Building a Dictionary of Image Fragments

Zicheng Liao †Ali Farhadi Yang Wang Ian Endres David Forsyth
University of Illinois at Urbana-Champaign †Carnegie Mellon University
{liao17, yangwang, iendres2, daf}@illinois.edu †afarhadi@cs.cmu.edu

Abstract

We show how to build large dictionaries of meaningful image fragments. These fragments could represent objects, objects in a local context, or parts of scenes. Our fragments operate as region-based exemplars, and we show how they can be used for image classification, to localize objects, and to compose new images. While each of these activities has been demonstrated before, each has required manually extracted fragments.

Because our method for fragment extraction is automatic it can operate at a large scale. Our method uses recent advances in generic object detection techniques, together with discriminative tests to obtain good, clean fragment sets with extensive diversity. Our fragments are organized by the tags of the source images to build a semantically organized fragment table.

A good set of fragment exemplars describes only the object, rather than object+context. Context could help identify an object; but it could also contribute noise, because other objects might appear in the same context. We show a slight improvement in classification performance by two standard exemplar matching methods using our fragment dictionary over such methods using image exemplars. This suggests that knowing the support of an exemplar is valuable. Furthermore, we demonstrate our automatically built fragment dictionary is capable of good localization. Finally, our fragment dictionary supports a keyword based fragment search system, which allows artists to get the fragments they need to make image collages.

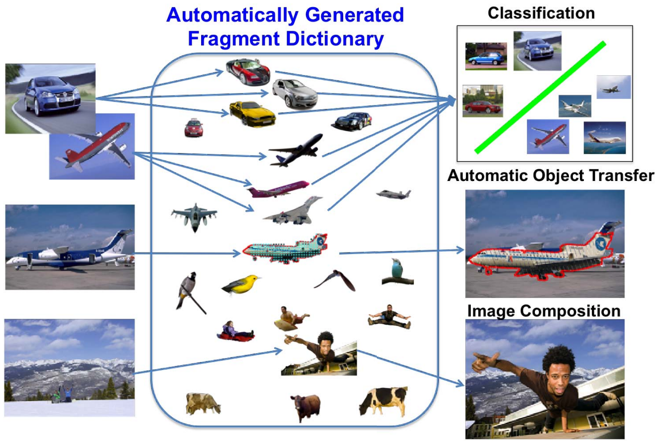
Automatically Generated Fragment Dictionary
Classification
Automatic Object Transfer
Image Composition

Figure 1. System Illustration. Given a fragment dictionary (middle column), we can classify images by matching to the fragment exemplars in the dictionary (top row), automatically transfer a fragment to an object in a test image by matching and alignment (middle row), or inserting fragment mattes to a new image (bottom row).

1. Introduction

Image fragments — regions that could represent a single object (e.g. "dog"), an object in context (e.g. "a person riding a motorbike") or a piece of a scene (e.g. "a piece of wood" or "a playing yard") — form a natural representation of objects. A rich dictionary of such fragments could be used in a variety of important applications. In particular, current natural applications of exemplar or data-driven methods would all likely work better if one could use fragment exemplars. For example, fragments could substitute for exemplar images in an exemplar-based classification method like that of [4], making this strategy viable for detection and for complex images. Exemplar matches would lead to good object localizations (for example, using the methods of [12], but without their need to mark up the exemplar in advance). Finally, a rich dictionary could support building convincing collages (like those of [8], again, without their need to mark up fragments by hand).

The key step is to build a large, rich dictionary of image fragments automatically. Extracting accurate fragments out of images is a challenging task. In this paper we show how to build a large, clean, and meaningful fragment dictionary. Our approach involves four steps. First, we use the method of category independent object proposal [3] to generate a set of fragment proposals from image sets. We then verify the qualities of the generated fragment proposals with a discriminative method. The selected fragments are grouped and indexed by the labels of their source images. After that, we use a clean-up procedure to remove anomalies from the dictionary within each category. Finally, we matte the resulting fragments out of the training images to get the best possible boundaries.

We then show that our fragment dictionary supports classification (section 4.1), localization (section 4.2) and collage building (section 4.3). In classification, we show intriguing evidence that two standard current exemplar matching methods perform slightly better when they use fragment exemplars, most likely because some context information is distracting (for example, both chairs and tables appear in dining rooms). It is well established that an exemplar-based matching method can localize, *if* one knows the location of the object in the exemplar image; for a fragment based method, this condition falls away. Finally, a rich enough fragment dictionary makes collage building easier, because an artist can easily search for fragments by keywords or phrases (e.g. "running dog"), select the desired ones and insert them into a new scene (Fig. 8).

The main contributions of this paper include: 1) a novel and scalable algorithm to build clean fragment dictionary; 2) a verification method to prune out bad fragments automatically; 3) experimental evaluations demonstrating the benefit of fragment-based representation compared with the conventional whole image representation. We plan to release our fragment dictionary. A large scale and clean fragment dictionary furnishes many novel research projects.

