Volume 0 (1981), Number 0 pp. 1–11 COMPUTER GRAPHICS *forum*

Data-Driven Automatic Cropping using Semantic Composition Search

A. Samii[1,2] and R. Měch[2] and Z. Lin[2]

[1]University of California, Berkeley
[2]Adobe Systems, Inc.
asamii, rmech, zlin@adobe.com

(a) (b) (c) (d)

Figure 1: Automatic cropping by searching a database of well-composed examples. By searching for similar spatial layouts, we can find exemplar images that have different low-level features but are similarly composed. (1a) A poorly composed photograph to be cropped. (1b) We search a large database of well-composed photographs. Shown are the nearest 30 matches to the input using our semantic descriptor. Similar to the input image, these exemplars all contain vegetation at the bottom of the frame, sky at the top, and many contain water in the center. (1c) A semantically similar but visually different exemplar is found. (1d) The input image is cropped to match the exemplar. The horizon line is in the same location, and the vegetation on the left is a grassy hill rather than a tree. The aspect ratio is set to match the exemplar's.

Abstract

We present a data-driven method for automatically cropping photographs to be well-composed and aesthetically pleasing. Our method matches the composition of an amateur's photograph to an expert's using point correspondences. The correspondences are based on a novel high-level local descriptor we term the "Object Context." Object Context is an extension of Shape Context: it is a descriptor encoding which objects and scene elements surround a given point. By searching a database of expertly-composed images, we can find a crop window which makes an amateur's photograph closely match the composition of a database exemplar. We cull irrelevant matches in the database efficiently using a global descriptor which encodes the objects in the scene. For images with similar content in the database, we efficiently search the space of possible crops using Generalized Hough Voting. When comparing the result of our algorithm to expert crops, our crop windows overlap the expert crops by 83.6%. We also perform a user study which shows that our crops compare favorably to an expert humans' crops.

Categories and Subject Descriptors (according to ACM CCS): I.4.9 [Computer Graphics]: Image Processing and Computer Vision—Applications

Keywords: image editing; photography; composition; cropping

1. Introduction

A photograph's composition can make the difference between an inappreciable image and an unforgettable piece of art. Cropping many photographs can be time-consuming and tedious. We address the problem of automating cropping to make a photograph more aesthetically pleasing. A fully-automatic cropping algorithm can serve as a first pass for a photographer, eliminating much of the work while still giving full control over the aesthetics. For extremely large datasets which cannot be manually cropped, the benefit of a fully-automated algorithm outweighs the cost of errors. This problem is challenging because it requires some notion of the subjective quality of a photograph's composition; for this reason, naive retargeting approaches such as scaling, cropping, black bars, or aesthetic-insensitive retargeting algorithms are not sufficient.

Most previous attempts at solving this problem have directly encoded composition rules and applied these rules to images, but we believe rule-based methods are not general enough to capture the diversity of possible photographs. Other approaches have used low-level features and saliency maps to learn from a training set, but these methods fail to capture higher-level semantics inherent in many photographs.

In order to capture these semantics, we present a new method of cropping based on semantic composition search. Our method is based on finding correspondences between a local descriptor we term the "Object Context," which describes the spatial arrangement of objects around a point based on a semantic labeling. This is motivated by our definition of composition: the spatial relationship between elements in a scene. Object Context is an extension of Shape Context [BMP02], which describes the arrangement of points in a binary line drawing. Using a database of well-composed exemplars, we can pick the crop with the best matching score to one or more exemplars using our composition distance function.

1.1. Method Overview

Our contributions are fourfold: the Object Context image descriptor, a composition distance function based on the descriptors, an efficient method for searching the space of possible crops and minimizing the composition distance, and an efficient database culling scheme. These contributions differ from prior data-driven methods in that (1) we use a higher-level descriptor which models composition semantically, and (2) we do not need to evaluate a large space of possible crops to find the best one. The cropping pipeline consists of searching the database for similar images, matching the composition of the input image to each of these exemplars by minimizing composition distance, and outputting the optimal crop/exemplar pair.

2. Related Work

2.1. Retargeting and Thumbnailing

Retargeting an image for different aspect ratios and sizes has received attention in several papers. The goal of these methods is to *summarize* the important data in the image without introducing artifacts. Aesthetics and composition are not accounted for.

Suh *et. al.* [SLBJ03] automatically crop an image to maximize the important regions in a thumbnail. Their goal is to make the thumbnail as recognizable as possible so a human can quickly find the desired fullsize image. There have been several other methods with similar goals [RSA09, WTSL08, SCSI08, AS07, STR*05, SS09].

Rubinstein *et. al.* [RGSS10] have analyzed several other image retargeting methods which aim to resize an image while minimizing distortion and missing content. Their study shows that combining several retargeting operators (such as warping, Seam Carving, and cropping) is often necessary to avoid visual artifacts. They also show that users often prefer just the cropping operator because it creates the least visual artifacts, which has been verified again more recently by Panozzo *et. al.* [PWS12]. A survey article by Vaquero *et. al.* [VTP*10] provides a summary of other retargeting techniques.

2.2. Optimizing Composition

There are many composition rules which can make a well-composed photograph. Liu *et. al.* [LCWCO10] rank a photograph based on how well they follow the Rule of Thirds, Diagonal Dominance, Visual Balance, and Region Size. Several other methods also encode different composition rules [Ban04, YSQ*12].

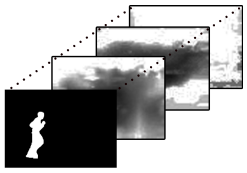
Santella *et. al.* [SAD*06] crop photographs based on eye-tracking data, zooming in on important regions. They are able to place important regions according to the Rule of Thirds, but found that it did not improve the quality of their crops.

Zhang *et. al.* [ZZS*05] use predetermined templates to determine where faces in photographs should be placed, and crop the image to match the templates. The method is limited to face images which have precomputed templates.

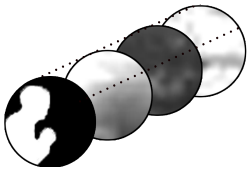
She *et. al.*use a data-driven method to encode well-composed saliency maps [SWS11]. They crop images to best match a saliency map in their database. However, a saliency map is unable to distinguish two overlapping objects with similar saliency.

Zhang *et. al.* [ZWH13] and Bhattacharya *et. al.* [BSS11] rely on segmentation algorithms to optimize the location of objects. While these methods are more flexible, they can suffer from artifacts such as incorrect lighting on moved objects.

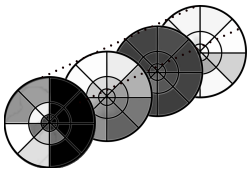
(a) We want to describe the spatial layout at the point indicated by the red circle.



(b) For a given input image, we run several object detectors.



(c) We look at a local region around the point in each object detection probability map.



(d) We compute log-polar histograms of the probability maps and concatenate the values of each bin.

Figure 2: Simplified steps for computing the Object Context.

Other works combine low-level features using a database of well-cropped images to determine which regions belong inside the crop window [AM11, NOSS09]. Neither of these methods take into account the composition: they do not discriminate between *where* in the frame each region belongs. Ahn and Mehta [AM11] compute several low-level feature vectors at each pixel and use a Conditional Random Field to determine which pixels should be inside the crop region. The final crop window is the smallest window which contains 90% of these pixels. Similarly, Nishiyama *et. al.* [NOSS09] combine local features into a large feature vector and use a classifier to rate the crop, then search for the optimal composition. We believe our Object Context descriptor is complementary to these methods; it will add higher-level semantic information to the classifiers.

3. Modeling Composition with Object Context

To describe the composition in an image, we first describe the composition of a point using the Object Context. The Object Context is a local descriptor of the semantic object labels in a circular region around a pixel. Figure 2 summarizes the algorithm; we describe it in detail below.

Given an image, run n object detectors which provide dense, per-pixel labels to get n probability heatmaps H_i, $i \in [1,n]$. Each pixel j in H_i is the probability that $H_i(j)$ contains object i.

The Object Context of a pixel is a log-polar histogram around that pixel. Each bin is an n-dimensional vector (one for each object heatmap). Thus, if we have p polar bins, and d log-distance bins, our descriptor size is $p \cdot d \cdot n$. Figure 2d visualizes this histogram.

Computing the Object Context naively on many pixels would be prohibitively slow. Since our bins are log-polar and not rectangular, we cannot use integral images. We present two solutions: an exact method for a small number of detected objects, and an approximate method if there are many objects.

3.1. Exact Computation using Convolution

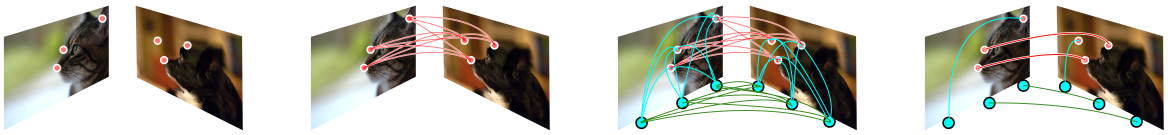
With a small number of object detectors (small n), we convolve each image pd times, where the kernel corresponding to each bin is 1 for any pixel location that falls into the bin, and 0 elsewhere. After each convolution,the Object Context can be computed in $O(pd)$, whereas the naive solution would require us to sum over $O(2^d)$ pixels. Therefore, if we want to compute the Object Context for m pixels, and our image contains S pixels, our runtime improves from $O(m2^d)$ to $O(pd\,(S\log S)+mpd)$. The $S\log S$ term is the runtime of computing the Discrete Fourier Transform (DFT) to transform the image into frequency domain, performing the convolution, and transforming back into the spatial domain. The DFT of the image is cached and re-used for each convolution, and the DFTs of each kernel are precomputed.

3.2. Approximate Computation

For large n, the convolution can become prohibitively slow. In this case, we first compute the top k labels per pixel, then compute an approximate log-polar histogram by only using these k labels. Although this only allows k objects per pixel, it does not place practical constraints on the total number of objects in the scene.

4. Computing the Composition Distance

To evaluate how similar two photographs are in terms of their composition, we define a new distance function called the Composition Distance. We find a correspondence between the Object Contexts in two photographs, and use the cost of this correspondence as our distance function. Each photograph can be viewed as one set in a bipartite graph. By sampling points from the photographs and creating a cost between pairs of points in the two sets, we can solve an instance of the Bipartite Graph Matching problem. Our algorithm is closely related to the Corresponding Boundary Maps algorithm [MFM04], which was used to compare the boundaries drawn by a human to those generated by a machine. The algorithm follows these steps:



(a) Select sample points in each image. Each point is a node in a bipartite graph. We visualize just a few.

(b) Compute the graph edge costs by differencing the Object Context descriptors between points.

(c) Create outlier nodes. Real-to-outlier edges are in blue. Outlier-to-outlier edges are in green.

(d) Solve the assignment problem for the minimum-cost graph.

Figure 3: The algorithm, as described in Section 4

1. **Object and Saliency Detection** We run several object detectors on the image (see Implementation Section 7.1 for details). We also generate a saliency map using the method of Margolin *et. al.* [MTZM13] (see Figure 2c).
2. **Compute Sample Points** We take the maximums of the derivatives of the object-probability heatmaps: $\text{argmax}_j \sum_i H_i'(j)$, and weight it by a smoothed saliency map. The saliency map must be smoothed so points near salient regions have high weight. (See Figure 3a.)
3. **Describe**: We compute the Object Context descriptor for each sampled point (see Section 3).
4. **Costs**: For every point in the first image, we compute the cost for matching it to every point in the second image. These are the weighted arcs of a bipartite graph. The cost function is the sum of square differences of their Object Context descriptors. In our implementation, we include all pairs of points between the two images; for larger images or more sample points, this can be limited to only allow nearby points. (See Figure 3b.)
5. **Outliers**: Given a bipartite graph with n_1 and n_2 nodes in the first and second set, respectively, add outlier nodes to each set such that the total number of nodes is $n_1 + n_2$. Arcs between two outlier nodes have no cost. Arcs between a real node and an outlier node have a cost proportional to the saliency of the real point. This ensures that highly salient regions have more precision in the matching. (See Figure 3c.)
6. **Solve**: Given the bipartite graph, we solve for the minimum-cost graph using a sparse assignment problem solver using Golberg's CSA package [CMT88]. (See Figure 3d.)

The Composition Distance is the average cost of the correspondences.

5. Minimizing Composition Distance

Given an input and an exemplar photograph, we want the input's composition to match the exemplar's. A brute force method would attempt every possible crop select the one with the least composition distance to the exemplar using the procedure in Section 4.

A generalized Hough voting optimization can handle a multimodal distribution of good crops, but it must be solved discretely and sampled at different crop window sizes [SLB*12, LLS04, Bal87]. A least-squares solution would find a weighted mean of a multimodal distribution, rather than a local optimum. We have found that one photograph can have several acceptable crops, resulting in a multimodal distribution for many photographs. We therefore perform an iteration of Hough voting to get near one local optimum, then an iteration of least-squares to get to the exact solution.

We enforce several constraints related to the saliency. A crop may not: (1) decrease the average saliency of the image, (2) intersect a highly salient region, or (3) remove a highly salient region. This prevents the algorithm from ignoring the foreground and finding a well-composed image in the background, and from cutting off important objects by placing them on the image boundary.

5.1. Scale-Invariant Object Context

The matching algorithm needs to use scale-invariant features to converge quickly. For example, if the optimal crop is half the width of the original image, then the Object Context will capture a region twice as large as it would in the final crop, creating very different descriptors between the input and exemplar points. We therefore compute a scale-invariant Object Context by only using the polar binning, where each bin sums over the entire region between two angles. Scale invariance is only used for the matching procedure; the cost function (Section 4) is scale-dependent. While this still allows objects outside of the crop window to influence the descriptor, it is invariant to the scale of the image.

5.2. Generalized Hough Voting

We generate a voting map between points in the input and exemplar. Using the differences between a scale-invariant

Object Context descriptor in the input and exemplar image, we construct a voting map of high-quality crop windows.

We use the Generalized Hough Voting algorithm as follows: between a point i in the input photograph to be cropped and j in the exemplar, denote the cost as $d(i,j) = ||OC(i) - OC(j)||$, where $OC(\cdot)$ is the Object Context descriptor. We then take a weighted vote for pixel i having the same location as pixel j when cropped with weight $d(i,j)$. We use the weights of all pairs of points, not just the corresponding points found in Section 4.

Let the location of point j be a continuous value in $[0,1]$: $(u_j, v_j) = (\frac{x_j}{w_j}, \frac{y_j}{h_j})$, where (x_j, y_j) is the discrete coordinate of the point, and w_j, h_j is the width and height of the exemplar. We then add a weighted vote for $u'_i = u_j$, where the prime indicates the coordinates in the cropped image:

$$\frac{x_i - c_l}{c_r - c_l} = \frac{x_j}{w_j}$$

where c_r and c_l are the x-coordinates of the right and left edge of the crop window, respectively. Given a fixed aspect ratio equal to that of the exemplar, we can now solve for c_l and in terms of the crop width, $c_r - c_l$:

$$c_l = -\frac{x_j}{w_j}(c_r - c_l) - x_i. \quad (1)$$

For each candidate width, we solve for c_r and c_l and add a vote with weight $d(i,j)^{-1}$. The procedure is similar for the vertical axis (with a fixed aspect ratio). We use this algorithm to construct several voting maps at various scales (see Section 7.3), where each location (x,y) in the voting map represents the sum of votes for location (x,y) being the center of the crop window at the given scale. To reduce noise, we smooth each voting map with a two dimensional Gaussian filter with a variance of two pixels.

We take the maximum of each smoothed voting map as candidate crop windows. For each of these candidate crops, we compute the composition distance to the exemplar.

The candidate crops produced by Hough Voting forms a tree where each branch is a proposed crop for its parent's node. We search this tree in an A* fashion, setting the cost of each node to distance between its parent node and the exemplar. We only follow branches which are sufficiently dissimilar to any other attempted crop. In our implementation, "sufficiently similar" is a crop with equal width and with a crop center within 2% of the image width. The least-squares solution will then provide higher precision.

5.3. Least-Squares Solution

We find the scale-invariant correspondences between points in each image, then solve for the crop window that most nearly places the corresponding points in the same location. Whereas in the Hough voting step we used all points in the images, here we only use the matching correspondences. We solve a problem of the form $Ax = b$, where A is a $2N$ by 4 matrix, and N is the number of correspondences. The rows in b correspond to c_r, c_l, c_b, c_t (the right, left, bottom, and top coordinates of the crop window).

Each pair of correspondences gives two rows of the system of equations:

$$\begin{pmatrix} u_j & (1-u_j) & 0 & 0 \\ 0 & 0 & v_j & (1-v_j) \end{pmatrix} \cdot \begin{pmatrix} c_r \\ c_l \\ c_b \\ c_t \end{pmatrix} = \begin{pmatrix} u_i \\ v_i \end{pmatrix}$$

This is a very simplified estimation of an affine transformation: it prevents rotation, reflection, and scaling, all of which are undesirable. To preserve the aspect ratio, we find the enclosing bounding box around these four coordinates with the correct aspect ratio. While it is possible to preserve aspect ratio in the system of equations by using only three free variables, we have found that this produces a less stable system.

6. Retrieving Exemplars from a Database

To find relevant exemplars for a given input image, we use a database of well-composed exemplars and look for semantically similar matches. If we crop our input image to have a small distance to a well-composed exemplar, then our cropped image will also be well-composed.

A naive algorithm would operate as follows. We would first minimize the composition distance between the input image and all images in the database, resulting in a set of suggested crops, one per database image. Then we compute the scale dependent Object Context distance between each crop and its corresponding database image. We would then pick one or more crops with the smallest composition distance. The run time of this algorithm for a large database would be too high, so we must select a useful subset of the database first.

6.1. Database Culling

To use a large database of images efficiently. We need a way of culling the database and ignoring most of the images. We need the resulting set of images to have similar detected objects so we can find matching Object Contexts. We therefore create a global descriptor that is comprised of three parts:

1. A normalized histogram of object labels, where each bin i is the percent of the image that contains the ith detected object.
2. A color histogram in HSV space with nine bins per channel
3. A 320-dimensional gist descriptor

When comparing two descriptors, we take the normalized sum of square differences of each of the three parts individually, then sum these three values with equal weights.

We precompute each of these feature vectors and find the top nearest neighbors as candidates. For over 20,000 exemplars, our database of filename-descriptor pairs can be stored in a single 47mb compressed file.

Note that after finding the optimal crop, the global descriptor will change- it will no longer contain information about parts of the image that have been cropped out. Using this new descriptor, it is theoretically possible that there are better matches in the database. In practice, however, the descriptor is robust to these changes and the match results are similar.

For each nearby candidates, we then perform the matching procedure (Section 5) and pick one or more matches with the least distance as the most aesthetic crop. Alternatively, we could use all of the exemplars and cluster them based on crop window coordinates, with each crop window weighted by the inverse of the composition distance to that exemplar. Using a weighted k-means clustering algorithm, the cluster center with the highest weight is chosen as the crop window. This produces results less sensitive to errors in the database of exemplars. The results shown in the supplementary material use this method (with $k = 2$).

Finally, since the matching procedure only requires the Object Context (both scale-dependent and scale-invariant) and saliency value for each chosen sample point, we precompute and cache these two values for each sample point in the database.

7. Implementation Details

We enumerate the parameter and data choices; these have been only empirically evaluated and may not necessarily be the optimal choices.

7.1. Computing the Object Context

We use the Finding Things model trained on the LabelMe+SUN dataset to obtain 232 object detections [TL13]. Their method predicts a single label per pixel by training one-vs-all Support Vector Machines (SVMs), then smoothing using a Markov Random Field (MRF). Since we want multiple labels per pixel, we omit the SVM and MRF and keep all 232 probabilities per pixel (most of which are zero). Since we omitted the noise suppression inherent in the SVM and MRF, we emulate it by multiplying each detected object by both the published accuracy of the detector and the frequency of occurrence in the LabelMe dataset. Though this noise suppression is ad-hoc, we have found it works well.

We then include face and eye labels detected using Haar Cascades [VJ01]. The saliency map is also treated as a detected object as a fallback for undetected scene elements. In total, this results in 235 object probability heatmaps per image: 232 labels from the Finding Things model, the face labels, the eye labels, and the saliency map. Since we have many detected objects, we use the approximate method described in Section 3.2 with $k = 5$ labels per pixel. The saliency map is always included in the Object Context.

For efficiency, all images are set to be 400 pixels wide.

For the Object Context, we bin radially every 30 degrees (for a total of $p = 12$ bins). The distance bins are computed by multiplying the Euclidean distance by a value $\alpha < 1$ so each bin captures a larger radius of pixels. We use $d = 3$ distance bins and $\alpha = 0.4$, so our object context captures the information in a circular region with a radius of 10% of the image width (20 pixels). Using larger values of p and d will increase the Object Context's specificity, which will be useful for more accurate object labels. Using smaller values will make the descriptor more general, allowing larger errors in the object labeling.

7.2. Composition Distance

We restrict all computed sample points to be at least 10 pixels apart. The cost of an arc between a real node and an outlier node is $.02\sigma$, where $\sigma \in [0, 100]$ is the saliency value of that pixel. Note that the maximum possible arc cost between two real nodes is one (the normalized sum of square differences of the Object Context).

7.3. Minimizing Distance and Retrieving Exemplars

Generalized Hough Voting must sample at discrete widths. At each iteration, we sample 50%, 75%, and 95% of the current image width. Finer sampling is not necessary since the least-squares solution will move closer to the optimal width in a single step. When navigating the A* search tree, we allow a maximum of six evaluations of the composition distance.

We perform the matching procedure on the nearest ten exemplars found in the database. We take the smallest distance as the solution. On a four core 3.3GHz machine, each photograph takes approximately one minute to get through the searching pipeline after the saliency and object detectors have been run. The saliency and object detectors we use take approximately an extra two minutes to run, so the total processing time is about three minutes.

8. Results

Our dataset is comprised of over 20,000 images. About 85% of the images in our database are downloaded from top rated images on Photo.net and dpchallenge.com. The remaining images are pulled from flickr.com to fill any missing content in database. We downloaded the first results for each of the following tags: beautiful, beach, city, composition, landscape, family, ocean, portrait, skyline, or travel, and we asked an affiliated, non-author expert photographer to remove poorly composed images. We compare our algorithm empirically against expert crops and other algorithms

	Experts	Ours	No Sem	Suh
Experts	83.8%	83.6 %	77.1%	78.2%
Turkers	81.2%	83.0 %	77.8%	74.8%

Table 1: Comparison of average crop overlaps with (top) expert photographers and (bottom) Turk workers. Columns correspond to expert crops, our predicted crops, our crops without semantic information, and the thumbnail crop from Suh *et. al.* [SLBJ03].

by computing overlapping crop regions and gathering votes from workers on Amazon Mechanical Turk [mtu].

For subjective evaluation, we also present a comparison of images with improved composition in Figure 9, which shows a comparison sequence between our algorithm and two other approaches [SLBJ03, LCWCO10]. The input images are from Liu *et. al.*'s published results. The disadvantage of rule based methods such as Liu's is that they can only capture a defined set of rules and thus are not flexible enough to capture different compositions.

In the supplementary material, we show more results for which Turk users agreed that our crop was more aesthetically pleasing than the crop from both Suh *et. al.*and an expert's crop.

8.1. Overlap Comparison

For our first evaluation, we compare the similarity of our crops to an expert crop. Each photograph can have several acceptable crops, so we asked four expert photographers to crop a set of 103 poorly composed photographs. We also asked ten Turk users to crop the same set of images. We calculate the amount each crop window overlaps with the nearest expert crop. A graph comparing our results to the experts is shown in Figure 4, and a table of the average overlaps to the expert crops (Row 1) and the Turker crops (Row 2) in Table 1; we discuss in detail below.

The baseline compares the average overlap between the crop of one expert photographer and the other three experts. For a fair evaluation, we then compare our method with three artists at a time, for all four combinations of three artists, and take the average of these four values. In the "no semantics" comparison, we replace the Object Context descriptor with a SIFT descriptor [Low99], choose sample points using SIFT Keypoints [Low04], and only use GIST and color histograms when culling the database. Finally, we test against the algorithm of Suh *et. al.* [SLBJ03].

These results show that the variance in our crops is similar to the variance between expert crops. While our framework without semantics is as effective as Suh's results, neither of these methods perform as well our framework with semantics.

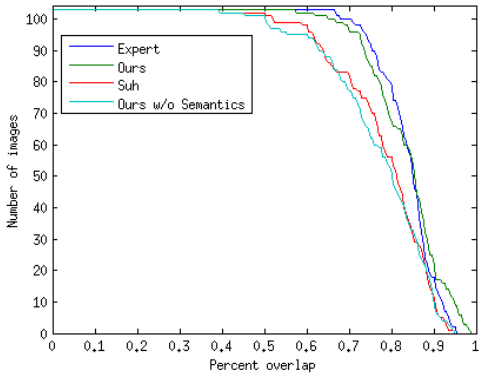


Figure 4: A comparison of crop overlaps with expert artists. The x-axis is the percent of overlapping crop windows. The y-axis is the number of images with at least that amount of overlap. (Blue) expert crops overlap with each other. (Green) our crops vs. experts'. (Red) our algorithm without semantic information. (Turquoise) Suh *et. al.*' crops.

8.2. User Study

We conduct a user studies on Amazon Mechanical Turk using the same dataset of poorly composed photographs. We present users with an image and three crops: our result, the result from Suh *et. al.* [SLBJ03], and one of expert photographers' crops (chosen randomly for each image). The order of the three images are presented randomly; each image is presented to 5 users. They are asked to pick one or more photographs which are most aesthetically pleasing.

Turkers selected experts crops 246 times, our crops 220 times, and Suh's 184 times. When ensuring consistency and agreement by counting a "vote" only when three or more users agree on the same result, experts have 36 votes, we have 51, and Suh has 33. We do not claim to have reached the quality of expert crops, as our failure cases are more obvious than a human's, but these results show that our good crops are comparable with an expert's.

When comparing our results to the best published results of Liu *et. al.* [LCWCO10], Turkers tend to favor their method. We presented three images: Liu's result, our result, and Suh's result. When counting votes as described in the previous paragraph, Liu had 28 votes, we had 18, and Suh had 11. We did not have access to Liu's method so we could not fairly compare it to arbitrary test images. Note that Liu's results also had manually-corrected saliency maps, whereas ours did not.

8.3. Advantages and Disadvantages

For further insight, we analyze the benefits of a data-driven method over rule-based, and show where data-driven meth-

ods can fail. The results shown here exemplify the behavior of the system.

