CHAPTER III

PROPOSED SOLUTIONS

3.1 A Neural Network Approach for Region's Image Depth Ordering

The goal of region image depth ordering works is to classify the foreground, middle ground and background regions. Actually, the foreground, middleground and background are the regions of focused or blurred images. Because the Depth-Of-Field(DOF) images include all foreground, middleground and background regions, we applied a supervised learning neural network to learn the relationship between the depth's order of focused and blurred regions in the DOF image. The depth's order of focused and blurred will be shown in the terms of foreground, middle ground and background regions image. Figure 3.1 shows the overview of the neural network learning and testing processes.

3.2 Depth-of-Field Image(DOF)

The focused and blurred regions that corresponds to foreground, middleground and background regions can be explained in terms of the optical geometry of the camera's lens.

When an image is staying in focus, all photons that are radiated by a point O pass through the aperture A and are refracted by the lens to converge at the point Q in the focused image plane as shown in Fig 3.2. The focused plane position v depends on the depth u of the object and the focal length f of the lens, according to the lens law:

$$\frac{1}{f} = \frac{1}{u} + \frac{1}{v} \tag{3.1}$$

However, when the point O is not in focus, its images on the far-focused plane I_1 and near-focused plane I_2 are not points but patches, which are indicated by blurred circles of diameters b_1 on plane I_1 and b_2 on plane I_2, respectively. The amount of blurring actually depends on the characteristic of lens system and the distance to the surface of exact focus. The blurred circle is generated from the incapable of lens to

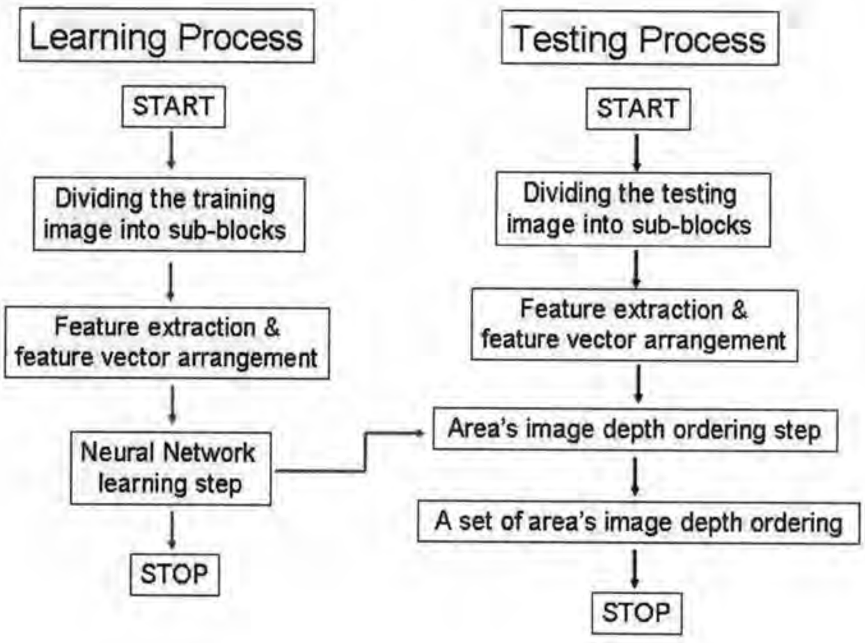

Learning Process
START
Dividing the training image into sub-blocks
Feature extraction & feature vector arrangement
Neural Network learning step
STOP
Testing Process
START
Dividing the testing image into sub-blocks
Feature extraction & feature vector arrangement
Area's image depth ordering step
A set of area's image depth ordering
STOP

Figure 3.1: Neural network learning and testing processes.

fuse the light reflection from the difference length of the objects in 3-D space into a 2-D space. In human vision, we know what region of image is blur or in-focus by considering lines and edges of an object in the image. When the lines and the edges of an object are sharp, it means that this object is located in the focal plane and the object is out of focus if the lines and edges are blurred. This is the blurred and in-focus object behavior. Thus, depth of field is defined as the distance in front of and behind your subject that will also be in-focus. Photographers often use this photographic technique to point their interest in the image or to help viewers understand the depth information from a two dimensional image. Figure 3.3 displays the depth-of-field of three sample images. The main different of the three sample images is the position of in-focus regions in an image, Fig. 3.3(a)-(c) shows the depth-of-field images that position of in-focus regions appeared in foreground, middleground and background regions of images, respectively.

