Object Recognition Using Hierarchical SVMs

Katarina Mele[1] and Jasna Maver[1,2]

[1]Faculty of Computer and Information Science, University of Ljubljana, Slovenia
katarina.mele@fri.uni-lj.si,
[2]Faculty of Arts, University of Ljubljana, Slovenia
jasna.maver@ff.uni-lj.si

Abstract

The paper deals with the object recognition problem. The objective is to localize and recognize the known objects in different orientations on cluttered background. As a learning tool we choose support vector machines (SVMs). To eliminate the problem caused by cluttered background we organize the image pixels in tree structures, which enable us to deal only with the object pixels. Both, one- and two-class SVMs are combined in the recognition process. One-class SVMs, used at the first stage, allow us to avoid the "non-object" class generation as required by two-class SVM for object localization task. Two-class SVMs are applied to further resolve the recognition process when necessary. As demonstrated by experimental results the proposed method reduces the number of erroneously recognized objects.

1 Introduction

One of the challenging problems in computer vision is to learn objects and to recognize them under different conditions. Illumination, viewing angle, varying background, different scale, noise, and occlusions are factors that can affect recognition rate.

The paper deals with the object recognition problem. The objective is to localize and recognize the known objects in different orientations on a cluttered background. As a learning tool we choose support vector machines (SVMs). SVMs have been used for a variety of learning and pattern recognition tasks such as face recognition [4], pedestrian detection [7], and also for world-wide web searching [5], etc. In [10] the authors also worked with the problem of localizing known object on cluttered backgrounds. While the work of Roobaert [10] relies on pedagogical learning our method uses the hierarchical organization of data pixels in images.

This paper is organized as follows. In section 2 we summarize the standard formalization of SVMs, pose the problem to be solved, briefly describe BW (abbreviation for the Black-White) method [10], and give mathematical foundation for one-class SVMs.

In section 3 we develop hierarchical SVMs. The method is based on the hierarchical organization of image pixels and relies on one- and two-class SVMs.

Comparison of the proposed hierarchical method vs. BW method is given in the section 4. At the end we give conclusions and some proposals for future work.

2 Theoretical background

2.1 Support Vector Machines

SVM is a binary classifier. It is also possible to use it for multi class problems [2]. The SVMs are convenient for classification in high dimensional space and consequently suitable for image classification. A set of N images of size $p \times r$ can be represented as a set of points $\mathbf{x_i}$; $i = 1, 2, ...N$ in $\mathbb{R}^n$; $n = p \times r$. Each $\mathbf{x}_i$ can be a member of only one of the two classes $y_i \in \{-1, 1\}$. Pairs $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ form a training set. The optimal separating hyper-plane (OSH) is defined with $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ as follows:

$$(\mathbf{w} \cdot \mathbf{x}_i) + b = 0. \tag{1}$$

The computation of $\mathbf{w}$ is achieved by minimizing $||\mathbf{w}||$ under correct classification of the training set, i.e.,

$$\forall i, \;\; y_i f(\mathbf{x}_i) \geq 1.$$

This is equivalent to maximizing the margin between the training points and the separating hyper-plane, and ensures good generalization property on the real population. It can be proven [2] that $\mathbf{w}$ is of the form $\sum_i \alpha_i y_i \mathbf{x}_i$ where α_i come from the following quadratic optimization problem: Minimize

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \mathbf{x}_i \mathbf{x}_j$$

under

$$\forall i, \quad \alpha_i \geq 0 \;\text{ and }\; \sum_i \alpha_i y_i = 0.$$

The value of b does not appear in the optimization. It has to be computed, given the α_i:

$$b = -\frac{\max_{y_i=-1} \sum_j \alpha_j y_j \mathbf{x}_i \mathbf{x}_j + \min_{y_i=1} \sum_j \alpha_j y_j \mathbf{x}_i \mathbf{x}_j}{2}.$$

Finally, using the expansion of $\mathbf{w}$, we can write the classification function as:

$$f(\mathbf{x}) = \sum_{\mathbf{i}} \alpha_{\mathbf{i}} \mathbf{y_i} \mathbf{x_i} \mathbf{x} + \mathbf{b}. \tag{2}$$

A new point $\mathbf{x}_i$ is classified according to the sign of $f(\mathbf{x})$, i.e., we test which side of the hyper-plane it belongs to.