Figure 5: (Left) original image; (Right) a poorly cropped image, due to the lack of information about gaze direction. The crop would be acceptable if the viewer were looking straight ahead, but a well-composed image should have the viewers gaze look across the frame.

Higher-Level Semantics The current data-driven approach cannot encode composition rules that have higher-level semantics such as gaze direction or depth information. For example, Figure 5 shows poorly composed image because the subject is gazing out of the frame rather than across it [Gar11]. These issues are shared by both data-driven and rule-based methods, but is easier to amend in the data-driven method: simply add another "object" which encodes depth, gaze direction, and any other desired semantic information, and the Object Context descriptor will handle it gracefully.

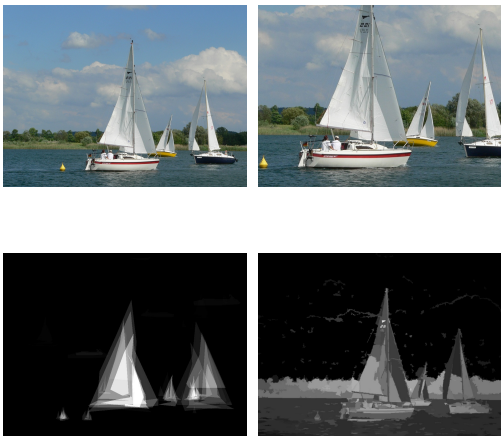


Figure 6: (Top left) original image; (Top right) a poorly-cropped image where the top of the boat is cut, due to missed regions in the saliency maps and object detector. (Bottom left) The probability of each pixel being a part of a boat- notice the top of the boat, where the crop window cuts, has low confidence. (Bottom right) The computed saliency map - notice that the top of the boat has relatively low saliency.

Incorrect Saliency Maps or Object Labels Figure 6 shows a failure case where the foreground object was cut by the crop window. This occurs when the saliency map fails to detect the object as important, and the object detector misses part or all of the object. In the rule-based method of Liu *et. al.*, these failure cases are avoided by manually fixing the saliency map. In a data-driven method, it is impractical to manually correct the entire database. By using both object detectors and a saliency map, we have a level of redundancy. When both fail, the cropping result is unlikely to be aesthetically pleasing.

Figure 7: (Left) original image; (Middle) a well-cropped image following a very specific composition rule of horizon placement: when there is reflective water, it is acceptable to place the horizon line in the center. (Right) The saliency map successfully detects the reflection in the water as salient.

Implicit Composition Rules Placing the horizon at the center of the frame is often considered to be poorly composed, but is considered acceptable when capturing reflective water [pla14]. While this rule is incredibly specific, the combination of our retrieval results and the object context is able to capture it despite our lack of a "reflective water" detector. Figure 7 shows an example where the horizon is placed in the center of the image, with the sunset just above it. In this case, the saliency map detected the reflected water as salient in both this image and the corresponding exemplar in the database.

Figure 8: (Left) original image; (Right) a well-cropped image following a very specific composition rule of leading lines: prominent lines, including curved lines, should converge in the center of the frame. The crop window is very conservative: it places the convergence point in the center of the frame without excessive cropping.

Another composition rule is to use converging prominent lines in the center of the frame to give the viewer "a sense of infinity" [lea14]. Our data-driven method naturally extends

this idea to curved lines based on the database's images. Figure 8 shows the convergence point of the curved lines being placed directly in the center of the image. A rule-based method would need to explicitly detect curves to encode this rule, rather than just straight lines.

9. Conclusion, Limitations, and Future Work

We have presented a novel composition descriptor and constructed a composition distance method based on correspondences of these descriptors in two images. The Object Context can be either weakly scale-dependent and completely scale-independent. We minimize the composition distance to match the composition between an input and exemplar photograph. We store a large database efficiently and can quickly cull it to find potential matches. We minimize the composition distance to each candidate and provide one or more crop recommendations.

Our method is complementary to current data-driven methods which use low-level cues such as intensity and texture statistics. It can also improve composition rating algorithms which explicitly encode rules such as diagonal dominance, visual balance, and the rule of thirds. Current algorithms are not able to efficiently search the parameter space of crops, resorting to brute-force searches [SAD*06] or Particle Swarm Optimizations [LCWCO10]. Our minimization algorithm can be used to speed up naive crop window searches.

As with all data-driven methods, we are limited by the images in our database. If we encounter a vastly different image, we may not handle it as gracefully as a rule-based method, and fail completely. However, the composition distances are large in these cases, so we know that we have produced a poor result.

The inaccuracy of object and saliency detectors will cause errors in our results. We attempt to mitigate these errors through redundancy, using several different detectors on each image. It is impractical to manually correct detection errors in the database; however, it may be beneficial to correct errors in the test images. Our results will improve as these computer vision algorithms become more accurate, and as the number of photographs in the database increases.

While we have stated our motivation for various design decisions throughout this paper, we have left a thorough evaluation of their importance to future work. Some open questions include regarding these choices include: Could fewer objects be used to speed up the precomputation time? How important are each of the features in the global descriptor? Because we have found that adding photographs to the database improves results, how many more photographs do we need before this improvement plateaus? How can we combine data-driven and rule-based methods, and will such a hybrid improve results?

Finally, as discussed in Section 8.3, the Object Context cannot alone capture all composition rules. Augmenting the Object Context with higher level descriptors, including depth estimation, pose estimation, and action recognition will capture additional composition rules that are present in the database.

Using a database instead of encoding rules, we can adapt to personal preferences and the ever-changing notion of aesthetics and beauty.

