Contour-Based Recognition *

Yong Xu[1], Yuhui Quan[1], Zhuming Zhang[1], Hui Ji[2],
Cornelia Fermüller[3], Morimichi Nishigaki[3] and Daniel Dementhon[4]
[1]School of Computer Science & Engineering, South China Univ. of Tech., Guangzhou 510006, China
[2]Department of Mathematics, National University of Singapore, Singapore 117542
[3]Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, U.S.A.
[4]Applied Physics Lab, Johns Hopkins University, Baltimore, MD 20723-6099, U.S.A.
{yxu@scut.edu.cn, yuhui.quan@mail.scut.edu.cn, z.zhuming@mail.scut.edu.cn, matjh@nus.edu.sg,
fer@umiacs.umd.edu, michi@cs.umd.edu, daniel@cfar.umd.edu}

Abstract

Contour is an important cue for object recognition. In this paper, built upon the concept of torque in image space, we propose a new contour-related feature to detect and describe local contour information in images. There are two components for our proposed feature: One is a contour patch detector for detecting image patches with interesting information of object contour, which we call the Maximal/Minimal Torque Patch (MTP) detector. The other is a contour patch descriptor for characterizing a contour patch by sampling the torque values, which we call the Multi-scale Torque (MST) descriptor. Experiments for object recognition on the Caltech-101 dataset showed that the proposed contour feature outperforms other contour-related features and is on a par with many other types of features. When combing our descriptor with the complementary SIFT descriptor, impressive recognition results are observed.

1. Introduction

While many recent object recognition studies have been based on interest point detectors and descriptors (*e.g.*, [1, 6, 8, 11, 12, 18, 19, 20]) tuned to texture-based features, some other powerful cues have not been sufficiently explored yet, and one of them is the cue of *contour*. Contours consist of curve or edge fragments, which present some meaningful geometric concepts. Contour features can effectively represent objects that can be clearly defined by shape (*e.g.*, a bottle or an LED monitor). It is clear that humans do recognize a wide range of objects based on their 2D outlines alone. Thus, contour features should play an important role in object recognition.

The contour-based approach is not as popular as the texture-based approach because of the complexity of detecting extended contours. A promising alternative approach is to use *contour patches* (fragments of contour) (*e.g.* [4, 10, 13, 16, 17]). There are three key components to contour patch based approaches: the *patch detector* that aims to find useful contour patches, the *local descriptor* that encodes the spatial distribution of edgels on the fragments into local features, and the *contour representation* of the overall contour or shape based on the spatial distribution of the local features. In the existing approaches, often the detection of contour patches is limited to fairly clean curves (*e.g.* [15]) that are sensitive to clutters. Some approaches detect simple elements like circles (*e.g.* [9]), whose discriminative power is weak, or represent shapes by the spatial distribution of local features and as a result, the stage of recognition becomes very complex (*e.g.* Hough like accumulators are involved in recognition in [16, 17]) with limited applications. This inspires us to develop a new contour-based detector with a trade-off between repeatability and discrimination and a feature descriptor, which provides very discriminative information of contours yet has a simple vector form to be easily used for recognition.

In this paper, we present a new contour-based feature based on the concept of *torque*. Torque, also called moment of force, is a physical concept that measures the tendency of an object to rotate around its axis. The torque measurement captures properties of the local shape structure of contours in a patch. A patch detector locates patches of largest/smallest torque value in a joint image-and-scale space. We call it the Maximal/Minimal Torque Patch (MTP)

*Y. Xu was partially supported by Program for New Century Excellent Talents in University(NCET-10-0368), the Fundamental Research Funds for the Central Universities(SCUT 2009ZZ0052) and National Nature Science Foundations of China 60603022 and 61070091. Cornelia Fermüller gratefully acknowledges the support of the European Union under the Cognitive Systems Program (project POETICON) and the National Science Foundation under the Cyberphysical Systems Program.

detector. The detected contour patches are then represented by a descriptor, which samples torque values in the neighborhood of the patch. It encodes the local variances of the contour fragments inside the patch. We call it the Multi-scale Torque (MST) descriptor.

The proposed contour feature was used for object recognition and tested on the Caltech-101 dataset. The experiments showed, using the contour cue as the only feature, our proposed method noticeably outperformed other contour-related features, and performed on a par with many existing methods using other types of cues. Combining it with SIFT, the resulting contour feature noticeably improved the classification performance of the SIFT-based approach.

1.1. Related work

There is an abundant literature on object recognition and classification. In the paradigm of current approaches, features are first extracted from images, and then these features are integrated for recognition. The methods differ in the choice of local features and the choice of integration. Since this paper focuses on the development of local image features, we only give a brief review on the integration of local features, which is followed by a more detailed review on image features.

Integrating extracted features for recognition. Early works simply matched individual local features to a feature pool collected from many known objects, e.g., Lowe [12] used this technique originally on the well-known SIFT feature. In recent years, the so-called *bag-of-features* (BoF) representation (*e.g.* [11, 19, 18]) has emerged as a powerful approach for integrating local features. The basic idea of BoF is to represent an image as a histogram with respect to a codebook built from the local features of known images. There are many variations of the BoF approach. Lazebnik *et al.* [11] proposed the so-called *spatial pyramid matching* (SPM) technique, in which an image is partitioned into increasingly finer spatial sub-regions and histograms of local features are computed from each sub-region. The SPM technique can effectively avoid the loss of spatial information in the BoF approach. To code an image more efficiently than in the simple vector quantization (VQ) scheme, Yang *et al.* [19] proposed an alternative soft and nonlinear coding scheme which balances reconstruction accuracy and sparsity of coding. Wang *et al.* [18] introduced a more general constraint regarding the locality of codes and developed the so-called *locality-constraint local coding* (LLC) scheme. Impressive recognition results have been reported in [18] using SIFT features. In this paper, we also adopt the LLC scheme for integrating our proposed features for recognition.

Image features for object recognition. The SIFT feature by Lowe [12] and its variations have been the most popular image features. SIFT captures the local structure of edge orientations in the neighborhood of an interesting image point, and has many attractive properties, including significant discriminative power and robustness to many types of environmental changes.

While SIFT is a texture feature, there also have been approaches using contour as the main cue for recognition. Belongie *et al.* [14] proposed the so-called shape context descriptor, that encodes the distribution of edgels in a histogram in log-polar coordinate system, and used it on segmented objects of simple shape. For recognizing objects in real scene, Jurie and Schmid [9] proposed a scale-invariant feature detector locating patches of local maximal saliency measured by the local convexity estimated from the energy and entropy of edgels. Then patches are described by the spatial distribution of points in a thin annular neighborhood of the circle. Fergus *et al.* [4] defined fragments of curve segments bounded by the bi-tangent points and used them in the constellation model for object retrieval. The descriptor is created by using a probabilistic likelihood term. The fragments used in these two methods either include only a few types of shapes or are not sufficiently dense.

An alternative is learning-based approach. Kumar *et al.* [10] proposed to learn contour fragments from video sequences in a Bayesian pictorial structure model and arranged them for the recognition of deformable objects. Shotton *et al.* [17] proposed to learn a fragment detector from random rectangles sampled from training segmentation masks. Opelt *et al.* [16] explicitly constructed fragments from a large fragment pool by simultaneously maximizing the occurrence in positive training and minimizing the occurrence in negative training sets. All these learning-based methods require very complex learning processes, and the invariance of the adaptive detectors is not very impressive.

Recently the contour grouping technique has emerged as a promising approach for contour-based recognition. Zhu *et al.* [21] proposed a set-to-set contour matching scheme for object detection. In their approach, the contour fragments are based on a bottom-up segmentation or contour grouping. Ferrari *et al.* [5] used groups of contour segments for objection detection in which local shape features are formed by chains of connected, roughly straight contour segments.

Finally, several approaches are proposed to combine multiple types of features, including both contour and texture. Zhang *et al.* [20] defined a distance measure of images using shape and texture features, and the approach developed by Boiman *et al.* [1] combined color, SIFT, shape context, and other descriptors for object classification.

2. Torque for image patches

In this section, we first give the definition of torque in image space, then we discuss its implications for contour-

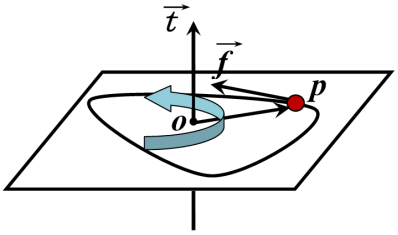


Figure 1. The torque defined in (1). The solid red point represents the edge point, while o represents the center.

based recognition.

