Object Recognition in Indoor Video Sequences by Classifying Image Segmentation Regions Using Neural Networks

Nicolás Amezquita Gómez and René Alquézar

Dept. Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Campus Nord, Edifici Omega, 08034 Barcelona, Spain
{amezquita, alquezar}@lsi.upc.edu
http://www.lsi.upc.edu/~alquezar/

Abstract. This paper presents the results obtained in a real experiment for object recognition in a sequence of images captured by a mobile robot in an indoor environment. The purpose is that the robot learns to identify and locate objects of interest in its environment from samples of different views of the objects taken from video sequences. In this work, objects are simply represented as an unstructured set of spots (image regions) for each frame, which are obtained from the result of an image segmentation algorithm applied on the whole sequence. Each spot is semi-automatically assigned to a class (one of the objects or the background) and different features (color, size and invariant moments) are computed for it. These labeled data are given to a feed-forward neural network which is trained to classify the spots. The results obtained with all the features, several feature subsets and a backward selection method show the feasibility of the approach and point to color as the fundamental feature for discriminative ability.

1 Introduction

One of the most general and challenging problems a mobile robot has to confront is to identify and locate objects that are common in its environment. Suppose an indoor environment composed by halls, corridors, offices, meeting rooms, etc., where a robot navigates and is expected to perform some helping tasks (in response to orders such as "bring me a coke from the machine in the corridor" or "throw these papers to the nearest waste paper basket"). To accomplish these tasks, the robot must be able to locate and identify different objects such as a beverage machine or a waste paper basket. Of course, a possible approach is to program the robot with recognition procedures specific for each object of interest; in this way, the knowledge about the object and its characteristics is directly injected by the programmer in the recognition code. However, this approach is somewhat tedious and costly, and a preferable one would be to show to the robot in an easy way what the object is from images taken in different views and to rely on the general learning abilities of the robot, which could be based on neural networks or other machine learning paradigms, to obtain a certain model of the object and an associated recognition procedure.

M. Lazo and A. Sanfeliu (Eds.): CIARP 2005, LNCS 3773, pp. 93 – 102, 2005.

A very important issue is to determine the type of object model to learn. In this respect, a wide range of object representation schemes has been proposed in the literature [1]. In our point of view, a useful model should be relatively simple and easy to acquire from the result of image processing steps. For instance, the result of a color image segmentation process, consisting of a set of regions (spots, from now on) characterized by different features (related to size, color and shape), may be a good starting point to learn the model. Although structured models like adjacency attributed graphs or random graphs can be synthesized for each object from several segmented images [2], we have decided to investigate first a much simpler approach in which the object is just represented as an unstructured set of spots and the spots are classified directly as belonging to one of a finite set of objects or the background (defined as everything else) using a feed-forward neural network.

The classification of segmented image regions for object recognition has been addressed in several works. In [3], *eigenregions*, which are geometrical features that encompass several properties of an image region, are introduced to improve the identification of semantic image classes. Neural networks are used in [4] not only to classify known objects but to detect new image objects as well in video sequences. In [5], objects of interest are first localized, then features are extracted from the regions of interest and finally a neural network is applied to classify the objects. Support vector machines are used in [6] to classify a segmented image region in two categories, either a single object region or a mixture of background and foreground (multiple object region), in order to derive a top-down segmentation method.

The rest of the paper is organized as follows. In Section 2, image acquisition, preprocessing and segmentation steps are described as well as the semiautomatic method to assign class labels to spots. In Section 3, the features computed for each spot are defined. Neural network training and test together with the experimental methodology followed are commented in Section 4. The experimental results are presented in Section 5, and finally, in Section 6, some conclusions are drawn and future work discussed.

2 Image Acquisition, Pre-processing and Segmentation

A digital video sequence of 88 images was captured by an RGB camera installed on the MARCO mobile robot at the *Institute of Robotics and Industrial Informatics* (IRI, UPC-CSIC) in Barcelona. The sequence shows an indoor scene with some slight perspective and scale changes caused by the movement of the robot while navigating through a room. The objects of interest in the scene were a box, a chair and a pair of identical wastebaskets put together side by side (see Figure 1), and the objective was to discriminate them from the rest of the scene (background) and locate them in the images.

