Representing shape with a spatial pyramid kernel

Anna Bosch
University of Girona
Computer Vision Group
17003 Girona, Spain
aboschr@eia.udg.es

Andrew Zisserman
University of Oxford
Robotics Research Group
OX1 3PJ Oxford, UK
az@robots.ox.ac.uk

Xavier Munoz
University of Girona
Computer Vision Group
17003 Girona, Spain
xmunoz@eia.udg.es

ABSTRACT

The objective of this paper is classifying images by the object categories they contain, for example motorbikes or dolphins. There are three areas of novelty. First, we introduce a descriptor that represents local image shape and its spatial layout, together with a spatial pyramid kernel. These are designed so that the shape correspondence between two images can be measured by the distance between their descriptors using the kernel. Second, we generalize the spatial pyramid kernel, and learn its level weighting parameters (on a validation set). This significantly improves classification performance. Third, we show that shape and appearance kernels may be combined (again by learning parameters on a validation set).

Results are reported for classification on Caltech-101 and retrieval on the TRECVID 2006 data sets. For Caltech-101 it is shown that the class specific optimization that we introduce exceeds the state of the art performance by more than 10%.

Categories and Subject Descriptors

I.4.8 [**Computing Methodologies**]: Image Processing and Computer Vision—*Scene Analysis*

General Terms

Measurement Performance

Keywords

Shape features, Spatial Pyramid Kernel, Object and video retrieval

1. INTRODUCTION

We consider the problem of image classification where our main goal is to explore how the spatial distribution of shape can benefit recognition. Much recent work has used a "bag of (visual) words" representation together with an SVM classifier in order to classify images by the objects they contain [6, 27]. These methods represent local appearance, but not shape directly. However, representations of shape using the spatial distribution of edges [19, 20, 21] often perform as well as or even better than local appearance patches, but recognition involves computing geometric consistency using Hough like accumulators – losing the simple vector representation of a bag of words. We introduce a new descriptor which has the advantages of both: it captures the spatial distribution of edges, but is formulated as a vector representation. Similarity on descriptor vectors between two images (for example measured using histogram intersection or χ^2) then measures the similarity of their spatial distribution of edges.

Our descriptor is mainly inspired by two sources: (i) the image pyramid representation of Lazebnik *et al.* [14], and (ii) the Histogram of Gradient Orientation (HOG) of Dalal and Triggs [7].

In essence, we wish to assess how well an exemplar image matches (the shape of) another image. As in [14] the intuition is that a strong match goes beyond a "bag of words" and also involves a spatial correspondence. To this end we represent shape in the form of edges, replacing the quantized appearance patches (visual words) of [14], and learn a class specific weighting for the levels of the hierarchical spatial histogram [11, 14]. This captures the intuition that some classes are very geometrically constrained (such as a stop sign) whilst others have greater geometric variability (e.g. dolphins, boats). The details of the descriptor, termed PHOG (for Pyramid of Histograms of Orientation Gradients) are given in Section 2, and the idea is illustrated in Fig. 1.

The flexibility of the spatial histogram level weighting means that a spectrum of spatial correspondences between two images can be represented. If only the coarsest level is used, then the descriptor reduces to a global edge or orientation histogram, such as used by [13, 24]. If only the finest level is used, then the descriptor enforces correspondence for tiles (spatial bins) over the image. This extreme is what is captured by [7, 23] where histograms are computed over local image regions. Other weightings of the spatial levels capture geometric consistency between these extremes. We compare the PHOG descriptor to a standard shape descriptor, Chamfer matching, in Section 2.3.

Having developed the PHOG descriptor we then introduce kernels, suitable for an SVM classifier, that combine both appearance (visual words) and edge (PHOG) descriptors. This is a form of feature combination and selection, but here the selection is at the kernel level. Again, in a class-specific learning step, the descriptors (appearance or shape or both) most suitable for a particular class are determined.

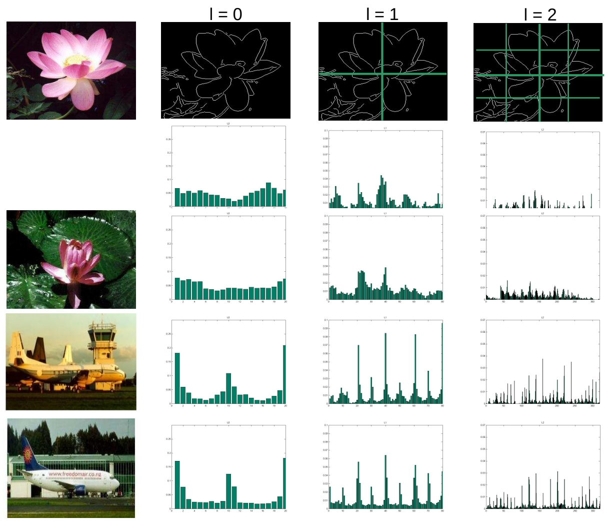

Figure 1: Shape spatial pyramid representation. Top row: an image and grids for levels $l = 0$ to $l = 2$; Below: histogram representations corresponding to each level. The final PHOG vector is a weighted concatenation of vectors (histograms) for all levels. Remaining rows: images from the same and from different categories, together with their histogram representations.

For example, a category such as *car* is best described by shape alone, *leopard* by appearance alone, and *buddha* by a combination of the two. This generalizes previous approaches that have used a fixed linear weighting to combine feature vectors [13]. The kernels are described in Section 3.