2.1. Definition of torque of edge points and image patches

Torque is a physical measurement of the tendency of a force to rotate an object around an axis. Let o denote the center point, then for any point p in space, its *torque*, denoted by $\vec{t}$, is defined as follows,

$$\overrightarrow{t_o}(p) = \overrightarrow{op} \times \overrightarrow{f}(p), \tag{1}$$

where $\vec{f}(p)$ denotes the force vector, $\overrightarrow{op}$ denotes the arm vector and $\times$ denotes the cross product operation. First, the force vector of each image point p with non-zero image gradients is defined as

$$\overrightarrow{f}(p) = \frac{\nabla I(p)^{\perp}}{|\nabla I(p)|}, \tag{2}$$

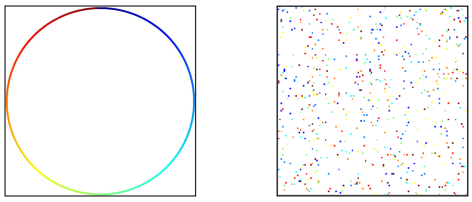
where $\nabla I(p) = (I_x, I_y, 0)$ denotes the gradient of p in the image, and $\nabla I(p)^{\perp}$ is the vector perpendicular to the image gradient (and parallel to the edge) measured counterclockwise, such that the brighter side is on its right and the darker side is on its left. $|\cdot|$ denotes the length of the vector. The magnitude of torque for an edge point p with a pre-defined center point o is defined as:

$$\tau_o(p) = |\overrightarrow{op}||\overrightarrow{f}(p)| \sin\theta = |\overrightarrow{op}| \sin\theta, \tag{3}$$

where θ is the counterclockwise angular distance from the arm vector to the force vector and is in the range from $0°$ to $360°$ degree. We emphasize here that the magnitude of torque defined in (3) is not the same as the length of $\vec{t}_o(p)$. It may take negative value. See Fig. 1 for an illustration of the torque and its magnitude. It can be seen that the magnitude of $\tau_o(p)$ is determined by the relative position of p with respect to the center o and the direction of its force vector.

The torque of an image patch is defined as follows. For a given patch P, let c denote the center of the patch. Then, the torque of the patch P is defined as

$$\vec{t}_c(P) = \sum_{p \in P} \vec{t}_c(p). \tag{4}$$



(a) μ_τ=0.785 (b) μ_τ=0.095

Figure 2. The torque magnitudes within a unit patch are calculated for two different cases. The edgels forming a circle with radius $r = 1/2$ in (a) result in a much larger μ_τ value than that of the edgels in (b) generated as a uniform distribution. Different colors of the edgels correspond to different orientations.

Notice that all $\vec{t}_c(p)$ are parallel to each other since they are perpendicular to the image plane. Thus, the magnitude of $\vec{t}_c(P)$, denoted by $\tau_c(P)$, can be expressed as

$$\tau_c(P) = \sum_{p \in P} \tau_c(p). \tag{5}$$

In other words, the magnitude of the torque of a patch is the sum of the magnitudes of torque of all points inside the patch. To achieve independence to patch size, we normalize to obtain :

$$\mu_\tau(P) = \frac{1}{2} \frac{\tau_c(P)}{\text{area}(P)}. \tag{6}$$

In the remainder of this paper, we refer to $\mu_\tau(P)$ as the torque magnitude of the patch P.

2.2. Discussion of the torque magnitude

Next we discuss how the torque magnitude μ_τ is related to the contours in the image patch P and how it can benefit a contour-based recognition.

First, the value of μ_τ will be larger when the edges in the patch tend to be in the order, regular and enclosed. On the contrary, if the edge segments are randomly distributed all over the patch, μ_τ will be very small. This is exactly parallel to what happens in mechanics. In order to rotate an object around an axis more efficiently, the force should be applied uniformly along the tangent direction of the rotation trajectory. See Fig. 2 for a comparison of the torque magnitude in two different patterns. Thus, μ_τ measures the orderliness of edges in a patch.

Second, the value μ_τ gives us some information on the relative size of the contour to the patch and its position within the patch. The larger the value of μ_τ, the tighter the patch boundaries will enclose the contour. A patch with large absolute value of μ_τ is likely to include convex contours. Thus, the torque magnitude μ_τ can be used to infer the existence of convex contours in the patch.

Lastly, according to our definition of the orientation of an edge, a contour that encloses a bright patch on dark back-

ground will have positive magnitude of torque, while a contour corresponding to a dark patch on bright background will have negative magnitude of torque. In summary, the measurement μ_τ defined in (6) implicates several attractive properties which describe the contours in image patches.

3. Contour related features using torque

3.1. MTP detector

As discussed in previous sections, the torque magnitude μ_τ is dependent on how tight the boundaries of a patches enclose regular salient contours. Thus, based on the value of μ_τ, we propose a local contour detector for finding local patches with regular contours. We define a patch as a maximal/minimal torque patch if its torque magnitude takes a maxima/minima among the torque magnitudes of all patches of multiple sizes but with the same center and is maximum/minimum among the spatial neighbors. We call this patch detector the MTP patch detector. A threshold is set to discard unreliable MTP patches resulting from low contrast regions. An outline of the algorithm is given in Alg. 1 and illustrated in Fig. 3.