2.2 Appearance based learning of 3-D objects with SVMs

The usual approach to appearance based learning of 3-D objects with SVMs is as follows [9]. The objects that are to be learned are individually placed on a turntable and images are taken for different orientations of the turntable. The acquired sets of images are then used to train SVMs. An example of such a training set is given in Figure 1.

By rotating the turntable the area in images belonging to object projection changes from view to view. Consequently, some pixels in images may for same viewing directions belong to an object while for the other may belong to a background. As we have seen in the previous subsection the SVM classifier treats images as points in $\mathbb{R}^n$. The location of a point is determined by both, the object pixels as well as the background pixels. Different scenes can drastically change the values of background pixels and can move the point to the wrong side of the hyper-plane.

2.3 BW Method

The BW method tries to be invariant to the cluttered background. To overcome the problem of cluttered background additional training images can be generated; the original background can be replaced with all possible backgrounds. The number of all such images is enormous, therefore Roobaert [10] proposes a data selection approach, so called pedagogical learning. He suggests to take as background values only extreme values. In practice this means that the training set consists of objects pasted onto a white and black background.

In the case of two-class SVMs (Eq. 2) everything we classify must belong to one class. In order to distinguish between the known objects and other parts of the scene a new class with the name "non-object" is introduced. In fact all images that do not represent the known object belong to the non-object class. Generating a representative non-object image set is a difficult task and unfortunately very important for appropriate classification. The consequence of an inappropriate training set of the non-object class is a large number of false positives. This means that the method detects the known objects at the places in the image where there is a background.

To avoid the problem of generating non-object class one can use the one-class SVM method.

2.4 One-Class SVM

It is reasonable to assume that images of an object cluster in a certain way and that images of the non-object do not because they can be almost anything. One-class SVM has been successfully used in situations where there is lack of information about some classes [1, 11]. The idea of one-class SVM is to put all the data of one-class into a hyper-sphere. The formulation of the problem is the following:

Consider a set of points $\mathbf{x}_i \in \mathbb{R}^n; i = 1, 2, \ldots, N$ belonging to the same class. Let Φ be a feature map $\mathbb{R}^n \to F$ such that the dot product in the image of Φ can be computed by evaluating some simple kernel:

$$k(\mathbf{x}, \mathbf{y}) = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y}))$$

We are looking for the smallest hyper-sphere with radius R and center c that include all $\mathbf{x}_i$:

$$\min_{R \in \mathbb{R}, c \in F} R^2$$
$$\text{subject to } \| \Phi(\mathbf{x}_i) - c \|^2 \leq R^2 \text{ for } i = 1, 2, \ldots, N.$$

We solve the dual problem:

$$\min_{\alpha} \sum_{ij} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}_i)$$
$$\text{subject to } 0 \leq \alpha_i, \quad \sum_i \alpha_i = 1.$$

The solution is:

$$c = \sum_i \alpha_i \cdot \Phi(\mathbf{x}_i)$$

and the decision function:

$$f(\mathbf{x}) = \text{sgn}(R^2 - \sum_{ij} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) + 2 \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}) - k(\mathbf{x}, \mathbf{x})). \quad (3)$$

R^2 is computed for Equation 3 such that for any $\mathbf{x}_i$ with $0 < \alpha_i$ the argument of the sgn is 0.

The function $f(\mathbf{x})$ (Eq. 3) is positive inside hyper-sphere and negative on the complement.

To accept also the points in the near vicinity of the hyper-sphere we relax the decision function (Eq. 3) by threshold t as follows:

$$f(\mathbf{x}) = \text{sgn}(R^2 - \sum_{ij} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) + 2 \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}) - k(\mathbf{x}, \mathbf{x}) - t). \quad (4)$$

3 Hierarchical One-Class SVM

Due to the fact that the BW method has a large false positive error rate (see Experiments), we decide to use one-class SVMs for the object detection and recognition tasks. One-class SVMs can not exploit the idea of pedagogical learning as it is done by the BW method. Black and white backgrounds spread the points in $\mathbb{R}^n$, which lead to large hyper-spheres and consequently to poor separability of different classes. To avoid the problem of cluttered background we organize the image pixels in a hierarchical structure which allow us to deal only with the object pixels.

3.1 Building a tree structure and training

We line up the images of an object according to a rotation angle. The reason for such an arrangement is the fact that images with close viewing angles are more similar.

