>
<td>0.9830</td>
<td>0.8151</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 46.15</td>
<td>47.62</td>
<td>48.21</td>
<td>49.02</td>
</tr>
<tr>
<td>Boat</td>
<td>90</td>
<td>MSE 0.0116</td>
<td>0.0154</td>
<td>0.0069</td>
<td>0.0024</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 67.47</td>
<td>66.27</td>
<td>69.72</td>
<td>74.30</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>MSE 0.2341</td>
<td>0.3271</td>
<td>0.1999</td>
<td>0.1116</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 54.44</td>
<td>52.98</td>
<td>55.12</td>
<td>57.65</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>MSE 1.4634</td>
<td>1.8176</td>
<td>1.0998</td>
<td>0.8011</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 46.48</td>
<td>45.54</td>
<td>47.72</td>
<td>49.09</td>
</tr>
<tr>
<td>Barbara</td>
<td>90</td>
<td>MSE 0.0145</td>
<td>0.0171</td>
<td>0.0079</td>
<td>0.0029</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 66.51</td>
<td>65.81</td>
<td>69.17</td>
<td>73.49</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>MSE 0.3925</td>
<td>0.3238</td>
<td>0.2287</td>
<td>0.1451</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 52.19</td>
<td>53.03</td>
<td>54.58</td>
<td>56.51</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>MSE 2.9232</td>
<td>1.7590</td>
<td>1.2279</td>
<td>1.0066</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 43.47</td>
<td>45.68</td>
<td>47.24</td>
<td>48.10</td>
</tr>
<tr>
<td>Fingerprint</td>
<td>90</td>
<td>MSE 0.0758</td>
<td>0.1413</td>
<td>0.0368</td>
<td>0.0114</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 59.33</td>
<td>56.63</td>
<td>63.25</td>
<td>67.56</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>MSE 2.3375</td>
<td>2.5713</td>
<td>0.8187</td>
<td>0.6911</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 44.44</td>
<td>44.03</td>
<td>48.99</td>
<td>49.74</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>MSE 15.4409</td>
<td>11.6338</td>
<td>4.0629</td>
<td>4.4065</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSNR 32.24</td>
<td>37.47</td>
<td>42.04</td>
<td>41.69</td>
</tr>
</tbody>
</table>
II. SURVEY OF IMAGE COMPRESSION TECHNIQUES...

Fig. 12. Percentage of Coefficients Retained vs PSNR for lena

Fig. 13. Percentage of Coefficients Retained vs PSNR for boat
Fig. 14. Percentage of Coefficients Retained vs PSNR for barbara

Fig. 15. Percentage of Coefficients Retained vs PSNR for fingerprint