Chapter 4. The Classification of Species and Colors of Finished Wooden Parts UsingRBFNs

4.1 Introduction

In Chapter 1, an introduction was given to the species and color classification problem of kitchen cabinets. The features of RBFNs, explained in Chapter 3, make them suitable for use in our classification problem. During this research, a number of methods are used with regard to choosing which network structure is used, how the input features are extracted from the original data, and how many RBF centers are used. Two kinds of network structures are employed. One of them is the traditional generalized RBFN, as shown in Figure 3.3. The second type of network is a modified version of the first type of network. A detailed description of the proposed structure is given in the following section.

In terms of input features used, four different input feature extraction techniques are used. First, average values of red, green, and blue in each image are used as the input of the RBFN. Second, *image partitioning* is employed in addition to the average values of red, green, and blue are calculated for four different regions in images. Then, these values are used as the input of the RBFN. Third, the covariance matrix of each image is involved in the feature extraction. In this case, average values of RGB colors and the covariance matrix of each image are employed as the inputs of the RBFN. Finally, the histogram of each image serves as the input of the RBFN. All input feature extraction methods are discussed in the following sections. The number of centers used in the RBFN is also a concern in the methods. The number of centers used in the RBFN takes the values 8, 23, 24, and 64 during this research.

Section 4.2 explains the two different network structures used in this research, the traditional generalized RBFN and the proposed *one-dimensional* RBFN. The different types of input feature extraction methods are introduced in Section 4.3. In the input feature extraction process, the average values of red, green and blue of images, the covariance matrix of these three colors over images, and the histograms of the images are examined. Section 4.4 gives all methods and their results accompanying the comparisons and discussions about these results. All results will be given in Section 4.4. The next section talks about the two different network structures used during this research.

4.2 Network Structures

In this section, the traditional generalized RBFN structure and proposed one-dimensional RBFN structure are explained in detail. Because the traditional generalized RBFN was explained in Section 3.7, we will not give many details here. The following section reintroduces the traditional generalized RBFN employed in our specific problem introduced in Chapter 1.

4.2.1 The Traditional Generalized Radial Basis Function Networks

In some methods, the generalized RBFN structure is employed. A detailed explanation for this network structure is given in section 3.7. The structure of the generalized RBFN is shown in Figure 4.1 for the specific problem we have.

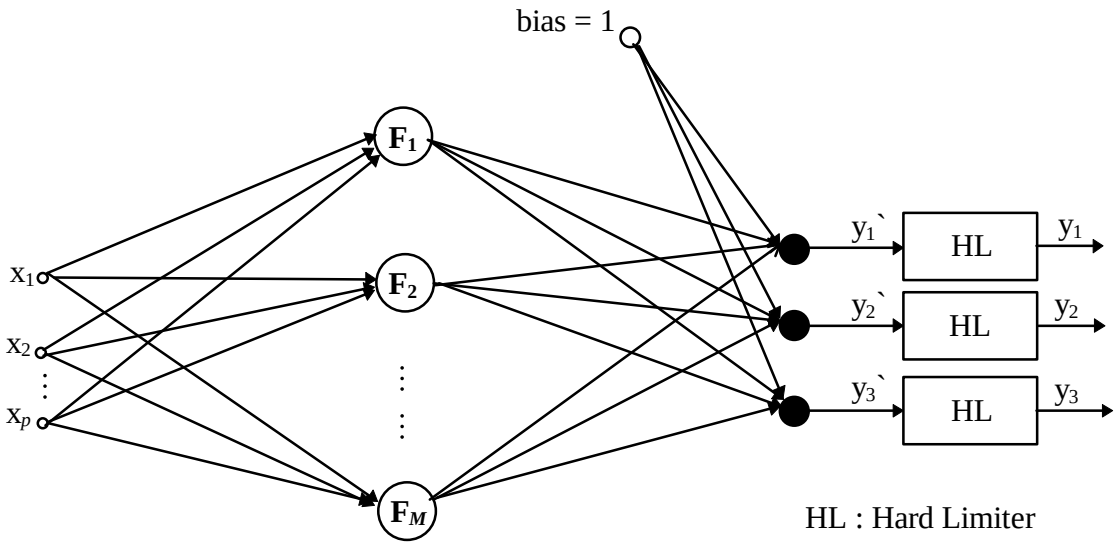

Figure 4.1. The traditional Generalized RBF Network Structure

In this work, all the network structures have three outputs because we have eight classes of wood. Binary numbers are employed in the output of the network. Therefore, the minimum number of outputs we can have is three. Using higher number of outputs (non-encoded) was tried but it did not provide any improvement in the results. To keep the computational complexity at a minimum we used three outputs in both the traditional generalized RBFN and the proposed one-dimensional RBFN. A typical output can be given as (0 0 0) representing the first class. All desired outputs are between (0 0 0) and (1 1 1). To have these values in the output of the network, a hard limiter is placed after each output. The hard limiter decides whether the output will be 1 or 0. If the output of the RBFN is less then 0.5, it decides that the output is 0. Otherwise, it decides that the output is 1. The hard limiter can be design to determine the "unknown" samples by dividing the interval [0,1] into 3 regions. For example, the values from 0 to 0.3 can be considered 0 and the values from 0.7 to 1.0 can be considered 1. The values between 0.3 and 0.7 can be considered unknown. The decision regions can be adjusted depending on the application.