Before segmentation, the images in the sequence were preprocessed by applying a median filter on the RGB planes to smooth the image and reduce some illumination reflectance effects and noise. Then, the image segmentation module was applied to the whole sequence, trying to divide the images in homogeneous regions, which should correspond to the different objects in the image or parts of them. We used an implementation of the Felzenszwalb – Huttenlocher algorithm [7], which is a pixel

merge method based on sorted edge weights and minimum spanning tree, to segment each image separately. Note that this is a method working on static images that does not exploit the dynamic continuity of image changes through the sequence.

The output of the segmentation process for each image consists of a list of regions (spots) that partition the image in homogeneous pieces, where each region is defined by the set of coordinates of the pixels it contains. For each spot, the mass center was calculated, and for all the spots whose mass center lied in some *region-of-interest* (ROI) rectangular windows, several features listed in Section 3 were computed as well. These windows were manually marked on the images with a graphics device to encompass the three objects of interest and a large region on the floor. Figure 1 shows one of the images and its segmentation together with the ROI windows on them.

The remaining set of spots, those with the mass center inside the ROI windows, was further filtered by removing all the spots with a size lower than 100 pixels, with the purpose of eliminating small noisy regions caused by segmentation defects. Hence, from the 88 images, a total number of 7853 spots were finally obtained.

In order to assign a class label to each spot, to be used as target for the spot pattern in the neural network training and test processes, a simple decision was made: each one of the four ROI windows constituted a class and all the spots in a window were assigned the same class label. Note that this is a rough decision, since several background spots are included in the ROI windows of the box, the chair and the wastebaskets, and therefore are not correctly labeled really. This is a clear source of error (incorrectly labeled patterns) that puts some bounds on the level of classification accuracy that the learning system, in this case the neural network, may reach. However, we preferred to carry out this simple but more practical approach instead of manually labeling each spot, which is obviously a very tedious task, although it would probably have raised the classification performance of the trained networks.

For illustrative purposes, the spots of Figure 1 that were assigned to each of the four classes are displayed in Figure 2; for the three objects (Figure 2 (a)-(c)), the union of selected spots is shown in the left and isolated spots that belong to the class are shown in the right. In addition, Figure 3 displays some of the ROI windows in other images of the sequence.

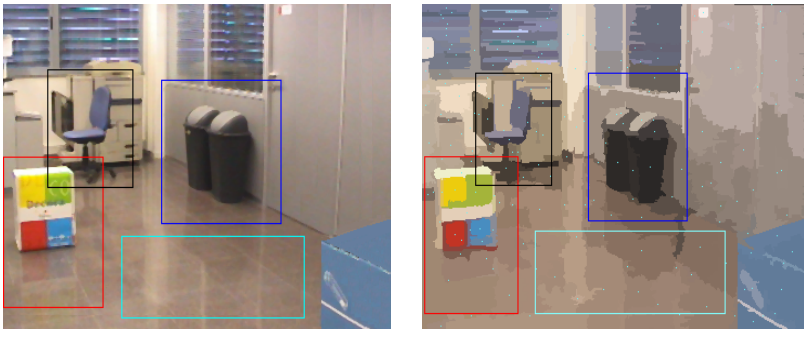
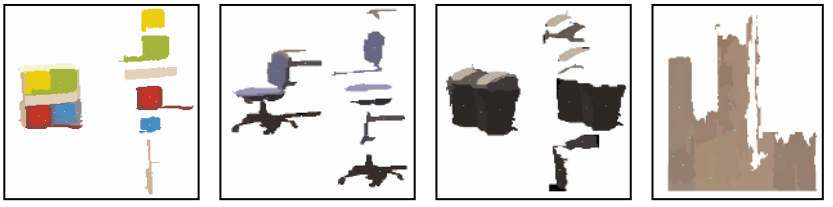
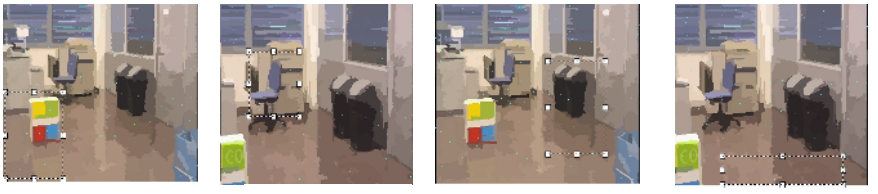


Fig. 1. One of the original images *(left)* and the corresponding segmented image *(right)*, with the four ROI windows marked on them. Spot mass centers are also displayed in the right image.



