Image representations beyond histograms of gradients: The role of Gestalt descriptors

Stanley Bileschi
MIT
Cambridge, MA, USA
bileschi@mit.edu

Lior Wolf
Tel Aviv Univ.
Ramat Aviv, Tel Aviv 69978
liorwolf@cs.tau.ac.il

Abstract

Histograms of orientations and the statistics derived from them have proven to be effective image representations for various recognition tasks. In this work we attempt to improve the accuracy of object detection systems by including new features that explicitly capture mid-level gestalt concepts. Four new image features are proposed, inspired by the gestalt principles of ***continuity****,* ***symmetry****,* ***closure*** *and* ***repetition****.*

The resulting image representations are used jointly with existing state-of-the-art features and together enable better detectors for challenging real-world data sets. As baseline features, we use Riesenhuber and Poggio's C1 features [15] and Dalan and Triggs' Histogram of Oriented Gradients feature [5]. Given that both of these baseline features have already shown state of the art performance in multiple object detection benchmarks, that our new mid-level representations can further improve detection results warrants special consideration. We evaluate the performance of these detection systems on the publicly available StreetScenes [27] and Caltech101 [11] databases among others.

1. Introduction

Computational approaches to perceptional tasks, visual and otherwise, can be divided into two components: representation of the input signal, and learning, e.g., optimization, modeling, or estimation. Generally, the systems with the best performance use standard learning techniques, and the key to their effectiveness is either a very large training set, or an effective signal representation.

For visual problems, a wealth of image representations have been proposed in the literature, but recently, with the emphasis on performance, there has been a rapid convergence into two prominent approaches: patch similarities, and edge orientation statistics. It is clear that these representations do not explicitly capture more intricate, high level concepts, which may be critical to the subsequent learning stages.

Our goal is to implement a few mid-level image representations based on Gestalt principles. If these new features discriminate between examples of object and non-object in new ways, then we can expect performance gains. This performance constraint is critical; the literature has many examples of symmetry detectors, saliency detectors and other mid-level features, *e.g.*, [14]. However, few have been shown *to add* to the discriminative power of a state-of-the-art detection system. The task of creating Gestalt-based features that improve upon cutting-edge descriptors remains mostly unexplored. We make no claims that our implementations are the best or only way to capture the target Gestalt concepts, only that they provide information which the classifier can then leverage.

Computationally, the implementation of these new features resembles the generalized Hough transform, but there are many significant differences, such as replacing summations with morphological operations. The importance of these non-linearities will be addressed in the experiments. Also, in the classical generalized Hough transform, thresholding is used to detect salient image structures. Instead, we pass a version of the voting space directly to the classifier. This design methodology will be elaborated as the paper progresses.

In each of Sections 3 through 5, we describe a new feature, describe its properties, and then evaluate the performance of the feature. Performance improvements on the Caltech 101-objects database in described in Sec. 6.

2. Current state of the art image features

Statistical learning methods require that images must first be converted into a mathematical object, some vector in $\Re^n$. A wide variety of image representations have been used in the past, including grayscale pixel values, wavelet coefficients, linear projections (i.e., PCA, LDA and variants), and many others. Each has strengths and weaknesses for certain

object types and modes of invariance. Two categories of image features currently dominate the object detection literature: *histograms of edge orientations*, and *patch based* features.

Histogram of edge orientations. This increasingly popular type of representation has demonstrated good discriminative power for many types of objects and tolerance for several common image transformations. It can be described as a weighted histogram wherein each histogram bin collects votes from gradients near particular locations and particular orientations. Examples include Lowe's SIFT [12], Riesenhuber and Poggio's C1 [15], Malik's geometric blur and shape context [2, 1], and Dalal and Triggs' HoG [5].

The SIFT feature, described in detail in [12], was designed to be used in concert with an interest operator in order to find correspondences in images of the *same* object under different lighting and view angles. It therefore possesses several properties that make it suboptimal for *generic* object recognition. It was noted in [12] that the SIFT feature is a general framework, and its parameters can be modified to suit individual tasks. This was done in [5], where the *Histogram of Gradients feature* (HoG) was introduced and shown to be much more suitable for generic object recognition.

In the HoG, the best results for a pedestrian detection task were obtained with one bin per 8 pixels in either spatial direction, and four or more orientations uniformly distributed from $0^\circ - 180^\circ$ (using rectified orientations). The magnitudes of the brightness gradient at each pixel are added to the appropriate histogram bins using linear interpolation between the nearest eight bin centers in location and orientation. Finally, a normalization step is applied. The performance of the HoG feature over a wide variety of parameters, histogram structures, and normalization techniques, is explored in [5]. In our experiments we use the version with the optimal set of parameters.

The C1 feature [15, 18] stems from a model of biological vision. In C1, the histogram bins are replaced by the concept of cells. Each cell computes a function of the oriented filter responses within its predefined receptive field, but, rather than each cell accumulating the associated responses into a sum, as in HoG or SIFT, only the maximum response is recorded.