The form of RBFs in the network is the Gaussian radial basis function as explained in equation (3.66). The form of the RBFs in our network structure can be given by

$$\mathbf{G}_{ji} = \exp[-\frac{1}{2}(\mathbf{x}_j - \mathbf{c}_i)^T \Sigma^{-1}(\mathbf{x}_j - \mathbf{c}_i)] \quad j = 1,..,N \quad i = 1,..,M \tag{4.1}$$

where $\mathbf{x}_j$ is the input vector, $\mathbf{c}_i$ is the ith RBF's center determined by the k-means algorithm, Σ is the covariance matrix of the input features of the images, and N, M are the number of training samples and the number of centers in the RBFN, respectively. The covariance matrix can be expressed by

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \sigma_p^2 \end{bmatrix} \tag{4.2}$$

where σ_i is the standard deviations of ith input feature obtained from the images and p is the number of input features in the input vectors. Off-diagonal elements are zero because the samples are independent from each other. Because of this independence, cross-covariance values become zero. The dilations of the RBFs are calculated by a statistical method in the following form:

$$\sigma_i = \left(\frac{1}{N_i}\right)\sum_{\mathbf{j=1}}^{N_i}(\mathbf{x}_j - \mathbf{c}_i)^2, \qquad i = 1,\ldots,p \qquad j = 1,\ldots,N_i \tag{4.3}$$

where N_i is the number of inputs in the$i$th class.

As given in Section 3.7, the weights of the output layer of the network are calculated by the following equations:

$$\mathbf{G}^+ = (\mathbf{G}^T \cdot \mathbf{G})^{-1} \cdot \mathbf{G}^T \tag{4.4}$$

and

$$\mathbf{w} = \mathbf{G}^+ \cdot \mathbf{D}_{\mathbf{out}} \tag{4.5}$$

where $\mathbf{G}$ is the N-by-(M+1) matrix containing the outputs of the hidden layer for all N input vectors; $\mathbf{G}^+$ is the (M+1)-by-N pseudoinverse of the matrix $\mathbf{G}$; $\mathbf{D}_{\mathbf{out}}$ is the N-by-3 matrix containing the desired outputs of the network response to N input vectors; and $\mathbf{w}$ is the (M+1)-by-3 output weight matrix of the RBFN. This calculation of the weights is equivalent to a least-mean-square algorithm, because the pseudoinverse gives the optimal weights in the least-mean-square sense. After we calculate the weights, the test images are tested by using the following equation:

$$\mathbf{D}_{\mathbf{out}} = \mathbf{G} \cdot \mathbf{w} \tag{4.6}$$

where $\mathbf{G}^+$ is the N_t-by-(M+1) matrix containing the outputs of the hidden layer for N_t input vectors obtained from the test samples and $\mathbf{D}_{\mathbf{out}}$ is the N_t-by-3 matrix containing the outputs of the network in response to the test images. As explained in Chapter 1, we have 480 images captured by a video camera; 95 (N) of them are used for training and 385 (N_t) of them are used to test the network designed. The following section explains the proposed network structure giving the reasons for changing the network structure.

4.2.2 Proposed One-dimensional RBFN Structure

The traditional generalized RBFN network has multivariate Gaussian radial basis functions. In these functions, the input vector of each function is multidimensional. The output of the function is affected by each element in the input vector. Since each radial basis function takes multidimensional input and gives one dimensional output, this step results in a loss of information. Moreover, the outputs of the functions may not be sensitive enough to each element of the input vector equally. For example, if one of the elements of the input vector is very high in quantity, the output will be determined by this specific element of the input vector. Thus, the other elements of the input vector will not be able to affect the output and the function will become insensitive to the other elements of the input vector.

