3D Object Recognition Using Multiple Features and Neural Network

XU Sheng, PENG Qi-cong
School of Communication and Information Engineering
University of Electronic Science and Technology of China
Chengdu, China
xs@uestc.edu.cn , qpeng@uestc.edu.cn

***Abstract*—To improve the performance of view-based three-dimensional object recognition system, we propose to extract multiple features from the 2D images of 3D objects, including texture characteristics, color moments, Hu's moment invariants, and affine moment invariants. Texture characteristics and color moments are used to distinguish objects of similar shape and different appearance. Hu's moment invariants have the invariance properties under rotation, scale and translation, and affine moment invariants have the invariance properties under affine transformation for the 3D objects in images. All these characteristics compose a 1-dimensional feature vector of 23 components for each 2D image of 3D objects, and then they are presented to a BP neural network for training. The trained BP network can be used to recognize 3D objects when provided the feature vectors of unseen views. We assessed our method based on both the original and noise corrupted COIL-100 3D objects dataset and achieved 100% correct rate of recognition when training views were presented every 10 degrees.**

***Keywords*—3D Object Recognition; texture analysis; color moments; Hu's moment invariants; Affine moment invariants; BP Neural Network**

I. INTRODUCTION

To give computer humanlike visual abilities so that robots can sense the three-dimensional environments in their two-dimensional views, the view-based (or appearance-based) three-dimensional object recognition has been widely and actively researched in recent years. In a two-dimensional image, the appearance of a three-dimensional object depends on its shape, reflectance properties, pose and the illumination conditions in the scene. View-based methods recognize objects by visual similarity, firstly learn or train a system with appearance of objects in two-dimensional images under different poses and illumination conditions. Then at recognition phase, presented a new two-dimensional image, this system is used to determine whether the target object exists in the new image.

Among the view-based 3D object recognition approaches, Poggio and Edelman showed that 3D objects could be recognized from the raw intensity values in 2D images using a network of generalized radial basis functions [1]. They argued and demonstrated that the full 3D structure of an object can be estimated if enough 2D views of the object were provided. Murase et al., proposed a parametric eigenspace methods to recognize 3D object directly from their appearance [2]. They developed a near real-time recognition system to recognize complex objects, and got accurate recognition results with an average pose estimate error of about 1.0 degree. Pontil et al., have used Support Vector Machines(SVM) to recognize 3D objects[3]. Instead of extract object features, they regarded images as points of a high dimensional space and performed recognition on images. Roobaert et al.[4] compared the performance of SVMs with different pixel-based input representations. Yang and Roth [5, 6] proposed a view-based algorithm using a network of linear units, the Sparse Network of Winnows (SNoW) learning architecture, to learn the objects representations and was specifically tailored for learning in the presence of a very large number of features. After converted color images to gray-level images of 32×32 pixels, they tested their method using pixel-based and edge-based representation of the objects respectively in large scale object recognition experiments. Recently Kostin et al.[7] proposed an object recognition scheme using graph matching and discussed matching two graph-based representations becomes a complicated process.

To the author's knowledge, although these methods have demonstrated excellent recognition performance, the computational cost involved in learning is extremely high, since these methods used pixel-based or edge-based representation of original objects images for system learning, the dimension of the input space was very high. In this paper, we propose a 3D object recognition method, which only use a few remarkable features extracted from each 2D image of 3D objects. Color moments[8] and texture characteristics[9] are used to distinguish 3D objects of similar shapes and different colors and texture. When 3D objects are projected to 2D images, the distortion of 3D objects is an inevitable problem. Hu's moment invariants[10] have been proven invariable under translation, rotation and scale of objects in 2D images, and affine moment invariants[11] have the properties of invariance under affine deformation when views of objects vary. Then we compose these characteristics to a 1-dimensional feature vector of 23 components for each image of 3D objects, and present the

vectors to a Back Propagation neural network for learning. The proposed method has been tested with 40 complex 3D objects selected from the Columbia Object Image Library (COIL-100) dataset [2], and achieved 100% correct rate of recognition when training views of 3D objects are presented every 10 degrees.

The paper is organized as follows. In section 2, we review the related basic theories of texture analysis, color and invariant moments. Section 3 discusses the implementation of our recognition system, and the obtained experimental results are illustrated. Finally, the conclusions and future work from our researches are summarized in Section 4.

II. THEORETICAL OVERVIEW

We will provide in this section a brief review about the related theories used in our method, including Gray Level Co-occurrence Matrix (GLCM) based texture analysis, color moments, Hu's moment invariants, affine moment invariants, and neural network recognition.

A. GLCM based Texture analysis

Texture is an important feature of objects in an image. Tow images with different content can usually be distinguished by their texture features even when the images share similar colors. Texture analysis is used in a variety of applications, including remote sensing, automated inspection, and medical image processing. When traditional threshold technical can not be used effectively, texture analysis can be helpful when objects in an image are more characterized by their texture than by intensity.

One of the most known texture analysis methods, Gray Level Co-occurrence Matrix (GLCM)[9], estimates image properties related second-order statistics. Gray level co-occurrence matrix $P_d(i, j)$ is constructed with each entry (i, j) corresponding to the number of occurrence of the pair of gray level i and j which are a distance d apart in original images. Haralick [9] proposed 14 statistical features extracted from GLCM to estimate the similarity of the gray level co-occurrence matrices with different distance d and different occurrence of the pair of gray level i and j. To reduce the computational complexity, we selected only 4 of these texture characteristics as features of 3D object in a 2D image as following:

1) Contrast: measure the local variations in the gray level co-occurrence matrix, i.e., a measure of the intensity contrast between a pixel and its neighbor over the whole image.

$$CON = \sum_{i,j} (i-j)^2 P_d(i,j) \tag{1.1}$$

2) Correlation: measure the joint probability occurrence of the specified pixel pairs, i.e., a measure of how correlated a pixel to its neighbor over the whole image.

$$COR = \frac{\sum_{i,j} (i-u_x)(j-u_y) P_d(i,j)}{\sigma_x \sigma_y} \tag{1.2}$$

Where $u_x, u_y, \sigma_x, \sigma_y$ are the average and standard variance of P_x, P_y separately. P_x is the sum of each row of $P_d(i, j)$, and P_y is the sum of each column of $P_d(i, j)$.

3) Energy: also know as uniformity or the Angular Second Moment, provides the sum of squared elements in the GLCM.

$$ASM = \sum_{i,j} P_d^2(i,j) \tag{1.3}$$

4) Homogeneity: measure the closeness of the distribution of the elements in the GLCM to the GLCM diagonal.

$$HOM = \sum_{i,j} \frac{P_d(i,j)}{1+|i-j|} \tag{1.4}$$

B. Color moments

Compared with geometric characteristics, the color of objects is quite robust, and insensitive to size and orientation of objects. Swain et al.[12] proposed histogram-based method for object representation. Their work is one of the earliest works which used color as object features for object recognition and image retrieval. They stored coarsely quantized color histograms of images. The histogram-based approach is simple, however changes in lighting and changes due to occlusion may cause relatively large change in their similar measures. Instead of storing the complete color distribution, Stricker et al.[8] proposed to store only the major features of images, i.e., to store the first three moments of each color channel of an image. For an image of RGB format or HSI format, only 9 numbers of moments are required.

A probability distribution is uniquely characterized by its moments according to probability theory. Color distribution of an image also can be regarded as a probability distribution, so color distribution also can be determined by its moments. The first moment, the second and the third central moment of each color channel can be used. The first moment is the average color of an image. And the second central moment of an image is the variance; the third central moment of an image is the skewness of each color channel. To make the value of the moments somewhat comparable, the standard deviation and the third root of the skewness of each color channel of an image are used, in this way all the values have the same unit.

To extract color features of objects, an appropriate color space should be selected first. The RGB color space is the most commonly used color model in computer image processing, while we prefer to use HSI color space (stands for Hue, Saturation, and Intensity) in our method. HSI color space is better suite for color moments calculation, because the hue component and saturation component are close to the manner of color sense of human being, and intensity component is independent with color information of images.