The image features described above share more in common than they differ. At the most basic level, each element of each feature vector simply encodes a statistic of a particular oriented filter response within a predefined receptive field. This makes these features good for describing the gross pattern of orientations within the image. However, they do not make explicit potentially useful higher level information, such as long range correlations of oriented responses.

Patch based features Along with local histograms of orientations, patch based features, too, have gained popularity in the computer vision community. A patch based feature vector is an image description which depends on comparing the image with set of stored image crops, also known as templates or fragments. Common examples of patch based image descriptors include those of Ullman and Sali [22], Freeman [21], and Liebe and Scheile [10]. Different implementations select different balances between invariance and the representation of geometric structure. It has been shown that patch based approaches can perform significantly better than detection via a single template, but they are still limited in representative power by the underlying measurement used to compare the image with the stored prototype.

3. Continuity based image descriptors

The detection of continuous edges has long been a focus of the vision community, having been a topic of discussion in studies of saliency, segmentation, and boundary detection, e.g., [19, 9], etc. Even the venerable Canny edge detection system [4] presupposes a notion of continuity to justify the use of two different thresholds and a hysteresis based detection scheme. While many modern object detection systems use edge gradients as feature vector elements, only a few, such as [14, 17, 6] have explicit encoding of the presence or absence of long continuous edges. The development our continuity feature was motivated by the need for a representation of long contour representation suitable for learning. We will show that performance can be improved by including this feature *along with* the state-of-the-art features mentioned in Sec. 2.

The traditional approach to detecting continuous lines in an image is to use the Hough transform. Gradient magnitudes are computed and thresholded at each pixel location, creating a set of edge-candidates. Each candidate is then mapped into the space of line parameters $(d, slope)$, where d is the distance from the line to some predetermined point in the image, often the center.

It would certainly be possible to then take the descretized parameter voting space, and use it directly as an image feature. This is unsatisfactory for several reasons, including the inability to represent lines of different lengths and that a horizontal line at the top of the image would have the exact same representation as a horizontal line at the bottom. The feature we propose below aims to address these issues.

Given an input image I, apply the following procedure:

For each $\theta \in \{0^\circ, 45^\circ, 90^\circ, 135^\circ\}$, repeat steps 1-4 below:	
1. Image rotation: Rotate the image by an angle of θ.	$\circlearrowleft_\theta$
2. Initial filter: Take the absolute derivative of I in the x direction.	$\lvert I_x(x,y)\rvert$
3. Perform the following sequence of morphological operators on I_x:	
a. Local minimization with a kernel of length 7 in the x direction.	$\ominus(7,1)$
b. Local maximization with a kernel of size $(x,y) = 7 \times 3$.	$\oplus(7,3)$
c. Perform a 2×2 subsumpling.	$\downarrow(2,2)$
4. Repeat step 3 for $i = 1 \ldots 6$ using $3c$ as input, record result of all repeats.	$R_{\theta,i}(x,y)$

Figure 1. Pseudocode for the Continuity descriptor. In the real implementation the filters and not the image are rotated.

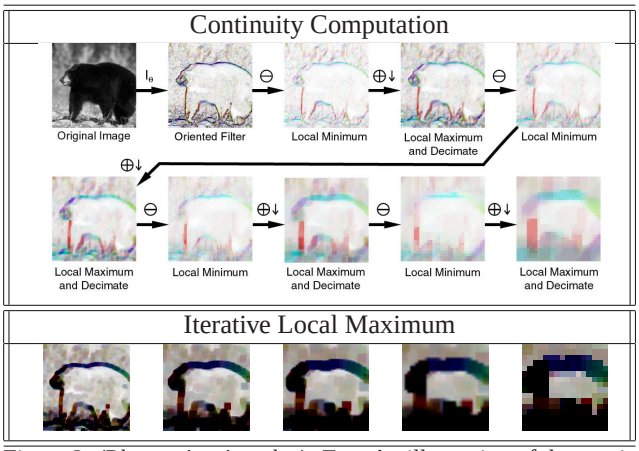

Figure 2. *(Please view in color)*: *Top:* An illustration of the continuity computation. After the original image, different colors indicate different orientations of contours, red is vertical, etc. Deeper color saturation indicates stronger contour evidence. Note that as the computation proceeds, the texture is filtered away and only the correct tint of the long contour remains. *Bottom:* If instead we iteratively take local maximums in the orientation space, the result, as shown here, is that all strong signals are spread out, no matter their local support. Since filters at many orientations are stimulated by strong edges, all orientations are strongly stimulated near the border of the bear.

3.1. The min-max continuity descriptor

In Fig. 1, we briefly describe the algorithm used to calculate the image continuity features. Matlab code is provided in the CVPR digital addendum. The computation consists of first filtering the image with oriented filters, rectifying the output, and then processing with alternating local dilation and erosion steps with orientation-specific kernels. Finally, the image is decimated to half resolution. This constitutes a round of processing, and each round defines a discrete step in a scale space of continuous contours. Representations of longer contours are found as co-linear concatenations of shorter contours. The output of this processing is a feature vector in which each element is sensitive to contours of a certain length, orientation, and spatial location, while being tolerant to some variability in each of these dimensions.