(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

Fig. 2. Labeling of the spots selected from the segmented image in Figure 1



Window of the class 1 | Window of the class 2 | Window of the class 3 | Window of the class 4

Fig. 3. Selection of ROI windows for two other images in the video sequence

3 Features Computed for the Image Regions

In order to be processed as a pattern by a neural network, a spot must be described by a feature vector. Table 1 displays the 14 variables that were initially considered to form the feature vector for training and testing the networks. In Section 5 we will also present results obtained from several different subsets of these 14 variables.

Two types of information were used in the computation of the spot features: color and geometry. With regards to color, average and variance values for each one of the three RGB bands were calculated for each spot on the basis of the corresponding intensity values of the spot pixels in the original image (not in the segmented image, for which spot color variance would be zero). This is, the result of the segmentation algorithm served to identify the pixels of every spot, but the color characteristics of these pixels were taken from the original RGB image.

The geometrical information may include features related to position, orientation, size and shape. Because of the robot movement, we were mainly interested in shape descriptors that were invariant to translation and scale, and to this end, we decided to use the seven invariant geometric moments defined by Hu [8]. In addition and since the range of variation of the objects' size was rather limited in the video sequence, we also calculated and used the size of each spot, i.e. its area measured in number of pixels.

For the calculation of the moments corresponding to a spot, all the pixels that form the spot are involved (not only its boundary pixels). More precisely, the seven invariant moments, independent of position and size of the region, that we used in this work are defined by the following equations:

$$I_1=N(2,0)+N(0,2) \tag{1}$$

$$I_2=(N(2,0)-N(0,2))^2+4(N(1,1))^2 \tag{2}$$

$$I_3=(N(3,0)-3N(1,2))^2 +(3N(2,1)-N(0,3))^2 \tag{3}$$

$$I_4=(N(3,0)+N(1,2))^2+(N(2,1)+N(0,3))2 \tag{4}$$

$$I_5=(N(3,0)-3N(1,2))(N(3,0)+N(1,2))[(N(3,0)+N(1,2))^2 -3(N(2,1)+N(0,3))^2] +(3N(2,1) –N(0,3))(N(2,1)+N(0,3))[3(N(3,0)+N(1,2))^2-(N(2,1)+N(0,3))^2] \tag{5}$$

$$I_6=(N(2,0) – N(0,2)) [(N(0,3)+N(1,2))^2-(N(2,1)+N(0,3))^2] +4N(1,1)(N(3,0)+N(1,2))(N(2,1)+N(0,3)) \tag{6}$$

$$I_7=(3N(2,1)-N(0,3)) (N(3,0)+N(1,2))[(N(3,0)+N(1,2))^2-3(N(2,1)+N(0,3))^2] +(3N(1,2)-N(3,0))(N(2,1)+N(0,3))[3(N(3,0)+N(1,2)^2)-(N(2,1)+N(0,3))^2] \tag{7}$$

where N (p, q) are the normalized central moments of order two, which are given by:

$$N(p, q) = MC(p, q) / MC^{\beta}(0, 0) \;\; ; \beta = ((p + q) / 2) + 1 \tag{8}$$

$$MC(p,q) = \sum\sum (x-X)^{p} (y-Y)^{q} f(x, y) \tag{9}$$

Table 1. Initial set of 14 variables that formed the feature vector for every spot and were used as input to the neural network for training and test