If p_{ij} is the pixel of a digital image $f(x, y)$ of $M \times N$ dimensional, A is the area of the image, then the three moments for each color channel can be defined as follows:

$$\mu = \frac{1}{A}\sum_i\sum_j p_{ij},\ \sigma = \left[\frac{1}{A}\sum_i\sum_j \left(p_{ij}-\mu\right)^2\right]^{1/2}, \qquad (1.5)$$
$$s = \left[\frac{1}{A}\sum_i\sum_j \left(p_{ij}-\mu\right)^3\right]^{1/3}$$

Since the color distribution information mainly concentrates at lower moments, we choose only the first moment and the second central moment for 3D object recognition to reduce computational cost.

C. Hu's moment invariants[10]

Given a density distribution function $f(x,y)$, its two-dimensional $(p+q)th$ order moments m_{pq} are defined in terms of Riemann integrals as:

$$m_{pq} = \int\int x^p y^q f(x,y)dxdy, \quad p,q = 0,1,2... \quad (1.6)$$

If it is assumed that $f(x,y)$ is piecewise continuous therefore bounded function, and that it can have nonzero values only in the finite part of the (x,y); then its moments of all orders exist and the moments sequence $\{m_{pq}\}$ is uniquely determined by $f(x,y)$; and conversely $f(x,y)$ is uniquely determined by $\{m_{pq}\}$。

For digital image $f(x,y)$ of discrete $M\times N$ dimensional, the $(p+q)th$ order geometry moments and central moments are defined as:

$$\begin{aligned} m_{pq} &= \sum_x\sum_y f(x,y)x^p y^q, \quad p,q = 0,1,2... \\ \mu_{pq} &= \sum_x\sum_y f(x,y)(x-\bar{x})^p (y-\bar{y})^q, \quad p,q = 0,1,2.. \end{aligned} \qquad (1.7)$$

Where $\bar{x} = m_{10}/m_{00}, \bar{y} = m_{01}/m_{00}$, is the center of gravity of an image. For an intensity image, m_{00} is its quality; for a binary image, m_{00} is its area. Both geometry moments and central moments can represent shapes of images, and central moments are invariants under translation.

Using the central moment of zero order to normalize all the central moments of other order, normalized central moments can be obtained as follow:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^r}, r = (p+q+2)/2,\ p+q = 2,3,4... \quad (1.8)$$

Using the linear combination of the second order and the third order normalized central moments, Hu M. K. proved 7 moment invariants under translation, rotation and scale of images. The 7 moments are called Hu's moment invariants and widely used for the discrimination of object shape.

$$\begin{aligned} M_1 &= \eta_{20}+\eta_{02}, \quad M_2 = (\eta_{20}-\eta_{02})^2 + 4\eta_{11}^2, \\ M_3 &= (\eta_{30}-3\eta_{12})^2 + (3\eta_{21}-\eta_{03})^2, \\ M_4 &= (\eta_{30}+\eta_{12})^2 + (\eta_{21}+\eta_{03})^2 \\ M_5 &= (\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})\left[(\eta_{30}+\eta_{12})^2 - 3(\eta_{21}+\eta_{03})^2\right] \\ &\quad +(\eta_{03}-3\eta_{21})(\eta_{03}+\eta_{21})\left[(\eta_{03}+\eta_{21})^2 - 3(\eta_{12}+\eta_{30})^2\right] \\ M_6 &= (\eta_{20}-\eta_{02})\left[(\eta_{30}+\eta_{12})^2 - (\eta_{21}+\eta_{03})^2\right] \\ &\quad +4\eta_{11}(\eta_{30}+\eta_{12})(\eta_{21}+\eta_{03}) \\ M_7 &= (3\eta_{21}-\eta_{03})(\eta_{30}+\eta_{12})\left[(\eta_{30}+\eta_{12})^2 - 3(\eta_{21}+\eta_{03})^2\right] \\ &\quad -(3\eta_{12}-\eta_{30})(\eta_{03}+\eta_{21})\left[(\eta_{03}+\eta_{21})^2 - 3(\eta_{12}+\eta_{30})^2\right] \end{aligned} \qquad (1.9)$$

D. Affine moment invariants

In realistic application, if we only extract Hu's moment invariants of 3D objects from 2D images, these objects features can not provide enough information to identify separate 3D objects accurately due to the affection of distance, weather, camera and visual angles etc. Jan Flusser et al.[11], proposed the affine moment invariants for characters identification and scene matching.