There is a need for a new type of network structure that does not decrease the dimensionality of the input in the hidden layer. Because of the reasons in the previous paragraph, a one-dimensional RBFN structure is proposed to ease the difficulty of having outputs equally sensitive to each element of the input vector. The proposed one-dimensional RBFN structure can be seen in Figure 4.2. In this case, instead of having M RBFs, the network has $M{\cdot}p$ RBFs because each input feature of the input vector has M RBFs in the hidden layer of the network. This proposed network structure is called a one-dimensional RBFN (O-RBFN) because all radial basis functions have one input and one output instead of having p inputs and one output. That is, all radial basis functions in the network are one-dimensional radial basis functions instead of being multidimensional radial basis functions. In the proposed network structure, $\mathbf{G}$ is a N_t-by-($M\,p$+1) matrix, while $\mathbf{G}^+$ is a ($M\,p$+1)-by-N_t matrix. Thus, the weight vector $\mathbf{w}$ has ($M\,p$+1) rows and 3 columns. The desired output vector $\mathbf{D_{out}}$ has the same form as in the traditional generalized RBFN

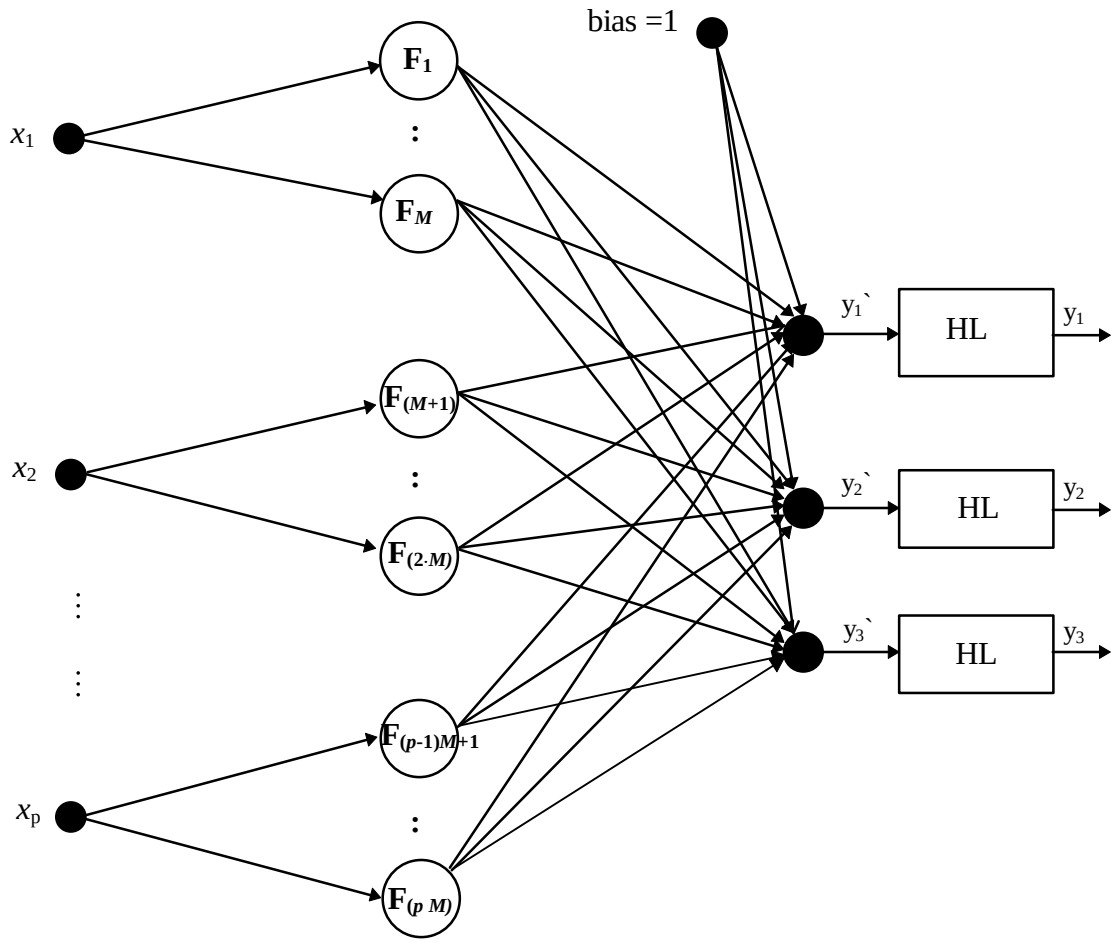

Figure 4.2. The proposed one dimensional radial function network. Every radial basis function has one input and gives one output in response to this input. For example first radial basis function has input x_1 and output $\mathbf{F_1}$.

The form of the outputs of the hidden units can be given by

$$\mathbf{F}_k = \exp(-(x_1 - c_{1k})^2 / \sigma_{1k}^2, \tag{4.7}$$

$$\mathbf{F}_{k+M} = \exp(-(x_2 - c_{2k})^2 / \sigma_{2k}^2, \tag{4.8}$$

$$\vdots$$

$$\mathbf{F}_{k+(p-1)M} = \exp(-(x_p - c_{pk})^2 / \sigma_{pk}^2, \quad k = 1, 2, ..., M \tag{4.9}$$

where $\mathbf{c}_{ik}$ is the center of the kth RBF for the ith element of the input vector. The dilation of the kth RBF for the ith element of the input vector is represented by σ_{ik}. The symbol x_i represents the ith element of the input vector.