3.3 Feature Extraction Based on the In-focus and Out-of-focus Region's Image

The prominent attribute of depth-of-field image is the three dimensional perceiving which directly relates to the in-focus and blur regions in an image (see image in Fig 3.3(a)-(c)). Usually, the in-focus and blur regions can be represented in spatial frequency domain. The in-focus regions correspond to the high frequency and blurred region correspond to

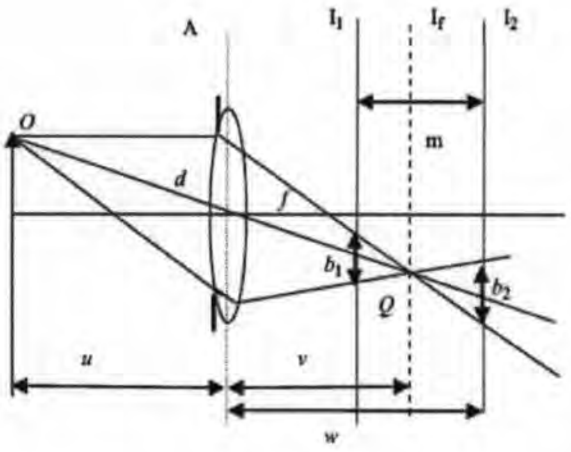


Figure 3.2: Optical geometry of a typical camera.

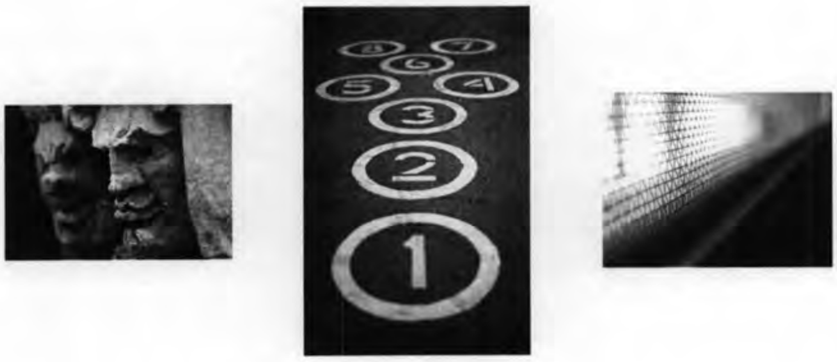
low spatial frequency. Wavelet transform is a mathematical tool that can be perfectly used in frequency analysis. The used feature in this work is the texture information that related to the spatial frequency, therefore, the wavelet transform was choosed to analyze the regions of in-focus and out-of-focus. We used Haar wavelet for analyze the depth-of-field image because this kind of wavelet is simple and sufficient for region classification. The method of estimator operation is the main difference of wavelet family, the results of Haar wavelet in one level estimator is not difference to the others. Feature extraction is the first essential stage before the neural network learning. The texture information of in-focus and blurred was used to be a features in the training set. In this work, the texture feature was extracted by using wavelet transform. Wavelets are mathematical tools for hierarchical decomposing functions [10]. They allow a function to be described in terms of a coarse overall shape, plus details that range from broad to narrow ranges. Regardless of whether the function of interest is an image, a curve or a surface, wavelets offer an elegant technique for representing the levels of detail present. Our proposed method uses two dimension Haar wavelet transform $W_f(x)$ to decompose the original image that is defined by :

$$W_f(x) = \sum\nolimits_{i=1}^{a} \sum\nolimits_{j=1}^{b} c_{i,j}\psi_{i,j}(x) \tag{3.2}$$

where $\psi_{i,j}(x)$ is the wavelet basis function and $c_{i,j}$ is the wavelet coefficient. The basis function of the wavelet transform is generated from the basic wavelet function $\psi(x)$ together with translation $(b : b \in \Re^{+})$ and dilation $(a : a \in \Re)$. The scaled and translated wavelet can be written as:

$$\psi_{a,b}(x) = f\left(\frac{x-b}{a}\right) \tag{3.3}$$

where $f(x)$ is translated by b units and dilated by a scaling factor a. The wavelet coefficient can be calculated by the inner products



(a) (b) (c)

Figure 3.3: Depth-of-field images. (a) The position of in-focus regions appeared in foreground. (b) The position of in-focus regions appeared in middleground. (c) The position of in-focus regions appeared in background.

$$c_{i,j} = \langle f(x), \psi_{i,j}(x) \rangle = \int_0^1 f(x)\psi_{i,j}(x)dx \qquad (3.4)$$

To get a sense for how wavelet transform work, let start with a simple example. Suppose we are given a one-dimensional "*image*" with a resolution of four pixels, having values

$$[\ 9 \quad 7 \quad 3 \quad 5\ ]$$

We can represent this image in the *Haar basis* by computing a wavelet transform. To do this we first average the pixels together, pairwise, to get the new lower resolution image with pixel values

$$[\ 8 \quad 4\ ]$$

Clearly, some information has been lost in this averaging process. To recover the original four pixel values from the two averaged values, we need to store some *detail coefficients*, which capture the missing information. In our example, we will choose 1 for the first detail coefficient, since the average we computed is 1 less than 9 and 1 more than 7. This single number allows us to recover the first two pixels of our original four-pixel image. Similarly, the second detail coefficients is -1, since $4 + (-1) = 3$ and $4 - (-1) = 5$. Thus, we have decomposed the original image into a lower resolution (two-pixel) version and a pair of detail coefficients. Repeating this process recursively on the average gives the full decomposition:

Resolution	*Averages*	*Detail coefficients*
4	[9 7 3 5]	
2	[8 4]	[1 -1]
1	[6]	[2]

Finally, we will define the *wavelet transform*(sometime call the *wavelet decomposition*) of the original four-pixel to be the single coefficient representing the overall average of the original image, followed by the detail coefficients in order of increasing resolution. Thus, for the one-dimensional Haar basis, the wavelet transform of our original four-pixel image is given by

[6 2 1 -1]

We can think of image as a piecewise constant function, a one-pixel image is just a constant function over the interval $[0, 1)$ in the space V^0. A two pixel image has two constant function over interval $[0, 1/2)$ and $[1/2, 1)$ in the space V^1. So, if we continue in this process, we will obtain the space v^j that include all piecewise function on the interval $[0, 1)$ with the interval divided equally into 2^j pieces. The basis function for each vector space V^j are called *scaling functions*, and are usually denoted by the symbol ϕ. A simple basis for V^j is given by the set of scaled and translated *box* functions:

$$\phi_i^j(x) = \phi(2^j x - i), \quad i = 0, ..., 2^j - 1$$

where

$$\phi(x) = \begin{cases} 1 & for\ 0 \leq x < 1 \\ 0 & otherwise \end{cases}$$

As an example, Fig.3.4 shows the four box functions forming a basis for V^2. The coefficients can be obtains from the inner product in Eq.3.4 with the box basis function, in this dissertation the box basis function is the *Haar wavelets Basis Function* that given by:

$$\psi_i^j(x) = \psi(2^j x - i), \quad i = 0, ..., 2^j - 1$$

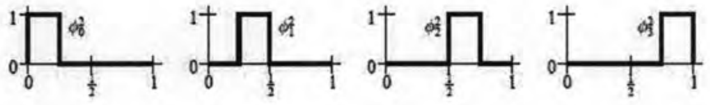


Figure 3.4: The box basis for V^2.

where i and j are indices in the intervals and levels

$$\psi(x) = \begin{cases} 1 & for\ 0 \leq x < 1/2 \\ -1 & for\ 1/2 \leq x < 1 \\ 0 & otherwise \end{cases}$$

Figure 3.5 shows the two Haar wavelets spanning W^1

We can describe the wavelet coefficients of the previous four original image $I(x)$ exam-

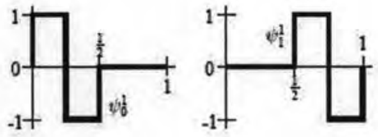
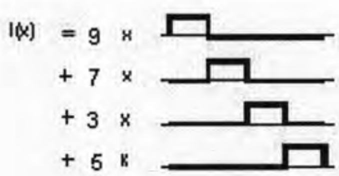


Figure 3.5: The Haar wavelets for W^1.

ple in the term of the linear combination of the box basis functions in V^2:

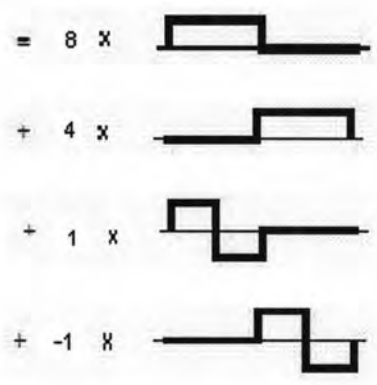
$$I(x) = c_0^2\phi_0^2(x) + c_1^2\phi_1^2(x) + c_2^2\phi_2^2(x) + c_3^2\phi_3^2(x)$$