2. Related Work

Automated object segmentation is a literature too vast to be surveyed here. The key ideas that we use are (a) that it is useful to work with more than one segmentation of a particular image, then choose good fragments ([13]); (b) that these multiple segmentations can yield estimates of support ([11]); and (c) that it is possible to identify segments that seem to form a single object, without knowing what the object is ([3, 1]).

Exemplar-based image classification is also a literature too vast to be surveyed here. There exist methods for image classification using region-based exemplar matching [5, 7] and image-based exemplar matching [4, 12]. Gu et al. use over-segmented image regions as image cues, and learn discriminative weights for the regions [5]. We offer a similar region-based matching solution for classification except that our fragments are meaningful object level segments. Malisiewicz *et al.* use whole images as exemplars and train a linear SVM classifier for every exemplar [12]. Their method offers promising result in classification; however, to localize they must use manually labeled object masks, while our fragments are automatically constructed.

Collages: It is now well established that one can compose impressive new pictures from image fragments. Lalonde et al. [8] use objects manually segmented by volunteers. Our main contribution is a method to make a larger set of matted fragments with more general coverage available to the artist. Cheng et al. [2] use hand drawn sketches together with object names to query a large image collection. Their system uses image fragments chosen to work together, whereas we simply generate a vocabulary of fragments. Their system uses the sketch drawn by the user as an important part of the fragment extraction process. In contrast, our fragments are general purpose fragments, made in advance of building a collage.

3. Building a Dictionary of Fragments

We assume we have a pool of tagged images, and wish to build a set of fragments for each tag. Notice that if this pool is heavily biased, then the set of fragments will be biased too. For example, careless collecting might result in a pool of near duplicate images for a particular tag; our algorithm cannot resolve this problem. We have built fragment tables out of the top k images from Google/Bing searches, and from Pascal images. In each case, the underlying collection method produces pools of images with quite diverse appearances (one might also use the methods of [10]). Our automatic fragment dictionary construction procedure consists of four major steps (Fig. 2).

Generating fragment proposals: Given a collection of images, the first step of our system is to generate a pool of image segments that are likely to be useful image fragments. Note that this is non-trivial, since we do not know what objects are in the image collection beforehand. We use the method of [3] that uses a generic object detector to identify potentially useful fragments. This method performs multiple segmentations of each image, then uses a learned model to rank the image segments so that the top-ranked image regions are likely to be foreground objects (called *object proposals*).

Fragment verification: Many fragments are not likely to be useful. However, quite simple tests can tell whether the fragment is likely to be reliable (for example, a good fragment tends to look quite different from the background, because otherwise the segment is likely to have gotten the boundary wrong). We train a classifier to determine whether the fragment is good. We quantitatively evaluate the performance of our classifier on three image sets: flickr, Caltech256 and Pascal VOC2010.

Dictionary clean-up: We associate a fragment that passes the verification stage with the label of its source image. It is common that not all of the fragments extracted from a image are directly related to the image label, i.e., a "dog" image may produce a good fragment of a nearby cat. We use an adapted asymmetric matching solution [7] to clean up within-class anomalies.

Matting dictionary fragments: For the final cleaned image fragments, we further produce soft mattes. For each fragment, we generate a trimap using morphological opera-

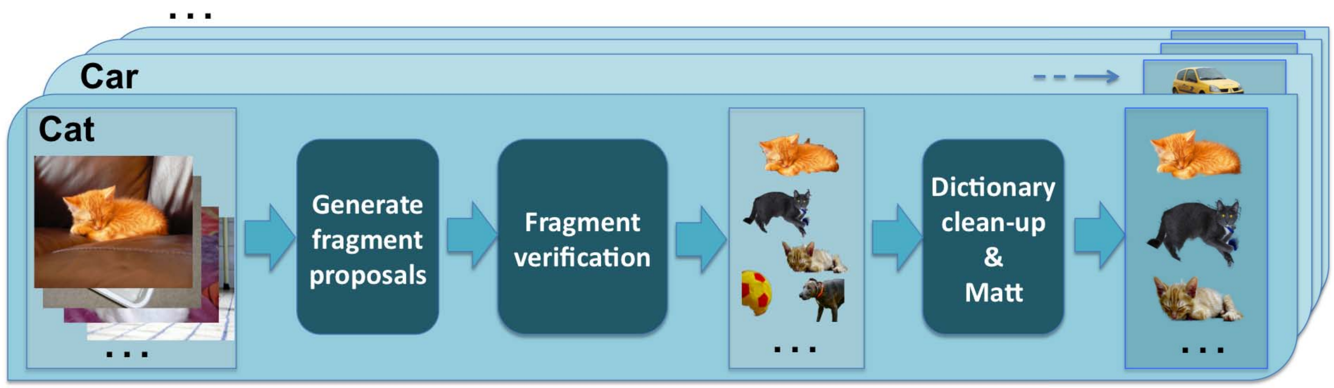

Figure 2. Fragment dictionary construction illustration: For each category, we collect images of that category and run a generic object proposal and fragment verification procedure to generate a set of plausible fragments, followed by a clean-up procedure to remove irrelevant fragments and a standard matting algorithm to produce soft mattes as final result.

tions (dilation and erosion), and use the closed form matting algorithm [9] to matte the fragment out of the image with the trimap.