Set of variables	
Number of variable	**Feature**
1	Size of spot
2	Average red plane
3	Average green plane
4	Average blue plane
5	I1RGB invariant moment
6	I2RGB invariant moment
7	I3RGB invariant moment
8	I4RGB invariant moment
9	I5RGB invariant moment
10	I6RGB invariant moment
11	I7RGB invariant moment
12	Variance red plane
13	Variance green plane
14	Variance blue plane

$$MC(0,0)= \sum\sum f(x,y) \tag{10}$$

where f (x, y) is the intensity value of the pixel (x,y) in the segmented image, as given by the average of the three planes RGB, and (X,Y) are the mean coordinates of the spot. It must be noted that, in this case, all pixels in the same spot share the same value f(x,y), which depends on the color assigned to the spot as result of the segmentation process.

4 Neural Networks and Experimental Methodology

Neural networks (NNs) are used for a wide variety of object classification tasks [9]. An object is represented by a number of features that form a *d*-dimensional feature vector $\mathbf{x}$ within an input space $X \subseteq R^d$. A classifier therefore realizes a mapping from input space X to a finite set of classes $C = \{1,...,l\}$. A neural network is trained to perform a classification task from a set of training examples $S = \{(\mathbf{x}^\mu, t^\mu), \mu = 1,...,M\}$ using a supervised learning algorithm. The training set S consists of M feature vectors $\mathbf{x}^\mu \in R^d$ each labeled with a class membership $t^\mu \in C$. The network typically has as many outputs as classes and the target labels are translated into *l*-dimensional target vectors following a local unary representation. During the training phase the network parameters are adapted to approximate this mapping as accurately as possible (unless some technique, such as early stopping, is applied to avoid over-fitting). In the classification phase an unlabeled feature vector $\mathbf{x} \in R^d$ is presented to the trained network and the network outputs provide an estimation of the *a-posteriori* class probabilities for the input $\mathbf{x}$, from which a classification decision is made, usually an assignment to the class with maximum *a-posteriori* probability [10].

In this work, we used a feed-forward 2-layer perceptron architecture (i.e. one hidden layer of neurons and an output layer) using standard backpropagation as training algorithm. For processing the full feature vectors, the networks consisted of 14 inputs, *n* hidden units and 4 output units, where *n* took different values from 10 to 200 (see Table 2). Hyperbolic tangent and sine functions were used as activation functions in the hidden layer and the output layer, respectively. A modified version of the PDP simulator of Rumelhart and McClelland [11] was employed for the experiments, setting a learning rate of 0.003 and a momentum parameter of zero for backpropagation, and a maximum number of 2,000 training epochs for each run.

As mentioned before, a dataset containing 7853 labeled patterns (spots) was available after the image segmentation and feature calculation processes described in previous sections. For each network tested, a kind of cross-validation procedure was carried out by generating 10 different random partitions of this dataset, each including 60% of the patterns for the training set, 15% for the validation set and 25% for the test set. The validation sets were used for early stopping the training phase. Actually, the network chosen at the end of the training phase was the one that yielded the best classification result on the validation set among the networks obtained after each training epoch. Then, this network was evaluated on the corresponding test set.

After the experiments with the whole set of features, we performed similar cross-validation experiments with different subsets of features (as indicated in Table 3),

using the same experimental parameters aforementioned, except that the number of hidden units was fixed to 160 (since this provided the best test result with all features) and that the number of inputs was obviously set to the dimension of the feature subset.

Finally, starting from the architecture selected for the full set of features, a sequential backward selection method [12] was applied trying to determine a good subset of input features by eliminating variables one by one and retraining the network each time a variable is temporarily removed. In this case, each partition of the cross-validation procedure divided the dataset in 60% of patterns for training and 40% for test (no validation set) and the training stop criterion was to obtain the best result in the training set for a maximum of 2,000 epochs.