The output layer weights are calculated by using the same equations as in the previous section. The only difference would be the dimensions of the matrices **G** and **w** because we have $M\,p$ centers in this network structure. The number of centers in this network structure is p times more than the traditional generalized RBFN structure has. In

Section 4.2, we talked about the network structures we used in our research. The next section introduces the types of input feature extraction methods.

4.3 Input Features

Having discussed the network, the types of input feature extraction methods are examined. This section talks about what types of input features are extracted from the images. The simplest feature extraction method is calculation of the average values of red, green and blue in each image. This feature extraction method is introduced in Section 4.3.1. In a classification system, classification can be performed better if there is more information in the input. Section 4.3.2 introduces the image partitioning as a feature extraction method to have more input features in the input vector. Section 4.3.3 explains another input feature extraction method that calculates the covariance matrix of the red, green, and blue in the images. Finally, the last input extraction method, which uses color histograms, is explained in Section 4.3.4.

4.3.1 Average of Colors

In some methods, the average values of red, green, and blue pixels in each image are used as the input vectors of the network. In our implementation, a Matlab m-file is written to read the image into a matrix and calculate the averages of colors of each pixel in the image. The Matlab m-file train.m calculates the average values of red, green, and blue pixels in the images and saves these values into a Matlab metafile. When the execution of the file train.m is finished, the metafile has three values for each image. At the beginning of the main program, this file is loaded to provide the inputs vectors of the RBFN.

4.3.2 The Image Partitioning

As mentioned in the previous section, having more features in the input vector might give better results. To extract more features from the images, an image partitioning method is used, dividing the images into four pieces, and then the average values of red, green and blue are calculated for each partition. Thus, in the input vector, there are 12 features. The image partitioning is represented in Figure 4.3. The reason for the partitioning is explained in the next paragraph. Partitioning the images makes it possible to identify some images that would not be correctly identified without partitioning. Figure 4.4 shows how the image partitioning can be useful in the sense that images having the same average values of colors can be separated.

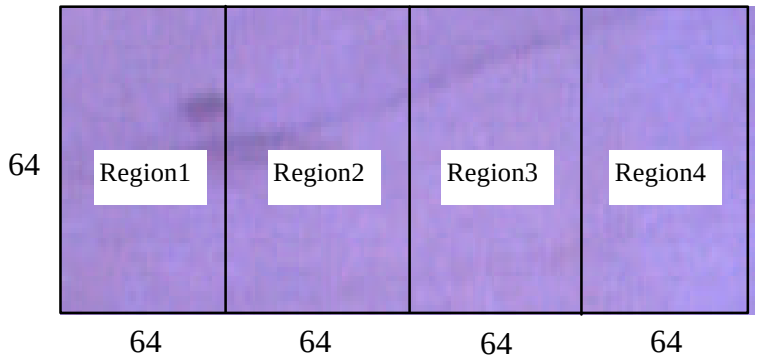

Figure 4.3. The representation of the image partitioning on an image of maple-frost kitchen cabinet.

After image partitioning is applied, the input vector takes the following form:

$$\mathbf{x} = [\ \mathbf{red1\ green1\ blue1\ .....\ red4\ green4\ blue4}]^T$$

where **x** is a 12-by-1 vector obtaining three values for each region. The first three values in the input vector represent the average values of red, green, and blue pixels in the first partition of the image. Similarly the second three, third three, and last three represent the average values of the colors in the second, third and fourth regions respectively.

To express the importance of the image partitioning, let us consider two gray scale images shown in Figure 4.4. In this case, our fundamental colors are black and white. The average values of the black and white are our input features. The first half of the image1 is black while the second half of the image is white. Image2 has exactly the opposite colors of image1's colors. When there is no image partitioning, the input feature vectors extracted from both image1 and image2 are equal to $\mathbf{x} = [\ 0.5\ 0.5]^T$. Thus, these two images are considered to be the same because of the averaging process, even though they are different images. When image partitioning is employed, the input feature vectors extracted from the images are different from each other as shown in Figure 4.4.b. In this case, image 1 is represented by the vector $\mathbf{x}_1 = [\ 1\ 1\ 0\ 0\ ]^T$ while image2 is represented by the vector $\mathbf{x}_2 = [\ 0\ 0\ 1\ 1]^T$. Because these two vectors are different from each other, the network can classify these two images easily.