Algorithm 1 Maximal/Minimal Torque Patch (MTP) Detector

Input: an image

1. **Torque calculation of patches**. The image is partitioned into multiple patches of different sizes, and the torque magnitude of each patch is calculated using (6).
2. **Extrema detection**. For each candidate patch, locate the candidate MTP patch whose torque magnitude takes the extreme value (maxima or minima) in its spatial-and-scale neighborhood.
3. **Patch thresholding**. Remove those patches from the set of all candidate MTP patches whose torque magnitudes are below some pre-defined threshold.

Output: The MTP patch set $\mathcal{R}$.

The MTP detector is inherently translation-invariant as it is based on the local coordinate system of a patch. The MTP detector is also scale-invariant. Note that either the amount of edgels (forces) or the length of arms of forces are proportional to the scale of the patch. Thus, the torque magnitude of a patch in (5) is proportional to the area of the patch, and the normalized torque magnitude of a patch in (6) is independent of the scale of the contour. To achieve robustness to rotation and affine transforms, 45 degree patch and rectangular patch can be considered.

Some examples of local contour patches detected by the MTP detector are shown in Fig. 4, in which both square patches and rectangular patches are employed. For clarity, we only show part of the detected patches. It is noted that there are two types of MTP patches based on the sign of the torque magnitude: one with the positive value of μ_τ called *bright patch*; the other with negative value of μ_τ called *dark patch*. These two types of patches are complementary to each other. If a concave contour cannot be detected by dark patches, it is very likely to be detected by its neighboring region as a bright patch. The odd columns and even columns in Fig. 4 illustrate this phenomena. Using complementary bright and dark patches allows us to locate most of the local patches with meaningful local contour information.

3.2. Fast computation of the torque

The MTP detector requires the calculation of μ_τ at every position and for multiple patch sizes (Step 1 in Alg. 1), which is time-consuming if computed straight-forward. Here we give another derivation of the torque of a patch (defined in (5)), such that the so-called integral image technique [3] can be applied to significantly speed up the computation. The basic idea is to pre-compute the force vectors and the torque values τ_o with respect to a fixed point, denoted as o :

$$\{\vec{f}(p); \tau_o(p)\}_{p\in\Omega},$$

where Ω denotes the image domain. We set o to be the left top corner of the image. Let $\vec{t}_c(P)$ denote the torque of patch P centered at the point c, as defined in (4). Then, we can rewrite $\vec{t}_c(P)$ as

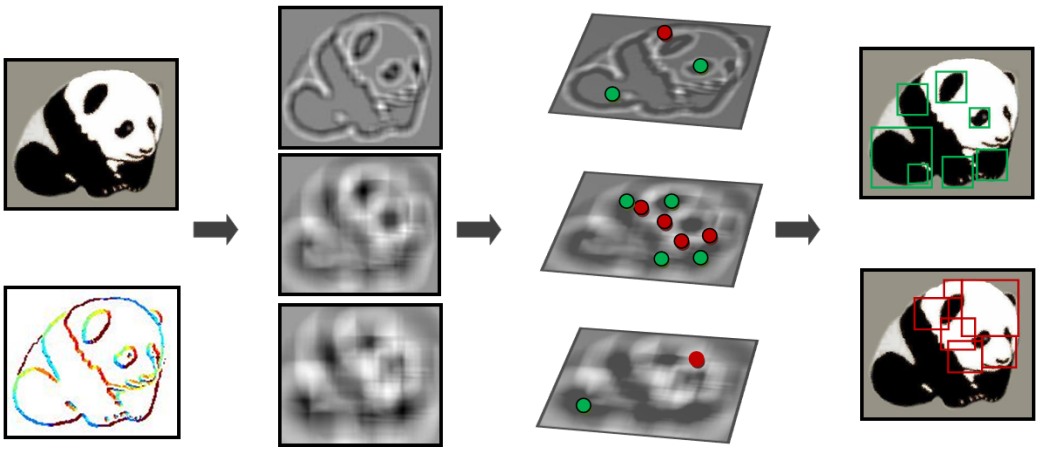
$$\begin{aligned} \vec{t}_c(P) &= \sum_{p\in P} \overrightarrow{cp} \times \overrightarrow{f}(p) \\ &= \sum_{p\in P} (\overrightarrow{co} + \overrightarrow{op}) \times \overrightarrow{f}(p) \\ &= \overrightarrow{co} \times \overrightarrow{f}(P) + \vec{t}_o(P), \end{aligned} \tag{7}$$

where $\overrightarrow{f}(P) = \sum_{p\in P} \overrightarrow{f}(p)$ is the sum of forces in the patch P, and $\vec{t}_o(P)$ is the torque of P with respect to the original point o.