Let X_i denote an image from the training set and let us form a set of binary images $B_i;\ i = 1, \ldots, N$ such that for each pixel of B_i

$$B_i(c,l) = \begin{cases} 1 & \text{if } X_i(c,l) \text{ is an object pixel} \\ 0 & \text{if } X_i(c,l) \text{ is a background pixel.} \end{cases}$$

First, the AND operation is performed on pixels of B_i:

$$M_1(c,l) = \bigwedge_{i=1}^{N} B_i(c,l) \quad c = 1, \ldots, p,\ l = 1, \ldots, r. \quad (5)$$

The computed intersection (Eq. 5) determines the mask of level 1 of the tree structure. The mask is a binary image with value 1 at the pixels marked as objects in all images and 0 otherwise. Next, we divide the image set in half and calculate the intersection mask for each half separately. Accordingly, level 2 is formed by two masks. Similarly, level 3 is formed by the intersection masks obtained by further partitioning of the image groups. Figure 2 represents such a tree structure of mask for the image set depicted in Figure 1.

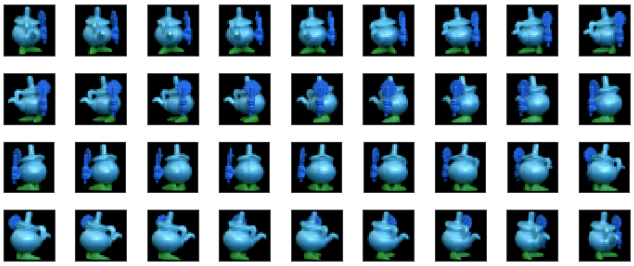

Figure 1: Acquired image set for a teapot.

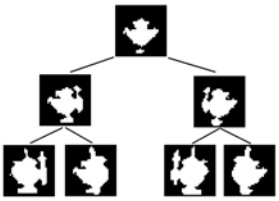

Figure 2: Tree structure of masks.

The active parts of the masks (pixels with value one) determine areas in images used in the process of learning at each level of the tree. Figures 3, 4 and 5 show three groups of masked images.

Next, we start with the training phase. For each group of masked images we compute a decision function determined by equation 4, i.e., each group is represented by its own hyper-sphere. Hence, for each object we have seven hyper-spheres.

We can notice that the active part of the mask at level 1 is a subset of the active part of the masks at level 2, and for each path separately, the active part of the mask at level 2 is a subset of the active part of the masks at level 3. Although, the hyper-spheres on lower level encompass less dimensions of $\mathbb{R}^n$ then the hyper-spheres on higher levels, they usually have bigger radius, which is the consequence of greater data set and consequently the greater variability of image pixels.

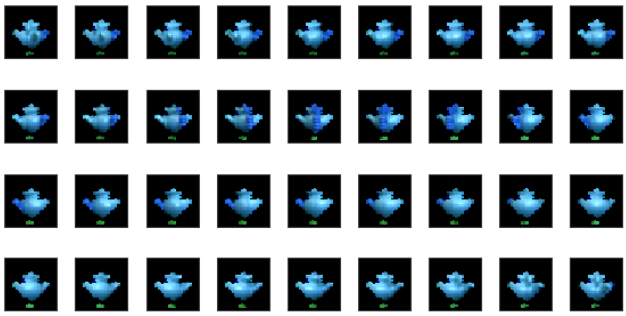

Figure 3: Masked images at level 1.

Figure 4: One of the sets of masked images at level 2.

Figure 5: One of the sets of masked images at level 3.

Thus, after the training phase the tree structure is built from masks and corresponding hyper-spheres. Tree structures have to be built for all objects.

3.2 Recognition stage

The recognition stage requires both localization and recognition of known objects in the image of the scene. To localize the known object we move the image window over the test image. At each location we perform recognition tests. The current pattern is first masked with the appropriate mask and then classified as a member of the class if it lies inside the corresponding hyper-sphere. We start with the hyper-sphere at level 1 and then continue with level 2 and level 3. The pattern is said to be the known object if there exists a path in the tree form root to the one of the leaves which gives the consistent classification on all levels. Figure 6 shows an example of a successful and also unsuccessful recognition test.

When searching for the particular known object in the given image, the level 1 of the tree structure allow us to quickly discard many patterns that certainly are not the known objects, while levels 2 and 3 further purify the recognition process.

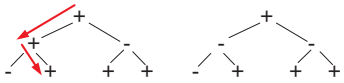

Figure 6: Two examples of recognition tests. Left tree shows an example where the current pattern is recognized as a known object while the one on the right shows an example where recognition fails.