Given an arbitrary curve $[x,y]$ in two-dimensional space, after affine transformation, it will be the curve $[x',y']$, accordingly the affine transformation is defined as follow:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = A\left[x,y\right]^T + B \quad (1.10)$$

The general formulation of affine moment invariants is μ_{00}^r divided by polynomial of μ_{pq}, where r is certain appropriate exponential. Jan Flusser et al.[11] proved following affine moment invariants of first three order:

$$\begin{cases} I_1 = (\mu_{20}\mu_{02} - \mu_{11}^2)/\mu_{00}^4 \\ I_2 = (\mu_{30}^2\mu_{03}^2 - 6\mu_{30}\mu_{21}\mu_{12}\mu_{03} + 4\mu_{30}\mu_{12}^3 \\ \quad +4\mu_{21}^3\mu_{03} - 3\mu_{21}^2\mu_{12}^2)/\mu_{00}^{10} \\ I_3 = (\mu_{20}(\mu_{21}\mu_{03} - \mu_{12}^2) - \mu_{11}(\mu_{30}\mu_{03} - \mu_{21}\mu_{12}) \\ \quad +\mu_{02}(\mu_{30}\mu_{12} - \mu_{21}^2))/\mu_{00}^7 \end{cases} \qquad (1.11)$$

E. Neural network recognition

Because the 3D object recognition problem is actually a classification problem according to the pattern recognition theory, neural network methods can be used in 3D object recognition intuitively. Compared with the traditional methods, using neural network methods to classify input feature vectors is proven more robust under noise condition, and can be parallel computed in high speed. We suggest in this paper to use the well-known feedforward BP neural network to classify the feature vectors of 3D objects to recognize each object.

Irie et al.[13] have proven three layer feedforward network are capable of approximating arbitrary functions, given that they have sufficient numbers of neurons in their hidden layers, so we choose BP neural network of three layers in our recognition system. To determine the initial threshold and weight of hidden-layer and output-layer, we suggest to use

Nguyen-Widrow initial algorithm[14], which makes the active area of neurons of all layers be distributed equally in the input space approximately. Compared with the traditional initial algorithms which randomly select values in $(-1,+1)$, Nguyen-Widrow initial algorithm wastes less neuron and reduces network training time since neurons exist in each area of input space approximately.

Generally, gradient descent algorithm is chosen for network training. This algorithm decreases fast in the first few steps, however the mean square error function decreases slowly when moving to the optimum value since the gradient tends to zero. Because Newton algorithm can generate a perfect searching direction near the optimum value, we suggest choosing Levenberg-Marquardt algorithm which consist gradient descent algorithm and Newton algorithm to improve network training. It does not need to calculate the Hessian matrix in Levenberg-Marquardt algorithm, so the computational cost is reduced, and the training time is also reduced 1~2 order of magnitude compared with the general gradient descent algorithm[15].

III. EXPERIMENTAL RESULTS

In this section, we extracted the features of each 3D object from its 2D images according to above theories and then use BP neural network for classification. We assessed our method based on both original and noise corrupted COIL-100 dataset.

A. COIL-100 dataset

The Columbia Object Image Library (COIL-100) is a public dataset and widely used to evaluate the performance of 3D object recognition algorithms. This dataset contains 7,200 true color images with 128×128 resolution of 100 objects (72 images per object). The objects have a wide variety of geometric and reflectance characteristics. Each object was placed on a motorized turntable against a black background. The turntable was rotated through 360 degrees to vary object pose in condition of a fixed color camera. Images of objects were taken at pose intervals of 5 degrees, i.e., 72 poses per object. Each of the 7200 images in COIL-100 was size normalized, and available online via http://www1.cs.columbia.edu/CAVE/.