3.1. Generating Fragment Proposals

We want to generate a pool of candidate regions that are likely to be foreground objects from a collection of images. We use the approach of [3], which involves two major steps.

Proposing regions: This step aims to generate a large and diverse set of proposals that are likely to be object regions. First, the occlusion boundary algorithm in [6] is applied on all the images. For each image, the algorithm in [6] outputs four successively coarser segmentations. It also outputs the probability of occlusion and figure/ground label for each boundary in the segmentation. The method in [3] uses the boundary strength and figure/ground likelihood of each segment as features and performs agglomerative grouping. In the end, we obtain a set of possibly overlapping image segments (called superpixels).

Each image segment is then used as a seed to identify other regions that might belong to the same object. To generate a proposal, we need to infer the foreground/background label over all the superpixels in an image. This is achieved by constructing a CRF that takes into account the affinity of each superpixel to the seed region, and the boundaries between adjacent superpixels.

Ranking proposals: Many of the proposals generated from the first step are not object regions. The next step of [3] is to rank all the proposals in an image so that object regions are ranked higher than non-object regions. The ranking algorithm will also encourage diversity with the goal of discovering all the objects in an image. The method in [3] uses a rank-SVM formulation based on various features computed from proposal regions, such as color, texture, geometric surfaces, boundaries, etc.

This method produces hundreds of object proposals. In practice, we found that it sufficient to use the top five object proposals as candidate fragments.

3.2. Fragment Verification

Not all candidate fragments are good. However, it is often quite obvious when a candidate is bad. For example, the matte may have a problematic spatial support (the whole image; too small a region; and so on). Alternatively, there may be cues that the segmentation process cannot be relied on. For example, the appearance of the candidate fragment may be similar to that of some or all of the background, which is a strong hint that the segmentation boundaries are poor. This suggests applying a classifier to features computed from the candidate fragments to test whether they are reliable. Fragments that do not pass the classifer are discarded.

We use a kernel SVM with an RBF kernel and regularization $c = 2.0$, trained using the *SVM-Light* software package. Training examples are labeled by hand. We use the features described in Table 1.

feature	*description*
1	normalized region size
2	normalized length of boundary intersection
3	entropy of the whole image
4	entropy of the interior region
5	entropy of the exterior region
6	entropy of the border region
7	interior/exterior color histo dist
8-13	top 6 shortest int/ext grid-wise color histo dist

Table 1. Features for fragment classification. For each 8 13, we partition the image domain into grids (10*10) and compute the color histogram distance of every interior/exterior grid pair, and return the top 6 shortest distances.

We demonstrate the effectiveness of our fragment verification procedure by experimenting on the following three image collections, each consisting of thousands of images.

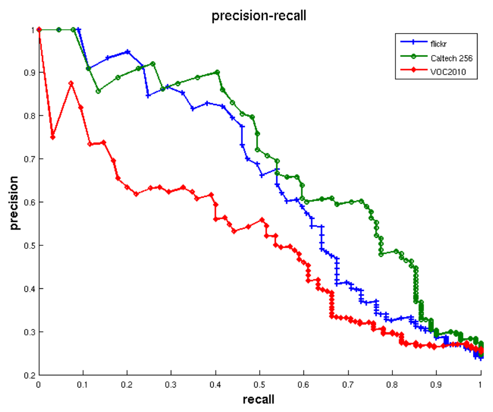

Figure 3. Precision-recall curve for the fragment verification classifier. The classifier is trained jointly with randomly sampled data from the flickr/Caltech256/VOC2010 datasets, and tested on each individual dataset. The training dataset consists of 3200 fragments. For each category, testing set consists of 400 fragments. Note that the classifier performance is dataset dependent. This figure shows that one can achieve highly accurate fragments by being conservative about the recall.

Flickr: These images are downloaded from Flickr. They are mainly about humans, objects, activities, pets, and familiar scenes (indoor and outdoor).

Caltech256: This dataset is widely used for object recognition in the computer vision literature.

PASCAL VOC2010: This is another widely used object recognition dataset. The images are more complex than those in Caltech256.

