To appear, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2010.

Asymmetric Region-to-Image Matching for Comparing Images with Generic Object Categories

Jaechul Kim and Kristen Grauman
University of Texas at Austin
{jaechul,grauman}@cs.utexas.edu

Abstract

We present a feature matching algorithm that leverages bottom-up segmentation. Unlike conventional image-to-image or region-to-region matching algorithms, our method finds corresponding points in an "asymmetric" manner, matching features within each region of a segmented image to a second unsegmented image. We develop a dynamic programming solution to efficiently identify corresponding points for each region, so as to maximize both geometric consistency and appearance similarity. The final matching score between two images is determined by the union of corresponding points obtained from each region-to-image match. Our encoding for the geometric constraints makes the algorithm flexible when matching objects exhibiting non-rigid deformations or intra-class appearance variation. We demonstrate our image matching approach applied to object category recognition, and show on the Caltech-256 and 101 datasets that it outperforms existing image matching measures by 10∼20% in nearest-neighbor recognition tests.

1. Introduction

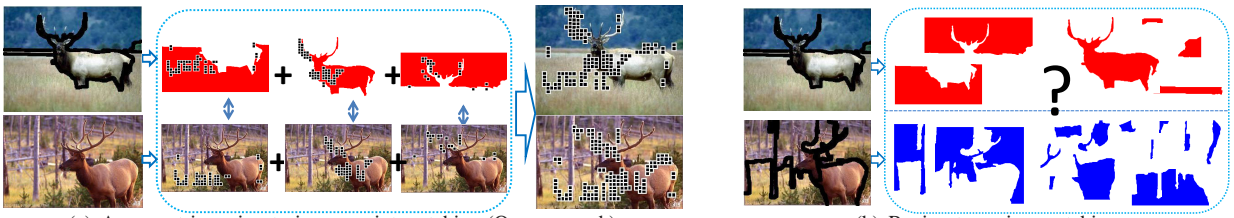
Finding corresponding points between images is a long-standing research problem in computer vision, and is especially important to today's object recognition and image retrieval methods that use local feature representations. Point-to-point matching methods with local descriptors (e.g., SIFT, shape context [1, 2]) are particularly valuable for these tasks, due to the features' robustness to partial occlusion, illumination changes, and clutter.

The locality of such appearance-based feature matches yields some noisy correspondences when used alone, and so additional geometric constraints are typically imposed to select the consistent matching points among the initial pool of point matches. Parameterized geometric constraints (e.g., an affine transformation between local regions) can be used for more reliable object instance matching and image retrieval [1, 3, 4, 5]. For generic categories, however, geometric consistency is less exact and the correct transformations are non-global, making the parametric constraints less amenable to category-level object matching (e.g., matching images of articulated giraffes, or different models of boats). Instead, non-parametric approaches that identify a group of matches having minimal geometric distortion may be preferable in order to establish correspondences at the category-level [6, 7]. In addition to measuring overall image similarity, the resulting correspondences are useful to localize the object within the two views.

However, there are two key limitations to current techniques. First, pre-computing the distortion for all tentative matching pairs is computationally expensive, making very densely sampled feature points off-limits in practice. As a result, most methods restrict the stronger geometric consistency measures to a sparsely sampled set of features (such as local maxima in scale-space, edge points, etc.). While generally effective for matching object instances, sparsely sampled interest points provide a weaker representation for category-level matching; much evidence suggests that a dense coverage of features is preferable [8].

Second, non-parametric methods that use pairwise measures of distortion typically identify a single group of corresponding points that undergo a common (low-distortion) transformation. Yet in typical real images, each part of a non-rigid object—or each instance of multiple objects in the image—can undergo a different transformation, suggesting that we should identify multiple groups of corresponding points, each with a different geometric configuration.

We propose a dense feature matching algorithm that exploits the grouping provided by bottom-up segmentation to compare generic objects with non-parametric geometric constraints. Our method takes two images as input and returns a score for their similarity. One input is left unsegmented, while the other is segmented. The matching process begins by finding correspondences between points within each region of the segmented image and some subset of those within the unsegmented image, while efficiently enforcing layout consistency in each of the region-to-image match groups via an objective solvable with dynamic pro-



(a) Asymmetric region-to-image point matching (Our approach) (b) Region-to-region matching

Figure 1. **(a)** The proposed asymmetric region-to-image matching method, and **(b)** the key contrast with region-to-region matching. In a *region-to-image* match, we use regions from the segmentation of one image (top row, left) to group points for which to seek matches in the second unsegmented image (bottom row, left). In our asymmetric strategy, we exploit the fact that a group of feature points (small squares, e.g., denoting dense SIFT) within the same segment often belong to the same object subpart, giving us an automatic way to generate groups which when matched to the second image, should have low total geometric distortion. For example, here, the elk horns, body, and grass are well-matched (center, larger images) even though the parts separately undergo different deformations to fit to a different category instance in the second image. In contrast, a *region-to-region* match that seeks correspondences between pairs of regions from *both* images' segmentations (two rows on right side), cannot exploit the bottom-up grouping as well, since it can be misled whenever the bottom-up segmentations for the two images lack agreement.

gramming. The union of these correspondence groups are then further evaluated for their mutual geometric consistency, at which point we favor low distortion matches for each segmented region, while allowing larger deformations *between* the segmented regions of the original image.

We call our image matching "asymmetric" because only one of the input images is segmented into regions, and its groups of points can match within any (consistent) portion of the other image. We find this deliberate imbalance to be an advantage when matching: we get the grouping power of low-level segmentation to assemble candidate regions of points, but without suffering in the inevitable event where the bottom-up cues produce incompatible regions for two images of the same object (see Figure 1).

We apply our matching algorithm to exemplar-based object category recognition on the Caltech-256 and 101 datasets. The results show our method works robustly under challenging appearance and pose variations. It provides 10-20% better accuracy over existing matching-based categorization techniques, and is competitive with methods that include more sophisticated learning components when using the same local feature types. To our knowledge, this is the first image matching algorithm tested on Caltech-256 that achieves a raw matching accuracy comparable to some classifier-based algorithms. We highlight the important conceptual differences between our approach and existing matching measures in the following section.

2. Related Work

One strategy to impose more "forgiving" geometric constraints for category-level feature matching is to generate semi-local constraints, building neighborhood features centered about each individual interest point [9, 10, 11]. Another idea is to minimize the total pairwise distortion between correspondences; while expensive to compute optimally, approximate methods have also been explored and yield good results [7, 6]. Nonetheless, the approximate methods still require computing pairwise candidate distortions prior to optimizing the assignment, which adds a significant computational overhead—$O(m^2n^2)$ for images with m and n points. We show that by representing the pairwise geometric relations between only adjacent points using multiple "strings" in a region, we can capture sufficient layout information without sacrificing efficiency; in contrast to the above, our dynamic programming method requires only $O(nm^2)$ time. The other important distinction between our approach and [7, 6] is that it identifies geometrically consistent correspondences in *groups*, not for the entire image at once, while allowing deformations between matches within different regions. This grants us more flexibility when matching each part of a non-rigid object, or when computing many-to-one matches.

To deal with non-uniform distortion patterns, a technique proposed in [12] uses agglomerative clustering to get multiple clusters of corresponding points, each of which exhibits an independent geometric configuration to the other. However, all pairwise geometric relations among candidate matching points must still be evaluated, and the method targets object instance matching rather than object categories.

The SIFT-flow algorithm [13] provides dense correspondences between every pixel of two images, and has shown promising results for scene matching. It is not clear whether it will correctly match images with significant non-rigid transformations or background clutter, since it enforces a smoothness constraint on the displacement between nearby corresponding points over the entire image.

Like our method, the "bundling features" algorithm [14] also uses a region as a unit for which geometric constraints are independently imposed. The method gives state-of-the-art results for instance-level image retrieval; however, since

the bundled points are taken from within affine invariant regions detected with MSER, they may be too local to cope with substantial distortion, and inter-region geometric relations are not used.

A spatial mismatch kernel [15] measures image similarity by counting the number of similar sub-strings within an allowed amount of mismatch. Our approach also integrates a form of string matching; however, unlike [15], it does not attempt to match strings from two images using identical ordering—which can only work well when the images have a globally similar layout (e.g., for scenes). Additionally, our strings are constructed within region primitives only.

Recent progress in region-to-region matching includes a bag-of-regions scheme [16], graph matching approaches using region hierarchies [17, 18, 19], and multiple segmentation-based methods (e.g., [20]). Hierarchical or multiple segmentations help reduce the risk of getting stuck with only poor segments, and the above methods give strong results for category recognition. In contrast to these methods, we use the region primitives in an asymmetric way, matching region-to-image rather than region-to-region. Figure 1 highlights the potential advantages. Essentially, by matching from a region in one image to some unrestricted (but geometrically consistent) portion of the second image, we can leverage the strength of bottom-up segmentation algorithms (keeping coherent pixels of object sub-parts in the same regions) while avoiding their known weaknesses (oversegmenting objects with homogeneous texture, and inconsistently decomposing different instances of the same generic category).

