

Detecting and Sketching the Common

Shai Bagon Ori Brostovski Meirav Galun Michal Irani
Dept. of Computer Science and Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel

Abstract

Given very few images containing a common object of interest under severe variations in appearance, we detect the common object and provide a compact visual representation of that object, depicted by a binary sketch. Our algorithm is composed of two stages: (i) Detect a mutually common (yet non-trivial) ensemble of ‘self-similarity descriptors’ shared by all the input images. (ii) Having found such a mutually common ensemble, ‘invert’ it to generate a compact sketch which best represents this ensemble. This provides a simple and compact visual representation of the common object, while eliminating the background clutter of the query images. It can be obtained from very few query images. Such clean sketches may be useful for detection, retrieval, recognition, co-segmentation, and for artistic graphical purposes.

1. Introduction

Given very few images (e.g., 3-5) containing a common object of interest, possibly under severe appearance changes, we detect the common object and provide a simple and compact visual representation of that object, depicted by a binary sketch (see Fig. 1). The input images may contain additional distracting objects and clutter, the object of interest is at unknown image locations, and its appearance may significantly vary across the images (different colors, different textures, and small non-rigid deformations). We do assume, however, that the different instances of the object share a very rough common geometric shape, of roughly the same scale ($\pm 20\%$) and orientation ($\pm 15^\circ$). Our output sketch captures this rough common shape.

The need to extract the common of very few images occurs in various application areas, including: (i) object detection in large digital libraries. For example, a user may provide very few (e.g., 3) example images containing an object of interest with varying appearances, and wants to retrieve new images containing this object from a database, or from the web. (ii) Co-segmentation of a few images. (iii) Artistic graphical uses.



Figure 1. **Detecting and sketching the common:** (a) The 4 input images provided to the algorithm. (b) The least trivial common part (the heart) is detected and sketched by the algorithm.

Our method is based on densely computed *Local Self-Similarity Descriptors* [14]. Our algorithm is composed of two main steps: (i) Identify the common object by detecting a similar (yet “non-trivial”) *ensemble of self-similarity descriptors*, that is shared by all the input images. Corresponding descriptors of the common object across the different images should be similar in their descriptor values, as well as in their relative positions within the ensemble. (ii) Having found such a mutually common ensemble of descriptors, our method “inverts” it to generate a compact binary sketch which best represents this ensemble.

It was shown in [14] that given a *single query image* of an object of interest (with very little background clutter), it is possible to detect other instances of that object in other images by densely computing and matching their local self-similarity descriptors. The query image can be a real or synthetic image, or even a *hand-drawn sketch* of the object.

In this paper we extend the method of [14] to handle *multiple query images*. Moreover, in our case those images are not centered around the object of interest (its position is unknown), and may contain also other objects and significant background clutter. Our goal is to detect the “*least trivial*” *common part* in those query images, and generate as clean as possible (region-based) sketch of it, while eliminating the background clutter of the query images. Such clean sketches can be obtained from *very few* query images,

and may be useful for detection, retrieval, recognition, and for artistic graphical purposes. Some of these applications are illustrated in our experiments.

Moreover, while [14] *received as an input* a clean hand-drawn sketch of the object of interest (and used it for detecting other instances of that object), we *produce* a sketch as one of our outputs, thereby also solving the “inverse” problem, namely: Given several images of an object, we can generate its sketch using the self-similarity descriptor.

A closely related research area to the problem we address is that of ‘learning appearance models’ of an object category, an area which has recently received growing attention (e.g., [4, 3, 5, 15, 8, 9, 12, 16, 18]), to name just a few). The goal of these methods is to discover common object shapes within collections of images. Some methods assume a single object category (e.g., [4, 5, 8, 15, 12, 16, 18]), while others assume multiple object categories (e.g., [3, 9]). These methods, which rely on weakly supervised learning (WSL) techniques, typically require tens of images in order to learn, detect and represent an object category. What is unique to the problem we pose and to our method is the ability to depict the common object from *very few images*, despite the large variability in its appearance. This is a scenario no WSL method (nor any other method, to our best knowledge) is able to address. Such a small number of images (e.g., 3) does not provide enough ‘statistical samples’ for WSL methods. While our method cannot compete with the performance of WSL methods when many (e.g., tens) of example images are provided, it outperforms existing methods when only few images with large variability are available. We attribute the strength of our method to the use of *densely computed region-based information* (captured by the local self-similarity descriptors), as opposed to commonly used *sparse and spurious edge-based information* (e.g., gradient-based features, SIFT descriptors, etc.) Moreover, the sketching step in our algorithm provides an *additional global constraint*.

Another closely related research area to the problem addressed here is ‘co-segmentation’ (e.g., [13, 1, 11]). The aim of co-segmentation is to segment out an object common to a few images (2 or more), by seeking segments in the different images that share common properties (colors, textures, etc.) These common properties are not shared by the remaining backgrounds in the different images. While co-segmentation methods extract the common object from *very few images*, they usually assume a much higher degree of similarity in appearance between the different instances of the object than that assumed here (e.g., they usually assume similar color distributions, similar textures, etc.)

The rest of the paper is organized as follows: Sec. 2 formulates the problem and gives an overview of our approach. Sec. 3 describes the component of our algorithm which detects the ‘least trivial’ common part in a collection of images, whereas Sec. 4 describes the sketching component of our algorithm. Experimental results are presented in Sec. 5.

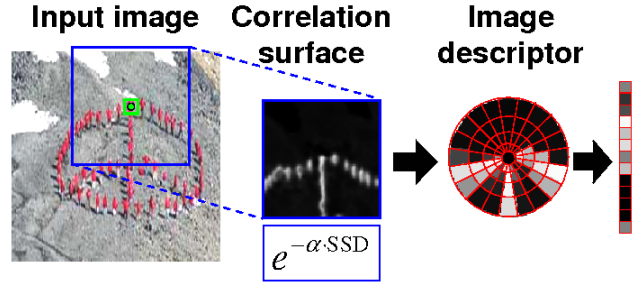


Figure 2. **The Local Self Similarity Descriptor:** (Figure taken from [14].) *The self-similarity descriptor for any given point (e.g., the green point in the left image), is computed by measuring the similarity of a 5×5 patch around the point with the surrounding 60×60 image region. This results in a ‘correlation’ surface (middle image). The correlation surface is quantized into a compact log-polar representation of 45 bins (15 angles, 3 radial intervals) to achieve invariance against small local affine and non-rigid deformations. The maximum values in each bin constitutes the value at the corresponding descriptor entry (right most image).*

2. Problem Formulation

Let I_1, \dots, I_K be K input images containing a common object under widely different appearances. The object may appear in different colors, different textures, and under small non-rigid deformations. The backgrounds are arbitrary and contain distracting clutter. The images may be of different sizes, and the image locations of the common object are unknown. We do assume, however, that the different instances of the object share a *very rough* common geometric shape, of roughly the same scale and orientation. Our output sketch captures this rough common shape.

Our approach is thus based on detecting ‘*common regions*’ (as opposed to ‘*common edges*’), using densely computed *Local Self-Similarity Descriptors* [14]. This descriptor (illustrated in Fig. 2) captures local shape information in the image vicinity where it is computed, while being invariant to its photometric properties (color, texture, etc.) Its log-polar representation makes this descriptor *insensitive* to small affine and non-rigid deformations (up to $\pm 20\%$ in scale, and $\pm 15^\circ$). It was further shown by [7] that the local self-similarity descriptor has a strong descriptive power (outperforming SIFT). The use of local self-similarity descriptors allows our method to handle much stronger variations in appearance (and in much fewer images) than those handled by previous methods. We densely compute the Self-Similarity descriptor in images I_1, \dots, I_K (at every 5-th pixel). ‘Common’ image parts across the images will have similar arrangements of self similarity descriptors.

Let c_1, \dots, c_K denote the unknown locations of the common object in the K images. Let $I_k^{c_k}$ denote a $w \times h$ subimage of I_k centered at c_k , containing the common object ($k = 1, \dots, K$) (need not be tight). For short, we will denote it by \tilde{I}_k . The sketch we seek is a binary image S of size $w \times h$ which best captures the rough characteristic shape of the common object shared by $\tilde{I}_1, \dots, \tilde{I}_K$. More formally,



Figure 3. **Sketching:** (a) Five input images. (b) Their joint sketch.

we seek a binary image S whose local self-similarity descriptors match as best as possible the local self-similarity descriptors of $\tilde{I}_1, \dots, \tilde{I}_K$. The descriptors should match in their *descriptor values*, as well as in their *relative positions* with respect to the centers $\{c_k\}$:

$$\begin{aligned} \text{Score}(S|\tilde{I}_1, \dots, \tilde{I}_K) &= \sum_{k=1}^K \text{match}(S, \tilde{I}_k) \quad (1) \\ &= \sum_{k=1}^K \sum_{i=1}^{w \cdot h} \text{sim}(d_i^S, d_i^k) \end{aligned}$$

where d_i^S is the i -th self-similarity descriptor computed at image location l_i in the sketch image S , d_i^k is the self-similarity descriptor computed *at the same relative position* l_i (up to small shifts) in the $w \times h$ subimage \tilde{I}_k , and $\text{sim}(d_1, d_2) = -\|d_1 - d_2\|_p$ measures how similar two descriptor vectors are (we experimented with L_p norms for $p = 1, 2$). Thus, the *binary sketch* we seek is:

$$\hat{S} = \underset{S}{\text{argmax}} \{ \text{Score}(S|\tilde{I}_1, \dots, \tilde{I}_K) \} \text{ s.t. } S(l) \in \{-1, 1\} \quad (2)$$

where $S(l)$ is the value of S at pixel l . This process is described in detail in Sec. 4, and results in a sketch of the type shown in Fig 3.

While edge-based detection and/or sketching [9, 18, 5] requires many input images, our region-based detection and sketching can be recovered from very few images. Edges tend to be very spurious, and are very prone to clutter (even sophisticated edge detectors like [10] – see Fig. 4.b). Edge-based approaches thus require a considerable number of images, to allow for the consistent edge/gradient features of the object to stand out from the inconsistent background clutter. In contrast, region-based information is much less sparse (area vs. line-contour), less affected by clutter or by misalignments, and is not as sensitive to the existence of strong clear boundaries. Much larger image offsets are required to push two corresponding regions out of alignment than to misalign two thin edges. Thus, region-based cues require fewer images to detect and represent the common object. Indeed, our method can provide good sketches from as few as 3 images. In fact, in some cases our method produces a meaningful sketch even from a *single* image, where edge-based sketching is impossible to interpret – see example in Fig. 4.

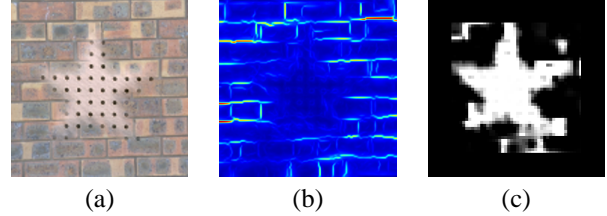


Figure 4. **Regions vs. Edges:** (a) a single input image. (b) The edge map generated by the method of [10]. (c) The binary sketch generated by our method when applied to the single input image (using all the self-similarity descriptors densely computed in that image). This illustrates the concept that region-based information is much richer than sparse edge-based information, and therefore appears to be more powerful for detection and for sketching.

In the general case, however, the locations c_1, \dots, c_K of the object within the input images I_1, \dots, I_K , are unknown. We seek a binary image S which sketches the ‘*least trivial*’ object (or image part) that is ‘*most common*’ to all those images. The ‘*most common*’ constraint is obvious: in each image I_k there should be a location c_k for which $\text{match}(S, I_k^{c_k})$ is high (where $\tilde{I}_k = I_k^{c_k}$ is the subimage centered at c_k). However, there are many image regions that are *trivially* shared by many natural images. For example, *uniform regions* (of uniform color or uniform texture) occur abundantly in natural images. Such regions share similar self-similarity descriptors, even if the underlying textures or colors are different (due to the invariance properties of the self-similarity descriptor). Similarly, strong vertical or horizontal edges (e.g., at boundaries between two different uniformly colored/textured regions) occur abundantly in images. We do not wish to identify such trivial (insignificant) common regions in the images as the ‘*common object*’.

Luckily, since such regions have good image matches in lots of locations, the *statistical significance* of their good matches tends to be low (when measured by how many standard deviations its peak match values are away from its mean match value in the collection of images). In contrast, a *non-trivial* common part (with non-trivial structure) should have at least one good match in each input image (could also have a few matches in an image), but these matches would be ‘*statistically significant*’ (i.e., this part would not be found ‘*at random*’ in the collection of images).

Thus, in the general case, we seek a binary sketch S and locations c_1, \dots, c_K in images I_1, \dots, I_K , such that:

- (i) S is ‘*most common*’, in the sense that it maximizes $\text{Score}(S|I_1^{c_1}, \dots, I_K^{c_K}) = \sum_{k=1}^K \text{match}(S, I_k^{c_k})$ of Eq. (1).
- (ii) S is ‘*least trivial*’, in the sense that its matches at c_1, \dots, c_K are *statistically significant*, i.e., it maximizes $\sum_{k=1}^K \text{StatSignificance}(\text{match}(S, I_k^{c_k}))$, where the significance of a match of S is measured by how many standard deviations it is away from the mean match value of S .

Our optimization algorithm may iterate between these two constraints: (i) Detect the locations $\{c_k\}_{k=1}^K$ of the least trivial common image part in $\{I_k\}_{k=1}^K$ (Sec. 3).

(ii) Sketch the common object given those image locations (Sec. 4). The overall process results in a sketch image, which provides a simple compact visual representation of the common object of interest in a set of query images, while eliminating any distracting background clutter found in those images.

3. Detecting the Common

We wish to detect image locations c_1, \dots, c_K in I_1, \dots, I_K , such that corresponding subimages centered at those locations, $I_k^{c_k}$, share as many self-similarity descriptors with each other as possible, yet their matches to each other are non-trivial (significant). The final sketch S will then be obtained from those subimages (Sec. 4).

Let us first assume that the dimension $w \times h$ of the subimages is given. We will later relax this assumption. Let \tilde{I} be a $w \times h$ image segment (this could be the final sketch S , or a subimage extracted from one of the K input images in the iterative process). We wish to check if \tilde{I} has a good match in each of the input images I_1, \dots, I_K , and also check the statistical significance of its matches. We ‘correlate’ \tilde{I} against all the input images (by measuring the similarity of its underlying self-similarity descriptors¹). In each image I_k we find the highest match value of \tilde{I} : $\max Match(\tilde{I}, I_k)$. The higher the value, the stronger the match. However, not every high match value is statistically significant. The *statistical significance* of $\max Match(\tilde{I}, I_k)$ is measured by how many standard deviations it is away from the mean match value of \tilde{I} in the entire collection of images, i.e.: $(\max Match(\tilde{I}, I_k) - \text{avgMatch}(\tilde{I})) / \text{stdMatch}(\tilde{I})$, where $\text{avgMatch}(\tilde{I})$ is the mean of all match values of \tilde{I} in the collection I_1, \dots, I_K , and $\text{stdMatch}(\tilde{I})$ is their standard deviation. We thus define the ‘Significance’ of a subimage \tilde{I} as: $\text{Significance}(\tilde{I} | I_1, \dots, I_K) = \frac{1}{K} \sum_{k=1}^K \text{StatSignificance}(\max Match(\tilde{I}, I_k))$.

Initially, we have no candidate sketch S . However, we can measure how ‘significantly common’ is each $w \times h$ subimage of I_1, \dots, I_K , when matched against all locations in all the other $K - 1$ images. We can assign a significance score to each *pixel* $p \in I_k$ ($k = 1, \dots, K$), according to the ‘Significance’ of its surrounding $w \times h$ subimage: $\text{Significance}(I_k^p | I_1, \dots, I_K)$.

We set c_k to be the pixel location with the *highest* significance score in image I_k , i.e., $c_k = \text{argmax}_{p \in I_k} \{\text{Significance}(I_k^p | I_1, \dots, I_K)\}$.

The resulting K points (one per image), c_1, \dots, c_K , provide the centers for K candidates of ‘non-trivial’ common image parts. We generate a sketch S from these image parts

¹We use the same algorithm employed by [14] to match ensembles of self-similarity descriptors, which is a modified version of the efficient “ensemble matching” algorithm of [2]. This algorithm employs a simple probabilistic “star graph” model to capture the relative geometric relations of a large number of local descriptors, up to small non-rigid deformations.

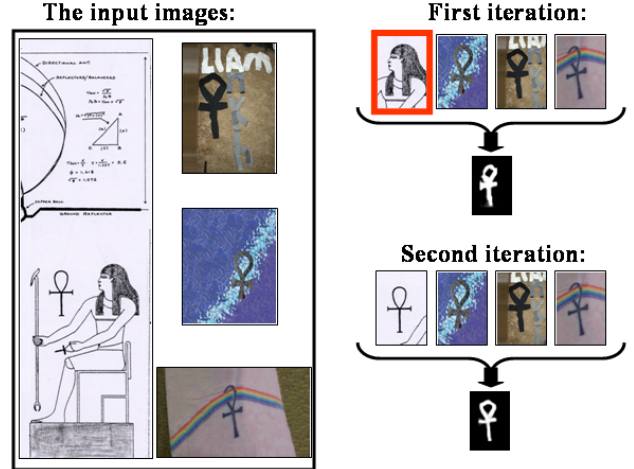


Figure 5. **Iterations of Detection & Sketching:** **Left:** The 4 input images. **Right:** The first iteration of the detection algorithm results in 4 detected image regions, of which 3 are correct and one is an outlier (marked by red). The resulting sketch produced from these regions is reasonably good (due to the robustness of the sketching to outliers – see Secs. 4 and 5), and is used for refining the detection in the input images. This results in 4 correct detections in the second iteration, and an improved sketch.

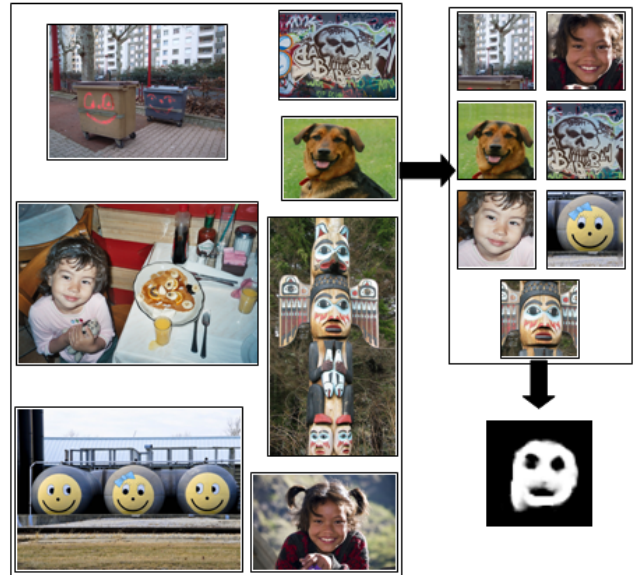


Figure 6. **Detecting and sketching the common:** (Left) The input images. (Upper-Right) The detected image regions of the common object, including one outlier. (Lower-Right) The resulting sketch.

(using the algorithm of Sec. 4).

We repeat the above process, this time for $\tilde{I} = S$, to detect its best matches in I_1, \dots, I_K . This should lead to improved detection and localization of the common object (c_1, \dots, c_K), and accordingly to an improved sketch S . This algorithm can be iterated several times. In practice, in all our experiments a good sketch S was recovered already in the first iteration. An additional iteration was sometimes useful for improving the detection. Fig. 5 shows two itera-

tions of this process, applied to 4 input images. More results of the detection can be seen in Fig. 6.

Handling unknown $w \times h$: In principle, when $w \times h$ is unknown, we can run the above algorithm “*exhaustively*” for a variety of $w = w_{min}, \dots, w_{max}$ and $h = h_{min}, \dots, h_{max}$, and choose “the best” $w \times h$ (with maximal significance score). In practice, this is implemented more efficiently using “integral images”, by integrating the contributions of individual self-similarity descriptors into varying window sizes $w \times h$.

Computational Complexity: The detection algorithm is implemented coarse-to-fine. The first step of the algorithm described above is quadratic in the size of the input images. However, since the number of images is typically small (e.g., 3 – 5), and since the quadratic step occurs only in the coarsest/smallest resolutions of the images, this results in a computationally efficient algorithm.

4. Sketching the Common

Let $\tilde{I}_1, \dots, \tilde{I}_K$ be the $w \times h$ subimages centered around the common object (detected and extracted from the input images using the algorithm of Sec. 3). The goal of the sketching process is to produce a binary image S , which best captures the rough characteristic shape of the object shared by $\tilde{I}_1, \dots, \tilde{I}_K$, as posed by Eq. (2). Namely, find S whose ensemble of self-similarity descriptors is as similar as possible to the ensembles of descriptors extracted from $\tilde{I}_1, \dots, \tilde{I}_K$. If we were to neglect the binary constraint $S(l) \in \{-1, 1\}$ in Eq. (2), and the requirement for consistency between descriptors of an image, then the *optimal solution* for the collection of self-similarity descriptors of S , $\{d_i\}_{i=1}^{w \cdot h}$, could be explicitly computed as:

$$\begin{aligned} d_i &= \text{median}_k \{d_i^k\} & \text{if } L_1\text{-norm} \\ d_i &= \text{mean}_k \{d_i^k\} & \text{if } L_2\text{-norm} \end{aligned} \quad (3)$$

We use the L_1 -norm to generate these ‘combined’ descriptors $\{d_i\}_{i=1}^{w \cdot h}$, because of the inherent robustness of the median operator to outliers in the descriptors (also confirmed by our empirical evaluations in Sec 5). Having recovered such a collection of descriptors for S , we proceed and solve the “inverse” problem – i.e., to generate the image S from which these descriptors emanated. However, the collection of descriptors $\{d_i\}_{i=1}^{w \cdot h}$ generated via a ‘median’ or ‘average’ operations is no longer guaranteed to be a valid collection of self-similarity descriptors of any real image (binary or not). We thus proceed to recover the simplest possible image S whose self-similarity descriptors best approximate the ‘combined’ descriptors $\{d_i\}_{i=1}^{w \cdot h}$ obtained by Eq. (3).

Self-similarity descriptors cover large image regions, with high overlaps. As such, the similarity and dissimilarity between two image locations (pixels) of S are *implicitly* captured by multiple self-similarity descriptors and in different descriptor entries. The self-similarity descriptor as defined in [14] has values in the range $[0, 1]$, where 1 indicates high resemblance of the central patch to the patches



Figure 7. **Computing attraction repulsion matrix W :** The log-polar self-similarity descriptor d_i is located at l_i (red cross). White bins signify image areas of high similarity to the central patch, dark bins signify image areas of dissimilarity to the central patch. The point l_j (blue cross), which is the center of descriptor d_j (not drawn), falls in a white bin of descriptor d_i (i.e., $0 < d_i(l_j) \leq 1$). The entry w_{ij} in the matrix W is determined accordingly: $w_{ij} = \alpha_{ij} (d_i(l_j) + d_j(l_i)) / 2$, where α_{ij} (the certainty assigned to this entry), is inversely proportional to the distance $\|l_i - l_j\|$ (the distance between the red and blue crosses). Similarly, the point l_k (green cross), which is the center of another descriptor d_k (also not drawn), falls in a dark bin of descriptor d_i , i.e., $-1 \leq d_i(l_k) < 0$, and $\alpha_{ik} < \alpha_{ij}$ (because the green cross falls farther away from the center of d_i , hence lower certainty).



Figure 8. **Detecting and sketching the common:** (a) Five input images. (b) The resulting sketch.

in the corresponding log-polar bin, while 0 indicates high dissimilarity of the central patch to the corresponding log-polar bin. For our purposes, we stretch the descriptor values to the range $[-1, 1]$, where 1 signifies “attraction” and -1 signifies “repulsion” between two image locations.

Let W be a $wh \times wh$ matrix capturing the attraction/repulsion between every two image locations, as induced by the collection of the ‘combined’ self-similarity descriptors $\{d_i\}_{i=1}^{w \cdot h}$ of Eq. (3). Entry w_{ij} in the matrix is the degree of attraction/repulsion between image locations l_i and l_j , determined by the self-similarity descriptors d_i and d_j centered at those points. $d_i(l_j)$ is the value of the bin containing location l_j in descriptor d_i (see Fig. 7). Similarly, $d_j(l_i)$ is the value of the bin containing location l_i in descriptor d_j . The entry w_{ij} gets the following value:

$$w_{ij} = \alpha_{ij} (d_i(l_j) + d_j(l_i)) / 2 \quad (4)$$

where $\alpha_{ij} = \alpha_{ji}$ is inversely proportional to the distance $\|l_i - l_j\|$ between the two image locations (we give higher weight to bins that are closer to the center of the descriptor, since they contain more accurate/reliable information).

Note that a ‘pure’ attraction/repulsion matrix W of a true binary image S contains only 3 types of values w_{ij} : $-1, 0, 1$. If l_i and l_j belong to the same region in S (i.e.,

both in foreground or both in background), then $w_{ij} = 1$; if l_i and l_j belong to different regions in S , then $w_{ij} = -1$, and if the points are distant (out of descriptor range), then $w_{ij} = 0$. In the general case, however, the entries span the range $[-1, 1]$, where 1 stands for “strong” attraction, -1 for “strong” repulsion and 0 means “don’t care”. The closer the value of w_{ij} to 0, the lower its attraction/repulsion confidence; the closer it is to ± 1 , the higher the attraction/repulsion confidence.

Note that W is different from the classical affinity matrix used in spectral clustering or in min-cut, which use non-negative affinities, and their value 0 is *ambiguous* – it signifies both *high-dissimilarity* as well as *low-confidence*. The distinction between ‘attraction’, ‘repulsion’, and ‘low-confidence’ are critical in our case, thus we cannot resort to the max-flow algorithm or to spectral clustering in order to solve our problem. An affinity matrix with positive and negative values was used by [17] in the context of the normalized-cut functional. However, their functional is not appropriate for our problem (and indeed did not yield good results for S when applied to our W). We therefore define a different functional and optimization algorithm in order to solve for the binary sketch S .

The binary image S which *best* approximates the attraction/repulsion relations captured by W , will minimize the following functional:

$$\min_S \sum_{i,j} w_{ij} (S(l_i) - S(l_j))^2 \quad \text{subject to } S(l) \in \{-1, 1\} \quad (5)$$

where $S(l)$ is the value of S at pixel l . Note that for a binary image, the term $(S(l_i) - S(l_j))^2$ can obtain only one of two values: 0 (if both pixels belong to foreground, or both belong to background), *or* 4 (if one belongs to the foreground, and one to the background). Thus, when w_{ij} is positive (attraction), $S(l_i)$ and $S(l_j)$ should have the same value (both 1 or both -1), in order to minimize that term $w_{ij}(S(l_i) - S(l_j))^2$. The larger w_{ij} (stronger confidence), the stronger the incentive for $S(l_i)$ and $S(l_j)$ to be the same. Similarly, a negative w_{ij} (repulsion) pushes *apart* the values $S(l_i)$ and $S(l_j)$. Thus, $S(l_i)$ and $S(l_j)$ should have opposite signs in order to minimize that term $w_{ij}(S(l_i) - S(l_j))^2$. When $w_{ij} \approx 0$ (low confidence), the value of the functional will not be affected by the values $S(l_i)$ and $S(l_j)$ (i.e., “don’t care”). It can be shown that in the ‘ideal’ case, i.e., when W is generated from a binary image S , the global minimum of Eq. (5) is obtained at S .

Solving the constrained optimization problem: The min-cut problem where only non-negative values of w_{ij} are allowed can be solved by the max-flow algorithm in polynomial time. However, the weights w_{ij} in the functional of Eq. (5) can obtain both positive and negative values, turning our ‘cut’ problem as posed above into an NP-hard problem. We therefore *approximate* Eq. (5) by reposing it as a quadratic programming problem, while relaxing the binary



Figure 9. **Detecting and sketching the common:** (a) The input images. (b) The resulting sketch.

constraints.

Let D be a diagonal matrix with $D_{ii} = \sum_j w_{ij}$, and let $L = D - W$ be the graph Laplacian of W . Then $\frac{1}{2} \sum_{i,j} w_{ij} (S(l_i) - S(l_j))^2 = S^T L S$. Thus, our objective function is a quadratic expression in terms of S . The set of binary constraints are relaxed to the following set of linear constraints $-1 \leq S(l) \leq 1$, resulting in the following quadratic programming problem:

$$\hat{S} = \arg \min_S S^T L S \quad \text{s.t.} \quad -1 \leq S(l) \leq 1 \quad (6)$$

Since L is not necessarily positive semi-definite, we do not have a guarantee regarding the approximation quality (i.e., how far is the achieved numerical solution from the optimal solution). Still, our empirical tests demonstrate good performance of this approximation. We use Matlab’s optimization toolbox (quadprog) to solve this optimization problem and obtain a sketch \hat{S} . In principle, this does not yield a binary image. However, in practise, the resulting sketches look very close to binary images, and capture well the rough geometric shape of the common objects.

The above sketching algorithm is quite robust to outliers (see Sec. 5), and obtains good sketches from very few images. Moreover, if when constructing the attraction/repulsion matrix W we replace the ‘combined’ descriptors of Eq. (3) with the self-similarity descriptors of a *single image*, our algorithm will produce ‘binary’ sketches of a single image (although these may not always be visually meaningful). An example of a sketch obtained from a single image (using all its self-similarity descriptors) can be found in Fig. 4.

5. Experimental Results

Figs. 1,3,6,8,9,10 show qualitative results on various image sets. In all of these examples the number of input im-










Input images	Output sketch	Input images	Output sketch
			
			
			
			
			

Figure 10. Sample results on ETHZ shapes [6] dataset: Detection and sketching using only 3 images (left), and using 6 images (right).

ages was very small (3 – 7), with large variability in appearance and background clutter. Our algorithm was able to detect and produce a compact representation (a sketch) of the common content.

We further conducted empirical evaluations of the algorithm using ETHZ shape dataset [6]. This dataset consists of five object categories with large variability in appearance: Applelogos, Bottles, Giraffes, Mugs and Swans (example images can be seen in Fig. 10). There are around 50 images in each set, with ground-truth information regarding the location of the object in each image, along with a single hand-drawn ground truth shape for each category. In order to assess the quality of our algorithm (which is currently not scale invariant, although it can handle up to $\pm 20\%$ scale variation, and $\pm 15^\circ$ rotations), we scaled the images in each dataset to have *roughly* the same object size (but we have not rotated the images, nor changed their aspect ratios).

Sketch quality score: Because our sketch S is continuous in the range $[-1, 1]$, we stretch the values of the ground-truth sketch S_{GT} also to this range, and multiply the two sketches pixel-wise. Our sketch quality score is: $Quality(S) = \langle S, S_{GT} \rangle / (\# \text{ of pixels})$. In places where both sketches agree in their sign (either white regions or black) the pixel-wise product is positive, while in

places where the sketches disagree, the product is negative. This produces a sketch quality score with values ranging between -1 (lowest quality) to $+1$ (highest quality). Note that even if our sketch displays a perfect shape, its quality will be smaller than 1, because it is not a perfect binary image. From our experience, sketch quality ≥ 0.8 are usually excellent-looking sketches.

We first assessed the quality of our algorithm to identify and sketch the common object correctly, as a function of the number of input images K ($K = 2, 3, \dots, 10$). We randomly sampled K images out of an object category set, applied our detection and sketching algorithm to that subset, and compared the resulting sketch S to the ground-truth S_{GT} . We repeated this experiment 15 times for each K , and computed mean sketch quality scores. Fig. 11 displays plots of the mean quality score for the 5 categories. It can be seen that from relatively few images ($K = 3$) we already achieve sketches of good quality, even for challenging sets such as the giraffes (although, with the increased number of example images, its legs tend to disappear from the sketch because of their non-rigid deformations). Examples for sketching results for some of these experiments can be seen in Fig. 10.

We next evaluated the robustness of the sketching com-

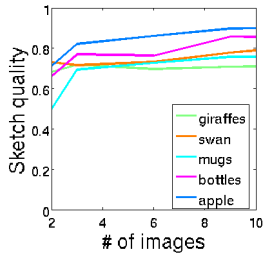


Figure 11. **Evaluating sketch quality:** Mean values of $Quality(S)$ as a function of the number of input images ($K = 2, \dots, 10$) randomly sampled from each set of ETHZ shape dataset [6].

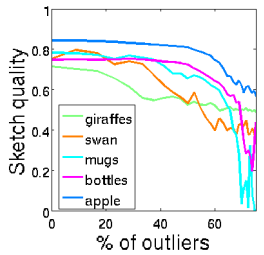


Figure 12. **Sketching in presence of outliers:** We “corrupt” a set of 10 “inlier” with n randomly chosen natural images. Graph shows mean values of $Quality(S)$ as a function of the percent of outlier images in the input set, i.e., $n/(10 + n)$.

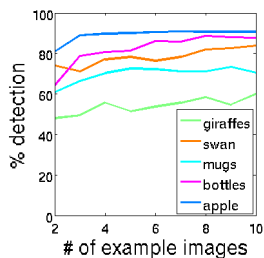


Figure 13. **Detection in new images:** We empirically evaluated how well the sketch generated from very few images ($K = 2, \dots, 10$) performs in detecting the common shape in new images.

ponent of our algorithm to outliers. Such robustness is important, since the detection algorithm often produces outlier detections (see Fig. 5). We used 10 “inlier” images which alone generate a good sketch with high sketch quality score. We then added to them $n = 1, \dots, 30$ outlier images (cropped at random from natural images). For every such $10 + n$ image set we generated a sketch, and compared it to the ground-truth. Each experiment was repeated 15 times. Fig. 12 displays plots of sketch quality vs. percent of outliers $n/(10 + n)$. Our sketching method is relatively robust to outliers, and performs quite well even in presence of 50% outliers (as expected due to the median operation in Eq. (3)).

In addition to sketch quality evaluation we tested the performance of our algorithm in the scenario described in the Introduction: given a very small number of example images, how useful is the output of our automatic detection & sketching algorithm for successfully detecting that object in *new images*. For $K = 2, 3, \dots, 10$, we randomly sampled K images out of an object category set, applied our detection & sketching algorithm to that subset, and used the resulting sketch to detect the object in the *remaining* $50 - K$ images of that category set. We consider an object in image I_n as “detected” if the location of $\max Match(S, I_n)$ (the detected center c_n of the object) falls *no farther away* than $1/4$ of the width or height of the bounding-box from the ground-truth center. We repeated each experiment 40 times and plotted the average detection rates in Fig. 13. For the Apples, Bottles, and Swans we get high detection rates

(for as few as $K = 3$ example images; a scenario no WSL method can handle to the best of our knowledge). However, our detection rates are not as good in the Giraffe set, since the giraffes undergo strong non-rigid deformations (they sometimes tilt their necks down, and their legs change positions). Our current algorithm cannot handle such strong non-rigid deformations.

Acknowledgement: This work was partially funded by the Israel Science Foundation.

References

- [1] S. Bagon, O. Boiman, and M. Irani. What is a good image segment? A unified approach to segment extraction. In *ECCV*, 2008. 2
- [2] O. Boiman and M. Irani. Detecting irregularities in images and in video. *IJCV*, 2007. 4
- [3] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: Finding a (thick) needle in a haystack. In *CVPR*, 2009. 2
- [4] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *CVPR*, 2007. 2
- [5] V. Ferrari, F. Jurie, and C. Schmid. From images to shape models for object detection. *IJCV*, 2009. 2, 3
- [6] V. Ferrari, T. Tuytelaars, and L. Van Gool. Object detection by contour segment networks. In *ECCV*, 2006. 7, 8
- [7] E. Hörster, T. Greif, R. Lienhart, and M. Slaney. Comparing local feature descriptors in pls-based image models. In *DAGM*, 2008. 2
- [8] L. Karlinsky, M. Dinerstein, D. Levi, and S. