Comparison of view–based object recognition algorithms using realistic 3D models

V. Blanz[1 2], B. Schölkopf[1 2], H. Bülthoff[1], C. Burges[2], V. Vapnik[2], T. Vetter[1]

[1] Max–Planck–Institut für biologische Kybernetik,
Tübingen, Germany, E–mail volker@mpik-tueb.mpg.de
[2] AT&T Bell Laboratories, Holmdel, NJ, USA

Abstract. Two view-based object recognition algorithms are compared: (1) a heuristic algorithm based on oriented filters, and (2) a support vector learning machine trained on low-resolution images of the objects. Classification performance is assessed using a high number of images generated by a computer graphics system under precisely controlled conditions. Training- and test-images show a set of 25 realistic three-dimensional models of chairs from viewing directions spread over the upper half of the viewing sphere. The percentage of correct identification of all 25 objects is measured.

In computer vision, view–based models of object recognition have become more and more influential in recent years. Moreover, psychophysical evidence has been found for a view–based representation of objects in humans (Bülthoff and Edelman, 1992). Unlike viewpoint-invariant representations using structural descriptions (e.g. Marr and Nishihara, 1978), viewpoint-dependent models do not require a three-dimensional representation (Poggio, Edelman 1990, Lades et.al., 1993). The present study compares two recognition algorithms that are explained in the following sections.

1 Recognition by Oriented Filters

If a three-dimensional object is rotated about a frontoparallel axis, orthographic projections of surface points will move in the image plane in a direction perpendicular to the axis. To a great extent this also applies to perspective projection under realistic viewing conditions. Thus, images of an object can be made insensitive to rotations about a particular frontoparallel axis by lowpass filtering in one direction.

In order to compensate for relatively large displacements, the lowpass filter operation extinguishes much of the high spatial frequency structure in one direction. Due to a centering process described below, the lowpass filtering has to account also for displacement components along the axis of rotation. As a consequence, performance cannot be improved significantly by choosing image resolutions higher than 16x16 pixels. In order to retain some of the high spatial frequency information from the initial image, the representation also contains images with an edge detection performed before downsampling.

The algorithm uses a set of stored views of each object. They are preprocessed and stored in a representation of low resolution. To classify a test image, it is preprocessed in the same way and compared one by one with all of the stored views.