3. Approach

Our method takes two images as input and returns a score for their similarity, as well as correspondences explicitly indicating their matching features. We first describe our algorithm in detail (Sec. 3.1), and then discuss some important implications of our choices (Sec. 3.2).

3.1. Region-to-Image Point Matching

We first decompose one of the two input images into regions using bottom-up segmentation. Each region is mapped to a set of local SIFT descriptors densely sampled at multiple scales on a regular grid. We denote each grid location as a "point". Then, we represent each region by strings of its constituent points. To more robustly represent the 2D layout using a 1D string, we extract two strings per region: a column-wise string and row-wise string. In the column-wise linkage of a string, we start from the top-left point in the region, and link nearest points along a column. When we arrive at the end of a column, the end-point is linked to the closest point in the next column. We repeat this process until we reach the end-point of the last column. Similarly, row-wise linkage links nearest points along a row.

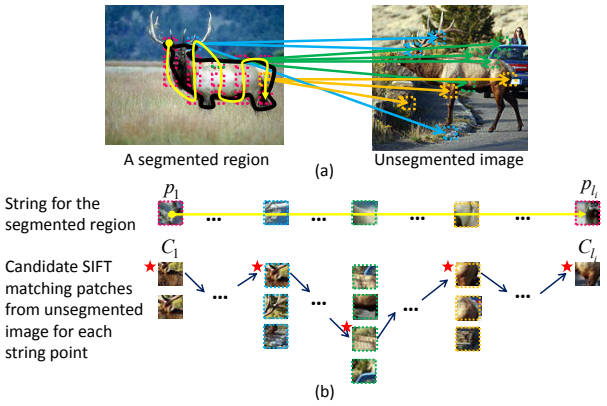


Figure 2. Illustration of an asymmetric string match. **(a)** A segmented region in the left image is represented by a column-wise string. For each point in the string, we identify candidate matches in the (unsegmented) right image by SIFT matching. **(b)** We use dynamic programming (DP) to solve for the assignment between the string and the candidates that minimizes the total geometry and appearance costs. Short arrows and stars denote the optimal matches selected with DP. (Note, this example does not include any null matches.) Best viewed in color.

When matching a region to the unsegmented image, for each point in the string, we first find a set of candidate matching points in the unsegmented image, as determined by the SIFT descriptor distances across multiple scales. Note that we only extract strings from one of the input images; the candidate matches may come from anywhere in the second (unsegmented) image. Given these candidate matches, we compute the optimal correspondence by using dynamic programming (DP) to minimize a cost function that accounts for both appearance and geometric consistency (to be defined below). See Figure 2 for an overview. We obtain the solution for both the row-wise linked string and the column-wise linked string, and take the union of the correspondence pairs to be the region-to-image matches for that particular region.

In the following, we define the cost function, and then explain how the resulting correspondences are scored to produce a single similarity value between the two images.

Match assignment cost function: The segmented image produces a string for every region. Each region's string consists of a set of points, $P_i = \{p_1 \ldots, p_{l_i}\}$, where l_i denotes the length of the i-th region's string, each p_k records the image coordinates for that point, and any (p_k, p_{k+1}) denotes a pair of neighboring points on the string. We normalize the image coordinates by the length of the longer side of the image, making the range of coordinate values between 0 and 1. Let C_k denote the set of candidate matching points for a point p_k; each point in C_k is a feature in the unsegmented image whose SIFT descriptor is close to the one associated with p_k (e.g., C_1 is the first column of patches in Figure 2(b)). Among the candidate sets

$[C_1, \ldots, C_{l_i}]$, we want to solve for the optimal matching $M^* = \{m_1, \ldots, m_{l_i}\}$, where each $m_k \in C_k$, such that the assignment minimizes the following cost function:

$$\mathcal{C}(P, M) = \sum_{k=1}^{l_i-1} w_g G(p_k, p_{k+1}, m_k, m_{k+1}) + \sum_{k=1}^{l_i} w_a A(p_k, m_k) + \sum_{k=1}^{l_i-1} w_o O(p_k, p_{k+1}, m_k, m_{k+1}) + \sum_{k=1}^{l_i} w_d D(p_k, m_k). \quad (1)$$

The cost function has two pairwise terms, $G(\cdot)$ and $O(\cdot)$, and two unary terms, $A(\cdot)$ and $D(\cdot)$. Each term has an associated weight (w_g, w_o, w_a, and w_d) that scales their relative impact. The input P is fixed; we optimize over the selections in M. We now define each component term.

The *geometric distortion* term,

$$G(p_k, p_{k+1}, m_k, m_{k+1}) = \|(p_k - p_{k+1}) - (m_k - m_{k+1})\|_2,$$

measures the pairwise geometric deformation between pairs of corresponding points. This gives preference to neighboring match pairs that have similar distortion.

The *ordering constraint* term, $O(\cdot)$, penalizes the pairs of correspondences when they violate the geometric ordering. Its value is 1 if the ordering of p_k and p_{k+1} is different from that of m_k and m_{k+1} (in either the horizontal or vertical direction), and 0 otherwise. This means, for example, that if point p_k is located left of the point p_{k+1}, its matching point m_k should be located left of m_{k+1}.

The *appearance similarity* term penalizes dissimilarity between the SIFT descriptors extracted at the two points, and is defined as:

$$A(p_k, m_k) = \frac{1}{1 + \exp\left(-\tau_a \left(\frac{1}{\mu_a}\|f_{p_k} - f_{m_k}\|_2 - 1\right)\right)}, \quad (2)$$

where f_{p_k} and f_{m_k} denote the SIFT descriptors at p_k and m_k, respectively, and $\|f_{p_k} - f_{m_k}\|_2$ is determined by the minimum distance among the descriptors at all scales extracted at the two points. The positive constants μ_a and τ_a simply adjust the shape of sigmoid function to make the values compatible with the other cost terms. Larger SIFT distances induce larger cost values.

Finally, the *displacement constraint* term $D(p_k, m_k)$ penalizes large displacement between the locations of corresponding points, thereby giving some preference for objects that occur within similar scene layouts:

$$D(p_k, m_k) = \begin{cases} \|p_k - m_k\|_2, & \text{if } \|p_k - m_k\|_2 > t \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

where t is a displacement threshold.

Our description thus far assumes that every point in the string has a corresponding point in the second image. However, some points may not have a reliable matching point. To handle this, we insert a "null match" candidate into each candidate set C_k. If a null match is selected when optimizing $\mathcal{C}(P, M)$, it means that the associated point does not have a real corresponding point. For every term where either m_k or m_{k+1} is a null match, the cost value in each sum of Eqn. 1 is set to a constant value χ. We set the constant χ to be larger than the typical matching costs for reliable correspondences.

For each region P_i in the segmented image, we use dynamic programming to minimize Eqn. 1 over the candidate selections for M, producing a set of corresponding points for each region. The union of those correspondences across all of the segmented image's regions is then the final set of matches between the two images. Let $\{(p_1, m_1), \ldots, (p_n, m_n)\}$ denote this final set of n total corresponding points.

Scoring the resulting correspondences: Given these correspondences, we want to assign a single match score between the two original images. Note that while each individual region's match is scored by $\mathcal{C}(P, M)$, the final match score is based on the union of the regions' matches, and must both summarize their appearance similarity as well as incorporate the geometric deformations *between* the component region matches. For two images I_1 and I_2, we define this score as a summation of match scores between all their corresponding points:

$$S(I_1, I_2) = \frac{1}{\sqrt{N_1 N_2}} \sum_{i=1}^{n} \omega_i \; s_a(p_i, m_i) \; s_g(p_i, m_i), \quad (4)$$

where N_1 and N_2 are the total number of points in each image, respectively, ω_i is a weight assessing the importance of the i-th matching pair (and will be defined below), and the s_a and s_g functions score each point match's appearance and geometric similarity, respectively. High values of $S(I_1, I_2)$ mean the two images are very similar.

The appearance similarity score is determined by the SIFT distance: $s_a(p_i, m_i) = 1 - A(p_i, m_i)$, where we are simply mapping the cost from Eqn. 2 to a similarity value.

The inter-point geometric consistency score $s_g(p_i, m_i)$ measures the average pairwise deformation between a matching pair (p_i, m_i) and all other matching pairs in the final set:

$$s_g(p_i, m_i) = \frac{1}{n-1} \sum_k \frac{1}{1 + \exp\left(\tau_g \left(\frac{G(p_i, p_k, m_i, m_k)}{\alpha_{ik}} - 1\right)\right)},$$

where $G(\cdot)$ is as defined above, and $k \in \{1, \ldots, n\} \backslash i$. This entire term gives lower similarity values to larger total distortions. The constant τ_g adjusts the shape of the sigmoid function, and α_{ik} weights the pairwise deformation differently according to whether the two points are in the same region. Specifically, if p_i and p_k are in the same region, $\alpha_{ik} = \alpha$; otherwise, $\alpha_{ik} = 2\alpha$, where α is a positive

constant. This doubling of the penalty for within-region matches enforces stronger consistency for pairs within the same region, while allowing more distortions for the matching pairs across different regions.

Finally, the weight ω_i in Eqn. 4 emphasizes the similarity of those correspondence pairs for which the point p_i has fewer initial candidates in the unsegmented image; the intuition is that those matches will be more discriminating. For example, a point in a textureless region will have a large number of SIFT candidate matches in the second image, but many will be less distinctive. Thus, we set $\omega_i = (1 + \exp(\tau_\omega(\frac{|C_i|/N_2}{\mu_\omega} - 1))^{-1}$, where $|C_i|$ denotes the number of initial matching candidate points for point p_i found in the unsegmented image, and N_2 denotes the total number of points in the second (unsegmented) image. The remaining constants simply serve to adjust the shape of the sigmoid, and provide a weight value $\omega_i \in [0.5, 1]$.

3.2. Discussion

We stress two important aspects of our approach. First, rather than consider only region-to-region matches between the two images' segmentations, we take the feature grouping given by each region in one image to find a geometrically consistent match in *any part* of the second image. This way, we can find the best matches whether or not the region in the second image would have happened to appear in a segmentation (see Figure 1).

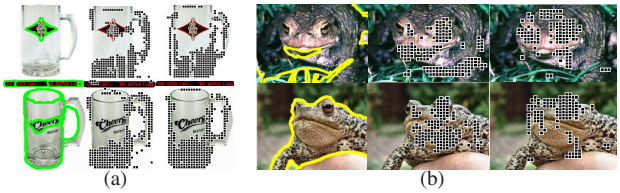
Second, this very idea is what lets us efficiently solve for matching regions using DP, without being susceptible to traditional string matching's sensitivity. Our matches are not string-to-string. Rather, we match a string from one region to a set of candidate points identified by local appearance similarity (see Figure 2). This means we avoid the pitfalls of traditional DP-based matching, namely, sensitivity to the strings' start and end points, and overly strict geometric consistency requirements. The upshot, as we demonstrate in Sec. 4, is that ours is the first image matching method to realize a dense matching with non-parametric geometric constraints—both aspects vital to matching images with generic object categories.

Compared to existing approaches that compute pairwise relations among all points, our approach substantially reduces the computational complexity of enforcing pairwise geometric constraints on the candidate match pairs. This complexity advantage does come at the price of considering geometric relations only between adjacent points on the string when solving for the optimal match. Nonetheless, we purposely mitigate the impact of this simplification by both (a) using two strings per region (column- and row-wise linked), as well as (b) using densely sampled local features and cross-scale matches.

Our current implementation is not fully scale or rotation invariant. While we do employ multiple scales of SIFT descriptors for matching, we sample feature points on a single grid. One could easily add scale invariance by matching across an image pyramid, or adding a multi-scale grid. We retain orientation information (which can be useful for object matching) since the geometric distortion term is computed with respect to 2D coordinates. When using our measure for example-based recognition, we assume a class's pose and scale variation will be represented via the exemplars.

(a) (b)

Figure 3. Example many-to-one and many-to-many matches. In both examples, the left image is segmented, and the right remains unsegmented. Points are color-coded according to their correspondence. **(a)** Different parts of an object (Mandolin's neck and base) are matched to a single instance of the object in the second image. **(b)** Multiple instances of the same category (dice) are matched.



(a) (b)

Figure 4. Examples showing robustness to the choice of which image is segmented. First columns in (a) and (b) show the segmentation of each image, and the remaining two columns show matched points (black dots) when we swap the segmentation used.

Figure 3 shows examples of many-to-one or many-to-many object matching with our method, which illustrates how our asymmetric approach can successfully match images that lack a strict global geometric consistency. This flexibility is useful for matching object categories whose instances have noticeable geometric variation.

A possible concern might be that because of the asymmetry, our matching results may vary depending on which image's segmentation is used for matching. We find in practice it typically produces similar final matching scores, which makes sense given that we only use a segment as a bounding area for imposing geometric constraints, not as a matching primitive. Figure 4 shows two examples illustrating this point. The matching points are not identical when we swap which image is segmented, yet we see that the quality of the assignments is quite similar.

Lastly, we find that SIFT descriptors of the points near the region boundary encode rich information about the shape of the region. To make full use of those near-

boundary points, we dilate a segmented region using a morphological operator, thereby including near-boundary points outside the region for building a string of the region.

4. Results

We test our algorithm on the publicly available Caltech-101 and 256 datasets. Both are among the largest benchmarks available for object category recognition.