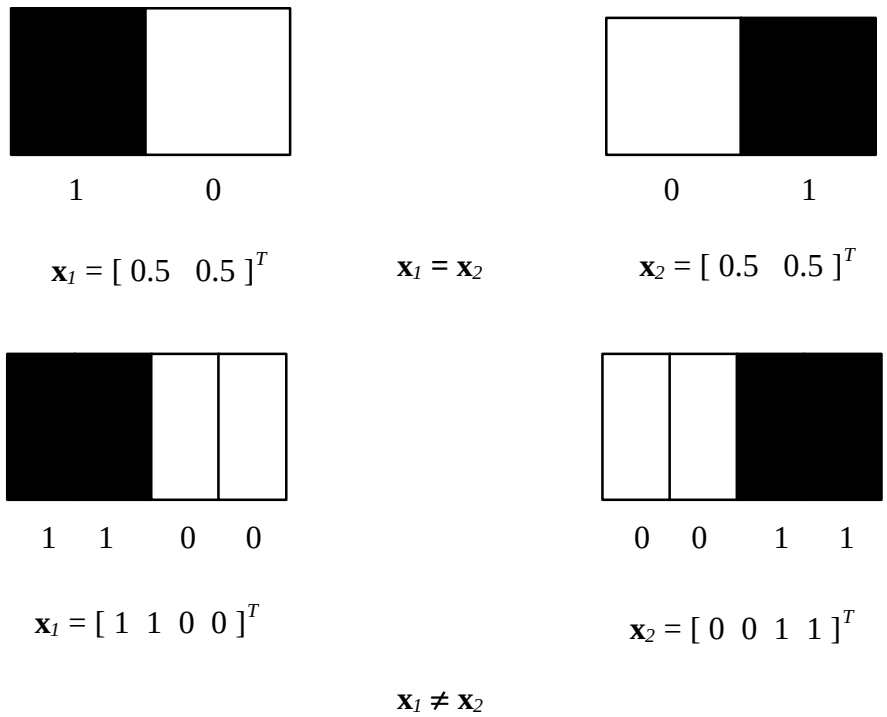

Figure 4.4. The effect of the image partitioning on two gray scale images; a) without image partitioning; b) with image partitioning.

Image partitioning is useful for certain type of images as shown in Figure 4.4. For some images, it may not be useful because the success of the image partitioning depends on how the image partitioning is applied. Especially, image partitioning will not give good results for images that have opposite geometrical shapes. For example, if the two images have been divided into four pieces horizontally, the image partitioning would create equal input vectors. Without the image partitioning, input vectors would be $\mathbf{x}$ = [.5 .5]. Similarly, using the image partitioning, the input vectors would be $\mathbf{x}$ = [.5 .5 .5 .5]. The next section introduces another feature extraction method that considers the variations of the colors in the images.

4.3.3 Usage of the Covariance Matrix

The usage of the covariance matrix of the red, green and blue colors in the input vector may improve the results even more because it has the information of the color variations in the images. First, the standard deviations of each color are used as input features with average values of the colors. In this case, the number of input features becomes six: three average values of the colors and three standard deviations of the colors. Second, the covariance matrix of the three colors in each image is calculated and used in the input vector. The covariance matrix has nine elements. Some of the elements have the same values in the covariance matrix. Since upper and lower diagonal elements are related to the square root of the diagonal elements of the diagonal elements, using the covariance matrix does not provide a lot of additional information. To calculate the

covariance matrix, the Matlab command "cov" is used. This command gives the covariance matrix of the three variables such as red, green, and blue. The covariance matrix has the following form:

$$\Sigma = \begin{bmatrix} \sigma_{rr} & \sigma_{rg} & \sigma_{rg} \\ \sigma_{rg} & \sigma_{gg} & \sigma_{gb} \\ \sigma_{rg} & \sigma_{gb} & \sigma_{bb} \end{bmatrix} \tag{4.10}$$

$$\sigma_{rr} = \sigma_r^2, \ \sigma_{gg} = \sigma_g^2, \ \sigma_{bb} = \sigma_b^2,$$

$$\sigma_{rb} = \sigma_r \sigma_b, \ \sigma_{rg} = \sigma_r \sigma_g, \ \sigma_{bg} = \sigma_b \sigma_g$$

In this case, the input vector has the following form:

$$\mathbf{x} = [\ Ra \ \ Ga \ \ Ba \ \ \sigma_{rr} \ \ \sigma_{gg} \ \sigma_{bb} \ \sigma_{rb} \ \sigma_{rg} \ \sigma_{bg}\ ]^T,$$

where *Ra*, *Ga*, and *Ba* are the average values of red, green, and blue, respectively.