To quantitatively evaluate our method, we need both training and testing datasets with ground truth labels. It is impossible to obtain a "ground truth" alpha matte for every possible matte of images from the web. However, our objective is to separate good fragments from bad ones. We set up the following criteria for good fragments and use them in the manual labeling process.

- A good fragment can be a whole object, a meaningful component of a larger object (e.g. tire of a car, head of a human being, a flower on a tree, etc), or a scene that consists of part of an scene (a sea shore without the sky, a crowd without background, a piece of woods without the surroundings, or a street background without the foreground objects or road).

- A good fragment's effective size should cover neither too little nor too much of the entire image domain. If a fragment covers too little of the image, it may contain little discriminative information and is unlikely to be useful for image compositing. On the other hand, if a fragment covers too much of the image (e.g. the whole image), its distinction from the whole image is small.

We have labeled 1200 fragment candidates (for 400 images) from each dataset. We train a classifier with data across the three datasets and evaluate the performance on each set's testing data. For testing on a particular set, say, Flickr, we use 2/3 of the labeled Flickr data and all of the labeled data from Caltech256 and VOC2010 as the training data, and test on the remaining 1/3 data from Flickr. Similar procedures are used for testing on Caltech256 and VOC2010 datasets.

Figure 3 shows the precision-recall curves on each dataset. Note that the performance of the classifier varies on different datasets, mainly due to the varying level of matting difficulty in different datasets. For a moderately well-behaved dataset, e.g. Flickr, we get 39% recall at 81% precision. The positive labeling rate for the Flickr dataset is about 24%. By setting the classifier's threshold at 39% recall, we expect to get about 35 positive responses with 28 true positives for every 100 images. For example, by applying the system to a 7663-image flickr dataset, we obtain 2477 positive fragments.

3.3. Dictionary Clean-up

Fragments associated with the same tag (e.g. "cat") in the source images are grouped as "cat" fragments. However, a cat image may produce a good quality dog fragment that passes the verification process. In this step we want to remove such within-class fragment anomalies from the dictionary. We start with a manually selected small set (15 in our experiment) of representative fragment exemplars for each category. For each new incoming fragment, we use an adapted asymmetric region-to-image matching algorithm [7] (see Appendix A) to measure its distance to the fragment set and count the top k (5 in experiment) best matches. We reject it if the score is below a threshold. False positive fragments from the verification step are also effectively removed as a by-product.

While this clean-up algorithm might seem simple, it is quite effective (Figure 4) for the same reason that exemplar based matching is effective: the matching procedure takes care of small deformations, and we expect that any reasonable exemplar is somewhat similar to some other exemplars. For example, the dictionary for "cat" can have quite high diversity because our algorithm only requires that each exemplar be somewhat similar to several other exemplars (by rejecting exemplars that do not have this property). If there are enough lateral (resp. frontal views), then the other lateral (resp. frontal) views are preserved.

3.4. Matting Dictionary Fragments

Finally, we can matte the fragments out of the image. We use the closed form method of [9]. The closed-form matting algorithm simplifies the matting equation with a local window color smoothness assumption, and transforms the