Implementation Details: For both datasets, we resize the images such that their longer side is 320 pixels, and then densely sample SIFT descriptors at every eight pixels over four scales, generating about 1200 points per scale for the typical image size (320 x 240). The descriptors are sampled in square patches of sizes 16, 24, 32, and 40 pixels. We use the segmentation method of [21], and the authors' code. The initial candidate matching points are those with a SIFT descriptor distance within $1.25\mu_a$ for each scale, among all scales. We use approximate nearest neighbor search to quickly find close SIFT descriptors, with code by [22].

We set the weight parameters in the cost function by visually inspecting the matches for ten same-class image pairs, mainly to understand the trade-off between the geometric and appearance terms. All were fixed after this initial inspection; we did not attempt to validate them with recognition accuracy. We lock the values for all experiments and all $\sim$15,000 images. The weights we use for the matching cost terms are: $w_d = 4.0$, $w_o = 1.5$, $w_a = 1.25$, $w_g = 1.0$. The remaining constants in Sec. 3.1 serve to massage the sigmoid function outputs into appropriate ranges, and can be found in the supplementary file.[1]

We use a simple near-neighbor classifier to perform exemplar-based category recognition with our matching. To avoid exhaustively matching against each of the exemplars, for each query, we first prune the pool of classes according to SIFT descriptor distances (we use the top 25 and 30 classes for the 256 and 101, respectively). Then, we apply our matching method to search for the top matched exemplars in only those classes. To categorize a query image, we take the sum of our method's matching scores for the k top-scoring exemplars in each category; the top-scoring category is the predicted label. All of the results use $k = 2$, which produced the best performance (for $k = 1, 2, 3$, accuracy varies only by 0.5-1%).

Caltech-101 Dataset: We randomly pick 15 exemplar and test images per category. Table 1 compares our algorithm to other image matching algorithms. Here we focus on those methods that are most closely related: all results listed use a nearest-neighbor scheme based on image-to-image matching scores. Our method clearly outperforms the existing methods by a large margin. In particular, it gives far stronger recognition accuracy than the method of [7], though both algorithms include related pairwise geometric constraints. This suggests that our asymmetric region-to-image matching is more effective for imposing geometric constraints than an image-to-image matching approach when dealing with intra-class variations, and also supports using more densely sampled descriptors (which is more efficiently done in our method).

[1] http://userweb.cs.utexas.edu/~jaechul/

Method	Accuracy (%)
SPM [23]	42.1 $\pm$ 0.81
GBDist [24]	45.2 $\pm$ 0.96
BergMatching [7]	48.0
GBVote [25]	52.0
Ours	61.3

Table 1. Comparison of our method to other image matching algorithms on the Caltech-101, for 15 training images per category. All methods listed here use nearest-neighbor classification based on an image matching score.

Feature	Method	Accuracy (%)
Single	PMK+SVM [26]	50.0
	SVM+kNN [27]	59.1 $\pm$ 0.6
	SPM+SVM [23]	59.3
	GBDist+SVM [24]	59.3 $\pm$ 1.0
	NBNN (1 desc.) [28]	65.0 $\pm$ 1.14
	Ours	61.3
Multiple	SKM [29]	57.3
	KTA [30]	59.8
	LearnDist [31]	63.2
	MKL [32]	70.0
	BoschTree [33]	70.4 $\pm$ 0.7
	NBNN (5 desc.) [28]	72.8 $\pm$ 0.39

Table 2. Comparison of our method to best existing recognition algorithms on the Caltech-101, for 15 exemplar images per class. The table divides the methods into two groups, depending on whether they use a **single** descriptor type or combine **multiple**.

Method	Accuracy (%)
Todorovic-GenLearn [17]	54.0
Todorovic-DiscLearn [18]	72.0
Gu et al. [16]	65.0
Ours	61.3

Table 3. Comparison of our method to region-to-region matching methods on the Caltech-101, for 15 training images per class.

Table 2 compares existing state-of-the-art algorithms, divided into two groups based on whether single (top) or multiple (bottom) descriptor types are used. When comparing to methods using a single local feature type as we do, our method is better than all previous methods, except for the method of [28], which measures an image-to-class distance without explicitly computing individual image-to-image distances. In contrast, our method computes both the similarity between images as well as the matching feature assignment. This can be seen as an advantage, since the lo-

Method	Accuracy (%)
Todorovic-GenLearn [17]	31.5
SPM+SVM [23]	34.1
NBNN (1 desc.) [28]	37.0
NBNN (5 desc.) [28]	42
BoschTree (No ROI) [33]	38.7 ± 1.3
BoschTree (ROI) [33]	43.5 ± 1.1
MKL [32]	45.8
Torodivic-DiscLearn [18]	49.5
Ours	36.3

Table 4. Comparison of our method to existing results on the Caltech-256, for 30 training images per category.

calization functionality is potentially useful for both detection in new images as well as for feature selection among training exemplars. Interestingly, the authors report that their accuracy decreases by 17% when they attempt an explicit image-to-image matching using the same protocol as the one used in their image-to-class matching [28].