4.3.4. Histogram

In addition to using average values of the three colors and the covariance matrix, the histograms of the images can also be used as inputs of the RBFN. The calculation of the histogram is done by using the uniform histogram method. Basically, the maximum range of each color (in our case 0-255) is divided into certain number of sections. Each section is called a histogram bin. Then the number of pixels in each bin is calculated. Finally, the number of pixels in each bin becomes an element of that image's histogram vector. The length of the histogram vector depends on the number of bins used. In this thesis, the length of the histogram vector is usually 24. The first eight elements of the vector represent red. The second eight elements of the vector represent green. The last eight elements of the vector represent blue. Figure 4.5 shows the histogram vector and the histograms of the red, green and blue colors for a sample image.

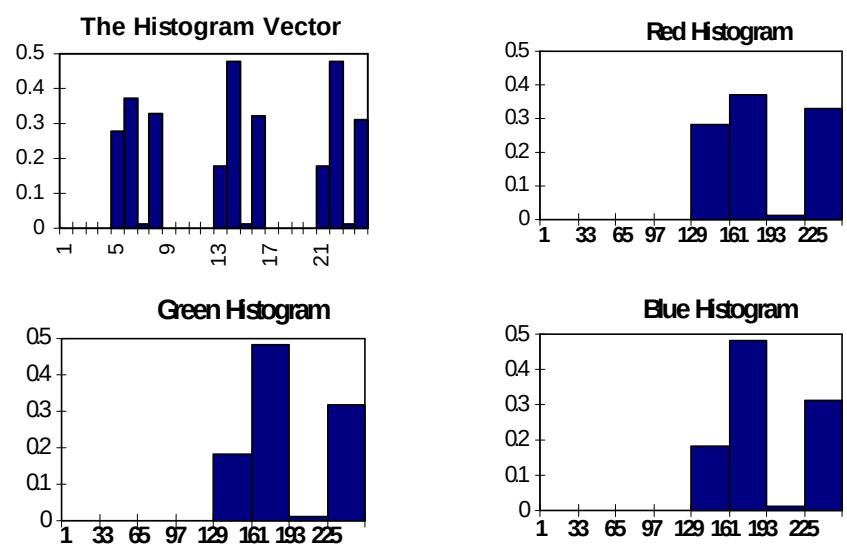

Figure 4.5. The representation of an example input to the RBFN: (a) Input vector with histogram representation; (b) red histogram; (c) green histogram; (d) blue histogram

In Figure 4.5, figure (a) shows the elements of the input vector obtained by th e histograms of the colors. The histogram vector has the values of all three color histograms. Figure (b) represent the histogram of the red pixels in the image of a maple-frost kitchen cabinet. Similarly, Figure (c) and Figure (d) represent the histogram of the green and blue pixels in the same image. These histogram values are normalized values. The maximum value that a histogram bin can take is 1 since each value is divided by the number of pixels in the images.

Increasing the number of centers can be useful for achieving better results because the weight matrix in the RBFN will be more nearly square. In the previous cases, only eight centers are used (each class has one center). To have more than one center for each class, the Matlab code of the *k*-means algorithm is modified. Since there will be more than one center for each class, the data need to be rearranged depending on centers. The function kmeann.m finds the centers and rearranges the data starting with the first center and class. Kmeann.m is modified version of thekmean.m.

This section explained the types of feature extraction methods used in this research such as averaging, image partitioning, using elements of the covariance matrix and histograms of the images. The next section introduces eleven different methods that differ from each other in terms of what type of network structure they used, what type of input feature extraction method is employed and how many RBFs they contain. The results of each method and the comparisons of the results are also presented.

4.4 Results

This section gives the descriptions of the methods used and provides some comparisons between the results. Methods are explained in terms of their network structure, the type of input feature extraction method used and the number of RBFs employed in the network. Method I and II differs from each other in terms of what type of input feature extraction method they used. Method III introduces an increase in the number of RBFs employed in the network. Methods IV, V, and VI introduce a new type of network structure (O-RBFN), shown in Figure 4.2, with different types of input feature extraction method. The histograms of the images are introduced in Methods VII and VIII. Methods IX, X, and XI use the histogram of the images. In these three methods, the fixed dilation is used in the RBFs in the network. These methods differ from each other in terms of how many input features they used. The following paragraphs provide detailed information about the methods.

Method I. In the first method, eight centers are used in the network with eight classes. The traditional (multidimensional RBF) structure is employed. Additionally, the average values of the color in the image are used as input. The network has three inputs, eight hidden units and three outputs. Since there are eight classes to identify, at least three output nodes are needed. This method gives **55.064%** correct classification of 385 test samples. This result is a surprisingly poor result. The possible reason for this result is that the average values of red, green, and blue do not provide enough information to manages good classification. Therefore, we might consider using more features in the

input vectors. The following method introduces the image partitioning and extracts more input features from the images.