Sections 4–6 describe the datasets used, implementation details, and the experimental procedure and results on classification for Caltech-101 [8] and retrieval for TRECVID 2006. It is shown that the set of innovations introduced here lead to a 10% performance improvement over the previous best result on Caltech-101 [26].

2. SPATIAL SHAPE DESCRIPTOR – PHOG

Our objective is to represent an image by its local shape *and* the spatial layout of the shape. Here local shape is captured by the distribution over edge orientations within a region, and spatial layout by tiling the image into regions at multiple resolutions. The idea is illustrated in Fig. 1. The descriptor consists of a histogram of orientation gradients over each image subregion at each resolution level – a Pyramid of Histograms of Orientation Gradients (**PHOG**). The distance between two PHOG image descriptors then reflects the extent to which the images contain similar shapes *and* correspond in their spatial layout.

The following sub-sections describe these two aspects (local shape and spatial layout correspondence) in more detail.

2.1 Local Shape

Local shape is represented by a histogram of edge orientations within an image subregion quantized into K bins. The contribution of each edge is weighted according to its magnitude, with a soft assignment to neighbouring bins in a manner similar to SIFT [17]. Implementation details are given in Section 5.

Each bin in the histogram represents the number of edges that have orientations within a certain angular range. This representation can be thought of as a traditional "bag of (visual) words", where here each visual word is a quantization on edge orientations. A similar representation is used in [2]. We will refer to the representation of each region as a Histogram of Orientated Gradients [7] (HOG).

2.2 Spatial Layout

In order to introduce spatial information we follow the scheme proposed by Lazebnik *et al.* [14] which is based on spatial pyramid matching [11]. Consider matching two images each consisting of a 2D point set, where we wish to determine soft matches between the point sets when the images are overlaid – for a particular point the strength of the match depends on the distances from its position to points in the other set. Each image is divided into a sequence of increasingly finer spatial grids by repeatedly doubling the number of divisions in each axis direction (like a quadtree). The number of points in each grid cell is then recorded. This is a pyramid representation because the number of points in a cell at one level is simply the sum over those contained in the four cells it is divided into at the next level. The cell counts at each level of resolution are the bin counts for the histogram representing that level. The soft correspondence between the two point sets can then be computed as a weighted sum over the histogram intersections at each level. Similarly, the lack of correspondence between the point sets can be measured as a weighted sum over histogram differences at each level.

In our case, a HOG vector is computed for each grid cell at each pyramid resolution level. The final PHOG descriptor for the image is a concatenation of all the HOG vectors. In forming the pyramid the grid at level l has 2^l cells along each dimension. Consequently, level 0 is represented by a K-vector corresponding to the K bins of the histogram, level 1 by a $4K$-vector etc, and the PHOG descriptor of the entire image is a vector with dimensionality $K \sum_{l \epsilon L} 4^l$. For example, for levels up to $L = 1$ and $K = 20$ bins it will be a 100-vector. In the implementation we limit the number of levels to $L = 3$ to prevent over fitting.

The PHOG is normalized to sum to unity. This normalization ensures that images with more edges, for example those that are texture rich or are larger, are not weighted more strongly than others.

Fig. 1 shows that images from the same category have a similar PHOG representation and that this representation is discriminative between categories. Note, PHOG is not the same as a scale space pyramid representation of edges [16] as there is no smoothing between levels of the pyramid, and all edges are computed on the high resolution image.

Similarity between a pair of PHOGs is computed using a distance function, with appropriate weightings for each level of the pyramid. In previous work [11, 14], the distance function was histogram intersection and the weightings were fixed and class independent. In this paper we learn the weightings for the levels, and show that a χ^2 distance has superior performance to histogram intersection.

2.3 What Is Being Represented?

In order to gain an intuition into what is represented by a difference between PHOG descriptors, we show here

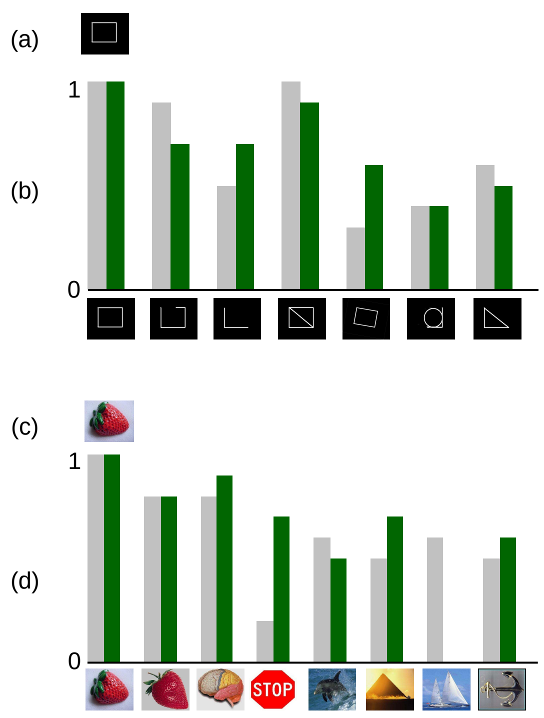

Figure 2: A comparison of difference in PHOG descriptors using χ^2 (gray bars) to Chamfer distance (dark green bars). Note, the y-axis shows 1−distance, so that a perfect match corresponds to unity, and a poor match to zero. (a) shows a synthetic model image and (b) its matching to the synthetic images (arranged on the x-axis). Similarly, (c) shows a real model image and (d) its matching to images from Caltech-101.

its relation to Chamfer distance between the edge maps. Comparing boundaries using Chamfer [3] has proven to be a reliable and efficient method for object recognition e.g. the pedestrian detector of Gavrila & Philomen [10] and the hand tracking of [22]. The chamfer distance between two curves measures the average over the closest distance between them. If the model and target curves are represented by the point sets $\{\mathbf{x}_m\}$ and $\{\mathbf{x}_t\}$ then the Chamfer distance can be computed as:

$$Chamfer = \frac{1}{N_m}\sum_m min_{\mathbf{x}_t}||(\mathbf{x}_m - \mathbf{x}_t)|| \qquad (1)$$

In addition for curves, rather than point sets, edges are only matched if they have similar orientations, and also distance is capped to reduce sensitivity to "outliers", such as missed edges through detector drop out [22].