A more graphical representation is



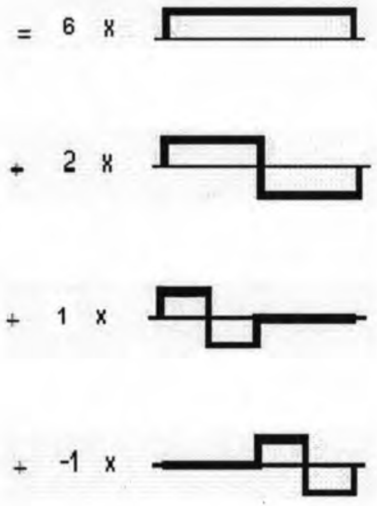
Note that the coefficients $c_o^2, ..., c_3^2$ are just the four original pixel values *9 7 3 5*. We can rewrite the expression for $I(x)$ in terms of basis functions in V^1 and W^1, using pairwise averaging and differencing:

$$I(x) = c_0^1\phi_0^1(x) + c_1^1\phi_1^1(x) + d_0^1\psi_0^1(x) + d_1^1\psi_1^1(x)$$



These four coefficients should look familiar as well. Finally, we will rewrite $I(x)$ as a sum of basis functions in V^0, W^0 and W^1:

$$I(x) = c_0^0\phi_0^0(x) + d_0^0\psi_0^0(x) + d_0^1\psi_0^1(x) + d_1^1\psi_1^1(x)$$



3.3.1 Two-dimension Haar wavelet transform

To obtain the standard decomposition of an image, we first apply the one-dimensional wavelet transform to each row of pixel values. This operation gives us an average value along with detail coefficients for each row. Next, we treat these transformed rows as if they were themselves an image and apply the one-dimensional transform to each column. The resulting values are all detail coefficients except for a single overall average coefficient. The algorithm below computes the standard decomposition [10].

```
Procedure Standard Decomposition (C: array[1..h,1..w] of reals)
- For row ← 1 to $h$ do
-     Decomposition(C[row,1...w])
- End for
- For col $\leftarrow$ 1 to $w$ do
-     Decomposition(C[1..h,col])
End procedure
```

The standard decomposition algorithm is call the following two pseudo code procedures accomplish this decomposition:

```
Procedure Decomposition (C: array[1..h] of reals)
- While $h > 1$ do
-     Decomposition Step(C[1..h])
-     $h \leftarrow h/2$
- End while
End procedure.
Procedure Decomposition Step (C: array[1..h] of reals)
- For $i \leftarrow 1$ to $h/2$ do
-     $C'[i] \leftarrow (C[2i-1]+C[2i])/\sqrt{2}$
-     $C'[h/2+i] \leftarrow (C[2i-1]-C[2i])/\sqrt{2}$
- End for
-     $C \leftarrow C'$
End procedure.
```

where h, w are the sizes of row an column in an image that varied by the decomposition step. Variables C and C' are the original image and its decomposition result.

The discrete wavelet transform provides information that is useful for texture analysis in an image [11]. In this work, we apply the single level of wavelet transform on the image and use the wavelet coefficients to be the texture information. Figure.3.6(a),(b), show a sample image of wavelet transformation where the original test image is shown in Fig.3.6(a) and the wavelet transform image is shown in Fig.3.6(b).

In this step, we prepared the training set of neural network by divide the im-

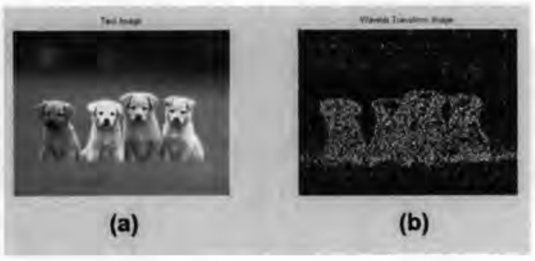


Figure 3.6: Sample images of wavelet transform, (a) The original test image. (b) The wavelet transform image.

age into 20-by-20 blocks. The number of rows and columns in each training image are r and c, then the size of each blocks is N_r x N_c, where $N_r = floor(\frac{r}{20})$ and $N_c = floor(\frac{c}{20})$. Then, the wavelet transformation was applied to the block images, the transformation results are the wavelet coefficients $c_{i,j}$, see Eq.3.2. We classify all of the blocks into two frequencies (High and Low) by using the threshold value computed from Eq.3.5. The wavelet coefficients in a small blocks in training set images comprise of the high and low spatial frequencies.