10. Acknowledgements

The authors would like to thank the following people for their help in this project: Joel Brandt and Gavin Miller for their early guidance on this project; Daichi Ito for helping remove poorly composed images from the database; and James F. O'Brien for support and encouragement.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE 1106400.

References

[AM11] AHN S., MEHTA S.: *Automated Crops Using Crowdsourced Data*. Tech. rep., 2011. 3

[AS07] AVIDAN S., SHAMIR A.: Seam carving for content-aware image resizing. *ACM Trans. Graph. 26*, 3 (2007), 10. 2

[Bal87] BALLARD D. H.: Readings in computer vision: Issues, problems, principles, and paradigms. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987, ch. Generalizing the Hough Transform to Detect Arbitrary Shapes, pp. 714–725. 4

[Ban04] BANERJEE S.: *Composition-Guided Image Acquisition*. PhD thesis, 2004. 2

[BMP02] BELONGIE S., MALIK J., PUZICHA J.: Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 24*, 4 (2002), 509–522. 2

[BSS11] BHATTACHARYA S., SUKTHANKAR R., SHAH M.: A holistic approach to aesthetic enhancement of photographs. *ACM Trans. Multimedia Comput. Commun. Appl. 7S*, 1 (Nov. 2011), 21:1–21:21. doi:10.1145/2037676.2037678. 2

[CMT88] CARPANETO G., MARTELLO S., TOTH P.: Algorithms and codes for the assignment problem. *Annals of Operations Research 13*, 1 (1988), 191–223. 4

[Gar11] GARDNER J. S.: *Aesthetics of spatial composition: Facing, position, and context, and the theory of representational fit*. University of California, Berkeley, 2011. 8

[LCWCO10] LIU L., CHEN R., WOLF L., COHEN-OR D.: Optimizing photo composition. *Computer Graphic Forum 29*, 2 (2010), 469–478. 2, 7, 9

[lea14] How to use leading lines for better composition, Mar. 2014. URL: http://digital-photography-school.com. 8

[LLS04] LEIBE B., LEONARDIS A., SCHIELE B.: Combined object categorization and segmentation with an implicit shape model. In *In ECCV workshop on statistical learning in computer vision* (2004), ECCV, pp. 17–32. 4

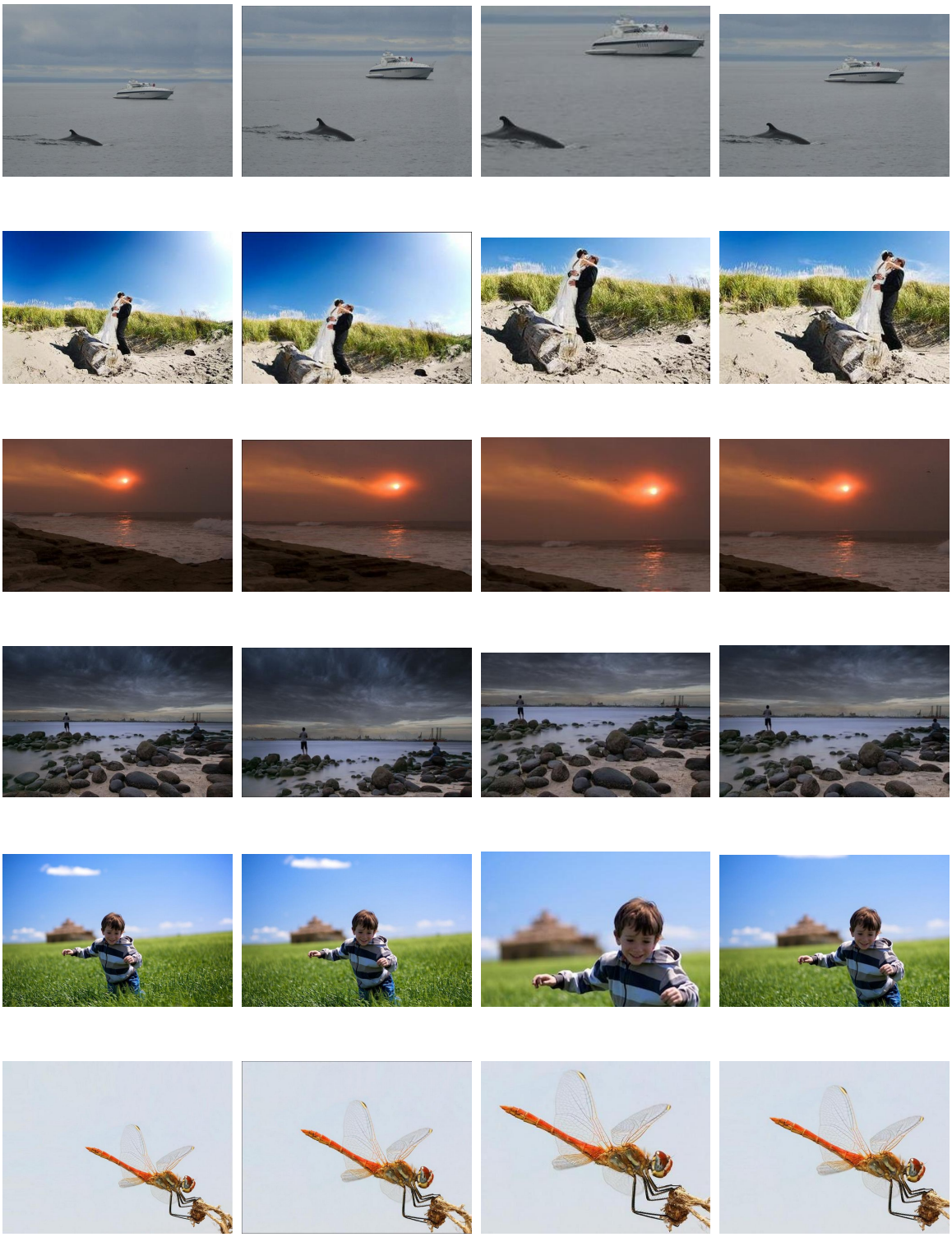


Figure 9: Comparison sequence of automatic composition improvement. The first column is the original image. The second is the result of Liu *et. al.*. The third is the result of Suh *et. al.*. The last column is our result.