To illustrate the similarity, in Fig. 2 we compare Chamfer distance to the difference between PHOG descriptors computed using χ^2. It can be seen in (a) that both have similar behaviours: both can tolerate missed edges to some extent (second and third examples where an edge is missing) and background clutter (fourth example where an edge is added). PHOG copes better with rotated images due to the additional slack given by computing orientation histograms over regions, whereas Chamfer will cease to find a matching edge that has the same orientation. Fig. 2(b) shows that for real images (Caltech-101), Chamfer and PHOG again have similar behaviour for the most part.

This raises the question of why it is necessary to introduce a new descriptor at all. The answer is that PHOG has three advantages over Chamfer: (i) insensitivity to small rotation (mentioned above). More significantly, (ii) PHOG is a compact vector descriptor suitable for use in standard learning algorithms with kernels. Chamfer matching can be reformulated as a distance between vectors as shown by Felzenswalb [9] (and similarly for Hausdorff matching). However, the vector has the dimension of the number of pixels in the image and will not be particularly sparse; (iii) the principal advantage is that Chamfer matching requires strict spatial correspondence whereas PHOG is flexible, since it builds in spatial pyramid matching, and is able to cope with varying degrees of spatial correspondence by design.

3. MODELING SHAPE AND APPEARANCE

In this section we describe the kernels that will be used in the SVM classifiers for classifying and retrieving images according to their class (e.g. containing a motorbike or a road). We begin by describing the kernel for the PHOG descriptor, and then introduce two kernels suitable for combining or selecting between appearance and shape representations.

3.1 Kernel Definition

If images I and J are represented by the PHOG feature vectors S_I and S_J, then the similarity between the images is defined as

$$K(S_I, S_J) = \sum_{l \epsilon L} \alpha_l d_l(S_I, S_J) \qquad (2)$$

where α_l is the weight at level l and d_l the distance between S_I and S_J at pyramid level l computed using χ^2 (though other distances could be used). This defines the kernel for PHOG similarity.

In the original spatial pyramid representation [14] each level was weighted using $\alpha_l = 1/2^{(L-l)}$ where L is the number of levels and l the current level. This means that histograms from finer resolutions are weighted more highly than those at coarser resolutions. However, this may not be the optimum weight choice for classification performance. We investigate two methods for learning the weights:
GLW – Global Level-Weights. Instead of giving a fixed weight to each pyramid level as in [14], we *learn* the weights α_l which give the best classification performance over all categories.
CLW – Class specific Level-Weights. Instead of learning weights common across all classes, the weights α_l are learnt for each class separately by optimizing classification performance for that class using one vs the rest classification. This means that for the 100 categories of Caltech-101 it is necessary to learn 400 parameter values (for $L = 3$ levels) instead of only 4 for GLW.

The advantage of learning class-specific level-weights is that classes then have the freedom to adapt if there is more or less intra-class spatial variation. The disadvantage is that the solution is sub-optimal since performance is not optimized over all categories simultaneously. Details and results are given in Section 6.1.

3.2 Merging Features

It was shown in Section 2 that shape represented by PHOG is a good measure of image similarity and thus for image

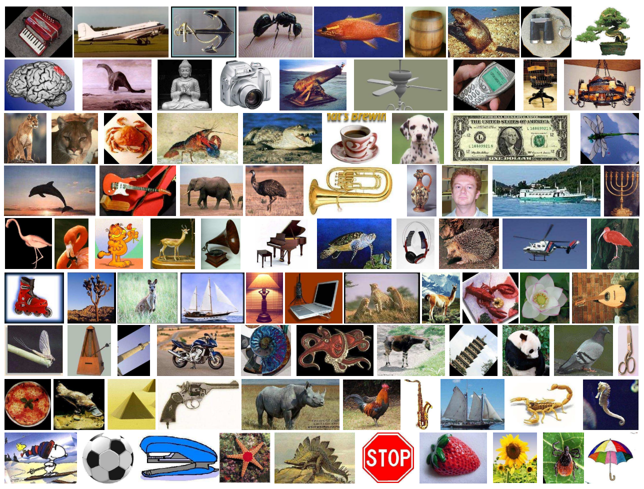

Figure 3: Images from the Caltech-101 dataset. One per category.

classification. However, shape features alone are not sufficient to distinguish all types of images, as is shown for example by the *strawberry* and *brain* examples in Fig. 2b. In this case, appearance [4, 14] is a better feature to distinguish them. Consequently, we investigate here kernels to combine these two features (shape & appearance). The appearance features and kernel are described in Section 5.2.

We consider two kernel combinations. The first is a simple linear blend, the second involves a hard choice between the feature types using a max operation.