3.3 Two-class SVM inside hyper-sphere

Since the decision function (Eq. 4) is computed on the basis of images that belong to only one object it can happen that the corresponding hyper-sphere also contains the points of other objects. In this case the computed hyper-spheres have nonempty intersections.

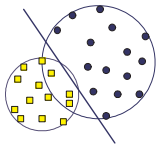

Figure 7: Combining one- and two-class SVMs.

Whenever two hyper-spheres intersect we have to approve the recognition process by two-class SVM. Again, a hierarchical principle is applied. The computation of masks for two-class SVMs and their composition into a tree structures is done in a similar way as for one-class SVM. The only difference follows from the image sets which include the images of two objects. The partitioning of the sets is done in accordance with the viewing angles as with the case of one-class SVM. An example of such a three structure of intersection masks is given in Figure 8.

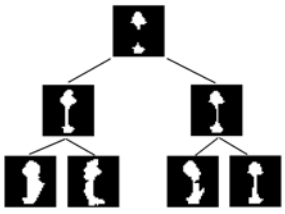

Figure 8: Tree structure of masks for a mouse and a ghost.

Let us demonstrate the procedure by an example. Assume that at level 1 of the tree structure the hyper-spheres of object 1, 2, and 3 intersect and we are searching for object 1. The approval of the recognition results obtained by one-class hierarchical SVM classifier also have to be done by hierarchical two-class classifiers, i.e., object 1 against object 2 and object 1 against object 3. Figure 9 gives an example of an unsuccessful recognition process. While the classifier *object 1 against object 2* is successful the classifier *object 1 against object 3* fails and the approval of object 1 is negative.

4 Experiments

In all given experiments data sets are linearly separable, therefore, there are no training errors. For SVM evaluation

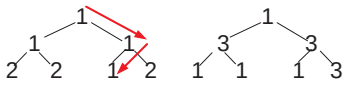

Figure 9: Successful and unsuccessful recognition test with two-class SVM.

we use the SMO [8] method from Statistical Pattern Recognition Toolbox for Matlab [3]. We deal with color images of size $32 \times 32 \times 3$.

4.1 Comparison of BW method vs. hierarchical classifier

4.1.1 Robustness to cluttered background In the first experiment we compare BW classifier against hierarchical classifier. We used four different objects: a teapot, a ghost, a mouse, and a pelican. For each object we acquired 72 images with 5 degree increments of a turntable. One half of the images (odd views) were used in the training phase, and the other half (even views) in the test phase. Here all the objects appear on a black background.

The training set for the BW method was extended by white and six additional color backgrounds. Colors of all backgrounds follow from all possible triples of values 0 and 255, e.g., $(0, 255, 255)$ and $(0, 0, 255)$.

A window of size 32×32 was shifted over the test images. At each location we performed a recognition test with both methods.

In the first example the test image was combined by objects on a black background (Fig. 10). One can notice that at some places the window does not cover only one object.

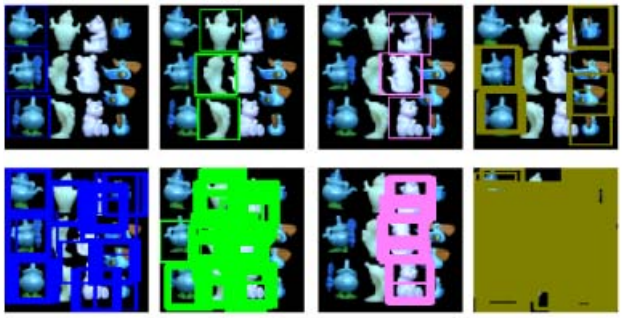

Figure 10: Example with a black background.

object method	teapot FP%(TP:FP)	ghost FP%(TP:FP)	mouse FP%(TP:FP))	pelican FP%(TP:FP)
HSVMs	0 (5:0)	0 (6:0)	0 (4:0)	61 (18:28)
BW	87 (27:179)	89 (41:340)	64 (58:102)	83 (294:1429)

Table 1: Recognition results: All objects are correctly recognized. False positives are represented separately for each object.

The first row of Figure 10 shows the results obtained by hierarchical SVMs, while the second row shows the results obtained by the BW method. The teapot, the ghost, the mouse, and the pelican were searched in the first, second, third, and fourth column, respectively. Squares show the places in images where the particular object was found. Table 1 gives the numerical evaluation of the recognition re-

sults.