B. Experiments based Original images

In this paper, given the use of COIL-100 3D objects dataset, it is assumed that the illumination conditions remains constant, thus we only consider object pose as variable. We selected first forty 3D objects in COIL-100 dataset to test the recognition rate of our method, totaled 2,880 images. The images of each object with view angel at 30^o are shown in Fig. 1. In the first experiment, we chose 36 views per object (10^o intervals) to consist the training set, amounted to 1440 training images (36 images per object). Our experimental steps are as follows:

1) To fully use the spatial information of images, we specified distance of 4, 8, and 12 pixels each with 4 directions of 0^o, 45^o, 90^o and 135^o (horizontal, vertical, and two diagonal) to calculate gray level co-occurrence matrices, then the values of contrast, correlation, energy and homogeneity were averaged from the gray level co-occurrence matrices with different distances and directions according Eq.(1.1)-(1.4), and were saved as texture feature vector of 12 elements.

2) To calculate the color moments, we need to convert the original COIL-100 training images of RGB format into HSI format. Then we could calculate color moments according to Eq.(1.5). We only used the first and second order moments of each H/S/I color channel, totaled 6 color moments for each training image.

3) The calculation of Hu's moment invariants and affine moment invariants are based on binary images, so the training images need binary conversion before calculation according to the appropriate threshold. We chose the approach of maximum square variance threshold to convert the original true color training images to binary images. The binary images of the selected objects with view angel at 30^o are shown in Fig. 2.

4) According to Eq. (1.7) we can get the central moments of the training images, and then Hu's moment invariants can be calculated from Eq.(1.8) and(1.9), affine moment invariants can be calculated from Eq.(1.11). Though Hu's moment invariants contains 7 components, we only used first 4 lower order invariants to reduce computational cost, and from our experimental results, this simplified selection did not affect the recognition performance at all. Also, we select the first order affine moment invariants.

5) At this step we could combine the 12 texture features, 6 color moments, 4 Hu's moment invariants and 1 affine moment invariant, totaled 23 components to a one-

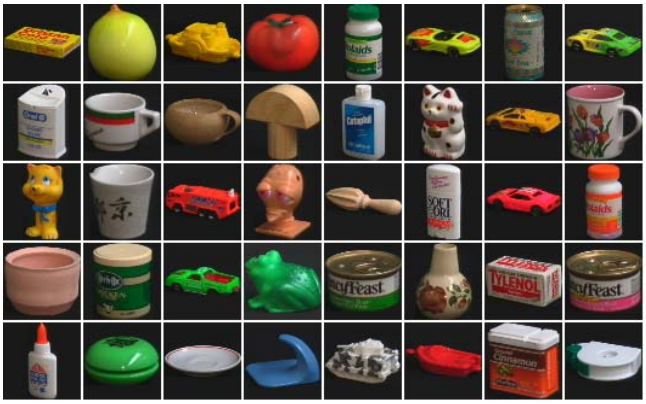

Figure 1. First 40 objects in COIL-100 dataset with view angle=30^o

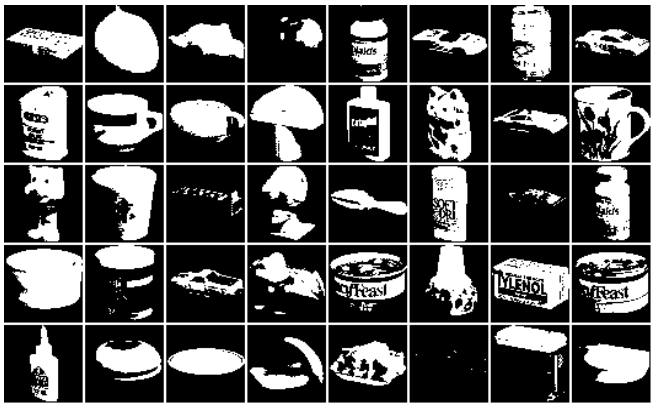

Figure 2. The binary images of the first 40 objects in COIL-100 dataset with view angle=30^o

dimensional feature vector for each 2D image of 3D objects. These one-dimensional feature vectors of 23 components would be the input of the BP neural network, so the number of neurons of input layer of this BP network was set to 23. We required this BP network to recognize 40 objects, so the number of neurons of output layer was set to 40.