The first merging kernel that we propose is based on the weighted sum of appearance and shape information:

$$K(x, y) = \alpha K_A(x_{App}, y_{App}) + \beta K_S(x_{Shp}, y_{Shp}) \qquad (3)$$

where α and β are the weights for the appearance and shape kernel (2) respectively. It has the capacity to give higher weights to the more discriminative features during learning. Moreover it also has the capability to ignore features which do not match well if $\alpha = 0$ or $\beta = 0$. It is a Mercer kernel [12]. Previous authors have considered similar merging kernels, but the optimization differs. We optimize performance over a validation set directly whereas in [1, 15] the interest is in efficient optimization of a proxy for classification performance.

The second merging kernel is based on taking the maximum value of the appearance or shape kernel:

$$K(x, y) = \max[K_A(x_{App}, y_{App}), K_S(x_{Shp}, y_{Shp})] \qquad (4)$$

This kernel has the ability to ignore the appearance or shape features if those features do not match well for a particular exemplar image. Note that it is not a Mercer kernel, but this has not proven to be a problem in practice.

For kernel learning we again consider two situations:
GFW – Global Feature-Weights. Optimize weights α and β in (3) over all classes together, so that all the classes will have the same weights for the kernel features used.
CFW – Class specific Feature-Weights. Optimize weights α and β in (3) for each class separately. Again, these weights are learnt by classifying that class against all others.

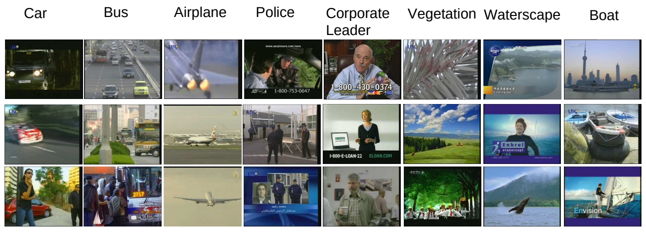

Figure 4: Training images from TRECVID. Each column shows three different examples. Note the high intra-class variability.

4. DATASETS

Caltech-101. This dataset (collected by Fei-Fei *et al.* [8]) consists of images from 101 object categories, and contains from 31 to 800 images per category. Most images are medium resolution, about 300 × 300 pixels. The significance of this database is its large inter-class variability. Fig. 3 shows images for each category from the dataset.

Fig. 6 shows examples of what is being represented by the spatial shape descriptor for this dataset. From the average image (averaged over 25 training images) it is evident that images from the same category are very well centred and do not suffer from much rotation. The average gradient and edge images show the strong alignment of the principal object edges within a class – note in particular the alignment of the cup and metronome boundaries. The gradient magnitude weighting of the histogram bins is particularly beneficial here as the strong gradients that are common within a class score highly in the HOG, in turn reinforcing the similarity of PHOGs for images of the same class. It is clear from the averaged orientation histograms for two levels that more local spatial-shape ($l = 3$) is able to distinguish between classes better than global ($l = 0$).

TRECVID[1]. We use the annotated training data from TRECVID 2006, which consists of 80 hours of video sequences. Video shots are annotated into 39 semantic categories which occur frequently in the database (e.g. *sports*, *entertainment*, *weather*). TRECVID also provides keyframes for each shot, and only these frames are used here for learning and shot retrieval. There are a total of 43907 keyframes. A sample is shown in Fig. 4. Note the difficulty of these images.

5. IMPLEMENTATION

For image classification we use the kernels defined in Section 3 in a SVM classifier [5]. Multi-way classification is achieved using a one-versus-all SVM: a classifier is learnt to separate each class from the rest, and a test image is assigned the label of the classifier with the highest response. For the retrieval results in TRECVID 2006 we use the probability estimate provided by [5] to rank the representative keyframes (shots).

5.1 Shape Implementation

Edge contours are extracted using the Canny edge detector on monochrome versions of the images (with intensity $I = 0.3R + 0.59G + 0.11B$). The orientation gradients are then computed using a 3 × 3 Sobel mask without Gaussian

[1]http://www-nlpir.nist.gov/projects/trecvid/

smoothing, since [7] showns that smoothing significantly decreases classification performance. The HOG descriptor is discretized into K orientation bins. The vote from each contour point depends on its gradient magnitude, and is linearly distributed across the two neighbouring orientation bins according to the difference between the measured and actual bin orientation. Histograms with K ranging between 10 and 80 bins are tested.

In the experiments two HOG descriptors are compared: one with orientations in the range $[0, 180]$ (where the contrast sign of the gradient is ignored) and the other with range $[0, 360]$ using all orientation as in the original SIFT descriptor [17]. We refer to these as $Shape_{180}$ and $Shape_{360}$ respectively.

5.2 Appearance Implementation

A dense representation is computed as described in Bosch *et al.* [4]. In short, SIFT descriptors are computed at points on a regular grid with spacing M pixels, here $M = 10$. At each grid point the descriptors are computed over circular support patches with radii $r = 4$, 8, 12 and 16 pixels. Consequently each point is represented by four SIFT descriptors. Multiple descriptors are computed to allow for scale variation between images. The patches with radii 4 do not overlap and the other radii do. The image can be represented using colour (termed App_{Colour}) or only monochrome information (termed App_{Gray}). For App_{Colour} the SIFT descriptors are computed for each HSV component. This gives a 128×3 D-SIFT descriptor for each point. In the case of App_{Gray} SIFT descriptors are computed over the gray image (again with intensity I = 0.3R + 0.59G + 0.11B) and the resulting SIFT descriptor is a 128 vector. Note that the descriptors are rotation invariant. The dense features are vector quantized into *visual words* using k-means clustering. The k-means clustering was performed over 5 training images per category selected at random. A vocabulary of 300 words is used here. Each image is then represented by a histogram of word occurrences. This forms the feature vector for an SVM classifier, here using the spatial pyramid kernel of (2).