$$T = \sum_{k=1}^{K} (std._k(c_{i,j}))/K \tag{3.5}$$

where T is the threshold value that computed from the mean of the standard deviation value of the wavelet coefficients $c_{i,j}$ in every blocks from k = *1 to the total amount of blocks(K)*. Because of the high and low frequencies inside the in-focus and out-of focus blocks can be describes as the variance of data in each block. The high and low frequency are relating to the high and low variance. Actually, the variance equation is the square of standard deviation equation, then, we can replace the variance equation with the standard deviation equation for making the threshold value. The standard deviation equation (std.) is the mathematical function of the vector data and its average, the standard deviation can be express here.

$$std = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

where index i is varying from i = 1 to the number of x and $\overline{x}$ is the average of vector data x. The threshold will separate the standard deviation of wavelet coefficients in each block into two groups, high frequencies and low frequencies groups. Equation 3.6 describes the high frequencies region and low frequencies region classification.

$$\begin{aligned} &\textit{In-focus} = c_{i,j} \geq T \\ &\textit{Out-of-focus} = c_{i,j} < T \end{aligned} \tag{3.6}$$

When the image is in-focus, this region of image will have the high details of texture information. On the other hand, when the image is staying out-of-focus region the low details of texture information will be appeared. The high and low details of texture information can be described in the form of high and low spatial frequencies. Then, the high and low frequencies regions from Eq. 3.6 is equivalent to the in-focus and out-of-focus region images [12]. We applied the classification approach in Eq. 3.6 with the depth-of-filed image for classifying the image into two regions (in-focus and out-of-focus). *Algorithm*1 described the separation process of the in-focus and out-of-focus regions.

*Algorithm*1 : In-focus, Out-of-Focus Region's Images Classification

```
- For $n$ = 1 to the number of training set images loop.
-   Dividing the training image into small block amount to 20-by-20 blocks.
-   Compute the Wavelet transformation of the training image block by using Eq.(3.2).
-    For $q$ = 1 to 20-by-20
-      Compute threshold value by using Eq.(3.5).
-      Prepare the two arrays (in-focus and out-of-focus) that have the same size with the
-      training image($n$) .
-      Classifying all small blocks into two regions (in-focus and out-of-focus) by using
-       Eq.(3.6).
-       Fill the in-focus array with the in-focus block from the classification process in step 7.
-       Fill the out-of-focus array with the out-of-focus block from the classification process in
-       step 7.
-     End for.
- End for.
```

After the algorithm 1, we obtain the wavelet coefficients($c_{i,j}$) of the two regions, in-focus and out-of-focus regions. Usually. it has three regions (in-focus, blur-near and blur-far) in the depth-of-field image and we design the three neural network to train the three regions in this work. Thus, we used the in-focus and out-of-focus regions to guiding the researcher for considering the regions before separated the two regions into three regions.

3.4 Training Set Preparation

This section is the continuity process of the feature extraction process, the task of this process is to prepare and arrange the three training sets of spatial frequency feature before neural network training step. We used the in-focus and out-of-focus regions image from *Algorithm 1* as the source of training images. Since, our scope of work is proposed the method that order the region of image into three regions. Then, we must to preparing the three neural networks and training sets for training the three regions. The first of three training set regions is the in-focus regions and we can used the in-focus region products from the Algorithm 1 to be the training set. The last two training set regions can be obtain by separating the out-of-focus regions image from the Algorithm 1 into two regions, out-of-focus near and far. This consider step is made by human judgment. Now we have three training set images, then, we consider and define the in-focus and out-of-focus near and far region images in order to foreground, middleground and background regions by the researcher. In this step, we will obtain the three regions foreground, middleground and background that made by the human selecting from the in-focus, out-of-focus near and out-of-focus far region of images. Most of the DOF images appeared the in-focus regions in the foreground and middleground regions and a few in background region images. Thus, we set the proportion of the three position of in-focus region in the training set image as same as the general proportion numbers of DOF images, the proportion numbers is 40-40-20 images. The training set images are including 40 images of foeground region, 40 images of middleground region, and 20 images of background region. Figure 3.7 shows the sample of three types of training images in this work. The regions of in-focus positions shown inside the thin circle of Fig 3.7(a),(b) and (c) appeared in foreground, middleground and background, respectively.