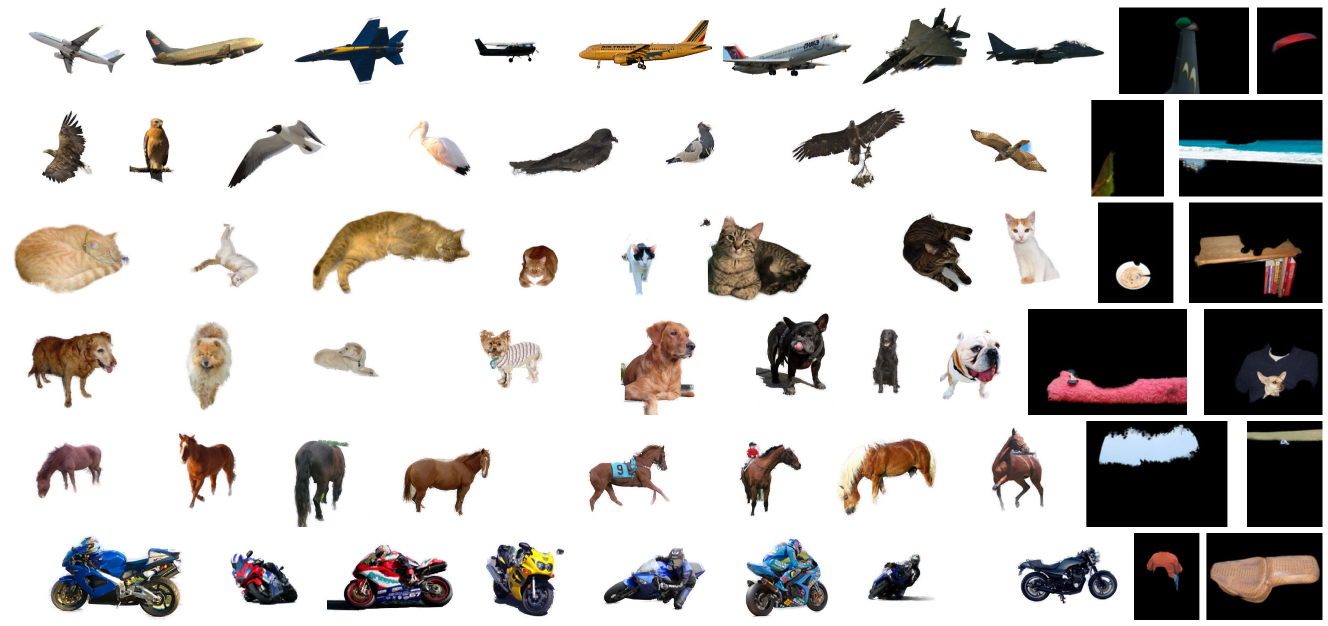

Figure 4. A snapshot of the cleaned fragment dictionary. For each row, the first eight examples are samples from the top ranked fragments in the category; the last two (shown in black background) are samples that are rejected in the dictionary clean-up step. Notice the appearance diversity in the dictionary; for example, the airplane entry contains lateral and three-quarter views, and both civil and military aircraft.

problem into a quadratic optimization problem, which can be solved efficiently via a sparse linear solver. This algorithm requires a user-supplied trimap to mark known foreground and background regions to solve for the alpha values of the unknown region. We replace this with a trimap generated automatically by eroding and dilating the fragment mask, yielding an entirely automatic pipeline.

4. Applications

In this section, we demonstrate the usefulness of our fragment dictionary.

4.1. Image Classification

We wish to know whether fragments make exemplars comparable in performance to images. We evaluate on a standard dataset (Pascal VOC2010), with two standard exemplar based classification methods (the local distance model of [4] and an adapted version of the asymmetric matching method of [7], see algorithm description in Appendix A and B). We build a fragment dictionary from the Pascal VOC2010 dataset. In this case, only 10 categories produce a large enough set of dictionary entries to be usable, because our dictionary collection method emphasizes throwing out training examples that are difficult to decide. Rather than collect other examples to fit the dictionary (and so work on 20 categories, but a non-standard training dataset), we chose to stick with the training dataset (and so work with only the remaining 10 categories), so that comparisons could be more accurate.

We can now evaluate fragments by comparing matching in two cases. The first matches fragment exemplars from an automatically built dictionary; the second matches image exemplars, using the whole image from which each fragment exemplar was taken. As table 2 shows, for each matcher fragments get results comparable to (and slightly better than) whole images. Fragments may have a small advantage because contextual information creates noise, or because our fragment selection procedure will prefer images with high contrast between fragment and background.

4.2. Object Localization

The spatial support of fragments can be used to accurately localize objects in a query image (Figure 5). The asymmetric matching method produces a one-to-one correspondence between the descriptors of the matching pair in the matched exemplar (fragment or image) and the query image. We compute a least-square solution for the affine transformation between corresponding descriptors, and use this to transfer the whole fragment to the query image. Figure 5 shows qualitative results of fragment transfer on both rigid objects (airplane, car, motorbike) and articulated objects (cow). The transferred object provides rich information in localizing the matching object in the query image and allows us to reason about local substructure correspondence that is unavailable from image-based matching methods.