6. EXPERIMENTAL RESULTS

Following standard procedures, the Caltech-101 data is split into 30 (25 for training and 5 for the validation set) training images (chosen randomly) per category and 50 for testing – disjoint from the training images. The validation set is used to optimize all the parameters (e.g. K, α_l). The final performance score is computed as the mean recognition rate per class, so that more populous (and easier) classes are not favoured. The classification process is repeated 10 times, (changing the training, validation and test sets), and the average performance score and its standard deviation are reported.

For the TRECVID data we used the most representative keyframe of each video shot (43907 keyframes). This data is also split into three disjoint sets: 27093 keyframes for training, 3900 (100 per category) for the validation set and 12914 for testing. In this case precision vs. recall graphs are computed and the average precision is reported as a performance measure, following the TRECVID guidelines.

6.1 Parameter Optimization – Caltech 101

Parameter optimization is carried out on the validation set. For example, weights α_l, (2) are learnt by maximizing the performance score on this set. The optimization is carried out by an exhaustive search over a range of values with granularity 0.1 for α_l, α and β, and granularity 10 for K.

Distance measures. We explore here three distance measures: histogram intersection, χ^2, and the normalized scalar product (cosine of angle). For this experiment $\alpha_l = 1$ and $K = 20$ using $Shape_{180}$. The best results are obtained with χ^2 (Fig. 5a), and consequently this distance is used for the rest of this section.

Number of bins – K. We change the value of K in a range $[10 \ldots 40]$ for $Shape_{180}$ and $[20 \ldots 80]$ for $Shape_{360}$. Note that the range for $Shape_{360}$ is doubled, so as to preserve the original orientation resolution. Performance is optimal with $K = 20$ orientations bins for $Shape_{180}$, and $K = 40$ for $Shape_{360}$ as shown in Fig. 5b. However, as can be seen, the performance is not very sensitive to the number of bins used.

Level-Weight – α_l. Since the final PHOG is normalized there are only three independent parameters which represent three of the ratios in: α_0:α_1:α_2:α_3. For GLW we optimize the ratios varying them in a range $[0, 2]$. Best performance is achieved with $\alpha_0 : \alpha_3 = 0.2$, $\alpha_1 : \alpha_3 = 0.4$ and $\alpha_2 : \alpha_3 = 0.4$. For CLW we optimize the same ratios and vary their value independently for each category. For the most part more weight is given to $l = 3$, however for *accordion* and *inline-skate* (and some others) more importance is given to $l = 2$ or $l = 1$. This is because at the higher levels regions are smaller and may miss the object entirely (capturing background) as the object position varies in the image (there is more intra-class pose variation).

Kernel features – α **&** β. For the GFW we learn the kernel weights in (3) by varying the ratio $\alpha : \beta$ in the range $[0 \ldots 2]$. In this case best performance is obtained with α:β = 1.5. This means that appearance has more influence. For CFW optimization there exist categories for which appearance works better (e.g. *leopards*, *barrel*) and others for which shape is best (e.g. *scissors*, *lamp*).

6.2 Classification Results – Caltech 101

Shape alone. The first four rows in Table 1 summarize the performances achieved using just the HOGs at one level of the pyramid (single level) as well as the performances when using the PHOG for GLW optimization. As can be seen in this table (and in Fig. 5a), very poor performance is obtained for the lowest level ($L = 0$). In this case we are representing the images with just one orientation histogram and this is not very discriminative between classes (see Fig. 6e). However performance increases by around 40% when introducing spatial information, so the PHOG becomes far more discriminative (see Fig. 6f). This means that objects can be more easily differentiated by their spatially local orientations than by global measures. Though matching at the highest pyramid level seems to account for most of the improvement, using all the levels together confers a statistically significant benefit.

Using $Shape_{360}$ we obtain the best result (69.0%). Including contrast sign information helps substantially for recognition tasks in Caltech-101, because this sign information is useful for the majority of objects in the dataset, like *motorbikes* or *cars*. This is in contrast to [7] where better results were obtained if the contrast sign of the gradient was ignored. This is because [7] is detecting pedestrians, and in