When compared to the methods using multiple types of local features (Table 2, bottom), our accuracy using a only single feature is a bit behind the state-of-the-art, which is perhaps expected given the known value of complementary feature types. At the same time, however, we do outperform some multi-feature learning methods [29, 30], and have comparable numbers to the method of [31], which includes a sophisticated learning stage to identify discriminative features. Thus, overall, these results demonstrate that our raw matching accuracy is quite strong.

Table 3 compares our method to those based on region-to-region matching. Numbers reported for other algorithms in Table 3 do not give their "raw" region-to-region matching accuracy, since all incorporate learning algorithms on top of the matching. Without learning, our algorithm outperforms that of [17], suggesting that the image-to-region match can indeed be more robust when matching with imperfect bottom-up segmentations. We obtain close accuracy to [16], though in our case without any discriminative region selection. Compared to [18], our algorithm is about 10 points less accurate. One interesting thing here is that both [17] and [18] rely on the same region-to-region matching algorithm, but the more recent [18] improves over [17] by about 20 points, apparently due to the switch from generative to discriminative learning. Since the matching algorithm is the same in both cases, this suggests that the significant jump may be largely attributable to the powerful learning algorithm, not the matching itself.

Caltech-256 Dataset: For the Caltech-256, we randomly pick 30 exemplar images and 25 test images per category. Table 4 compares our algorithm to existing methods. To our knowledge, we are the first to attempt to use raw image matching to perform recognition on this challenging dataset.

Figure 5. Examples of matching results on the Caltech-256. Leftmost image in each row is a query, following six are nearest exemplars found by our method among 7680 total images. Last two rows show queries whose nearest exemplar has the wrong label, and yet seem to be fairly intuitive errors.

Compared to methods using a single type of local feature, our nearest-neighbor accuracy improves over the SVM-based method used in [23], and gives very similar results to that of [28]. Compared to region-to-region matching-based methods, we achieve better accuracy than the method of [17], which learns generative models for each category based on the matching. Compared to methods using multiple feature types [33, 32], our method lags by only about 2.4-9.5%. Given that our recognition accuracy is based solely on raw image matching with a single feature type, these are very encouraging results.

Finally, for qualitative evaluation we show the nearest neighbors for some image queries (see Figure 5). We selected query images from some of the hard categories for which previous recognition methods produce below average accuracy [23]. Our algorithm shows powerful matching results for those difficult object categories under notable intra-class variations or background clutter. Further, even for the failure cases (as judged by the true category label of the nearest exemplar), we find that the top matched images often show very similar geometric configurations and appearance, meaning the mistakes made are somewhat intuitive (see last two rows).

Computational Cost: Using the dense sampling described above, we get about 4800 features per image. It takes about 5-15 seconds to match two such images using our MATLAB implementation on a 2.5GHz CPU. Computation time varies depending on how many initial matching candidates are obtained via SIFT search. Segmentation using [21] required about 3-5 minutes per image, though the authors report that much faster parallel implementations are possible ($\sim$2 s). When using our matching to compare to exemplars for recognition, note that the segmentation cost is a one-time thing; due to the asymmetry, we only need to have segmented exemplars, and can leave all novel queries unsegmented.

To compare the cost of our method to other matching algorithms, we tested the computation time of the spectral matching algorithm [6], which is known as the most efficient matching algorithm among ones using a pairwise geometric constraint. In our MATLAB implementation, it takes more than 10 minutes to match images of 320 by 240 pixels with dense features. Most of that time was consumed by the pairwise distortion computation; once those are computed, the spectral matching itself runs fast. Another popular and effective matching algorithm ([7]) has been reported in [6] to be limited in practice to running with under 50-100 features, due to the expense of the linear programming step. Thus, the existing most related matching algorithms that enforce geometric constraints do not seem amenable for dense point matching with large datasets.

5. Conclusions

The proposed method offers a novel and efficient way to exploit segmentation for image matching, using an asymmetric region-to-image strategy. Our experiments with two datasets demonstrate its effectiveness in practice. We show that when used within a simple nearest neighbor recognition application, the matching is powerful enough to produce results better than a number of existing related methods, and competitive with the state-of-the-art.

Acknowledgements: This research is supported in part by NSF EIA-0303609, Microsoft Research, the Luce Foundation, and a Fellowship from the ILJU Foundation, Korea. Thanks to Yong Jae Lee and Sung Ju Hwang for interesting discussions, Marius Muja for the FLANN code, the Berkeley group for the segmentation code, and the anonymous reviewers for their helpful comments.