Method II. Because the first method gives unsatisfactory results, image partitioning is employed in the second method. After image partitioning, the network has an input vector with 12 features as described in Section 4.3.2. The network structure and the number of centers used are the same as with Method I. The only difference from Method I is that the number of input nodes is 12 for Method II. The network therefore has more information about the images than in Method I. After using image partitioning, the result is improved by about **10**%. The correct classification is found to be **66.75**% with this method. The result in the second method is very near to the results found in the work by Zhao [1]. His result was **67.0**% correct classification for the case that average colors are used as input features with minimum distance classifier. Because there are more input features in the second method than in the first method, the second method is more computationally complex. The number of input features can be further increased, but in that case there will be more computation. One of the limitations is computation time because the wooden cabinets are continuously moving on a conveyor. Because increasing the number of input features results in more computational complexity, the next method introduces an increase in the number of RBFS used.

Method III. In Method III, the traditional generalized RBF network is employed with three input features, the average values of red, green, and blue pixels in the images. This method differs from Method I in the sense that it employs more centers than Method I. The number of centers in the network is increased up to 23 step by step. After having 23 centers, it was not logical to increase the number of centers in the network again because we have 95 training samples. In this case, every center and dilation is calculated by using three or four samples. If we used more centers, then there would not be enough samples to calculate dilations of the RBFs properly. Since we calculate the dilations using a statistical method, expressed in equation (4.3), we need to have enough samples to provide a good guess for the dilations. Increasing the number of centers helps the network to define classes more accurately. Instead of using one RBF (center) for each class, more than one RBF is used for each class. This helps to define the class regions more accurately. After using 23 with three input features, the average values of red, green, and blue pixels in the images, the results got even better. Method III performed with 67.79% correct classification. Even though this result is better than results in [1], better results might be obtained by using a different RBFN structure. Three methods have already been introduced with the same network structure. Because of the computational complexity, we cannot increase either the number of centers or the number of input features. Therefore, the new network structures should be defined. The following paragraph introduces a new type of network structure.

Method IV. Because the results did not get better using the multidimensional RBFN, illustrated in Figure 4.1, we next consider changing the network structure. In the new network structure, every input feature has separate RBFs and centers. The new network structure is presented in Figure 4.2. This network structure can be called a one-

dimensional radial basis function network (O-RBFN) because all radial basis functions have one input and one output instead of having three inputs and one output. As explained in Section 4.2.2, in the traditional RBFN structure, the outputs are not sensitive for each input feature in the input vector. In Method IV, the network has three input features, the average values of red, green, and blue pixels in the images. The number of centers used for each input feature is eight and the entire network has 24 centers. After applying the new network structure to the problem, results were better than the results in Methods I, II, III. The percent correct classification is **72.2**% using method IV. This result is **5**% higher than the result in [1] with three input features and minimum distance classifier.

Method V. In this method, the number of input features is increased up to six by calculating the standard deviations of red, green and blue colors in each image. Using standard deviations of the colors gives the network more information about the images. Now, the input vectors have six features: three features from the average values of red, green, and blue pixels in the images; and three features from the standard deviations of these colors in the images. The one-dimensional RBFN structure is employed. Eight centers are used for each input feature in this method for less computational complexity. The percent correct classification is **5**% more than the method IV performed. The network identified**77.66**% of the images correctly

Method VI. Because it is assumed that more information gives more accurate results, every element of the covariance matrix can be used as an input feature. Using these information, the input vector has three more variables in it. Now we have nine input features including the cross-covariance values of the colors as explained in Section 4.3.3. The detailed input structure is explained in the previous section. After using these new variables, the network gave almost the same results. The percent correct classification over test images is **78.12%** in Method VI. Because the cross variances are related to the variances of each color, actually the cross-variances do not add a lot of information into the input vector. For this reason, it is natural that the network gives similar result. Increasing the number of features in the input does not improve the system performance significantly. This method also has the one-dimensional RBFN structure with nine input features and 72 (9x8) centers. The following paragraph introduces a new type of input feature extraction method.

Method VII. If these kinds of statistical methods are used to extract features of images, the maximum number of features that can be in an input vector is limited by nine because there are no more than six different variables in the covariance matrix. To have more information, other feature extraction methods should be used. The most popular and efficient way is by calculating histograms of the images. Histogram is also a type of statistical method. Detailed information about histograms is given in Section 4.3.4. In this method, the input vector contains 24 elements. Eight red histogram values, eight green histogram values and eight blue histogram values are employed in the input vectors. The uniform histogram method is used. Eight centers are used with the traditional generalized RBFN structure. With this configuration, the network gave **77.14**% correct

classification. This result is almost the same as given by Method V. Even though the result is not better than Method VI, it is better than the result in [1] with the same information. His result was **75.8**% correct classification using eight bins for each color. The result using RBFNs is almost **2**% better than the result using the Minimum Distance Classifier in [1].