5 Experimental Results

The results obtained for the full set of features with the different networks tested are displayed in Table 2. For each one of the three sets (training, validation and test set), the classification performance is measured as the average percentage of correctly classified patterns in the ten cross-validation partitions, evaluated in the networks selected after training (the ones that maximize the performance on the validation set). Although the classification performance is shown for the three sets, the main result for assessing the network classification and generalization ability is naturally the classification performance in the test set. Hence, a best correct classification rate of 75.94 % was obtained for the architecture with 160 hidden units, even though the generalization performance was rather similar in the range from 60 to 200 hidden units. It appears to be an upper bound for both the training and test sets that might be caused in part by the incorrectly labeled patterns mentioned in Section 2.

Table 2. Classification performance obtained for different network configurations (hidden layer sizes) for the full set of 14 features using 10-partition cross-validation and early stopping

Classification performance (all features)			
Hidden units	**Training**	**Validation**	**Test**
200	80.27 %	77,00 %	75.63 %
180	80.04 %	77.47 %	75.81 %
160	80.33 %	77.38 %	**75.94** %
140	80.24 %	77.46 %	75.74 %
120	80.03 %	77.06 %	75.23 %
100	79.74 %	76.94 %	75.74 %
80	79,43 %	77,12 %	75,54 %
60	79,22 %	77,30 %	75,77 %
40	77,86 %	76,08 %	74,33 %
20	74,92 %	74,84 %	73,46 %
10	72.11 %	71.60 %	70.18 %

Table 3. It presents the classification results for several groups of selected variables to assess the relative importance of the different types of features (size, color averages, color variances and shape invariant moments)

Classification performance (with feature subsets)			
Feature Subsets	**Training**	**Validation**	**Test**
spot size, average and variance r,g,b	79.69	78.02	**76.17**
all variables	80.33	77.38	75.94
spot size and average r,g,b	77.77	77.49	75.92
spot size, average r,g,b and three first invariant moments	79.32	77.68	75.90
average r,g,b and seven invariants	77.51	77.23	75.38
average r,g,b and three first invariants	76.90	76.91	74.83
spot size and variance r,g,b	45.12	45.82	45.61
spot size, variance r,g,b and three first invariant moments	45.10	45.59	45.08
Seven invariant moments, variance r,g,b	40.95	41.12	40.79
Seven invariant moments	30.30	30.34	29.96

Table 4. It presents top-down the order of the variables eliminated in the sequential backward selection process and the associated performance in the training and test sets after each step

Backward selection process – Classification performance			
Num. var.	**Feature eliminated**	**Training**	**Test**
BASELINE	No variable removed	79.88	75.64
5	I1RGB invariant moment	80.14	76.40
13	variance of green plane	80.16	76.22
14	Variance of blue plane	79.77	76.55
11	I7RGB moment invariant	80.04	76.26
12	variance of red plane	78.90	**77.13**
9	I5RGB moment invariant	78.60	76.24
8	I4RGB moment invariant	78.07	75.99
7	I3RGB moment invariant	78.14	75.73
6	I2RGB moment invariant	77.81	75.89
10	I6RGB moment invariant	77.04	74.18
1	spot size	73.35	72.49
3	average of green plane	65.89	63.96
4	Average of blue plane	40.36	39.86
2	Average of red plane	30.48	30.83

The results obtained for different subsets of features are displayed in Table 3, ordered decreasingly by test classification performance. It can be noted that similar results are obtained if the average color features are taken into account, but the performance falls down dramatically when they are not used. The best result here was

76.17% test classification performance for a subset comprising color features (both RGB averages and variances) and spot size (and with the shape invariant moments removed). Using only the seven invariant moments, the performance is almost as poor as that of a random classification decision rule.

The results of the sequential backward feature selection, shown in Table 4, clearly confirmed that RGB color variances and invariant moments were practically useless (they were the first features removed without a significant performance degradation, indeed the test classification rate grew up to a 77.13% after the elimination of six of these variables) and that RGB color averages provided almost all the relevant information to classify the spots.

6 Conclusions and Future Work

A simple approach to object recognition in video sequences has been tested in which a feed-forward neural network is trained to classify image segmentation regions (spots) as belonging to one of the objects of interest or to the background. Hence, objects are implicitly represented as an unstructured set of spots; no adjacency graph or description of the structure of the object is used.