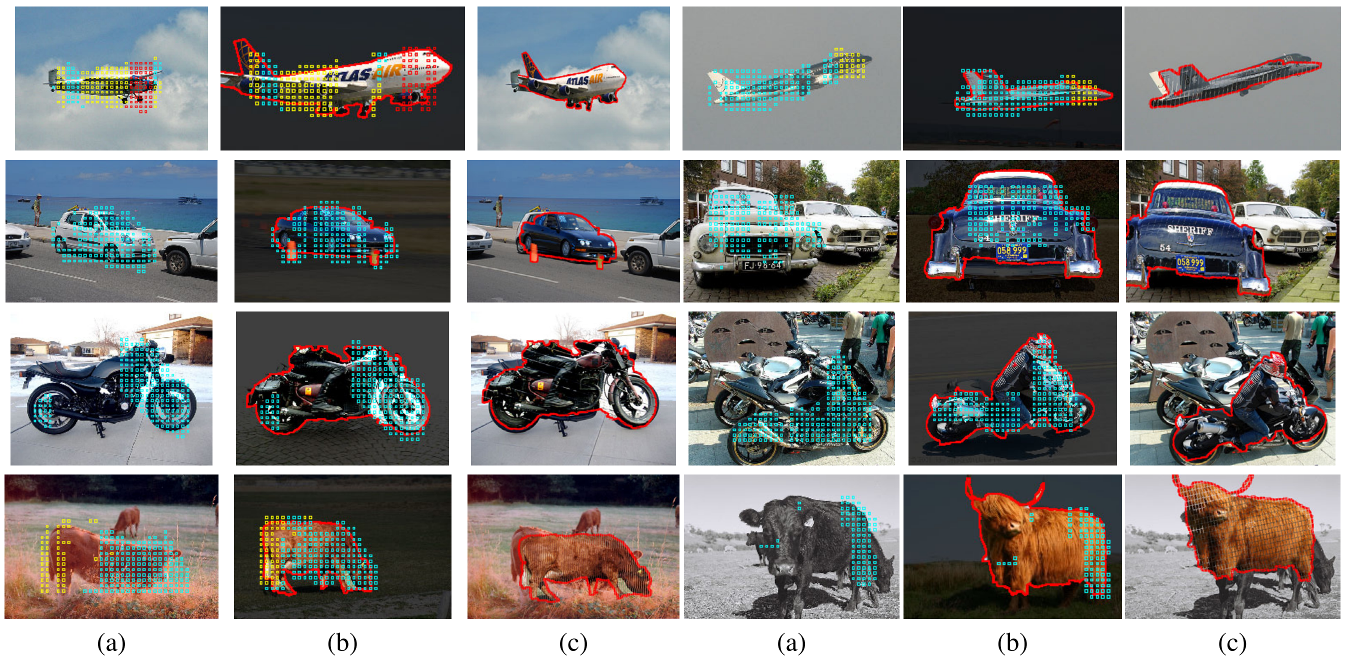

Figure 5. Object transfer by affine estimation. Each row displays two example sets and each set contains three images: (a) query image; (b) matched fragment (boundary marked in red); (c) transformation result. The colored dots are visualization of matched descriptors in the query image and matching fragment; cyan indicates excellent matching, yellow indicates moderately good matching and red indicates weak matching. Note that the affine transformation successfully transfers matched fragments of different locations, scales and orientations to the right regions in the query images. We would like to emphasize that the fragments in (b) are obtained *automatically*. Previous work on object transfer (e.g. [12, 8]) usually require manual segmentation.

	Exemplar	aeroplane	bird	car	cat	cow	dog	horse	motorbike	sheep	tv/monitor	average
I	Image	0.405	0.080	0.507	0.352	0.073	0.071	0.065	0.456	0.515	0.279	0.280
	Frag step1	0.568	0.025	0.371	0.096	0.179	0.166	0.181	0.434	0.116	0.325	0.243
	Frag step2	0.503	0.083	0.451	0.236	0.057	0.042	0.514	0.421	0.188	0.255	0.274
	Frag final	0.642	0.156	0.231	0.404	0.089	0.046	0.259	0.434	0.319	0.292	**0.287**
II	Image	0.387	0.053	0.273	0.047	0.336	0.041	0.169	0.116	0.571	0.364	0.236
	Frag step1	0.675	0.004	0.098	0.027	0.442	0.007	0.203	0.087	0.391	0.006	0.194
	Frag step2	0.490	0.087	0.273	0.533	0.138	0.074	0.060	0.214	0.594	0.625	0.309
	Frag final	0.724	0.102	0.281	0.593	0.363	0.007	0.040	0.218	0.398	0.520	**0.325**

Table 2. Classification accuracy comparison. Method I is the local distance algorithm, method II is the asymmetric region-to-image matching algorithm. For each algorithm, we compare the classification accuracy of the image based model (first row) and fragment based model (row 2-4). 25 exemplars per category are used for the training table. For method I, 100 geometric blur descriptors per exemplar are sampled for training and testing. For method II, SIFT features are sampled uniformly with step size 8 at square patches of widths 6, 8, 10, 12 for each grid point.

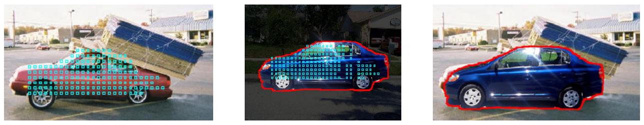

Figure 6. Mapping a good fragment through the affine transform implied by the match gives a good estimate of the likely location of occluded object parts, in this case the back of the car covered by a load. **Left:** the query image; **center**, the fragment matched; and **right**, the matched fragment overlaid on the query image.

Our transfer method is comparable to that of [12]; however, our fragments have been obtained automatically. Notice that our method gets good enough bounding contour estimates that the affine transformation sketches out a good region of support for the object in the matched image. Furthermore, because we have good fragment estimates, we can identify interesting occluded regions through matching (Figure 6).