Intuitively, the erosion step only leaves a strong response if **all** edge elements along a contour are strong, and then the maximum filter propagates that signal to the neighborhood. It is important to note that the kernel of the dilation operator is wider than that of the erosion operator in order detect straight edges which are not precisely at the represented orientations, and edges which slowly curve.

For illustration, Fig. 2 depicts our continuity operator applied to an image of a bear: different colors indicate different contour orientations and intensity indicates contour strength. Note that long continuous contours become enhanced and remain represented through gross image downsampling. Also note that dilation alone will not preserve contour orientation, as noisy edge elements cause strong contour detections in all orientations.

3.2. Experimental validation

The utility of the continuity feature in object detection tasks is demonstrated in three separate object detection experiments. In each, an existing system is improved by adding the continuity features to the existing feature pool. The first experiment demonstrates improved results in detecting cars, pedestrians, and bicycles in the StreetScenes [27] database. The second experiment involves discriminating images which contain an animal from those which do not, where the animal may be at any scale or position within the image, without windowing. Finally, improved results are demonstrated on a three-class database consisting of low-resolution images of cars, mid-size vehicles, and trucks.

Object detection with no clutter For this experiment, we use the StreetScenes data set [27], discriminating cars, pedestrians and bicycles from background. The objects in the positive data are standardized in terms of position and scale within the images. The background class consists of image crops known not to contain any of these object types. We should note that this is a difficult task owing to the large amount of internal class variability. For instance, the car data set includes vehicles of many different types, from small coupes to large busses, and at many different poses. Example images are shown in Fig. 3. Note that the performance measures are made on predefined crops from these larger images, and multiple train-test splits are used. The data set contains includes 5,001 cars, 1,449 pedestrians, and 209 bicycles. Random splits included $\frac{2}{3}$ of examples for training and $\frac{1}{3}$ for testing.

In [25] it was shown that, for these three data sets, the C1 features exhibit outperform several other state-of-the-art feature sets, notably, the two systems described by Torralba and Leibe in [21] and [10]. We therefore compare only to the C1 features[1] and to the similarly successful HoG feature set. The relative performance of these two feature sets depends on which object is used in the test.

For both features, significant performance improvement is seen when the continuity features are included in addition. In table 1 the performance of these classifiers both with and without the additional continuity features is listed. These statistics were collected by randomly splitting the data into training and testing 100 times, training gentleBoost classifiers [7] until convergence and then recording statistics of the resulting ROC curve on the test data. We report the equal-error rate and the true-positive-rate when

[1] Code is available at `http://cbcl.mit.edu/software-datasets`.

Figure 3. Examples from some of the databases used to test the new features. *Top Left*: An image humans classify easily as non-containing-animal, but is difficult for the system of [18]. *Bottom Left*: 12 Examples of car vs. SUV vs. truck images from the car database mentioned in Sec. 3.2. *Middle and Right*: Examples of the StreetScenes database images and corresponding labels [25].

the false-positive-rate $= 1\%$, since the low-false-positive region is more interesting for detection applications. It can be seen that continuity features have significantly improved the power of the classifier.[2]

Finally, to illustrate the necessity of the non-linear morphological operations in the continuity operator, we include the results of a more traditional linear version of the operator in the same table. This operator uses conventional oriented filters in the place of the morphological operations (sums replace max and min). It can be seen that this linear version helps the classifiers, but not nearly as much as the algorithm described in Fig. 1.

Animal detection in clutter In this experiment, the goal is to discriminate between natural images which either do or do not contain an animal. A database of $1,200$ grayscale images was provided for training and testing, as well as a previous results for both humans (using a rapid exposure setting), and an SVM based classifier operating on the neuromorphic features of [18]. The types of mistakes that the machine would make that the human would not were typically false positives on urban scenes, such as that in the top left of Fig. 3.

It should be noted that this is a difficult task, and the provided system was already outperforming several state-of-the-art benchmarks. A significant performance increase in the area under ROC curve statistic, from $71.4 \pm 2.6\%$ to $74.2 \pm 1.8\%$ was noted when the SVM was trained on a combined feature set consisting of the original features concatenated with the continuity features. The conclusion to be drawn from these results is that the new features provide additional information that was not immediately available from the baseline features.

Car type identification The car type data set consists of 480 images of private cars, 248 of mid-sized vehicles (such as SUV's), and 195 images of trucks. These are small 20×20 pixel images, collected using an automatic car detector on a video stream taken from the front window of a moving car. The task is to classify the three types of cars for a safety application. Taking into account the low resolution and the variability in the three classes, this is a difficult task (see Fig. 3).