The criterion for human to select the image to be a training image is the position of in-focus region. It has three kinds of the in-focus region position, appeared in the foreground, middle ground and background regions. The sample images in Fig.3.7 show the three kinds of training images separated by the positions of in-focus regions. The objective of training set preparation is to prepare the training image set into three

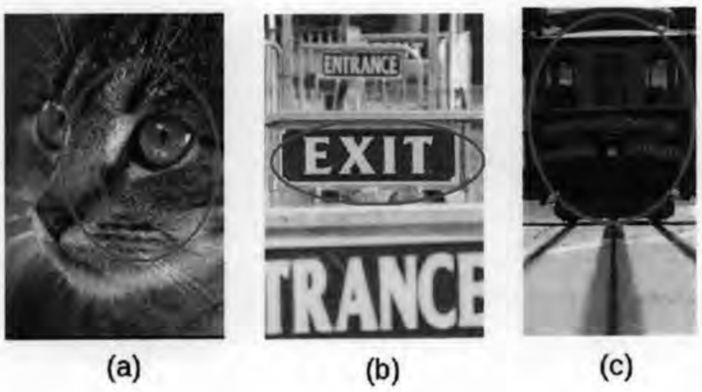


Figure 3.7: The sample of three types region's image depth's ordering. Each type of depth's ordering is specified by the position of in-focus region that appeared in foreground, middleground and bckground region image. (a) In-focus regions appeared in foreground. (b) In-focus regions appeared in middleground. (c) In-focus regions appeared in background.

regions which are foreground, middle ground and background regions. Then, we transform the three regions image to three feature vectors and three targets. The mechanism of transformation process is the standard deviation computation at each block in all regions of the foreground, middleground and background regions image. The result of this process is a feature's vector that can be described as:

$$feature\ vector(1, q) \ = \ std.(Wt - Coef.)_q \tag{3.7}$$

where Equation.3.7 is the standard deviation operation of the wavelet coefficients $(Wt - Coeff.)$, we use this operation to transform every blocks from q = 1 to the total number of blocks (400), the size of the transformation results (*feature vector*) is *1-by-400*. *Algorithm*2 is described the three training and target vectors building processes.

*Algorithm*2 : Training and Target Vector Building

```
1. For n = 1 to the number of training set images loop.
2.   Dividing the foreground, middle ground and background regions image into the 20-by-20
-    blocks.
3.   Prepare the three training's vectors and the three target's vectors size 1-by-400.
4.     For q = 1 to 20-by-20.
5.       Computed the standard deviation value in every blocks of the wavelet transform image
-        in foreground, middleground and background regions by using Eq.(3.7).
6.       Save each standard deviation value of each block in foreground, middleground and
-        background regions from step 5 to foreground, middleground and background training's
-        vectors.
7.       Target's vector generating by examined the standard deviation value of foreground,
-       middleground and background training's vectors from step 6,
-       $If$ the $training's\ vector_q \geqslant$ zero
-       $Then$ the $target's\ vector_q =$ to 1
-       $Else$ the $target's\ vector_q =$ to $-1$
-       *End if*
8.      Save each target's vector of each block in foreground, middleground and background
-      regions from step 7 to foreground, middleground and background target's vectors.
9.      *End for.*
10. *End for.*
```

The training and target vectors from *Algorithms 2* are not ready to be a training set for neural network learning. Because the extracted feature of the blurred regions in the image are similar. We must make a different in the orders of the vector before the network learning. The training set arrangement was used to solve this problem. We arranged all of foreground, middle ground and background regions of training vectors. The algorithm for training set arrangement is described as follows:

$Algorithm3$: Training and Target vectors Arrangement

```
1. For n = 1 to the number of training vectors
2.    Defined fore[1 to length of feature] = foreground training vector[1 to legth of feature,n].
3.    Defined mid[1 to length of feature] = middleground training vector[1 to legth of feature,n].
4.    Defined back[1 to length of feature] = background training vector[1 to legth of feature,n].
5.      For q = 1 to 400.
6.       For qq = 1 to 400.
7.        push fore(qq) to arrange-fore[1,q].
8.        arrange arrange-fore[2 to qq,q] with fore[1 to qq].
9.        arrange arrange-fore[qq+1 to 400,q] with fore[qq+1 to 400].
10.        push mid(qq) to arrange-mid[1,q].
11.        arrange arrange-mid[2 to qq,q] with mid[1 to qq].
12.        arrange arrange-mid[qq+1 to 400,q] with mid[qq+1 to 400].
13.        push back(qq) to arrange-back[1,q].
14.        arrange arrange-back[2 to qq,q] with back[1 to qq].
15.        arrange arrange-back[qq+1 to 400,q] with back[qq+1 to 400].
16.       End for.
16.      End for.
18.     define start length t1 = ((n-1)*400)+1.
19.     defined stop length t2 = (n*400).
20.     foreground training set(1 to 400,t1 to t2) = arrange-fore(1 to 400,1 to 400).
21.     middle ground training set(1 to 400,t1 to t2) = arrange-mid(1 to 400,1 to 400).
22.     background training set(1 to 400, t1 to t2) = arrange-back(1 to 400,1 to 400).
23. End for.)
```

For examples, supposed we have the wavelet coefficients array size 1-by-10 that obtained from the wavelet transform.