[Low99] LOWE D.: Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on* (1999), vol. 2, IEEE, pp. 1150–1157 vol.2. 7

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2 (Nov. 2004), 91–110. 7

[MFM04] MARTIN D., FOWLKES C., MALIK J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 26*, 5 (2004), 530–549. 3

[mtu] Amazon mechanical turk. http://www.mturk.com. 7

[MTZM13] MARGOLIN R., TAL A., ZELNIK-MANOR L.: What makes a patch distinct? In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (June 2013), IEEE, pp. 1139–1146. doi:10.1109/CVPR.2013.151. 4

[NOSS09] NISHIYAMA M., OKABE T., SATO Y., SATO I.: Sensation-based photo cropping. In *Proceedings of the 17th ACM international conference on Multimedia* (New York, NY, USA, 2009), MM '09, ACM, pp. 669–672. 3

[pla14] Placing the horizon line, Mar. 2014. URL: http://www.fodors.com/travel-photography/article-placing-the-horizon-line-48/. 8

[PWS12] PANOZZO D., WEBER O., SORKINE O.: Robust image retargeting via axis-aligned deformation. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, ECCV, pp. 229–236. 2

[RGSS10] RUBINSTEIN M., GUTIERREZ D., SORKINE O., SHAMIR A.: A comparative study of image retargeting. In *ACM SIGGRAPH Asia 2010 papers* (New York, NY, USA, 2010), SIGGRAPH ASIA '10, ACM, pp. 160:1–160:10. 2

[RSA09] RUBINSTEIN M., SHAMIR A., AVIDAN S.: Multi-operator media retargeting. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2009) 28*, 3 (2009), 1–11. 2

[SAD*06] SANTELLA A., AGRAWALA M., DECARLO D., SALESIN D., COHEN M.: Gaze-based interaction for semi-automatic photo cropping. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2006), CHI '06, ACM, pp. 771–780. 2, 9

[SCSI08] SIMAKOV D., CASPI Y., SHECHTMAN E., IRANI M.: Summarizing visual data using bidirectional similarity. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), pp. 1–8. 2

[SLB*12] SHEN X., LIN Z., BRANDT J., AVIDAN S., WU Y.: Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (2012), pp. 3013–3020. 4

[SLBJ03] SUH B., LING H., BEDERSON B. B., JACOBS D. W.: Automatic thumbnail cropping and its effectiveness. In *Proceedings of the 16th annual ACM symposium on User interface software and technology* (New York, NY, USA, 2003), UIST '03, ACM, pp. 95–104. 2, 7

[SS09] SHAMIR A., SORKINE O.: Visual media retargeting. In *ACM SIGGRAPH ASIA 2009 Courses* (New York, NY, USA, 2009), SIGGRAPH ASIA '09, ACM, pp. 11:1–11:13. 2

[STR*05] SETLUR V., TAKAGI S., RASKAR R., GLEICHER M., GOOCH B.: Automatic image retargeting. In *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia* (New York, NY, USA, 2005), MUM '05, ACM, pp. 59–68. 2

[SWS11] SHE J., WANG D., SONG M.: Automatic image cropping using sparse coding. In *Pattern Recognition (ACPR), 2011 First Asian Conference on* (2011), pp. 490–494. 2

[TL13] TIGHE J., LAZEBNIK S.: Finding things: Image parsing with regions and per-exemplar detectors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (2013), IEEE. 6

[VJ01] VIOLA P., JONES M.: Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (2001), vol. 1, IEEE, pp. I–511. 6

[VTP*10] VAQUERO D., TURK M., PULLI K., TICO M., GELFAND N.: A survey of image retargeting techniques. In *Proc. SPIE* (2010), vol. 7798, Society of Photo-Optical Instrumentation Engineers, p. 779814. 2

[WTSL08] WANG Y.-S., TAI C.-L., SORKINE O., LEE T.-Y.: Optimized scale-and-stretch for image resizing. In *ACM SIGGRAPH Asia 2008 papers* (New York, NY, USA, 2008), SIGGRAPH Asia '08, ACM, pp. 118:1–118:8. 2

[YSQ*12] YAO L., SURYANARAYAN P., QIAO M., WANG J., LI J.: Oscar: On-site composition and aesthetics feedback through exemplars for photographers. *International Journal of Computer Vision 96*, 3 (2012), 353–383. 2

[ZWH13] ZHANG F.-L., WANG M., HU S.-M.: Aesthetic image enhancement by dependence-aware object recomposition. *Multimedia, IEEE Transactions on 15*, 7 (Nov 2013), 1480–1490. doi:10.1109/TMM.2013.2268051. 2

[ZZS*05] ZHANG M., ZHANG L., SUN Y., FENG L., MA W.: Auto cropping for digital photographs. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on* (2005), IEEE, pp. 4 pp.–. 2