CHAPTER 3

HYBRID AFFECTIVE WAVELET TRANSFORM (HAWT) METHOD

3.1 INTRODUCTION

After exploring the previous researches and techniques that have been done for facial expression recognition, this chapter will discuss in details the methodology and the framework to be used in this thesis. The following sections describe the mathematical background of Discrete Wavelet Transform (DWT), and K-Nearest Neighbor (KNN). In this study, the new methodology will consider the several methodologies that had been done by previous researchers that relates to the thesis. To achieve better performance, new technique has been developed based on methodologies given in this chapter.

3.2 NEW HUMAN EMOTION RECOGNITION SYSTEM

The overview of the new Hybrid Affective Wavelet Transform (HAWT) approach for human emotional state analysis is depicted in this section. In order to achieve powerful recognition system, hybrid feature analysis is introduced by employing approaches such as DWT, UWT and KNN classifier. It is also exploited for e-learning application to assist the teachers on the way to automatically analyze the emotions of learners. The block diagram of the new system along with E-learning environment is shown in figure 3.1. The obtained HAWT features are stored in database, which is indicated as feature database. The HAWT system consists of two important stages; feature extraction and classification.
Figure 3.1 Block diagram of the New Human Emotion Recognition System
This research develops an automatic human emotion recognition system to classify human emotion state into happiness, sadness, surprise, anger, disgust, and fear and neutral. Automated processing and classification of human emotion has become an increased need due to its various applications. This demand has led to the development of several techniques to classify the human emotion state.

The steps involved in this research are:

STEP 1: Significantly review the available human emotion recognition/classification schemes, identify the Discrete Wavelet Transform (DWT), Undecimated Wavelet Transform (UWT) and K-Nearest Neighbor(KNN) classifier which are applied to extract the features and classify the emotion state. These terms relate to the algorithms used.

STEP 2: Developing a Hybrid Affective Wavelet Transform(HAFT) based human emotion recognition system which fuses the features of DWT and UWT.

STEP 3: Extract frequency domain features and store them in a database for further processing.

STEP 4: Identify the significant features needed for the classification of emotional state of facial images from the database created and verify the percentage of classification.
STEP 5: Train the nearest neighbor classifier to facilitate the correct classification from the database using the optimum number of extracted salient features, which can be used as feature vectors.

STEP 6: Evaluate the performance of the emotion recognition system in comparison with DWT and UWT.

STEP 7: Apply the hybrid affective wavelet transform based human emotion recognition system in e-learning applications.

3.3 FEATURE EXTRACTION

In digital image processing, there are numerous transforms are applied on digital images to analyze and manipulate the images. The most commonly used transforms are, Discrete Fourier Transform(DFT), Discrete Cosine Transform(DCT), Haar Wavelet Transform, 2-D Haar Wavelet Transform. This research work is based on Discrete Wavelet Transform(DWT) and Undecimated Discrete Wavelet Transform(UWT) and HAWT which is of fusion of DWT and UWT.

3.3.1 DISCRETE FOURIER TRANSFORM

The Fourier transform is a special case of the Z-transform. The discrete Fourier transform (DFT) is the sampled Fourier transform that requires an input signal that is discrete. Such inputs are often created by sampling a continuous signal. It is used to decompose an image into its sine and cosine components. A digital image contains point samples of a band-limited continuous 2D signal, according to the uniform sampling
theorem. A signal is band-limited if it contains no energy at frequencies higher than some band limit. The output of the transform represents the image in the frequency domain (or Fourier domain), while the input image is the spatial domain equivalent. In the image equivalent of the Fourier coefficients, each point represents a particular frequency contained in the spatial domain image [THK2008]. For an image $X$ of size $N \times N$, the 2-D discrete Fourier transform is given by (3.1)

$$X(a,b) = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m,n) e^{-j2\pi \frac{am}{N}} e^{-j2\pi \frac{bn}{N}}$$  

(3.1)

The set of Fourier Coefficients corresponding to the digital image can be retransformed to the spatial domain using the inverse Fourier transform, which is given by (3.2)

$$X(a,b) = \frac{1}{N} \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} X(a,b) e^{j2\pi \frac{am}{N}} e^{j2\pi \frac{bn}{N}}$$  

(3.2)

### 3.3.2 DISCRETE COSINE TRANSFORM

The Discrete Cosine Transform [ANK1974] is a Fourier related transform, similar to the discrete Fourier transform, which uses only real numbers. It expresses a signal in terms of a sum of cosine functions oscillating at different frequencies. An obvious distinction between a discrete cosine transform and a discrete Fourier transform is that the discrete cosine transform uses only cosine functions, while the discrete Fourier transform uses both cosine and sine functions. The Discrete cosine transforms are utilized in many applications, with the most notable being the MP3 music format, as well as the JPEG
image format. For a 2-D signal, like an image, \( x \) of size \( N \times N \), the Discrete Cosine Transform [THK2008] is given by (3.3)

\[
C(u,v) = \frac{1}{N^2} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m,n) \cos\left(\frac{(2m-1)u}{2N}\right) \cos\left(\frac{(2n-1)v}{2N}\right)
\]

while the inverse 2D discrete cosine transform is given by

\[
x(m,n) = \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u,v) \cos\left(\frac{(2m-1)u}{2N}\right) \cos\left(\frac{(2n-1)v}{2N}\right)
\]

where

\[
a(k) = \begin{cases} 1, & k = 0 \\ \frac{1}{N}, & k \neq 0 \\ \frac{2}{N}, & k = 0 \\ \frac{1}{N}, & k \neq 0 \\ \frac{2}{N}, & k = 0 
\end{cases}
\]

### 3.3.3 DISCRETE WAVELET TRANSFORM

The best way to introduce the wavelet transform mathematical derivation is to review the familiar Fourier transform given by Richard Andrew Muyshondt (1995). The Continuous Time Fourier transform \( (CTFT = \mathcal{F}(\cdot)) \) is defined as the inner product or projection of the function \( f(x) \) onto the basis function \( e^{\pm i\omega t} \). In mathematical terms, the \( CTFT \) [RRS2004, ANI1989]
where, \(<,>\) denotes the inner product, \(x\) is the location variable, and \(\omega\) is the frequency variable. The inner product in equation (3.6) denotes the Fourier coefficients. Therefore, a function \(f(x)\) can be written exactly as

\[
  f(x) = \int e^{i\omega x} f(x) e^{-i\omega x} d\omega

equation (3.7)
\]

Equation (3.7) says that \(f(x)\) is the integration of \(e^{i\omega x}\) multiplied or weighed by the appropriate Fourier coefficient. Although the Fourier transform is a very strong mathematical and analytical tool, it has some properties which are not desirable for image processing techniques. As can be seen from equation (3.7), all past and future time information is needed to accurately reconstruct a signal \(f(x)\). If all this time information is not known, the reconstructed signal will only be an approximation of the original. In addition, in order for the Fourier transform not to have an infinite number of coefficients, the signals being transformed must be stationary. Since the Fourier transform can be written as the projection of the signal onto cosines in the real plane and sines in the imaginary plane, discontinuities in a signal would only be represented with many sine and cosine terms. Therefore, the decomposition of a discontinuous signal would contain many coefficients which are spread out over the entire frequency axis.

In order to mitigate this coefficient frequency spreading, the \(CTFT\) was modified to form the Short Time Fourier transform (\(STFT\)). The \(STFT\) which is sometimes referred
to as the windowed Fourier transform, is similar to the CTFT except for the addition of a window function \((w)\) that limits the existence of the transform. The window function is usually a Gaussian, and it’s shifting over the time axis results in a time frequency description of the signal. In general, the STFT can be mathematically written as

\[
STFT: F(a, b) = \int_{-\infty}^{\infty} f(x) e^{i\omega x} w(x-b) dx
\]  

(3.8)

where, \(x\) is the location variable, \(\omega\) is the frequency variable, \(w(\cdot)\) is the window function, and \(b\) is the location shifting parameter for the window function [BAR1994].

The STFT eliminates the need for complete past and future signal information. However, the STFT still contains a drawback. The window function used in the transform has a fixed window size. Therefore, it cannot adapt to the characteristics of signals at certain points. That is, it does not give a good description of signals with widely changing frequency spectra. The Continuous Time Wavelet Transform (CTWT) mitigates the limitations of the CTFT and STFT by having a basis function which can be both shifted and dilated or contracted. The CTWT basis function is

\[
a_{a,b}(x) = a^{-\frac{1}{2}} \tilde{\chi}_{a,b}(x)
\]

(3.9)

where, \(\tilde{\chi}(x)\) is a zero mean band pass function, and the transform is defined as

\[
CTWT (a,b) = a^{-\frac{1}{2}} \int_{-\infty}^{\infty} f(x) \tilde{\chi}_{a,b}(x) dx
\]

(3.10)
where, \( a_{a,b} \) are the wavelets basis function, \( a \) is the scaling factor, and \( b \) is the shifting factor (Barnard 1994). \( a_{a,b} \) are real and oscillatory and fade away when they approach plus or minus infinity. \( \Psi \) itself is called the mother wavelet, and the shifted and scaled conglomeration of \( \Psi \) form an ortho normal basis which as mentioned above are called wavelets.

A close examination of equation (3.10) shows that increasing 'a' causes \( T \) to be stretched, and thus, the formed wavelets can act as low frequency windows. On the other hand, decreasing 'a' causes \( W \) to be shrunk, and thus, these wavelets act as high frequency windows. Consequently, one can narrow and widen the time-frequency window as well as shift it over the time domain via 'b' in order to adjust to the frequency characteristics of any signals [DAU1988, MMP1992].

In addition, due to the decay property of the wavelets, one does not need all past and future signal information in order to accurately represent any signal. In general, any signal \( f(x) \) can be represented as

\[
f(x) = \frac{1}{C_{a,b}} \int_{-\infty}^{\infty} f(x) \, d\tau (a,b)a^{\frac{1}{2}} \left( x - b \right) d\tau a \, dB a, a^2
\]

(3.11)

where, \( a, b \) are continuous and \( C_{a,b} \) is a constant. Since there are no restrictions on \( a \) and \( b \), there exist infinite combinations of possible wavelets. In order to reduce this redundancy, \( a \) and \( b \) are discretized by letting

\[
a = a_0^m
\]

(3.12)
and

$$ b \equiv n a_0^m b_0 \quad (3.13) $$

where, $a$, $b$, $a_0$ and $b_0$ are constants. By substituting these equalities into equation (3.11), the basis functions now become

$$ m_{m,n}(x) \equiv a_0^{-m} x^n b_0 \quad (3.14) $$

and the new "sampled" CTWT, now called the continuous time wavelet series (CTWS), is defined as [DAU1988, MMP1992, ALI1994].

$$ CTWS_{m,n}(x) \equiv d_{m,n}(x) \int_{m,n} f(x) \, dx \equiv m_{m,n}(x), f(x) \quad (3.15) $$

and

$$ f(x) \equiv d_{m,n} m_{m,n}(x) \equiv m_{m,n}(x), f(x) \equiv m_{m,n}(x) \quad (3.16) $$

As mentioned above, wavelets are zero mean band pass functions and have zero DC coefficients. Thus, in order to completely represent a function $f(x)$ via a pure wavelet expansion, one needs an infinite number of wavelet resolutions [ALI1994]. This is impracticality in image compression where the goal is to reduce the amount of data needed to represent the total image.
To avoid using an infinite number of coefficients for the wavelet representation \( f(x) \) is represented by a low pass or low resolution version, \( f_j(x) \) and a detail version, \( D \). The detail version \( D \) is simply a finite wavelet representation. The low pass version, on the other hand, is a result of the inner product between \( f(x) \) and a scaling function \( f_j(x) \). Thus, if the scaling functions weight coefficients are

\[
C_{j,n} \int_0^a \cdots \int_0^{a^j} \cdots \int_0^{a^j} x^{nb} \cdots dx \cdots \int_0^{a^j} (x), f(x) \quad (3.17)
\]

and the function at resolution \( j \) (i.e. \( f_j(x) \)) is

\[
f_j(x) = \int_0^a \cdots \int_0^{a^j} \cdots \int_0^{a^j} x^{nb} \cdots dx \cdots \int_0^{a^j} (x), f(x) \quad j_n(x) \quad (3.18)
\]

then any arbitrary function \( f(x) \) can be represented as [DAU1988, MMP1992, ALI1994]

\[
f_j(x) = C_{j,n} \int_0^{a^j} \cdots \int_0^{a^j} x^{nb} \cdots dx \cdots \int_0^{a^j} (x), f(x) \quad j, n \quad (3.19)
\]