[0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0]

Step 1: Check the feature vector lengths, this example the length is equal to 10.

Step 2: Rearrange the feature vector
- for i = 1, the rearranged vector are:
[0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0]
- for i = 2, the rearranged vector are:
[0.2 0.1 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0]
- for i = 3, the rearranged vector are:
[0.3 0.1 0.2 0.4 0.5 0.6 0.7 0.8 0.9 1.0]
- for i = 4, the rearranged vector are:
[0.4 0.1 0.2 0.3 0.5 0.6 0.7 0.8 0.9 1.0]
- for i = 5, the rearranged vector are:
[0.5 0.1 0.2 0.3 0.4 0.6 0.7 0.8 0.9 1.0]
- for i = 6, the rearranged vector are:
[0.6 0.1 0.2 0.3 0.4 0.5 0.7 0.8 0.9 1.0]
- for i = 7, the rearranged vector are:
[0.7 0.1 0.2 0.3 0.4 0.5 0.6 0.8 0.9 1.0]
- for i = 8, the rearranged vector are:
[0.8 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.9 1.0]
- for i = 9, the rearranged vector are:
[0.9 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 1.0]
- for i = 10, the rearranged vector are:
[1.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9]

Step 3: The results after rearrange vector is the training set for neural network learning.

[0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
0.2 0.1 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
0.3 0.1 0.2 0.4 0.5 0.6 0.7 0.8 0.9 1.0
0.4 0.1 0.2 0.3 0.5 0.6 0.7 0.8 0.9 1.0
0.5 0.1 0.2 0.3 0.4 0.6 0.7 0.8 0.9 1.0
0.6 0.1 0.2 0.3 0.4 0.5 0.7 0.8 0.9 1.0
0.7 0.1 0.2 0.3 0.4 0.5 0.6 0.8 0.9 1.0
0.8 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.9 1.0
0.9 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 1.0
1.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9]

3.5 Network Architecture

Region's image depth ordering neural network model is constructed using MLP networks with back propagation learning algorithm. The model consists of three neural visual networks for foreground, middleground and background regions detection. Three essential region features,i.e., foreground regions, middleground and background regions, are considered. Each feature will be located using one MLP network, then the three MLP networks must be introduced for all features in each neural visual network. Figure 3.8(a) and (b) shows the overview

of the three neural visual networks learning and testing steps in this work. The neural network learning and testing processes block diagram are shows in Fig. 3.1. Each of the neural visual network used in this work is a multilayer perceptron (MLP) composed of an input layer, a double hidden layers with 20 nodes, and an output layer. Each layer is fully connected to the succeeding layer. The outputs of nodes in one layer are transmitted to nodes in another layer through links. The link between nodes indicates flow of information during recall. During learning, information is also propagated back through the network and used to update connection weights between nodes.
Let o_j be the output of the previous layer and w_{ij} the connection weight between the i^{th} node in one layer and j^{th} node in the previous layer. The total input to the i^{th} node of a layer is:

$$net_i = \sum\nolimits_j w_{ij} o_j$$

In this research, we used the two hidden layers with the hyperbolic tangent function

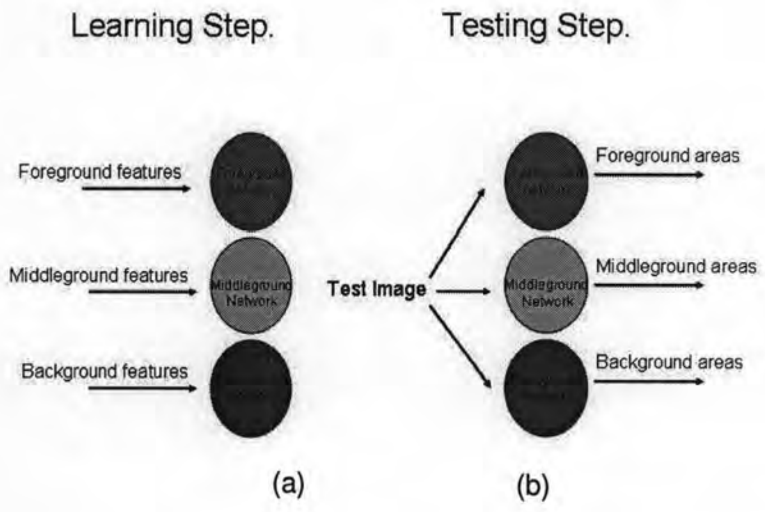


Figure 3.8: The overview of the three neural visual networks learning and testing steps. (a) The three neural networks learning step. (b) The three neural networks testing step.