Notice that we can pre-compute $\overrightarrow{f}(p)$ and $\vec{t}_o(p)$ for all N pixels in the image in $O(N)$ time. Once they are pre-computed, $\overrightarrow{f}(P)$ and $\vec{t}_o(P)$ can be calculated for any patch P in $O(1)$ time by integral images [3]. After $\vec{t}_c(P)$ is calculated using (7) for all patches, the torque magnitude $\mu_\tau(P)$ of the patch P, defined in (6), can be obtained easily.

3.3. MST descriptor

For a given contour patch, we propose a torque-based descriptor to describe the density and variance of the local edge structure in a multi-scale manner. We call it the Multi-scale Torque (MST) descriptor. The basic procedure is as follows. For a given patch we consider all patches having an overlap with the patch along the eight axes at discrete space intervals as shown in Fig. 5 (a). The MST descriptor is



(a) Edge extraction (b) Torque magnitudes (c) Finding extrema (d) Locating patches

Figure 3. Outline of the MTP detector. From left to right: (a) The original image and its edge extraction. Different colors represent the corresponding orientations of the edgels. (b) The torque magnitudes at every point at multiple patch sizes are computed. (c) Extremal torque magnitudes are detected. (d) The corresponding contour patches are localized.

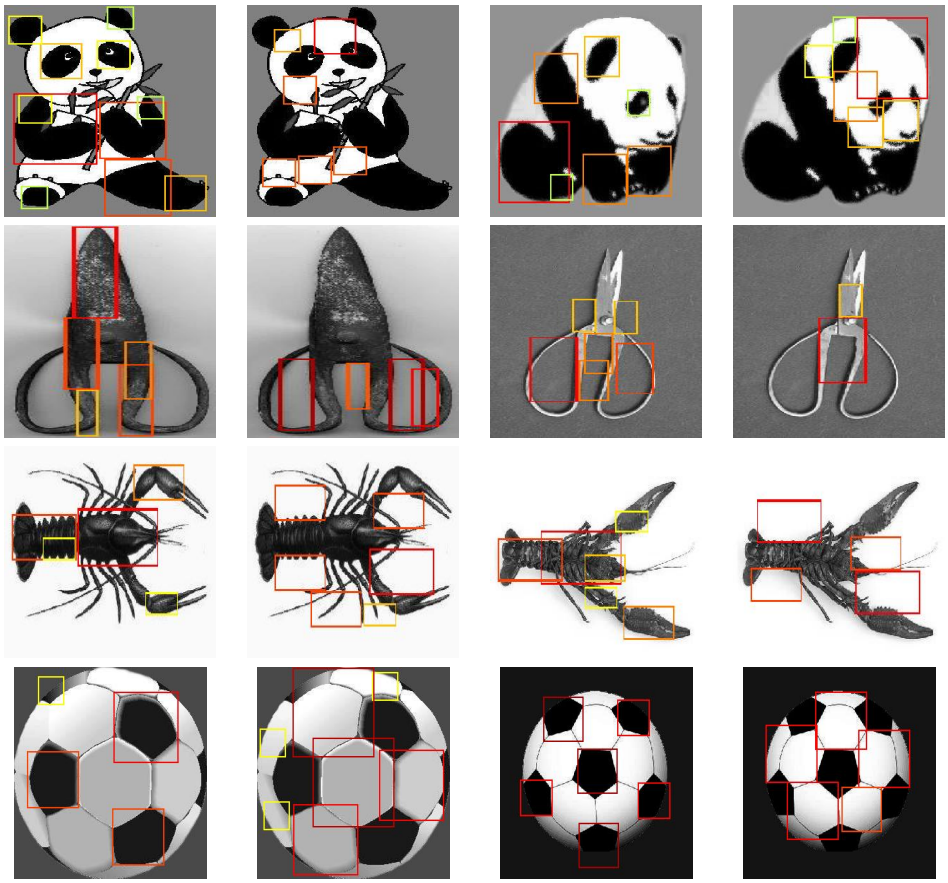


Figure 4. Examples of applying the MTP detector on four object categories in the Caltech-101 dataset. For each object category two samples are shown, and for each sample the two types of patches are detected. The odd columns show the dark patches while the even columns show the bright ones. The colors of the patches denote their size. Note, that for clarity not all of the detected patches are shown.

the concatenation of the torque magnitudes of these patches. To keep the number of selected patches the same for all patches, we sample the torque magnitudes with an adapted step size according to the patch size. To achieve rotation invariance, the patch is rotated such that its x-axis becomes the direction closest in direction to the vector pointing from the center to the centroid of the edges inside the patch. This can be done by circular shift. The outline of the MST descriptor is illustrated in Fig. 5.