In order to provide labeled examples for the supervised training of the network, a semiautomatic procedure for assigning object labels (classes) to spots has been carried out based on the manual definition of graphical region-of-interest windows. However, this procedure produces some incorrectly labeled examples that affect negatively the learning of the objects and the posterior classification performance.

Spot RGB color averages and, to a less extent, spot size have been determined empirically in this work as adequate features to discriminate objects based on segmentation regions, whereas spot shape invariant moments and spot RGB color variances have been shown to be of very little help. The obtained classification results are rather good taking into account the simplicity of the approach (for instance, two very similar spots could perfectly belong to different objects) and the presence of incorrectly labeled patterns in the training and test sets caused by the semiautomatic labeling procedure.

In order to increase the classification performance, there are several actions that can be attempted. First, the labeling procedure may be improved to reduce (or even eliminate) the presence of incorrectly labeled spots. Second, some model of the structure of the object can be used in the learning and test phases; for instance, attributed graphs and random graphs with spots as nodes may be tried [2]. Third, a better image segmentation algorithm may be used, for instance, one based on the dynamic sequence of images (instead of using only single images separately) may be more robust.

In the long-term, our purpose is to design a dynamic object recognition method for video sequences by exploiting the intrinsic continuity in the object views represented by the successive images the mobile robot capture while navigating in an indoor environment.

Acknowledgements

We would like to thank the people of the Learning and Vision Mobile Robotics group led by Dr. Alberto Sanfeliu at the *Institute of Robotics and Industrial Informatics* (IRI) in Barcelona for providing us with the video sequence data and for their constant support, especially to Juan Andrade Cetto and Joan Purcallà. This work was partially funded by the Spanish CICYT project DPI 2004-05414.

References

1. Pope A. R. "Model-Based Object Recognition. A survey of recent research", University of British Columbia, Vancouver, Canada, Technical Report 94-04, January 1994.
2. Sanfeliu A., Serratosa F., Alquézar R., "Second-order random graphs for modeling sets of attributed graphs and their application to object learning and recognition", Int. Journal of Pattern Recognition and Artificial Intelligence, Vol. 18 (3), 375-396, 2004.
3. Fredembach C., Schröder M. and Süsstrunk S., "Eigenregions for image classification", IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 26 (12), pp. 1645-1649, 2004.
4. Singh S., Markou M., Haddon J., "Detection of new image objects in video sequences using neural networks", Proc. SPIE Vol. 3962, p. 204-213, Applications of Artificial Neural Networks in Image Processing V, Nasser M. Nasrabadi; Aggelos K. Katsaggelos; Eds., 2000.
5. Fay R., Kaufmann U., Schwenker F., Palm G., "Learning object recognition in a neurobotic system". In: Horst-Michael Groß, Klaus Debes, Hans-Joachim Böhme (Eds.) *3rd Workshop on SelfOrganization of AdaptiVE Behavior (SOAVE 2004).* Fortschritt -Berichte VDI, Reihe 10 Informatik / Kommunikation, Nr. 743, pp. 198-209, VDI Verlag, Düsseldorf, 2004.
6. Wang W., Zhang A. and Song Y., "Identification of objects from image regions", IEEE International Conference on Multimedia and Expo (ICME 2003), Baltimore, July 6-9, 2003.
7. Felzenszwalb P. and Huttenlocher D., "Efficiently computing a good segmentation". In IEEE Conference on Computer Vision and Pattern Recognition, 98-104, 1998.
8. Hu M-K., "Visual pattern recognition by moment invariants", IRE Trans. on Information Theory, Vol. 8 (2), pp. 179-187, 1962.
9. Fiesler E. and Beale R. (eds.), *Handbook of Neural Computation*, IOP Publishing Ltd and Oxford University Press, 1997.
10. Bishop C.M., *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.
11. Rumelhart D.E., McClelland J.L. and the PDP Research Group (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, MIT Press, 1986.
12. Romero E., Sopena J.M., Navarrete G., Alquézar R., "Feature selection forcing overtraining may help to improve performance", Proc. Int. Joint Conference on Neural Networks, IJCNN-2003, Portland, Oregon, Vol.3, pp.2181-2186, 2003.