The protocol used in the experiments is as follows. In each one of 20 repeats, 100 training images and 95 testing images were randomly drawn from each class. A multi-class SVM was trained, and the average success rate along with the standard deviation was reported. The gray-level features score $59.86\% \pm 3.35\%$ on this three class problem, the $C1$ features score $41.89\% \pm 3.62\%$ and the continuity features score $67.51\% \pm 2.92\%$. This success supports the postulation that classification is best performed on this dataset by using long horizontal and vertical edges. We also tried combining feature sets: gray-level combined with C1 scores $59.65\% \pm 4.24\%$, while gray-level and continuity together scores the highest score among our experiments; $72.35\% \pm 2.47\%$ correct.

4. Circularity and Form based descriptors

There is a well known result in studies of statistics of natural images that, given the location of one oriented edge, the most likely locations for another oriented edge are those in which the two edges would have the property of *cocircularity* [8]. Similarly, in studies of saliency, it is known that humans tend to group together edge elements which form common forms such as circles or quadrilaterals [23]. In order to capture the notion of form in an explicit image feature, we have developed the algorithm detailed in Sec. 4.1. We will show that this mid-level image feature captures information about the image which is not immediately available in other state-of-the-art image representations. It should be noted here that this feature, along with the others, could be computed in many different ways. The goal was to build an explicit representation of the existence closed forms within the image, and not to segment out the salient form or to find an optimum such representation.

4.1. The circle and form descriptor

Given an input image I, apply the following procedure:

1. *Initial filter:* For each $\theta \in \{0^\circ, 45^\circ, 90^\circ, 135^\circ\}$, $D(x, y, \theta) = |\nabla_\theta(I)|$
2. *Non-maximal suppression:* if $D(x, y, \theta_1) < D(x, y, \theta_2)$: set $D(x, y, \theta_1)$ to 0
3. *Initialize 4-D voting space to zero:* $H(x, y, \theta, s) = 0$
4. *For each radius* $r \in \{2^1, 2^2, \ldots, 2^5\}$ *and orientation* θ*:*
   a. *Define an elongated gaussian shaped voting kernel:* $G_{(\theta,r)}$ of size $2r \times 2r$ with a diagonal covariance matrix with values $3r$ and r.
   b. *Rotate:* $G_{(\theta,r)}$: by θ
   c. *Define the translation:* $t_x = r \times \sin(\theta), t_y = r \times \cos(\theta)$
   d. *Collect votes by convolving the voting kernel and translated filter response:*
   $H(\mathbf{x}, \mathbf{y}, \theta, r) = H(\mathbf{x}, \mathbf{y}, \theta, r) + (\mathbf{G}_{(\theta,r)} \circledast D(\mathbf{x} - t_x, \mathbf{y} - t_y, \theta))$
   $H(\mathbf{x}, \mathbf{y}, \theta, r) = H(\mathbf{x}, \mathbf{y}, \theta, r) + (\mathbf{G}_{(\theta,r)} \circledast D(\mathbf{x} + t_x, \mathbf{y} + t_y, \theta))$
4. *Maximal suppression:*
   if $\forall \theta_2, H(x, y, \theta_1, r) \geq H(x, y, \theta_2, r)$ then set $H(x, y, \theta_1, r)$ to 0
5. *Sum the voting matrix H over orientations*
6. *Reduce image resolution:* using bilinear interpolation by a factor of 8

Figure 4. Algorithm to compute the circularity feature vector of an image.

[2] Results are similar with linear SVMs.

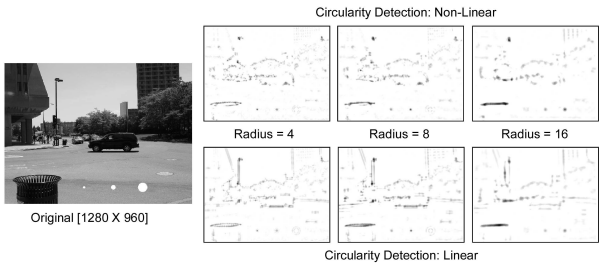

Figure 5. *Left:* An image of a common street scene with a car and some buildings. White circles of radius 4, 8, and 16 have been added artificially for illustrative purposes. *Right-Top:* The circle feature output at several different radii. *Right-Bottom:* step 4 has been removed from the algorithm to make the computation similar to the standard Hough transform. **Note** especially that the linear version gives strong circle detections over the straight edges in the telephone post, building, and car shadow. The non-linear version does not. Note also that the detection of the car tires is obscured by the car shadow edge in the linear version.

In Figs. 4 and 5 we describe in pseudo-code and diagrams the algorithm used to calculate the circle feature. Each element in the vector output by this algorithm collects the evidence that there is a convex form (circle, rectangle or otherwise) of an approximate scale centered at an approximate (x, y) location in the image. The algorithm presented here is in some ways similar to the generalized Hough transform used to detect circles, though there are also distinct differences. In the usual framework, boundary elements "vote" in a non-parametric way into the circle-parameter space. Similarly, in our system, each edge element spreads its vote, modulated by the edge magnitude, into data structure indexed by (r, x_c, y_c), the postulated circle's radius and center. However, since our edge elements contain orientation information, they vote for circle-centers only if they would be tangential that circle. A wide, elongated voting kernel is used in order to represent shapes which are not precisely circular and to prepare for the coarse descretization necessary to restrict the feature vector length to a manageable size.

In the usual Hough framework all edge elements vote equally into the parameter space. The natural extension would be to have all oriented edges contribute equally into the parameter space, according to their magnitude. Instead, for each point in the parameter space we remove all votes from the strongest contributing orientation. This is done to reduce the noise from long straight contours in the feature which is meant to only represent closed forms. The effect is that only forms with support from more than one orientation are retained.

4.2. Experimental validation

Again we demonstrate performance on the data set provided to us by the authors of [25] by appending the baseline features with the circularity features. As can be seen in table 1, for each object category, the inclusion of the circle features significantly improves the detection score. An analysis of statistical significance showed that the circularity feature significantly improves detection for all three StreetScenes object classes at p=.01.

In order to demonstrate the importance of the non-linear *sum-minus-max* step, we also show the results of a similar algorithm which does not remove the support from the dominant orientation. For all three classes tested, the non-linear circle representation is significantly more powerful. This likely due to the superfluous form detections caused by the prevalence of straight edges in the images, a feature which is already captured in the baseline representation.

5. Repetition and Symmetry based image descriptors

It has been shown that humans are adept at detecting symmetry in natural images. As symmetry is a relatively rare phenomenon in images, it is likely that it is a useful cue in the detection of objects that exhibit symmetric characteristics. A measure of symmetry is defined at a position p, scale s, and orientation θ in an image. Strong symmetry at (p, s, θ) means that if we imagine a dividing line with orientation θ passing through p, the image on one side is in some way similar to a reflection of the image on the other side, at scale s. Previous object detection studies studying notions of symmetry include [16, 26, 13].

The notion of repetition is similar to symmetry but does not require a reflection. Fig. 7 illustrates the concepts of repetition and symmetry as addressed in this text. Below we will describe an image features to represent image repetition and symmetry, and measure their utility in object detection.

5.1. The Symmetry Feature

The symmetry descriptor takes an input grayscale image and outputs an explicit measure of mirror symmetry for local regions within the image. Specifically, this operator measures symmetry by looking for axes of *vertical mirror* symmetry. While we leave out many other types of symmetry, such as mirror symmetries at other axis, axial symmetry, etc., there is a great deal of evidence that humans are far more sensitive to this type of symmetry than others [24]. Briefly, our symmetry operator produces a feature vector in which each element estimates a symmetry score for a certain image position and scale. So that the number of samples in the feature vector is small, each sample must be tolerant to an appropriate range of positions and scales.

The algorithm, in pseudo-code, is listed in Fig. 8. Furthermore, in Fig. 6, we show the results of the symmetry detection algorithm on some test images. It is easy to see that the symmetric structures are well represented by the feature.

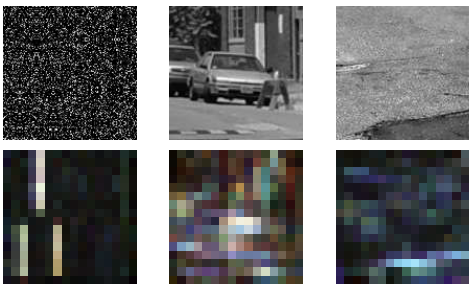

Figure 6. An illustration of the symmetry detection algorithm on three test images. In the symmetry response images, (*bottom*), brightness indicates strength and the color indicates the scale of the detected symmetry, with blue representing smaller regions than red (white regions have symmetry detected at all scales). The leftmost toy image contains three regions of perfect vertical symmetry and three regions of perfect horizontal symmetry. The vertical symmetries are strongly detected in the response images. The symmetry of the car image is detected as a region of strong symmetry at the center of the image. As can be seen in the third column, random textures have little symmetry, and continuous lines have symmetry at a small scale, but not larger.

The algorithm detects symmetry by accumulating evidence from pairs of matching patches. Each patch is compared to mirrored images of patches located across a posited line of symmetry. By taking the maximum of the match strength over a small pool of locations we build tolerance to variations in the depth of the symmetric object and small rotations of the symmetry axis. Afterwards, all match strengths which would contribute to the same line of symmetry at the same location are summed. Thus, two mirror patches near to the line of symmetry and a pair far from the line of symmetry both contribute to the symmetric stimulus. Finally, after accumulating evidences of symmetry at all points within the image and at a discreet set of scales, the data is reduced in such a way so as to retain the essential qualities of the symmetric information, but fit within a feature vector of reasonable size. This is done through bilinear resizing of the matrix after taking a local maximum, since we are not concerned with the exact pixel location of the line of symmetry so much as the fact that at least one strong symmetry signal exists within reasonably sized region.

The symmetry feature is very expensive to compute, requiring approximately 80 seconds per 128×128 image, using the implementation described here. This time can be cut down significantly by calculating the patch matching scores in grayscale, as opposed to an oriented filter space, although this affects performance. The major cost is in computing the matching scores for all the patches, so it is likely that the speed can be improved greatly by using a form of approximation, such as PCA, to rapidly compute the large matrix S.

Given an input image I, perform the following algorithm

1 Initial filtering: Compute absolute derivatives of I in four orientations	$D(x, y, \theta)$
*2 Loop over scales:*For a discrete set of scales $\sigma \in \{2^{\frac{0}{3}}, 2^{\frac{1}{3}}, \ldots, 2^{\frac{15}{3}}\}$	
a: Let $\overline{D}$ equal D resized by a factor of σ in x and y	$D \rightarrow \overline{D}$
b: *Calculate Local similarities:* $\forall\{x, y, d_y, d_x\}, d_y \in \{-7 \ldots 7\}, d_x \in \{0 \ldots 25\}$	
b_i: Let P_1 be the $7 \times 7 \times 4$ patch of $\overline{D}$ centered at $(x - dx, y)$	
b_{ii}: Let P_2 be the $7 \times 7 \times 4$ patch of $\overline{D}$ centered at $(x + dx, y + dy)$	
b_{iii}:Let $S(x, y, d_x, d_y)$ = NormCrossCorr(P_1,***mirror***(P_2))	$S(x, y, d_x, d_y)$
c: Half-wave rectify S	$\max(S, 0)$
d: Retain only maximum of S over d_y	$S(x, y, d_x)$
e: Take sum of S over all d_x	$S(x, y)$
f: Resize $S(x, y)$ to 128×128 and store as a layer of G	$G(x, y, \sigma)$
3 Reduce feature vector length:	
a: Local min of $G(x, y, \sigma)$ with a kernel of size 8 in y	$\ominus(1, 8, 1)$
b: Local max of $G(x, y, \sigma)$ with a kernel of size 8 in x	$\oplus(8, 1, 1)$
c: Perform a 8×8 subsampling in (x, y)	$\downarrow(8, 8, 1)$
d: Bilinearly resize $G(x, y, \sigma)$ to $16 \times 16 \times 4$	$\Downarrow$
4. $G(x, y, \sigma)$ is our 128 dimensional feature vector.	$G(x, y, \sigma)$

mirror:
Given a multi-layer image $I(x, y, \theta)$ where x ranges from 0 to x_{max}, and θ ranges from 0 to 180, the mirror image $\hat{I}$ is defined as $I(x_{max} - x, y, 180 - \theta)$.

Figure 8. Algorithm for computing image symmetry features. A feature vector of reasonable length is computed from an image of arbitrary size.

5.2. The Repetition Feature

Given an input image I, perform the following steps:

1. Initial filtering: Compute absolute derivatives of I in four orientations.	$D(x, y, \theta)$
2. Local similarities: Compute local similarities of every $5 \times 5 \times 4$ image patch to all neightberhood patches of a distance $4 \leq d_x, d_y \leq 25$.	$S(x, y, d_x, d_y)$
3. Perform the following sequence of morphological operators:	
a. Local maximization with a kernel of size 3×3 in (d_x, d_y)	$\ominus(1, 1, 3, 3)$
b. Local summation with a kernel of size 16×16 in (x, y).	$\sum(16, 16, 1, 1)$
c. Perform a 8×8 subsampling in (x, y).	$\downarrow(8, 8, 1, 1)$
d. Local maximization with a kernel of size 7×7 in (d_x, d_y).	$\oplus(1, 1, 7, 7)$
e. Perform a 7×7 subsampling in (d_x, d_y).	$\downarrow(1, 1, 7, 7)$
4. Obtain a 4D result.	$R(x, y, d_x, d_y)$

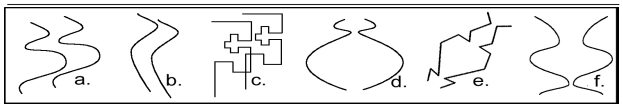

Figure 7. *Top*: Algorithm for computing image repetition features. *Bottom*: Examples of repeated (**a**–**c**) and symmetric (**d**–**f**) figures.

The algorithm to describe image repetition is detailed in the pseudocode in Fig. 7. This feature is designed to capture whether or not there exists a consensus of relative locus of similarity within spatial sub-regions of the image, i.e., if a cluster of pixels all agree that they are similar to a certain shifted sub-region of the image. First, the image is filtered and rectified into a set of 4 orientation images. In stage 2, each $(5 \times 5 \times 4)$ image patch is compared to all patches of a similar size within its neighborhood. L_2 distances of the edge-response patches is used as the similarity measure so that edges only match similar magnitude edges of the same orientation. In stage 3 the local maximum of this similarity measure within a (3×3) region of the relative transformation is taken in order to allow a small amount of distortion in the relative location computation. Next, for each relative location separately, a local sum is taken in the space of the image plane. If a group of nearby patches agree that the image is similar at this relative location, then the sum will produce a peak at the center of the group.

Finally the data structure is appropriately subsampled. This reduces the feature vector to a manageable size, and al-

lows tolerance to the exact location and nature of the repetition. The responses in the image plane are pooled by adding them locally, in order to compute a consensus of similarity, but the responses in the (d_x, d_y) direction are pooled by taking the maximum. The intuition behind this is that we are only interested in the translation that gives the best evidence of repetition, not in the average repetition over all translations.

5.3. Experimental validation

Referring once again to table 1, we see that the addition of the repetition and symmetry features enable the object detectors to achieve significantly better results on all three object databases. Note that if the dilation operations are linearized, then the entire repetition-detection algorithm can be collapsed into one sum on the original data structure, i.e., each feature would simply be the sum of a set of individual patch similarities.

6. Experiments on the 101 objects dataset

A final supporting result is given on the popular 101 objects dataset [11]. The results reported here are the average and standard deviation, taken over all 101 classes plus the background class, of the object recognition performance obtained from 20 independent trials. In each trial 15 random training and 50 random testing images were selected from each class, fewer testing images were used if not enough were in the database. In the final score, all errors are weighted such that each class has the same contribution regardless of the number of the testing images. Note that by using all of the testing images at once, one gets a much better performance, due to the over-representation of some easy classes. Therefore, to compare with previous results, we use the $15/50$ protocol with re-weighting.

Using this protocol, Berg et al.[3] report 45% correct detections on the **non-duplicated** version of the dataset. Serre et al. [18], report an average performance of 35% using $10,000$ "global" standard model C2 features, but using their publicly available code, combined with a per-feature variance normalization step we were able to improve this performance to $36.86\% \pm 1.64\%$ using only $3,000$ such features. (All of the results were obtained using a one vs. all linear SVM).

Using the same splits, the $C1$ features gave $30.93\% \pm 1.20\%$ by themselves, and $44.18\% \pm 0.76\%$ when added to the $3,000$ C2 features. Continuity gave, when combined with C2 $44.59\% \pm 0.93\%$, and 30.45 ± 0.34 by itself. Circularity gave, combined with C2 $41.48\% \pm 0.52\%$ (26.96 ± 0.88 alone). Repetition by itself received $17.10\% \pm 0.35\%$, and did not help C2 (37.47 ± 1.05). All the features taken together C2, C1, Continuity, Circularity and Repetition produced a score of $48.26\% \pm 0.91$, significantly higher than the performance without the gestalt features. The additional gain over a system which already performs well is valuable, by the law of diminishing returns.

7. Conclusion

The novel features presented in this work collect image information that is not necessarily concentrated in space (HoG, SIFT, C1) or in scale (C1), but instead along other modes of image structure. Evidence pooled along lines or circular arrangements is combined to support hypotheses of image structures that are unlikely to be coincidental. Patch-based similarities are grouped according to spatial patterns of similarity in order to build an effective representations of repetition and symmetry.

The idea that more meaningful image representations can produce significantly better recognition results is attractive, but it is not trivial to demonstrate. In this work, we revisit the principles that were used to build effective histograms-of-orientations-based features and use them to derive mid-level features. Histograms consisting of spatial bins are used, just like in SIFT and HoG, and non-linearities are employed in a way which is not unlike the maximization in C1. Changing the details is what enables us to describe mid-level concepts.

The four novel image statistics improve substantially the performance of state-of-the-art detectors. The performance gain is consistent across several challenging real world datasets, indicating that the framework is not simply noise.

References

[1] S. Belongie, J. Malik, J. Puzicha: Shape matching and object recognition using shape contexts. PAMI (2002) 2

[2] A. Berg, J. Malik: Geometric blur for template matching. In: Computer Vision and Pattern Recognition (CVPR). (2001) 2

[3] A.C. Berg, T.B., J. Malik: Shape matching and object recognition using low distortion correspondences. In: CVPR. (2005) 7

[4] J. Canny: A computational approach to edge detection. PAMI **8** (1986) 2

[5] N. Dalal, B. Triggs: Histograms of oriented gradients for human detection. In: CVPR. (2005) 1, 2

[6] V. Ferrari, L. Fevrier, F. Jurie, C. Schmid Groups of Adjacent Contour Segments for Object Detection INRIA Tech Report 2006 2

[7] J. Friedman, T. Hastie, R. Tibshirani: Additive logistic regression: a statistical view of boosting. Ann. Statistics (2000) 4

[8] W. S. Geisler, J. S. Perry, B.J.S., Gallogly, D.P.: Edge co-occurence in natural images predicts contour grouping performance. Vision Research **41** (2001) 4

[9] M. Kass, A.W., D. Terzopoulos,: Snakes: Active contour models. IJCV. (1987) 2

Dataset	Car		Pedestrian		Bicycle	
measure	EER	tp@fp = .01	EER	tp@fp = .01	EER	tp@fp = .01
C1	5.62±1.1	81.73±4.1	18.41±2.2	32.83±6.1	8.57 ± 2.2	59.79±10.0
C1 + Cont	4.58±1.0	86.88±3.9	8.07±1.2	71.02±4.5	7.28±2.1	78.33±7.2
C1 + LinCont	4.87±1.0	86.72±4.0	10.14±1.7	56.33±6.4	8.51±2.3	66.31 ±8.0
C1 + Circ	4.68± 0.9	86.89±3.9	8.27±1.7	67.98±4.8	6.79 ± 2.1	79.36±5.5
C1 + LinCirc	4.92±1.0	85.55±3.8	9.79±1.3	61.90±5.8	8.27±2.2	68.26±9.9
C1 + Par	5.07±0.9	84.24±3.6	10.50±1.3	58.23±7.9	8.87±2.5	65.39±9.7
C1 + LinPar	4.87±1.0	86.72±4.0	10.14±1.7	56.33±6.4	8.51±2.4	66.31±8.0
C1 + Symm	4.92±0.9	86.01±3.4	9.00±1.5	57.05±5.2	6.37±1.9	76.52±9.0
C1 + All Four	3.6±0.7	90.9±2.6	4.8±1.0	85.2±3.2	6.2±2.6	84.7±6.7
HoG	8.62±1.1	61.36±6.7	9.81±1.5	62.62±6.5	12.14 ± 2.7	52.90±9.3
HoG + Cont.	6.93± 0.9	69.88±5.7	7.67±1.0	72.81±6.1	7.84±2.9	73.29±7.7
HoG + LinCont	6.69± 0.7	68.96±5.8	7.71±1.2	67.90±6.6	9.87 ± 3.2	65.44 ±10.7
HoG + Circ	5.82±1.1	75.11±4.7	6.66±1.4	76.84±5.1	6.99±1.8	76.14±8.1
HoG + LinCirc	7.06± 0.9	69.09±5.2	8.18±1.3	70.09±5.6	9.19±2.4	68.02±8.4
HoG + Par	6.78±1.0	73.83±5.7	7.49±1.5	71.75±5.8	9.87±3.2	65.44±10.7
HoG + LinPar	6.63±0.7	69.06±6.6	7.74±1.2	68.18±6.4	10.27±3.1	61.31 ±9.8
HoG + Symm	5.88±0.8	75.83±5.3	7.20±1.1	71.31±7.2	8.27±2.9	65.88±9.1
HoG + All Four	4.9± 0.9	81.9±4.4	6.4±1.2	79.4±4.3	6.4±1.9	79.0±7.9

Table 1. Object detection results obtained by the proposed methods for the public StreetScenes Car, Pedestrian and Bicycle data sets [27]. Each row indicates a feature-set. Each column lists ROC statistics in percentage: either equal-error-rate, or true-positive-rate when the false-positive-rate is set to 1%. All statistics were generated by randomly splitting the data 100 times into training and testing sets.

[10] B. Leibe, L. Schiele: Interleaved object categorization and segmentation. In: BMVC. (2003) 2, 3

[11] F Li., R. Fergus, P. Perona: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: CVPR, Workshop on Generative-Model Based Vision. (2004) 1, 7

[12] D. G. Lowe: Distinctive image features from scale-invariant keypoints. IJCV (2003) 2

[13] G. Marola: Using symmetry for detecting and locating objects in a picture. Comput. Vis. Graph. Image Process. (1989) 5

[14] A. Opelt, A. Pinz, and A. Zisserman: A Boundary-Fragment-Model for Object Detection Proceedings of the In: ECCV. (2006) 1, 2

[15] M. Riesenhuber, T. Poggio: Hierarchical models of object recognition in cortex. Nature Neuroscience **2** (1999) 1019–1025 1, 2

[16] E. Saber, A. Tekalp: Face detection and facial feature extraction using color, shape and symmetry based cost functions. In: ICPR (1996) 5

[17] A. Selinger, R. Nelson: A Perceptual Gropuing Hierarchy for Appearance-Based 3D Object Recognition In: CVIU (1999) 2

[18] T. Serre, L.W., T. Poggio: Object recognition with features inspired by visual cortex. In: CVPR. (2005) 2, 4, 7

[19] A. Shashua, S. Ullman: Grouping contours by iterated pairing network. In: NIPS-3: (1990) 2

[20] P. Soille: Morphological Image Analysis. Springer (2003)

[21] A. Torralba, K. Murphy, W. T. Freeman.: Sharing features: efficient boosting procedures for multiclass object detection. In: CVPR. (2004) 2, 3

[22] S. Ullman, E. Sali: Object classification using a fragment-based representation. In: Biologically Motivated Computer Vision, (BMCV), (2000) 2

[23] M. Usher, Y. Bonneh, D.S., Herrmann, M.: Mechanisms for spatial integration in visual detection: a model based on lateral interactions. Spat Vision. (1999) 4

[24] J. Wagemans: Detection of Visual Symmetries In: Spatial Vision (1995) 6

[25] Wolf, L., Bileschi, S.: A Unified System For Object Detection, Texture Recognition, and Context Analysis Based on the Standard Model Feature Set. In: BMVC. (2005) 3, 4, 5

[26] T. Zielke, M.B., von Seelen, W.: Intensity and edge-based symmetry detection applied to car-following. In: ECCV. (1992) 5

[27] http://cbcl.mit.edu/software-datasets/streetscenes/ 1, 3, 8