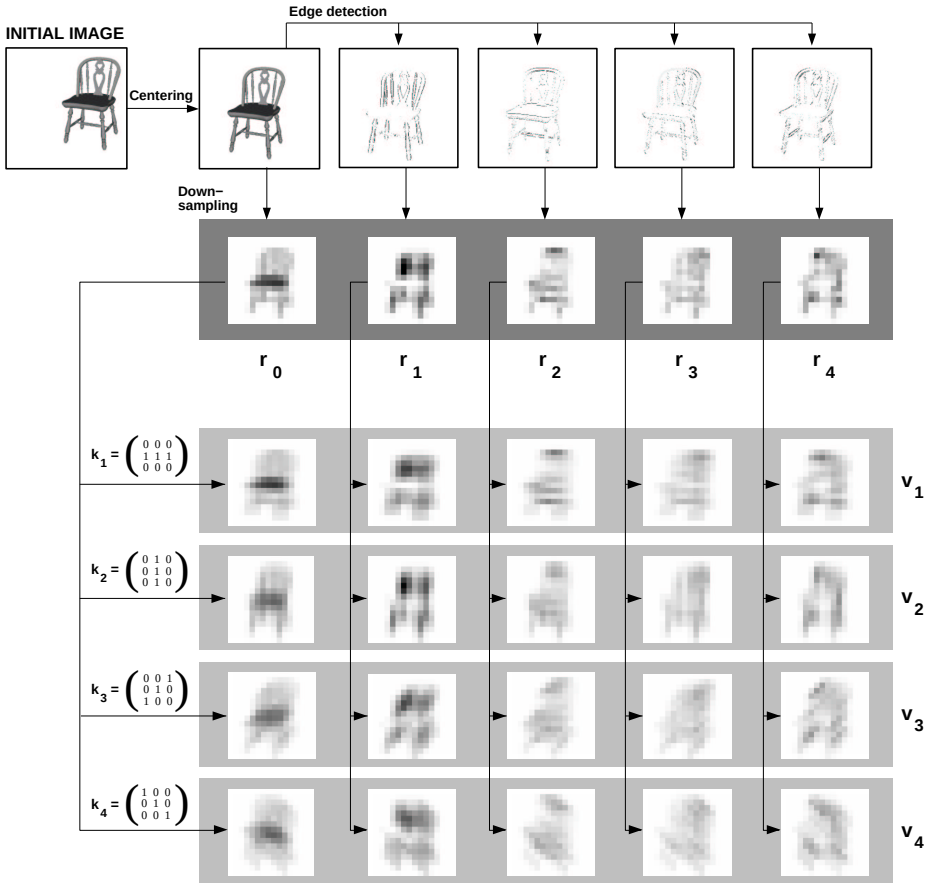

Fig. 1. Recognition by Oriented Filters: In preprocessing, a representation consisting of five low-resolution images $r_0 ... r_4$ is generated. For a comparison, these are convoluted with matrices $k_1 ... k_4$. The results are combined to 5*16*16-dimensional vectors $v_1 ... v_4$. Euclidean distance between vectors v_i is used to measure similarity between images.

The algorithm performs the following steps:

Preprocessing:

1. **Centering:** The picture is centered with respect to the center of mass of the binarized image. All objects are shown on a white background, so the binarized image segregates figure from ground.
2. **Edge detection:** Four one-dimensional differential operators (vertical, horizontal, diagonal) are applied to the image and the modulus is taken.
3. **Downsampling:** Reducing resolution of all five images from 256x256 to 16x16 pixels, we obtain images $r_0 ... r_4$. In this representation, each view requires 5*16*16=1280 bytes. In our simulation, we stored 25 views per object, summing up to a total of 32kB per object.

Image comparison: To compare a given image that has been preprocessed to vectors $r_0 ... r_4$ with a stored view $r'_0 ... r'_4$, we perform the following steps:

1. **Oriented filters:** Images are lowpass filtered in four directions, using the filter matrices $k_1 ... k_4$ shown in figure 1. Each of them is applied to all five low resolution images of a view. The resulting images are combined to a vector
$$v_i = (k_i \otimes r_0,\ k_i \otimes r_1,\ k_i \otimes r_2,\ k_i \otimes r_3,\ k_i \otimes r_4)\ ,\ i = 1...4. \tag{1}$$
2. **Euclidean distance:** As a measure of similarity of two views, we compute sums of squared differences of corresponding pixel values. This yields four distance values
$$d_i = \|v_i - v'_i\|,\ i = 1..4. \tag{2}$$

Training: During training, a set of views of each object is preprocessed and stored. For each object, the same viewing directions are used. The selection of these views is done externally, but different ways for an automatic selection process are conceivable. For each of the four distances d_i of each view of each object, a threshold is calculated and stored. It is found by comparing the given view with all views of all other objects and choosing a value that leads to a false alarm rate below a fixed value on the training set.

Decision rule: If two images show the same object, at least one distance value of one object should be below threshold. It has proved to be most reliable to compute distance values with all stored views and then choose the object with the highest number of below-threshold distances.

2 Support Vector Learning Machines

To construct decision rules that generalize well, the support vector algorithm uses the Structural Risk Minimization (SRM) principle (Vapnik, 1995). SRM is based on the result that the error rate on an independent test set is bounded by the sum of the training error rate and a term which depends on the so–called VC(Vapnik–Chervonenkis)–dimension of the learning machine. By minimizing the sum of both quantities, high generalization performance can be achieved. For linear hyperplane decision functions $f(\mathbf{x}) = \mathrm{sgn}\left((\mathbf{w} \cdot \mathbf{x}) + b\right)$, the VC–dimension can be controlled by controlling the norm of the weight vector $\mathbf{w}$ (Vapnik, 1995). Given training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)$, $\mathbf{x}_i \in \mathbf{R}^N, y_i \in \{\pm 1\}$, a separating hyperplane which generalizes well can be found by minimizing (Cortes & Vapnik, 1995)
$$\|\mathbf{w}\|^2 + \gamma \cdot \sum_{i=1}^{\ell} \xi_i \tag{3}$$
subject to
$$\xi_i \geq 0, \quad y_i \cdot ((\mathbf{x}_i \cdot \mathbf{w}) + b) \geq +(1 - \xi_i) \quad \text{for } i = 1, \ldots, \ell \tag{4}$$
(γ is a constant which determines the trade–off between training error and VC–dimension).