4. Experiments

4.1. Implementation details

In order to evaluate the contour feature for object recognition, we followed the *bag of features* (BoF) representation paradigm. The basic procedure is as follows. The MST features from all the images are clustered as a codebook using the K-Means algorithm and are represented as codes via the LLC coding scheme [18]. Then each image is represented as a normalized histogram with respect to its codes using the SPM pooling technique [11]. The reason for using the LLC scheme is that it works well with simple linear classifiers, and there exists an approximated version for fast computation ([18]). The SPM pooling technique is used, because it showed good performance in many recent state-of-the-art image classification systems (*e.g.*, [11, 18, 19]). The details of our image representation are as follows:

Feature extraction. Each image is converted to a collection of local contour features, *i.e.*, we compute the MST descriptor on each patch extracted by the MTP detector. Taking account of efficiency and effectiveness, we use square and rectangular patches of fixed aspect ratios, rotated by 0 and 45 degree.

Codebook generation. For each image in the training set, we cluster its contour features to build a codebook. The bright and dark MTP patches are coded in two codebooks, which are processed separately.

Image representation. Given an image, its features (descriptors) are quantized as codes w.r.t. the codebook using LLC (in practice we use its approximated version), in which each descriptor is projected into its local-coordinate system using the locality constraint. Multiple codes are integrated via SPM and max pooling as a normalized histogram. This histogram is the feature vector of the image.

Training stage. Once each image is represented as a vector, numerous learning-based approaches can be used to train a classifier (*e.g.*, KNN, SVM). A plain SVM is used as the classifier in our implementation.

4.2. Classification on the Caltech-101 dataset

We evaluated the performance of our proposed feature for object classification on the widely used Caltech-101 object dataset.

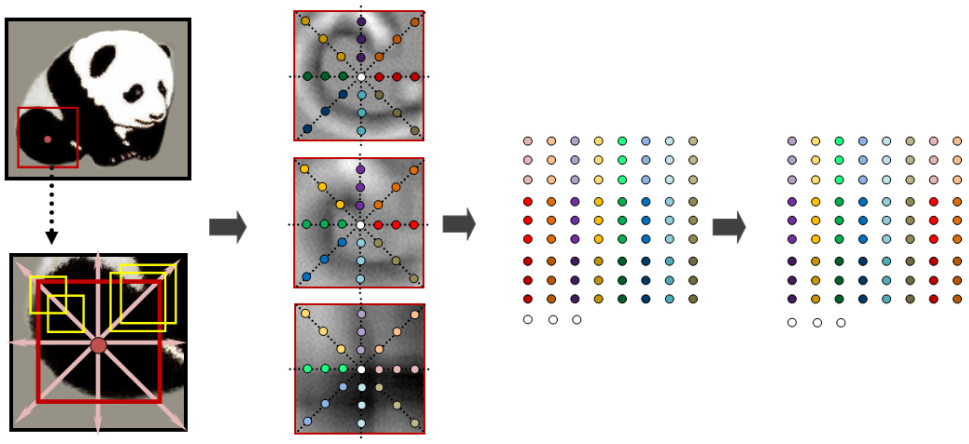
Configuration. Caltech-101 [2] is a large dataset with 8677 images from 101 object categories with different shapes and appearances, and with 467 images from an additional background category. The number of images per category varies greatly from 31 to 800. We follow the experimental configuration suggested by the original dataset, and also used in [20, 7, 18]. Images were resized to a maximum of 300*300 gray-scale pixels with preserved aspect ratio. For each category, 5, 10, 15, 20, 25 and 30 images were randomly picked for training, and no more than 50 images were randomly picked for testing from the remaining images. Performance was measured using average classification accuracy over all classes.

Methods for comparison. First we compared our features to the other two state-of-the-art contour-related features for which source codes are available, the *k*AS feature [5] and the extended shape context in [1]. Note that these two features were originally designed for matching, not for recognition. To eliminate the effect of the image representation framework on recognition performance, we ran all three contour-related features on the same BoF-based image representation framework. In the comparison the two methods are denoted "*k*AS + BoF" and "Boiman shape + BoF", respectively.