4.3. Image Composition

Having a system that can automatically produce accurate image fragments makes photo editing as simple as putting these pieces together. We construct a fragment dictionary from a Flickr image collection using the algorithm de-

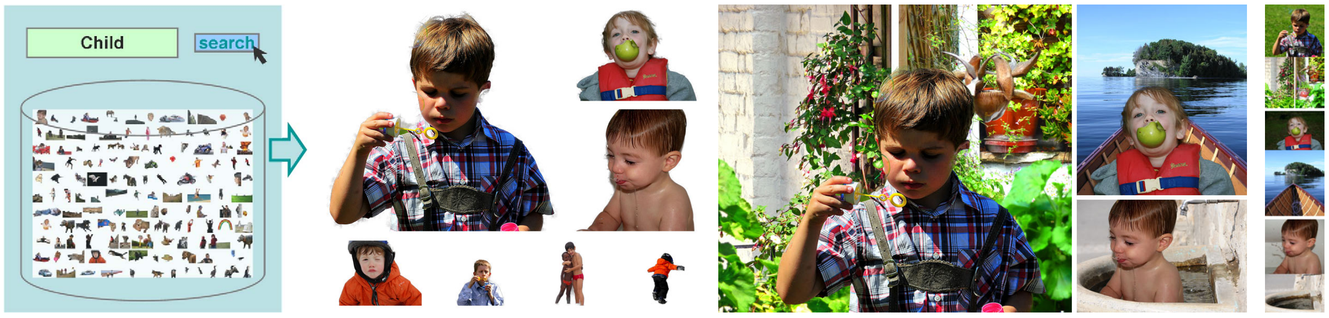

Figure 8. A demonstration of our image composite system: artists can easily search for image fragments they need from our fragment dictionary by keywords, e.g., "child" (Left), then select from a set of relevant results (Middle left). They can then compose new images by using the selected fragments (Middle right). Right: source images of the fragments and backgrounds used in the 3 compositions.

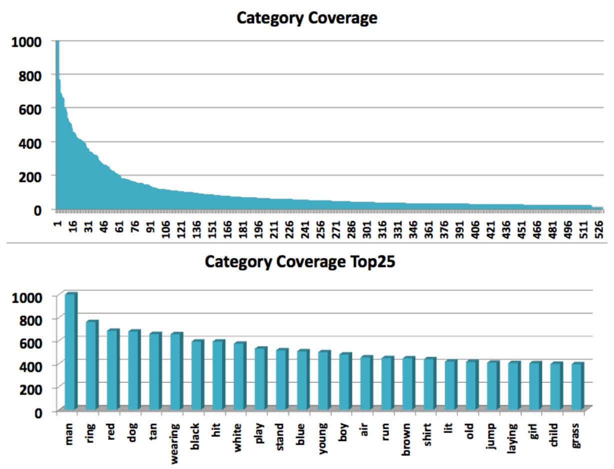

Figure 7. Category coverage histogram: the number of positive mattes returned for the categories on our flickr dataset (minimum number = 10). 531 categories (keywords from image annotations) are generated from the dataset. Upper: the coverage histogram for all categories. Bottom: the coverage histogram for the top 25 categories. X axis are the categories, Y axis are the number of positive responses.

scribed in section 3 (the dictionary clean-up step is skipped due to the large number of categories present in this task). The Flickr images are annotated with textual descriptions, for example, "a dog running in snow". The resultant fragments are then associated with *keywords* in the source image's annotation. A keyword may be an object, a scene, an action or an adjective word. Since the annotation of an image may contain multiple keyword, fragments are also associated with multiple keywords. This allows us to do fragment search not only by a single keyword (i.e. "child"), but also by complex phrases (i.e. "running dog" or "girl on a sunset beach") by taking the intersection of items in each keyword category. See Fig. 8 for an example showing an artist searching "child" fragments from our fragment base and making novel image compositions.

To provide a sense of the coverage of our system, we show the number of fragments we have in our dictionary (Fig. 7). The bottom histogram in Fig. 7 shows the top 25 categories based on the average per class fragment verification scores.

Once a user selects the fragments from our proposals, putting fragments together is straightforward. Since our matting algorithm produces high accuracy fragments, image composition involves using the alpha mattes to blend fragments with a background image. Users may select multiple fragments from the dictionary, rotate, resize and rearrange them to make novel image montages. Figure 8 shows an example of an artist using the fragment dictionary, searching for fragments and creating desired images. Figure 9 shows more examples of such compositions.

5. Conclusions and Future Work

We have introduced a solution of automatically constructing a fragment table from image set by generating fragment proposals, fragment verification, anomaly removal, and matting fragments. A good quality fragment table is useful in many computer graphics and vision applications. We demonstrate its applicability on two sets of applications: fragment-based classification and object localization, and image composition. There are other potential applications we can build upon our fragment table. One is to use the highly localized information of fragment-based matching to do object detection. In our scenario, detection can be done further on substructure of objects given labels of the substructure of the fragments in the table, without having to train a part detector for each individual substructure of every category. Another is to maintain the table size and diversity when we scale up to larger dataset. We would also like to point out that more sophisticated image editing algorithms are needed to resolve lighting or tone inconsistency problem in our image composition application, while in the current version we rely on the artists to select the right fragment for a given background image. We would like to deal with these issues in future work.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1029035, and by the Office of Naval Research under a MURI award.