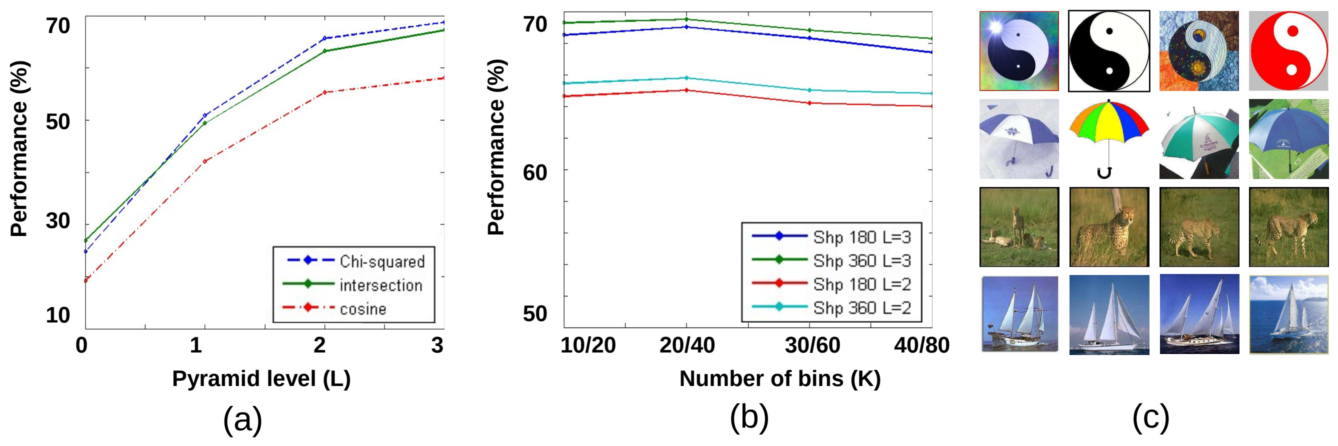

Figure 5: (a) Caltech-101 validation set performance for different distance measures over pyramid Levels $L = 0$ to $L = 3$; and (b) over the number of bins (K). (c) Intra-colour variability for a few different categories.

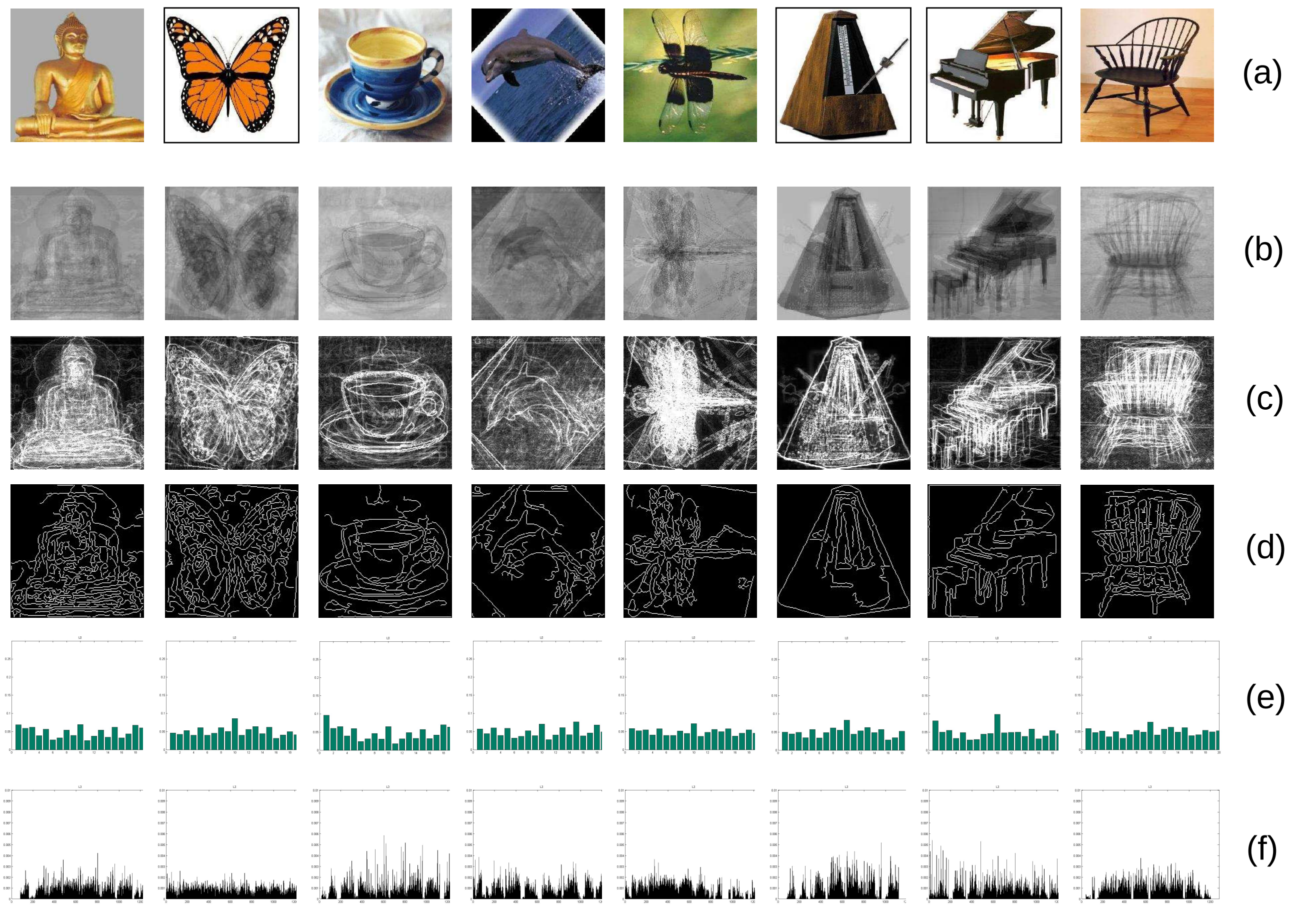

Figure 6: Caltech-101 training set: (a) representative image for the class; (b) average over images; (c) average over gradient images; (d) average over edge images; (e-f) orientation histograms at $l = 0$ and at $l = 3$. Note: (i) the gradient and edge images illustrate that a spatial orientation histogram with some tolerance to translation will capture these classes well; (ii) the classes have quite similar global (level $l = 0$) orientation histograms, but differ in their finer (level $l = 3$) spatial layout.