6) The transfer functions of hidden layers were set to $TF_1(x) = \frac{1}{1+e^{-x}}$ for Levenberg-Marquardt training algorithm working better and since the outputs of BP network were judgments of 0 or 1, the transfer functions of output layers were set to $TF_2(x) = \frac{2}{1+e^{-2x}} - 1$.

7) After the values of weights and thresholds of the BP network were initialized using Nguyen-Widrow algorithm, the training for the BP network began. During training, Levenberg-Marquardt was used to update the values of weight and threshold. The network iterated 300 epochs in each training.

8) After training, at this step we could present all the 2880 images of the 40 objects to the trained BP network to test its recognition performance.

In the first experiment, half of the 2880 images of the 40 3D objects were utilized as training set, the correct rate of recognition of this trained BP network achieved 100%, shown in the first column of Table 1.

In order to study our method in more realistic situation, we compared the performance of our method when fewer numbers of views of the 3D objects were presented during training. We reduced images in the training set from 36 views per object (10^o intervals) to 2 views per object (180^o intervals). The case of 4 training views of one object is demonstrated in Fig. 3. We repeated our experiments from step1 to step8 for 4 times, each time with fewer training views, and reported the recognition results in other columns of Table 1.

Under these more challenging experimental setups, although it is not surprising to see from Table 1 that the correct rate of recognition decreased as the number of available views decreased during training, it is worth noticing that when the number of training views per object were reduced to 18 (20^o interval), the BP network achieved 99.86% correct rate of recognition, i.e., only 4 poses were not recognized in the total 2880 testing images. At the most hardness setup, we reduced the number of training view to 2, only 0^o and 180^o view angles were used, the correct rate of recognition of the BP network also achieved 72.92%.

C. *Experiments based noise corrupted images*

In order to assess the robustness of our method under noise environment, we added Gaussian white noise of zero-mean and variance 0.05 and 0.1 respectively, to the original COIL-100 object images, showing in Fig. 4 and Fig. 5. Under each noise variance condition, also we changed the number of views in the training set from 36 views per object (10^o intervals) to 2 views per object (180^o intervals). Again we repeated our experiments from step1 to step8 for 5 times, each time with fewer training views.

TABLE I. CORRECT RATE OF RECOGNITION WITH VARYING VIEW ANGLES

Number of views/object	36	18	9	4	2
Original images	100%	99.86%	99.69%	86.32%	72.92%

Figure 3. The views that making up the training set for an object, in the case of 4 training views per object

From the obtained experimental results in Table 2, when noise corrupted training views were presented at 10^o intervals in the case images are corrupted by Gaussian white noise of zero-mean and variance 0.05, our method also achieved 100% correct rate of recognition. In the case images were more seriously corrupted by Gaussian white noise of zero-mean and variance 0.1, the BP network achieved 99.97% correct rate of recognition, i.e., only 1 poses was not recognized in the total 2880 testing images. Other results of different number of training views were similar with the results reported in Table 1.

Figure 4. First 40 noise corrupted objects with view angle=30^o, Gaussian white noise of mean=0 and variance=0.05

Figure 5. First 40 noise corrupted objects with view angle=30^o, Gaussian white noise of mean=0 and variance=0.1

TABLE II. CORRECT RATE OF RECOGNITION WITH VARYING VIEW ANGLES FOR CORRUPTED IMAGES WITH GAUSSIAN WHITE NOISE OF ZERO-MEAN

Number of views/object	36	18	9	4	2
variance=0.05	100%	99.76%	99.44%	91.04%	75.9%
variance=0.1	99.97%	99.27%	98.12%	88.09%	75.69%

It can easily be inferred that our method is very robustness even in the presence of large number of image noise.