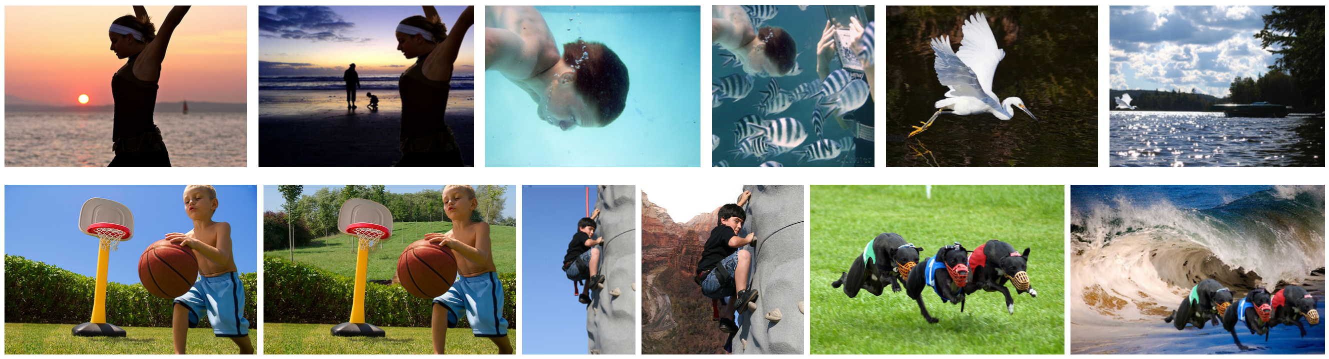
Figure 9. A few more image composition examples. For each pair of examples, the first one is the original image that produces the fragment matte, the second one is the composited image with the fragment inserted.

A. Asymmetric Matching

Our adapted asymmetric region-to-image matcher works as follows: for every pair of exemplars (fragments or images), the matching is to find an optimal matching string pair that minimizes the following cost function using dynamic programming:

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

where $P = \{p_1, p_2, \cdots, p_l\}, M$ are the matching strings of exemplar pair. $G(\cdot)$ and $O(\cdot)$ are the pairwise terms that measure the geometry and order cost, $A(\cdot)$ and $D(\cdot)$ are the unary terms that measure the appearance and displacement cost. A string P is defined by concatenating the 2D uniformly sampled SIFT descriptors in a fragment columnwise and row-wise into a 1D list. Refer to [7] for the definition of each term.

Note that the original asymmetric matching algorithm [7] does segmentation on one image. it then finds a matching for each segmented region, and generate final matching by summarizing the local matches. In this paper, the algorithm is adapted in the way that either the fragment region or the whole image region is used for matching.

B. Local Distance Model

The local distance model [4] uses a large-margin formulation to learn a set of local weights for each training exemplar such that the local distance between images of the same category is small and large otherwise. The local distance from image i to image j is defined as:

$$D_{ji} = \sum_{m=1}^{M} w_{j,m} d_{ji,m} \qquad (1)$$

Where M is the number of local descriptors in image j, $d_{ji,m}$ is the distance of the best matching descriptors in image i to the m-th descriptor in image j, and $w_{j,m}$ is the learned local weight for the m-th descriptor of image i.

References

[1] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010. 2

[2] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu. Sketch2Photo: Internet image montage. *ACM Transactions on Graphics*, 2009. 2

[3] I. Endres and D. Hoiem. Category independent object proposals. In *ECCV*, 2010. 1, 2, 3

[4] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, 2007. 1, 2, 5, 8

[5] C. Gu, J. J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, 2009. 2

[6] D. Hoiem, A. N. Stein, A. A. Efros, and M. Hebert. Recovering occlusion boundaries from a single image. In *ICCV*, 2007. 3

[7] J. Kim and K. Grauman. Asymmetric region-to-image matching for comparing images with generic object categories. In *CVPR*, June 2010. 2, 4, 5, 8

[8] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi. Photo clip art. *ACM Transactions on Graphics*, 2007. 1, 2, 6

[9] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. *PAMI*, 2008. 3, 4

[10] L.-J. Li and L. Fei-Fei. Optimol: Automatic online picture collection via incremental model learning. *IJCV*, 88:147–168, 2010. 2

[11] T. Malisiewicz and A. A. Efros. Improving spatial support for objects via multiple segmentations. In *BMVC*, 2007. 2

[12] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011. 1, 2, 6

[13] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006. 2