The results show that the hierarchical method has a lower false positive rate. The only false positives are due to the fact that the pelican's back side is very similar to the central part of the teapot.

We notice that the BW method is less sensitive to shift. This is the reason why the squares around the objects in the second row are normally thicker than they are around objects in the first row.

The second example (Figure 11) is done with a color background. The color of the background of the test image is reddish with slightly varying intensity. Objects stay in the same composition as previously.

The results show that the hierarchical method is almost insensitive to the change of the background color while the BW method is not. Here the false positive rate of the BW method is quite different specially for the teapot and the pelican (see Table 2).

Figure 11: Example with a color background.

object method	teapot FP%(TP:FP)	ghost FP%(TP:FP)	mouse FP%(TP:FP))	pelican FP%(TP:FP)
HSVMs	0 (4:0)	0 (6:0)	0 (4:0)	69 (19:42)
BW	62 (26:42)	87 (38:260)	47 (54:47)	69 (174:382)

Table 2: Recognition results: All objects are correctly recognized. False positives are represented separately for each object.

In the third example (Fig. 12) the background is taken from the real world. Also in this case, the hierarchical method outperforms the BW method (Table 3).

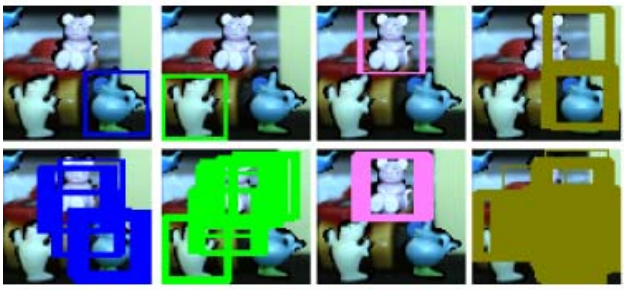

Figure 12: Example with background from the real world. First row: hierarchical method; second row: BW method.

4.1.2 Robustness to noise Both methods were tested on noisy images. The experiment were done with uniform noise. In the first experiment (Fig. 13) a random value

object method	teapot FP%(TP:FP)	ghost FP%(TP:FP)	mouse FP%(TP:FP))	pelican FP%(TP:FP)
HSVMs	0 (3:0)	0 (4:0)	0 (3:0)	100 (0:29)
BW	57 (39:52)	87 (11:75)	0 (33:0)	100 (0:302)

Table 3: Recognition results: All objects are correctly recognized. False positives are represented separately for each object.

from the interval $[-25, 25]$ was added to each image pixel. The hierarchical method failed only to recognize the teapot, while the BW method successfully recognized all three objects, but has a huge number of false positives specially for the pelican which was not present in the scene.

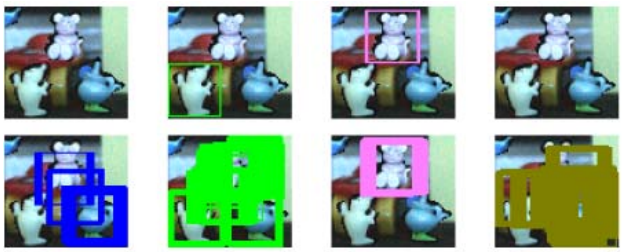

Figure 13: Example with uniform noise $[-25, 25]$. First row: hierarchical method; second row: BW method.

In the second example we enlarge the amplitude of uniform noise. A random value from the interval $[-50, 50]$ (Fig. 14) was added to each image pixel. The recognition results were similar. The hierarchical method was not capable of recognizing the teapot while the pelican was recognized as the teapot. The BW method again gave huge number of false positives.

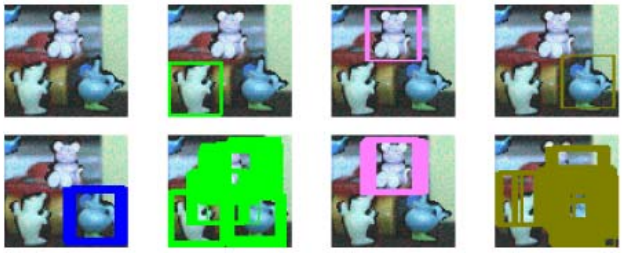

Figure 14: Example with uniform noise $[-50, 50]$. First row: hierarchical method; second row: BW method.