IV. CONCLUSIONS

In this paper, we proposed to extract multiple features of 3D objects from its 2D images to consist 1-dimensional feature vectors, each containing 12 texture characteristics, 6 color moments, 4 Hu's moment invariants and 1 affine moment invariants. Then these feature vectors are used for BP neural network training and recognition. The experiments which are based on both original and noise corrupted COIL-100 dataset were performed with different number of view angles as training set. The 100% correct rate of recognition could be achieved when the training set were presented with view angle of every 10^o. And when the number of training views were reduced, the correct rate of recognition was also good. The theoretical analysis and the remarkable good experimental results indicate our method can be used for view-based 3D object recognition.

Based on our work, one can further research a) looking for other feature extracting methods to extract local characteristic of 3D objects to correct the disadvantage of moment methods which only can detect global characteristic of objects; b) to recognize more 3D objects, beside above mentioned moment invariants, color and texture as object features, more features having other physical or mathematical properties, can be added into the feature vectors for BP neural network training.

ACKNOWLEDGMENT

The authors thank the creators of COIL-100 3D object dataset for giving the permission to use this dataset in research works.

REFERENCES

[1] T. Poggio and S. Edelman, "A network that learns to recognize three-dimensional objects," Nature, vol. 343, no. 6255, 1990, pp. 263-266.

[2] H. Murase and S.K. Nayar, "Visual learning and recognition of 3-d objects from appearance," International Journal of Computer Vision, vol. 14, no. 1, 1995, pp. 5-24.

[3] M. Pontil and A. Verri, "Support vector machines for 3D object recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 6, 1998, pp. 637-646.

[4] D. Roobaert and M.M. Van Hulle, "View-based 3D object recognition with support vector machines," IEEE International Workshop on Neural Networks for Signal Processing, 1999, pp. 77-84.

[5] M.H. Yang, D. Roth and N. Ahuja, "Learning to Recognize 3D Objects with SNoW," Proceedings of the Sixth European Conference on Computer Vision, 2000, pp. 439-454.

[6] D. Roth, M.H. Yang and N. Ahuja, "Learning to Recognize Three-Dimensional Objects," Neural Computation, vol. 14, no. 5, 2002, pp. 1071-1103.

[7] A. Kostin, J. Kittler and W. Christmas, "Object recognition by symmetrised graph matching using relaxation labelling with an inhibitory mechanism," Pattern Recognition Letters, vol. 26, no. 3, 2005, pp. 381-393.

[8] M.A. Stricker and M. Orengo, "Similarity of Color Images," Storage and Retrieval for Image and Video Databases (SPIE), 1995, pp. 381-392.

[9] R.M. Haralick, K. Shanmugam and I.h. Dinstein, "Textural features for image classification," IEEE Transactions on Systems, Man and Cybernetics, vol. SMC-3, no. 6, 1973, pp. 610-621.

[10] H. Ming-Kuei, "Visual pattern recognition by moment invariants," IEEE Transactions on Information Theory, vol. 8, no. 2, 1962, pp. 179-187.

[11] J. Flusser and T. Suk, "Pattern recognition by affine moment invariants," Pattern Recognition, vol. 26, no. 1, 1993, pp. 167-174.

[12] M.J. Swain and D.H. Ballard, "Color indexing," International Journal of Computer Vision, vol. 7, no. 1, 1991, pp. 11-32.

[13] B. Irie and S. Miyake, "Capabilities of three-layered perceptrons," Proceedings of the IEEE International Conference on Neural Networks, 1988, pp. 641-648.

[14] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," International Joint Conference on Neural Networks, 1990, pp. 21-26.

[15] M.T. Hagan and M.B. Menhaj, "Training feedforward networks with the Marquardt algorithm," IEEE Transactions on Neural Networks, vol. 5, no. 6, 1994, pp. 989-993.