References

[1] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2), 2004.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape Matching and Object Recognition Using Shape Contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(24):509–522, 2002.

[3] V. Ferrari, T. Tuytelaars, and L.B. Gool. Simultaneous Object Recognition and Segmentation from Single or Multiple Model Views. *IJCV*, 67(2):159–188, April 2006.

[4] J. Kannala, E. Rahtu, B. Brandt, and J. Heikkila. Object Recognition and Segmentation by Non-Rigid Quasi-Dense Matching. In *CVPR*, 2008.

[5] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases. In *CVPR*, 2008.

[6] M. Leordeanu and M. Hebert. A Spectral Technique for Correspondence Problems using Pairwise Constraints. In *ICCV*, 2005.

[7] A. Berg, T. Berg, and J. Malik. Shape Matching and Object Recognition Low Distortion Correspondences. In *CVPR*, 2005.

[8] E. Nowak, F. Jurie, and B. Triggs. Sampling Strategies for Bag-of-Features Image Classification. In *ECCV*, 2006.

[9] T. Quack, V. Ferrari, B. Leibe, and L.B. Gool. Efficient Mining of Frequent and Distinctive Feature Configurations. In *ICCV*, 2007.

[10] Y. J. Lee and K. Grauman. Foreground Focus: Unsupervised Learning from Partially Matching Images. *IJCV*, 85(2), May 2009.

[11] J. Sivic and A. Zisserman. Video Data Mining Using Configurations of Viewpoint Invariant Regions. In *CVPR*, 2004.

[12] M. Cho, J. Lee, and K.M. Lee. Feature Correspondence and Deformable Object Matching via Agglomerative Correspondence Clustering. In *ICCV*, 2009.

[13] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. Freeman. Sift Flow: Dense Correspondences across Different Scenes. In *ECCV*, 2008.

[14] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling Features for Large Scale Partial-Duplicate Web Image Search. In *CVPR*, 2009.

[15] Z. Lu and H. Ip. Image Categorization with Spatial Mismatch Kernels. In *CVPR*, 2009.

[16] C. Gu, J. Lim, P. Arbelaez, and J. Malik. Recognition using Regions. In *CVPR*, 2009.

[17] S. Todorovic and N. Ahuja. Extracting Subimages of an Unknown Category from a Set of Images. In *CVPR*, 2006.

[18] S. Todorovic and N. Ahuja. Learning Subcategory Relevances for Category Recognition. In *CVPR*, 2008.

[19] S. Todorovic and N. Ahuja. Region-Based Hierarchical Image Matching. *IJCV*, 78(1):47–66, January 2008.

[20] T. Malisiewicz and A. Efros. Recognition by Association via Learning Per-Example Distances. In *CVPR*, 2008.

[21] P. Arbelaez, M. Marie, C. Fowlkes, and J. Malik. From Contours to Regions: An Empirical Evaluation. In *CVPR*, 2009.

[22] M. Muja and D. Lowe. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In *VISAPP*, 2009.

[23] G. Griffin, A. Holub, and P. Perona. Caltech-256 Object Category Dataset. Technical report, CalTech, 2007.

[24] M. Varma and D. Ray. Learning the Discriminative Power-Invariance Trade-Off. In *ICCV*, 2007.

[25] A. Berg. *Shape Matching and Object Recognition*. PhD thesis, Compute Science Division, Berkeley, 2005.

[26] K. Grauman and T. Darrell. The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features. In *ICCV*, 2005.

[27] H. Zhang, A. Berg, M. Marie, and J. Malik. SVM-KNN: Discriminative Nearest Neighbor Classfication for Visual Category Recognition. In *CVPR*, 2006.

[28] O. Boiman, E. Shechtman, and M. Irani. In Defense of Nearest-Neighbor Based Image Classification. In *CVPR*, 2008.

[29] A. Kumar and C. Sminchisescu. Support Kernel Machines for Object Recognition. In *ICCV*, 2007.

[30] Y. Lin, T. Liu, and C. Fuh. Local Ensemble Kernel Learning for Object Category Recognition. In *CVPR*, 2007.

[31] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning Globally-Consistent Local Distance Functions for Shape-Based Image Retrieval and Classificaiton. In *ICCV*, 2007.

[32] P. Gehler and S. Nowozin. On Feature Combination for Multiclass Object Classification. In *ICCV*, 2009.

[33] A. Bosch, A. Zisserman, and X. Munoz. Image Classification using Random Forests and Ferns. In *ICCV*, 2007.