4.2 Test on COIL-100 database

The second experiment was performed on COIL-100 database [6]. Figure 15 shows 20 selected objects used in the experiment. Only the hierarchical SVM method was tested. Experiments were performed on two different test images (Figs. 16 and 17).

In both examples all 20 objects were searched. In the first example (Fig. 16) all four objects in the test image were correctly recognized. There were no false positives.

In the second example (Fig. 17) there are six known objects (obj.: 2, 3, 30, 33, 35, 50) in the test image. All the object were correctly recognized. On six images we can also notice erroneously recognized objects, i.e., false positives.

The proposed hierarchical SVMs are insensitive to clut-

Figure 15: 20 selected object from COIL-100 database.

Figure 16: Results of recognition process for the first test image. Obj1 (first row left), obj20 (first row right), obj24(second row left), and obj47(second row right) are correctly recognized.

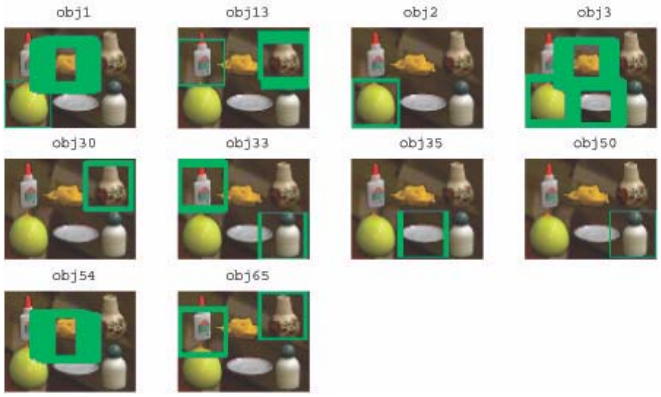

Figure 17: Results of recognition process for the secod test image. Text above the images gives the label of searching object.

tered background and very precisely locate objects in the image. This could also represent a disadvantage in comparison to the BW method since it requires recognition tests at each pixel location in the test image. On the other hand it can be faster than the BW classifier because it can refuse non-object patterns at the first level of tree structure.

5 Conclusion and further work

In the paper we propose a method for hierarchical SVMs. The hierarchical structures allow us to avoid the problem of cluttered background. Hierarchical trees were built from one- and two-class SVMs. The combination of both improves the recognition process and reduces the number of false positives. The hierarchical SVM allow us also to recognize the sides of the object, e.g., front side, back side, left side, and right side.

The method can be improved by adapting the number of levels in tree structure to the complexity of the learning objects. Objects with complex 3D shape require a representation tree with more levels, while symmetrical objects (like obj47 in figure 15) do not need hierarchical representation. In the future it would also be interesting to develop and analyze hierarchical kernel SVM.

References

[1] Y. Chen, X. Zhou, and T. Huang. One-class SVM for learning in image retrieval. In *Proceedings of the 2001 IEEE International Conference On Image Processing (ICIP-01)*, pages 34–37, Thessaloniki, Greece, Oct. 7–10 2001. IEEE.

[2] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines (and Other Kernel-Based Learning Methods)*. CUP, 2000.

[3] V. Franc and V. Hlavac. Statistical pattern recognition toolbox for matlab, 2000.

[4] B. Heisele, P. Ho, and T. Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Proceedings of the Eighth International Conference On Computer Vision (ICCV-01)*, pages 688–694, Los Alamitos, CA, July 9–12 2001. IEEE Computer Society.

[5] I. Kkai and A. Lrincz. Fast adapting value estimation-based hybrid architecture for searching the world-wide web. *Applied Soft Computing*, 2(1):11–23, 2002.

[6] S. Nene, S. Nayar, and H. Murase. Columbia object image library: Coil, 1996.

[7] C. Papageorgiou and T. Poggio. Trainable pedestrian detection. In *Proceedings of the 1999 International Conference on Image Processing (ICIP-99)*, pages 35–39, Los Alamitos, CA, Oct. 24–28 1999. IEEE.

[8] J. C. Platt. Sequential Minimal Optimizer: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, 1998.

[9] M. Pontil and A. Verri. Support vector machines for 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.

[10] D. Roobaert. *Pedagogical Support Vector Learning: a pure learning approach to object recognition*. PhD thesis, Royal Institute of Technology (KTH), Dept. of Numerical Analysis and Computing Science (NADA), Stockholm, Sweden, 2001.

[11] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.