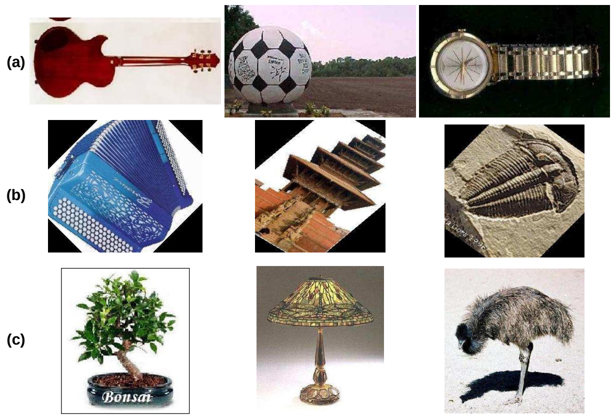

Figure 7: Examples of categories that are confused using shape alone: (a) images confused with a guitar; (b) images confused with an accordion; (c) images confused with a bonsai.

this case the wide range of clothing and background colours presumably makes the contrast sign uninformative. If we use the CLW (class specific level-weight optimization) the score increases as far as 69.8% for $Shape_{180}$, and to 70.6% for $Shape_{360}$.

Fig. 7 samples some examples of class confusions, and it is evident that the confusion is understandable, and arises here (and in others we have looked at, but are not shown) from shape similarity. The best classified classes are *octopus*, *metronome* and *inline skate* with a score of 100% and the worst are *brontosaurus* and *cannon*, with 25.0% and 25.7% respectively.

Appearance alone. The last two rows of Table 1 summarize the appearance performance for GLW optimization. For Caltech-101, App_{Gray} works better than App_{Colour} (68.1% vs 66.5%). If we use CLW then the score increases to 71.6% for App_{Gray} and to 68.2% for App_{Colour}. However, in some individual categories (e.g. ketch, leopards) colour information is very relevant and class confusion is reduced if it is used, whilst for others (e.g. yin-yang, umbrella) it is not. Fig. 5c shows examples of each case.

Shape & Appearance. We first use the kernel in (3) with GLW and GFW. When merging App_{Gray} and $Shape_{180}$ the performance is 70.7%, and this increases to 71.5% when merging App_{Gray} and $Shape_{360}$. For GLW and CFW performances increase to 72.8% for App_{Gray} and $Shape_{180}$, and to 73.5% for App_{Gray} and $Shape_{360}$. The best results are obtained using both class-specific optimizations (CLW & CFW): 76.2% for App_{Gray} and $Shape_{180}$, and 76.6% for App_{Gray} and $Shape_{360}$. That merging with $Shape_{360}$ is better than with $Shape_{180}$ is expected, since $Shape_{360}$ alone performs better. Using the kernel in (4) and both class-specific optimizations, performances are slightly better at 76.6% and 76.7%.

We have at our disposal two kind of appearances cues (App_{Gray} and App_{Colour}) and two kinds of shape representation ($Shape_{180}$ and $Shape_{360}$). If we merge all the kernels representing these cues using CLW and CFW, then we obtain the best performance overall: 77.8%. Table 2 gives a summary of the performances using different feature combinations. For just one feature CLW is used. Both class-specific optimizations (CLW & CFW) are used for merging

	Single level			
	l=0	l=1	l=2	l=3
S_{180}	23.2 ± 0.5	47.3 ± 0.7	61.7 ± 0.5	64.3 ± 0.8
S_{360}	25.0 ± 0.4	49.4 ± 0.7	62.1 ± 0.7	66.9 ± 0.9
	Pyramid – PHOG			
	L=0	L=1	L=2	L=3
S_{180}	23.2 ± 0.4	49.3 ± 0.6	64.1 ± 0.6	67.2 ± 0.5
S_{360}	25.0 ± 0.5	51.4 ± 0.8	64.2 ± 0.7	$\mathbf{69.0} \pm 0.6$
A_G	55.3 ± 0.4	64.6 ± 0.3	67.0 ± 0.5	$\mathbf{68.1} \pm 0.6$
A_C	52.2 ± 0.5	63.1 ± 0.7	65.3 ± 0.9	66.5 ± 0.8

Table 1: Caltech-101 performance using appearance and shape separately with a χ^2 kernel. Single level means that only a HOG from level l is used. For PHOG, GLW is used to find the α_l. Note that S_{360} has slightly better performance than A_G.

$Shape_{180}$	$Shape_{360}$	App_{Gray}	App_{Colour}	Perform
✓				69.8 ± 0.5
	✓			70.6 ± 0.6
		✓		71.6 ± 0.6
			✓	68.2 ± 0.8
		✓	✓	75.3 ± 0.6
✓		✓		76.2 ± 0.6
	✓	✓		76.6 ± 0.8
✓	✓	✓	✓	77.8 ± 0.8

Table 2: Comparison of performance scores for Caltech-101 when using different cues: top part of the table is for single features; bottom part is when using different feature combinations; ✓ means the cue is used. Feature selection is carried out using the optimization process CLW and CFW. The kernel defined in (3) is used for merging cues.

cues. If we use the kernel defined in (4) to merge all the appearance and shape features, then a slightly worse result of 77.5% is obtained. The conclusion is that the kernel in (3) works better than the kernel in (4) when a large number of kernel features are used.

Table 3 compares our results to those reported by other authors. Our best performance is 77.8%. This outperforms the results reported by Zhang *et al.* [26] that to our knowledge are the best until now.

6.3 Retrieval Results – TRECVID

The appearance representation is computed in a similar manner to that described in Section 5.2. A vocabulary of 1500 words is used for App_{Colour}. The number of bins for PHOG is set to $K = 40$ using $Shape_{360}$ (we have not optimized K here, as it is demonstrated in Section 6.1 that it does not have much affect on the final results). Level weights (α_l) and kernel weights (α and β) are learnt on the validation set. We used χ^2 distance together with a spatial pyramid kernel with $L = 1$, which means that the inputs for the discriminative classifiers have a dimension of $1500 \times 5 + 20 \times 5$. In TRECVID spatial distribution is very variable between images of the same category. Consequently we only used up to $L = 1$ for this data set.

We learn the appearance and shape classifiers for all 39 required topics. In general, most of the categories are best retrieved when using App_{Color} features alone (e.g *face* – AvPr = 99.4%, *Weather* – AvPr = 95.3%, *Sky* – AvPr =

[18]	[11]	[25]	[14]	[26]	Ours
56.0	58.23	63.0	64.6	66.0	77.8 ± 0.8

Table 3: Classification of Caltech-101 with 30 training images per class.

91.6%). However, there are some categories like *truck* or *screen* where $Shape_{360}$ works better. Concretely there is a significant increase in performance over App_{Colour} for *building* (from 7.8% for App_{Colour} to 44.5% for $Shape_{360}$) and *road* (from 5.4% to 18.6%). Overall, the best results are obtained when using the appearance and shape merging kernel (3) with both class-specific optimizations (CLW & CFW). In this case there is a significant performance improvement when retrieving: *crowd* – 84.2%, *entertainment* – 74.9%, *people walking* – 65.4%, *sports* – 61.1%, *mountain* – 59.3% and *military* – 49.9%.

7. SUMMARY & CONCLUSIONS

We have introduced a new descriptor, PHOG, which flexibly represents the spatial layout of local image shape. For the case of the Caltech-101 dataset, we can attribute how much each representational and feature increment contributes to the overall performance. Compared to using a single level ($l = 3$) shape representation, PHOG increases performance by 2% using weightings common across all classes. This increase rises to 4% using class-specific level weighting. The combination of the PHOG and appearance descriptors achieves an 11% improvement (compared to the single level shape representation) using the class-specific feature kernel. This demonstrates that the shape and appearance descriptors are complementary. We conclude that complementary spatial pyramid based descriptors, together with class-specific optimization of pyramid weights and class-specific kernel selection for merging are all important for good performance.

Acknowledgements

This work was partially funded by the research grant BR03/01 from the University of Girona. We are grateful for support from the Royal Academy of Engineering, EU Project CLASS, the EU Network of Excellence PASCAL and the MEC Project $TIN2006-08035$.

8. REFERENCES

[1] F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proc. ICML*, 2004.

[2] A Bissacco, M. H. Yang, and S. Soatto. Detecting humans via their pose. In *NIPS*, 2006.

[3] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE PAMI*, 10(6):849–865, 1988.

[4] A. Bosch, A. Zisserman, and X. Muoz. Scene classification via plsa. In *Proc. ECCV*, 2006.

[5] C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at urlhttp://www.csie.ntu.edu.tw/cjlin/libsvm.

[6] G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[7] N. Dalal and B Triggs. Histogram of oriented gradients for human detection. In *Proc. CVPR*, 2005.

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *IEEE CVPR Workshop of Generative Model Based Vision*, 2004.

[9] P. F. Felzenszwalb. Learning models for object recognition. In *Proc. CVPR*, 2001.

[10] D. Gavrila and V. Philomin. Real-time object detection for “smart” vehicles. In *Proc. ICCV*, 1999.

[11] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. ICCV*, 2005.

[12] D. Haussler. Convolution kernels on discrete structures. Technical Report UCS-CRL-99-10, 1999.

[13] A. K. Jain and K. Karu. Learning texture discrimination masks. *IEEE PAMI*, 18:195–205, Feb 1996.

[14] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, 2006.

[15] D. Lewis, T. Jebara, and W. Noble. Nonstationary kernel combination. In *Proc. ICML*, 2006.

[16] T. Lindeberg and J. Gårding. Shape-adapted smoothing in estimation of 3-D depth cues from affine distortions of local 2-D brightness structure. In *Proc. ECCV*, 1994.

[17] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[18] J. Mutch and D. Lowe. Multiclass object recognition using sparse, localized features. In *Proc. CVPR*, 2006.

[19] A. Opelt, A. Pinz, and A. Zisserman. Incremental learning of object detectors using a visual shape alphabet. In *Proc. CVPR*, 2006.

[20] E. Seemann, B. Leibe, K. Mikolajczyk, and B. Schiele. An evaluation of local shape-based features for pedestrian detection. In *Proc. BMVC.*, 2005.

[21] J. Shotton, A. Blake, and R. Cipolloa. Contour-based learning for object detection. In *Proc. ICCV*, 2005.

[22] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Filtering using a tree-based estimator. In *ICCV*, 2003.

[23] M. Szummer and R. W. Picard. Indoor-outdoor image classification. In *ICCV Workshop on Content-based Access of Image and Video Databases*, 1998.

[24] J. Thureson and S. Carlsson. Appearance based qualitative image description for object class recognition. In *Proc. ECCV*, 2004.

[25] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *Proc. CVPR*, 2006.

[26] H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proc. CVPR*, 2006.

[27] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *IJCV*, 2007.