\[
= f_j(x) + D \quad (3.21)
\]
Furthermore, by letting $a_0 = 2$ and $b_0 = 1$, one can pick special $\square(x)$ such that for fixed $m$, $\square_{m,n}(x)$ and $\square_{m,n}(x)$ individually form an orthonormal basis. In addition, both $\square_{m,n}(x)$ and $\square_{m,n}(x)$ are orthogonal complements of one another, $\square_{m,n}(x), f(x)$ and $\square_{m,n}(x), f(x)$ represent the information lost by going to a lower resolution via [ALI1994] and $\square_{m,n}(x), f(x)$. Note that the wavelet representation of a function has been established by the above, one should be aware of the containment restrictions placed on both $\square(x)$ and $\square(x)$. Since the scaling function projects the function $f(x)$ to a lower resolution plane (i.e., resolution $j$ in the above discussion), it is easy to visualize that all of the information in any lower resolution is contained by the top resolution of the scaling function. Thus, the containment restriction on $\square(x)$ is

$$\square(x) \square 2^n h_0(n) \square (2x)^n$$

(3.22)

where $h_0(n)$ are called interscale basis coefficients. Similarly, the containment restriction on $\square(x)$ is

$$\square(x) \square 2^n h_1(n) \square (2x)^n$$

(3.23)

where, $h_1(n)$ are the expansion coefficients. In general, it can be proved that

$$h_n$$

64
In addition if one defines

\[
\int_{-2^n}^{2^n} (3.24) \quad dx
\]

\[
\int_{x^n}^{2x} (2x) \quad dx
\]
\[ g_1 = (-1^j) h.\cdot j + l \]  \hfill (3.25)

Then the scaling coefficients \( C_{j,n} \) and wavelet coefficients \( d_{m,n} \) can be represented by

\[ C_{j,n} = \bigotimes_k h_{2n \cdot k} C_{j+1,k} \]  \hfill (3.26)

and

\[ d_{m,n} = \bigotimes_k g_{2n \cdot k} C_{m+1,k} \]  \hfill (3.27)

Equations (3.26) and (3.27) are used for the computation of the multi resolution wavelet transform, and \( h \) is a low pass filter and \( g \) is a band pass filter [DAU1988, MMP1992, ALI1994]

The detail or information lost going to the lower approximation of the original function, \( f(x) \) by taking the inner product of \( f(x) \) with \( \bigotimes j,n \) is given by the projection (or inner product) of the original function in the \( \bigotimes j,n \) subspace. Thus, the discrete representation of a discrete function \( a_{j-1,l}(f) \) at resolution \( j - 1 \) is

\[ C_{j\cdot 1,l}(f) = \bigotimes_n \left( g_{2n \cdot 1} d_{m,n}(f) \bigotimes h_{2n \cdot 1} C_{j,n}(f) \right) \]  \hfill (3.28)

where \( g_{2n-1} \) and \( h_{2n-1} \) are the discrete high and low pass filters, respectively, and \( C_{j,n} \) and \( d_{m,n} \) are the lower \( j \) resolution discrete signal and the details lost by going to the lower resolution signal, respectively by [MMP1992].
It follows that one can represent an arbitrary signal by its lower resolution and the detail lost going to the lower resolution with proper choices of the basis functions. Furthermore, one can extend this concept to a multi resolution approach where the original signal is represented by consecutively decomposing it to lower resolutions, while at the same time storing all the detail information between each consecutive resolution.

This multi resolution approach can be extended to a 2-D. In the 2-D multi-resolution wavelet transform, one must introduce a scaling function \( \square(x,y) \) and a wavelet function \( \square(x,y) \) that are separable. That is

\[
\square(x,y) \triangleq \square(x) \square(y) \tag{3.29}
\]

and

\[
\square(x,y) \triangleq \square(x) \square(y) \tag{3.30}
\]

In addition, the combinations of these functions are defined as

\[
\square(x,y)_l \triangleq \square(x) \square(y) \tag{3.31}
\]

\[
\square(x,y)_h \triangleq \square(x) \square(y) \tag{3.32}
\]

\[
\square(x,y)_v \triangleq \square(x) \square(y) \tag{3.33}
\]
\( \square(x, y) \square(x) \square(y) \) (3.34)
Using a similar extension of the properties, one can now define the filters $h$ and $g$ for the basic functions. The filters are $h_r, h_c, g_r,$ and $g_c$ for $\square(x), \square(y), \square(x)$ and $\square(y)$ respectively. The subscripts $r$ and $c$ stand for row and column and indicate the direction in which the filters are applied. So, the low pass and detail images are given by

\[ C_1 = h_r h_c C_0 \] \hspace{1cm} (3.35)

\[ d_h = d_{1,1} = h_r g_c C_0 \] \hspace{1cm} (3.36)

\[ d_v = d_{1,2} = g_r h_c C_0 \] \hspace{1cm} (3.37)

and

\[ d_d = d_{1,3} = g_r g_c C_0 \] \hspace{1cm} (3.38)

Computationally, the 2-D WT of an image is taken in two parts. First, the 1-D wavelet transform is taken along the image pixel rows by multiplying each row by the appropriate low and high pass filters $h$ and $g$. The low pass and detail groups are then down sampled by two. The second steps in the 2-D wavelet transform is accomplished by taking the 1-D wavelet transform along the columns of each of the reordered low and high pass filtered groups. These is again accomplished by multiplying both the low pass and high pass filtered groups by the same low pass and high pass filters discussed above. The columns are again down sampled by two. The result of this operation is a decomposition which has a low pass $L$ image in quadrant one, a vertical error image $D_v$ in quadrant two, a horizontal error image $D_h$ in quadrant three, and a diagonal error image $D_d$ in quadrant four. Thus, the first wavelet decomposition of the image is complete. Further wavelet
transformations of the resulting low pass image will result in multiresolutional wavelet transform decomposition.

### 3.3.4. HAAR WAVELET TRANSFORM

In mathematics, the Haar wavelet is a certain sequence of functions. It is now recognized as the first known wavelet. This sequence was new in 1909 by Alfred Haar. Haar used these functions to give an example of a countable orthonormal system for the space of square integrable functions on the real line. The study of wavelets, and even the term "wavelet", did not come until much later. The Haar wavelet’s mother wavelet function \( \square(x) \) can be described as

\[
\begin{align*}
\square(x) &= \begin{cases} 
0 & x \leq \frac{1}{2}, \\
1 & \frac{1}{2} < x < 1, \\
0 & \text{otherwise}
\end{cases} \\
\end{align*}
\tag{3.39}
\]

and its scaling function \( \square(x) \) can be described as

\[
\begin{align*}
\square(x) &= \begin{cases} 
1 & x \leq 1, \\
0 & \text{otherwise}
\end{cases} \\
\end{align*}
\tag{3.40}
\]
3.3.5 2D HAAR WAVELET TRANSFORM

It has been shown in previous section how one dimensional image can be treated as sequences of coefficients. Alternatively, consider images as piecewise constant functions on the half-open interval \([0, 1]\). To do so, the concept of a vector space is used. A one-pixel image is just a function that is constant over the entire interval \([0, 1]\). Let \(V^0\) be the vector space of all these functions. A two pixel image has two constant pieces over the intervals \([0, 1/2]\) and \([1/2, 1]\). Call the space, containing all these functions \(V^1\). If it is continued in this manner, the space \(V^j\) will include all piecewise-constant functions defined on the interval \([0, 1]\) with constant pieces over each of \(2^j\) equal subintervals. Consider every one-dimensional image with \(2^j\) pixels as an element, or vector, in \(V^j\).

Note that because these vectors are all functions defined on the unit interval, every vector in \(V^j\) is also contained in \(V^{j+1}\). For example, it is always described a piecewise constant function with two intervals as a piecewise-constant function with four intervals, with each interval in the first function corresponding to a pair of intervals in the second. Thus, the spaces \(V^j\) are nested; that is, \(V^0 \sqsubseteq V^1 \sqsubseteq V^2 \ldots\) is nested set of spaces \(V^j\) is a necessary ingredient for the mathematical theory of multiresolution analysis. It guarantees that every member of \(V^0\) can be represented exactly as a member of higher resolution space \(V^1\). The converse, however, is not true: not every function \(G(x)\) in \(V^1\) can be represented exactly in lower resolution space \(V^0\); in general there is some detail lost.

Now define a basis for each vector space \(V^j\). The basic functions for the spaces \(V^j\) are called scaling functions, and are usually denoted by the symbol \(\square\). A simple basis for \(V^j\) is given by the set of scaled and translated box functions.
\[ \Box^i(x) \Box x \Box i \quad i \in \{0, 1, 2, \ldots, 2^j\} \quad (2^j) \]

where \( \Box(x) \) is given by equation (3.40). The wavelets corresponding to the box basis are known as the Haar wavelets, given by

\[ \Box^i(x) \Box x \Box i \quad i \in \{0, 1, 2, \ldots, 2^j\} \quad (2^j) \]

where \( \Box(x) \) is given by equation (3.39).

Thus, the DWT for an image as a 2D signal will be obtained from 1D DWT. To get the scaling function and wavelet function for 2D by multiplying two 1D functions. The scaling function is obtained by multiplying two 1D scaling functions given by equation (3.40). The wavelet functions are obtained by multiplying two wavelet functions or wavelet and scaling function for 1D. For the 2D case, there exist three wavelet functions that scan details in horizontal, vertical and diagonal directions. This may be represented as a four channel perfect reconstruction filter bank as shown in Figure 3.2. Now, each filter is 2D with the subscript indicating the type of filter (HPF or LPF) for separable horizontal and vertical components. By using these filters in one stage, an image is decomposed into four bands. There exist three types of detail images for each resolution: horizontal (HL), vertical (LH), and diagonal (HH). The operations can be repeated on the low frequency (LL) band using the second stage of identical filter bank.
Assessing the performance criteria of the applied feature extraction process requires selecting an appropriate classification algorithm or metric dissimilarity measures. The differentiation between one classification method and another resides in their ability to overcome feature nonlinearity in order to distinguish the differences between compared texture regions; however, no matter how good the applied feature extraction process is if a poor classification design was implemented. In the training or learning stage, there are two kinds of learning methods supervised learning and unsupervised learning. Unsupervised learning means that no human intervention during the learning or little knowledge about the training data is required. However, supervised learning means that every training sample is labeled with the class to which this sample belongs. To sum up, the performance of a pattern recognition system depends on the preprocessing steps, feature extraction and classifier selection. The feature extraction module is the part where most face image processing techniques differ.

Figure 3.2 one level 2D DWT

3.4 KNN CLASSIFICATION
In classification, the decision would be to what class the data belongs. Computer learning falls under two broad categories: supervised and unsupervised learning. Unsupervised learning is used when there is no class data available for a dataset. In this case objects are partitioned so as to best cluster the data.

Supervised learning is used in situations where there is some sample data available with appropriate decisions that can be used as a training set. Classifiers often operate in two phases. The training phase is where the relationship between certain features and outcomes is determined and optimized. This is often a long and computationally intensive process. The operating phase is when the training data is put to use to classify an object. This is usually much quicker. Possibly the most important component of a classification routine is the feature vector. The feature vector is a set of scalar quantities that describe an object. The choice of a feature vector is vital to the success of a machine learning algorithm. The algorithms work by comparing the feature vector of a test object with those of objects already classified. If the data in the feature vector is not appropriate for the classification task, it will fail.

Usually the initial choice of a feature vector is not the best one. Some features may not contribute to the classification task or might be made redundant by other features. Attempting to classify with these features can not only significantly increase computation time, but can make classifications less accurate. To mitigate this problem, a feature reduction step should take place. A good feature reduction process will result in faster learning due to less data, higher accuracy and better generalization to other data sets. There are two approaches of choosing a feature vector from all available features, top-down and bottom-up. The top-down approach takes a vector of all features and removes
them one by one, testing the classification accuracy at each step. The bottom-up approach does the opposite. It starts with an empty vector and adds features to it one by one.

Classifiers can be either soft or hard. A Hard classifier classifies an object without giving a probability. The assumption is made that an object that meets a certain criteria always belongs to a particular class. Soft classifiers give a probability of their classification. The assumption made is that sometimes objects with similar features may belong to different class. In pattern recognition, the K-Nearest Neighbor Algorithm (K-NN) is a method for classifying objects based on closest training examples in the feature space. K-NN is a type of instance-based learning where the function is only approximated locally and all computation is deferred until classification. In K-NN, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of its nearest neighbor. The neighbors are taken from a set of objects for which the correct classification is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A measure in which to determine the distance between two scenarios is established and simply pass through the data set, one scenario at a time, and compare it to the testing scenario. Let our training data set is represented as a matrix $D^{N \times P}$, containing $P$ scenarios $s_1, \ldots, s^P$ where each scenarios $s^i$ contains $N$ features $s^i \in \{s_1^i, \ldots, s_N^i\}$. A vector with length $P$ of output values $\hat{y}_1, \ldots, \hat{y}^P$ accompanies this matrix, listing the output value $\hat{y}_i$ for each scenario $s^i$. 
It should be noted that the vector \( \mathbf{r} \) can also be seen as a column matrix; if multiple output values are desired, the width of the matrix may be expanded. KNN can be run in these steps:

**STEP 1**: Store the output values of the \( M \) nearest neighbors to query scenario \( q \) in vector \( \mathbf{r} = \{r^1, \ldots, r^M\} \). By repeating the following loop \( M \) times:

- Go to the next scenario \( s^i \) in the data set, where \( i \) is the current iteration within the domain \( \{1, \ldots, P\} \)
- If \( q \) is not set or \( q \neq d(q, s^i) : q \neq d(q, s^i) : t \neq i \).
- Loop until reach the end of the data set \( \text{i.e.} i \neq P \)
- Store \( q \) into vector \( \mathbf{c} \) and \( t \) into vector \( \mathbf{r} \).

**STEP 2**: Calculate the arithmetic mean output across as follows

\[
r = \frac{1}{M} \sum_{i=1}^{M} r_i
\]

(3.43)

**STEP 3**: Return \( \mathbf{r} \) as the output value for the query scenario \( q \).

**STEP 4**: The closeness can be measured by various distance techniques such as Euclidean, City Block, Cosine and Correlation distance.
3.4.1 EUCLIDEAN DISTANCE

Let us consider \( u = x_1, y_1 \) and \( v = x_2, y_2 \) are two points. The Euclidean distance between these two points is given by

\[
\text{Euclidean Distance} \left( u, v \right) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{3.44}
\]

If the points have \( n \)-dimensions such as \( u = x_1, x_2, x_3, \ldots, x_n \) and
\( v = y_1, y_2, y_3, \ldots, y_n \) then the generalized Euclidean distance formula between these points is

\[
\text{Euclidean Distance} \left( u, v \right) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \tag{3.45}
\]

3.4.2 CITY BLOCK DISTANCE

Let us consider \( u = x_1, y_1 \) and \( v = x_2, y_2 \) are two points. The city block distance between these two points is given by

\[
\text{Cityblock} \left( u, v \right) = |x_1 - x_2| + |y_1 - y_2| \tag{3.46}
\]

If the points have \( n \)-dimensions such as \( u = x_1, x_2, x_3, \ldots, x_n \) and
then the generalized city block distance formula between these points is

\[
\text{Cityblock } (u, v) = \sum_{i=1}^{n} |x_i - y_i|
\]
3.4.3 COSINE DISTANCE

Let us consider \( X = [x_1, x_2, x_3, \ldots, x_n] \) and \( Y = [y_1, y_2, y_3, \ldots, y_n] \), then

\[
\cos \theta \text{ may be considered as the cosine of the vector angle between } X \text{ and } Y \text{ in } n \text{ dimension.}
\]

The cosine of the vector angle between \( X \) and \( Y \) is given by

\[
\text{Cosine } X, Y = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}} \tag{3.48}
\]

One important property of cosine angle is that it gives a metric of similarity between two vectors unlike Manhattan distance and Euclidean distance, both of which give metrics of dissimilarities. Also \( \text{Cosine } X, Y \in [0, 1] \), this makes it easy to combine distance between two images using multiple features.

3.4.4 CORRELATION DISTANCE

A correlation is a single number that describes the degree of relationship between two variables \( X \) and \( Y \) where \( X = [x_1, x_2, x_3, \ldots, x_n] \) and \( Y = [y_1, y_2, y_3, \ldots, y_n] \). The correlation \( r \) between \( X \) and \( Y \) is defined by

\[
r = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}} \tag{3.49}
\]
\[
N x^2 \quad N y^2 \quad x^2 \quad y^2
\]
3.5 SUMMARY

In this chapter, multi resolution analysis based on DWT is discussed. This transform is applied to various image processing applications such as compression, denoising, enhancement and edge detection. KNN is a well known machine learning approach to automated learning of pattern classifiers. It has been proven to give highly accurate results in classification systems. The next chapter deals with the human emotion recognition system based on DWT and UWT with KNN.