The 2% of improvement on the results is not really significant. The reason for this is that the structures of the histograms are not entirely appropriate for radial basis functions. Because of the image structures, some of the bins in the histogram have zero values. When the program calculates the dilation of each class, some of the dilations go to zero because of the zeros in the histogram vectors as illustrated in Figure 4.5. The outputs of some of the radial basis functions always stay at infinity or a huge number because of the position of the dilations (σ) in the radial basis functions as expressed in equation (3.48). In other words, these large numbers are caused by division of zero or very small number because dilations are in the denominator in the radial basis functions. These large numbers will have much more influence in determining the outputs of the network than the other RBF's outputs. Whatever outputs of the other RBFs are, the network will probably give the same results, determined by those large numbers. At this point, the network becomes insensitive for some of the input features. The next paragraph introduces a way to get rid of this difficulty.

Method VIII. In this method, the standard RBFN network structure is employed with the same input features as Method VII. Because there is an insensitivity in the network, as explained above, white noise is added to input vectors in method VIII. In this way, the network will have strictly nonzero elements in most dilation vectors. This method gave some improvement in the results. The percent correct classification became **78.15**%. Because this particular histogram structure is not well suited to radial basis functions, there is no need to increase the number of features in the input vectors or to increase the number of centers. The methods with more centers or more input features did not give better results. That is why they are not mentioned as different methods here.

Method IX. In previous methods, the dilation of each class is calculated by a statistical method, expressed in equation (4.3). In this way, generally good results are obtained. In the methods where a histogram is used, the statistical method is not really efficient because some of the values go to zero. Thus, this creates insensitivity in the network. To avoid this insensitivity, the use of constant dilation for every radial basis function is preferred in method IX. Matlab's neural network toolbox has its own routines for RBFN with fixed dilation. The standard RBFN structure is used with original Matlab functions, "SOLVERB" and "SIMURB". In this method, the histogram values of the images using 8 bins for each color are employed. The number of centers used is 30. This number is determined by finding the best classification result. The program determines this number automatically minimizing the error between desired output and the network's output. Using the fixed dilation in the network causes an increase on the number of RBFs needed. To better define the classes, the network needs more RBFs because the network can not adjust the dilations of the RBFs. The results are very good for training data but

not really good for test data in method IX. The percent correct classification is **77.34**% which is almost the same as given in method VII. As we explained in method VII, some elements of the histogram vector are zero while some elements have high values. The large differences between the elements of the histogram vector cause difficulty in the network. The radial basis functions cannot fit this kind of variation in the input vector very well. The next paragraph explains how to overcome this problem.

Method X. An example of a histogram vector can be given as follows:
$\mathbf{x}$ = [13589 1234 500 5 0 0 0 0 13597 1225 499 6 0 0 0 0 13452 1300 490 12 0 0 0 0]^T.
As can be easily seen, although the first three elements are very large, the next five elements are very small. The next eight elements and the last eight elements have the same pattern. In this case, there are three peak points and there are three local minima in the histogram vector. It is hard to fit this kind of variation using RBFs with fixed dilations. Even though some of the dilations need to be zero because of the histogram structure, the network has the fixed dilations due to its structure. As we pointed out in previous paragraphs, the first eight elements represent the histogram of red portion of the image, the second eight elements represent the histogram of green portion of the image, and finally the last eight elements represent the histogram of blue portion of the image. To overcome this difficulty, a normalization process is applied to the histogram vectors but the normalization process was not able to ease this problem either. Then, we tried using only the histogram of one color as the inputs of the network. When the histograms of only the red portion of the images are used, the network gave **79.74**% correct classification with 21 radial basis functions or centers. When the histograms of only the green portion of the images are used, an **80.25**% correct classification is obtained with 19 centers. Finally, when the histograms of only the blue portion of the images are used, the network performed an **81.81**% correct classification with 16 radial basis functions. As can be easily seen, using histograms separately leads to better results because only one colors histogram is easier to fit than the regular histogram. This result is more than **6**% better than the results in [1]. In this method, the number of input features used is one-third of the number of features method IX used. Therefore method X has even less computational complexity with better results.

Method XI. Because using only one color's histogram gives better result, using more bins in the histogram should improve the result. In this method, only the histogram of the blue component is calculated, using 64 bins. Therefore, there are 64 elements in input vectors of the network. The network gave **84.41**% correct classification with 80 radial basis functions or centers. With 64 bins for each color's histogram, [1] has **76.6**% correct classification. In total, 192 input features are used in [1]. Method XI gives better results even though it uses fewer input features than [1] used. The result with RBFN is almost 10% better than the results with the distance measurement method in [1].