We also compared our methods to other recognition methods using different types of cues, specifically the methods: [20, 11, 7, 1, 8, 6, 19, 18]. Furthermore, we combined our proposed feature with the popular SIFT feature to see how much additional improvement can be gained by adding the proposed contour feature to the classic texture-based approach. Specifically, we added our proposed feature into the SIFT-based method by Wang *et al*. [18], denoted as "Ours+SIFT.

Parameter setting. For the MTP detector we set the patch threshold to 0.3. During detection, we employed square patches and four types of rectangular patches, whose aspect ratios were 1:$\sqrt{2}$, 1:2, $\sqrt{2}$:1 and 2:1 respectively. The scales of the patches were defined as a series of integers from $1/50$ to $4/5$ of the image size, increasing by a factor of $\sqrt[3]{2}$. For the MST descriptor, we used 3 scales: the current scale of the described patch and one neighbor scale above and below. We sampled 15 torque magnitudes along each axis at each scale, resulting in a 363-dim ($3 \times 8 \times 15 + 3 = 363$) descriptor. For codebook generation, the codebook size was fixed at 2048. For the approximated version of LLC, we set our parameters the same as in [18]. In the SPM pooling, we employed 4×4, 2×2 and 1×1 sub-regions.

The implementation of the extended shape context and *k*AS was as follows. For the extended shape context, we computed a 192-dim (8 angular bins multiply 3 radius bins multiply 8 edge orientation bins) descriptor for each patch.



(a) An MTP patch (b) Torque magnitudes (c) Multi-scale sampling (d) Orientation alignment

Figure 5. Outline of the MST descriptor. From left to right: (a) An interesting patch is detected by the MTP detector. (b) The torque magnitudes of the patches centered at points inside the detected patch are computed. (c) The torque magnitudes are down-sampled along 8 directions at several scales. The sampled values are collected and concatenated as the local feature of the MTP patch. (d) The orientation of the feature is aligned by circular-shifting.

	5	10	15	20	25	30
*k*AS + BoF	24.47	31.80	35.86	38.64	40.64	41.99
Boiman shape + BoF	40.48	49.43	54.56	57.59	59.92	61.54
Dense patch + MST	37.67	47.62	52.75	56.59	58.82	60.61
Ours (MTP + MST)	**48.17**	**57.65**	**62.33**	**65.32**	**67.39**	**68.97**

Table 1. Classification accuracy for methods using single contour feature on the Caltech-101 dataset.

In *K*AS, we used k=1,2,3,4, resulting in 4 types of descriptors. Each descriptor was represented as a feature vector and then the vectors were concatenated. Considering the low dimension of the *k*AS descriptor, we reduced the codebook size to 64 for *1*AS and 1024 for the remainders.

4.3. Results and discussion

As discussed in Sec. 3.2, since we use integral images, the computation is very efficient. The average running time for MTP and MST implemented purely in matlab is about 11 seconds for one image in Caltech-101 on a PC with 1.6 GHZ Intel CPU.

The experimental results for three contour feature based methods are reported in Table 1. As can be seen, our approach outperformed the other two contour-related features under the same BoF image representation framework. This result is not surprising, considering the fact that these contour-related features were designed originally for matching and not for recognition.

To see the power the MST descriptor, we also extracted the MST descriptor from patches densely located at every 6th pixel in the image using patches of 7 scales. The result (denoted as 'Dense patch + MST') is also shown in Table 1. Clearly, when using the MST descriptor directly without the elaborate selection by the MTP detector, the recognition performance declines. Even so, it still performed a par with other state-of-the-art contour features.

Referring to the results of comparison to other feature descriptors in Table 2, it can be seen that our approach outperforms several state-of-the-art methods, including [20], [11], [7] and [6], but did not perform as well as [1], [8], [6] and [19]. This result is also not surprising to us since the contour cue is only one of many visual cues for recognition. There is a large diversity of images in the Caltech-101 dataset, and a significant amount of images have significant texture content, not used in our contour-based feature. One single visual cue apparently is not sufficient to characterize all types of images in the Caltech-101 dataset. In comparison, [1, 8, 6, 19] use SIFT-based features and thus efficiently utilize the texture information and salient image points for recognition.