The solution of this problem can be shown to have an expansion $\mathbf{w} = \sum_{i=1}^{\ell} \lambda_i \mathbf{x}_i$, where only those λ_i are nonzero which belong to an $\mathbf{x}_i$ precisely meeting the constraint (4) — these so–called *Support Vectors* lie closest to the

Fig. 2. The dataset of 25 3D-models of chairs.

decision boundary. The λ_i are found by solving the quadratic programming problem defined by (3) and (4). Finally, this method can be generalized to non-linear decision surfaces by first mapping the input nonlinearly into some high–dimensional space, and finding the separating hyperplane in that space (Boser, Guyon & Vapnik, 1992). This is achieved implicitely by using different types of symmetric functions $K(\mathbf{x}, \mathbf{y})$ instead of the ordinary scalar product $(\mathbf{x} \cdot \mathbf{y})$. This way one gets decision functions

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{\ell} \lambda_i \cdot K(\mathbf{x}, \mathbf{x}_i) + b\right). \tag{5}$$

In the present study, we used polynomial classifiers $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^n$ of degree $n = 5$, and a value $\gamma = 10$. Other choices of K allow the implemetation of neural networks and radial basis function classifiers. In handwritten digit recognition, the support vector set has empirically been shown to be largely independent of the type of support vector machine constructed, and it contains all the information necessary to solve the classification task (Schölkopf, Burges, and Vapnik,

1995).

To construct the multi–class classifier needed for our purposes, we simply combined binary classifiers which were trained to recognize individual objects. This is done by choosing as the output of the multi–class classifier the class where the argument of the decision function (5) is maximal.

3 Experimental Results

The object database consisted of 25 different 3D-models of chairs (figure 2). All of them had a uniform grey surface. They were rendered in perspective projection in front of a white background on a Silicon Graphics workstation using Inventor software. The initial images had a resolution of 256x256 pixels.[1] In all viewing directions, image plane orientation was such that the vertical axis of the object was projected in an upright orientation. Both in training and test set, only views on the upper half of the viewing sphere were used. The training set consisted of 89 equally spaced views of each object. The test set contained 100 random views of each object. In both algorithms, the images ($r_0 ... r_4$) were rescaled based on their variances on the training set. Performance was measured in terms of correct identification of all 25 objects from all viewing directions.

Results for oriented filters: We stored only 25 equally spaced views per object, but used the full training set for calculation of the thresholds. Under these conditions, a classification error of 4.7% was achieved. Ignoring all data ($r_1 ... r_4$) from edge detection and relying only on r_0 increased the error to 21%.

Results for a Support Vector Learning Machine: Trained on the rescaled images, the support vector machine had an error rate of 1.0%. Without rescaling, error rate increased to 1.6%. Using only the images, i.e. r_0, the error was 8.4%.

Discussion: Images generated by means of computer graphics provide a useful basis for studying and comparing object recognition algorithms. However, generalizations of absolute performance values from simulations to real-world problems may be problematic. For the algorithms used in this work, noise should have only small effect because most of the processing is performed on a low spatial frequency domain. Much more impact has to be expected from a realistic, non uniform background. On the other hand, objects with different albedo and color can faciliate recognition significantly.

For both algorithms, performance data for this relatively difficult classification task were below 5% – with a fully connected feed-forward neural network with one hidden layer, we were not able to get error rates below 10%. Given the simple design of the oriented filter algorithm, its recognition rate was surprisingly high. A closer investigation of some of the image vectors ($r_0 ... r_4$) shows that vectors of a single object change drastically with viewpoint. As it seems, the support vector machine is very much capable of dealing with such a complicated decision surface.

[1] For benchmarking with other recognition algorithms, we will make the set of images available on our ftp-server.

References

Boser, B. E., Guyon, I. M., Vapnik, V.: A training algorithm for optimal margin classifiers. Fifth Annual Workshop on Computational Learning Theory, Pittsburgh ACM (1992) 144–152.

Bülthoff, H. H., Edelman, S.: Psychological support for a two–dimensional view interpolation theory of object recognition. Proc. Natl. Acad. Sci. USA **89** (1992) 60–64

Cortes, C., Vapnik, V.: Support Vector Networks. Machine Learning **20** (1995) 1–25

Lades, M, Vorbrüggen, J. C., Buhmann, J., Lange, J., von der Malsburg, C, Würtz, R. P., Konen, W. Distortion Invariant Object Recognition in the Dynamic Link Architecture. IEEE Trans. Comp. **42** 3 (1993) 300–311

Marr, D., Nishihara, H. K. Representation and recognition of the spatial organization of three dimensional structure. Proceedings of the Royal Society of London B, **200** (1978) 269–294

Poggio, T., Edelman, S. A network that learns to recognize three-dimensional objects. Nature **343** (1990) 263–266

Schölkopf, B., Burges, C., Vapnik, V.: Extracting support data for a given task. In: Fayyad, U. M., Uthurusamy, R. (eds.): Proceedings, First International Conference on Knowledge Discovery & Data Mining, AAAI Press, Menlo Park, CA (1995)

Vapnik, V.: The Nature of Statistical Learning Theory. Springer Verlag, New York (1995)

This article was processed using the LTEX macro package with LLNCS style