as an activation function in every networks. The topology of the multi-layer perceptron neural network is illustrated in Fig. 3.9.

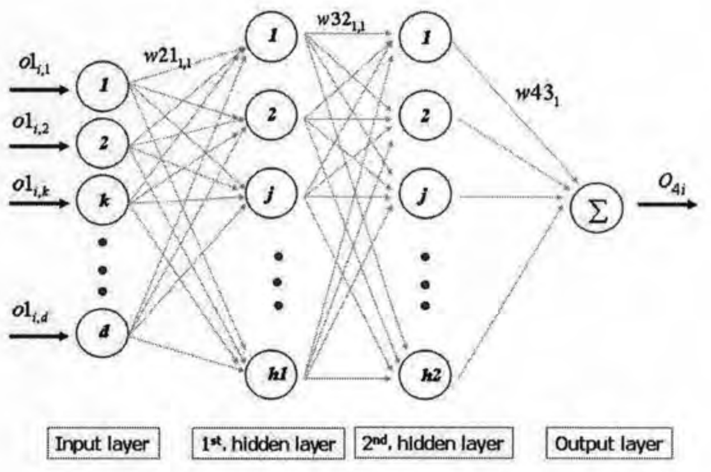


Figure 3.9: The topology of the neural visual network.

$\mathbf{o1_i}$	$=$	$[o1_{i,1}, \ldots, o1_{i,d}]^T$	: input vector of the i^{th} pattern
$\mathbf{o2}$	$=$	$[o2_1, \ldots, o2_h]^T$	: output vector from 1^{st} hidden unit
$\mathbf{o3}$	$=$	$[o3_1, \ldots, o3_h]^T$	: output vector from 2^{nd} hidden unit
$\mathbf{o4_i}$	$=$	$[o4_{i,1}, \ldots, o4_{i,h}]^T$	: output vector of the i^{th} pattern
$\mathbf{w21}$	$=$	$[w21_{1,1}, \ldots, w21_{h,d}]^T$	: weight vector from input layer to 1^{st} hidden layer
$\mathbf{w32}$	$=$	$[w32_1, \ldots, w32_h]^T$	: weight vector from 1^{st} hidden layer to 2^{nd} hidden layer
$\mathbf{w43}$	$=$	$[w43_1, \ldots, w43_h]^T$	: weight vector from 2^{nd} hidden layer to output layer
d			: number of input dimension
$h1$			: number of 1^{st} hidden unit
$h2$			: number of 2^{nd} hidden unit
p			: number of learning pattern
i, j, k			: indices of dimensions

In the learning phase for such a network, we present the training pattern $T = O1_i$, where $O1_{i1}$ is the d^{th} node in the input layer, and ask the network to adjust the weights in all the connecting links such that the desired outputs D_k are obtained at the output nodes. Let O_k be the evaluated outputs of the network in its current state. For a training pattern the mean squared error of the system can be written

$$mse = \frac{1}{Q}\sum_{k=1}^{Q}(D_k - O_k)^2$$

The error is calculated as the difference between the target output and the network output. The goal is to minimized the average of sum of these errors. The generalized delta-rule learning algorithm is applied to adjust the weights such that the error E is a minimum. A detailed derivation of the learning procedure can be found in reference [27]. Table 3.1 summarizes the

T 24699354

details of the neural visual networks in this work.

Table 3.1: The summary of the neural visual network in this work.

Items	Details
Number of networks	Three networks
Type of networks	Multi-layer perceptron
Type of learning rule	Backpropagation
Weight adjustment	Gradient Descent method
Type of activation function	The hyperbolic tangent activation function
Type of error measure calculation	The mean square error
Number of hidden layer	Double hidden layers
Number of input nodes	Four hundred nodes
Number of the first hidden nodes	Twenty nodes
Number of the second hidden nodes	Twenty nodes
Number of output nodes	One node
Learning rate	0.8
Error goal	0.001