To evaluate whether our proposed contour-related feature can improve existing recognition methods, we combined our contour-based feature with the texture-related SIFT feature in a straightforward way. The implementation of the SIFT feature followed that of [18], and we combined the two features by concatenating them into a single vector and weighing them 1:2 (ours v.s. SIFT). This weighting

	5	10	15	20	25	30
Zhang *et al.* [20]	46.6	55.80	59.10	60.20	-	66.20
Lazebnik *et al.* [11]	-	-	56.40	-	-	64.60
Griffin *et al.* [7]	44.20	54.50	59.00	63.30	65.80	67.60
Boiman *et al.* [1]	-	-	65.00	-	-	70.40
Jain *et al.* [8]	-	-	61.00	-	-	69.10
Gemert *et al.* [6]	-	-	-	-	-	64.16
Yang *et al.* [19]	-	-	67.00	-	-	73.20
Wang *et al.* [18]	51.15	59.77	65.43	67.74	70.16	73.44
Ours	48.17	57.65	62.33	65.32	67.39	68.97
Ours + SIFT	**53.60**	**64.01**	**69.15**	**72.40**	**74.52**	**76.22**

Table 2. Classification accuracy for different methods on the Caltech-101 dataset.

scheme was chosen because our feature vector is 2 times as long as the SIFT vector. We refer to it as "Ours+SIFT ". It can be seen that there is an additional $2.45\% - 4.66\%$ accuracy gain over the best results of other methods with respect to different sizes of the training set. The results demonstrate that our proposed contour-based feature does capture meaningful information of object contour and is a useful addition to object recognition. It is noted that a better performance (72.8% when using 15 for training) is reported in [1]. However, this approach is based on multiple features including SIFT, simple luminance, color, extended shape context and the self-similarity descriptor, while our result is based on two types of features only.

5. Conclusion

In this paper we proposed a new contour-based feature coding scheme for object recognition. It includes a contour patch detector (MTP patch detector) and a contour feature descriptor (MST descriptor). We evaluated the scheme on the Caltech-101 dataset, and the results showed its performance to be on a par with many other methods when using it as a single cue. When used in combination with the SIFT-based feature, it provides a more effective image representation that outperformed other methods in object recognition.

References

[1] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. *CVPR*, 2008. 1, 2, 6, 7, 8

[2] Caltech-101 http://www.vision.caltech.edu/Image_Datasets/Caltech101/ 6

[3] B. Catanzaro, B. Y. Su, N. Sundaram, Y. Lee, M. Murphy, and K. Keutzer. Efficient, high-quality image contour detection. *ICCV*, 2009. 4

[4] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for Google images. *ECCV*, 2004. 1, 2

[5] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *PAMI*, 30(1): 36 - 51, 2008. 2, 6

[6] J. Gemert, J. Geusebroek, C. Veenman, and A. Smeulders. Kernel codebooks for scene categorization. *ECCV*, 2008. 1, 6, 7, 8

[7] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. *Technical Report 7694*, California Institute of Technology, 2007. 6, 7, 8

[8] P. Jain, B. Kullis, and K. Grauman. Fast image search for learned metrics. *CVPR*, 2008. 1, 6, 7, 8

[9] F. Jurie and C. Schmid. Scale-invariant shape features for recognition of object categories. *CVPR*, 2004. 1, 2

[10] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Extending pictorial structures for object recognition. *BMVC*, 2004. 1, 2

[11] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. *CVPR*, 2006. 1, 2, 6, 7, 8

[12] D. Lowe. Distinctive image features from scale invariant keypoints. *IJCV*, 60(2): 91 - 110, 2004. 1, 2

[13] K. Mikolajczyk, A. Zisserman, and C. Schmid. Shape recognition with edge-based features. *BMVC*, 2003. 1

[14] G. Mori, S. Belongie, and J. Malik. Efficient shape matching using shape contexts. *PAMI*, 27 (11): 1832-1837, 2005. 2

[15] R.C. Nelson and A. Selinger. A cubist approach to object recognition. *ICCV*, 1998. 1

[16] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. *ECCV*, 2006. 1, 2

[17] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. *ICCV*, 2005. 1, 2

[18] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. *CVPR*, 2010. 1, 2, 6, 7, 8

[19] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. *CVPR*, 2009. 1, 2, 6, 7, 8

[20] H. Zhang, A. C. Berg, M. Maire, and J. Malik. SVM-KNN: discriminative nearest neighbor classification for visual category recognition. *CVPR*, 2006. 1, 2, 6, 7, 8

[21] Q. H. Zhu, L. M. Wang, Y. Wu, and J. B. Shi. Contour context selection for object detection: a set-to-set contour matching approach. *ECCV*, 2008. 2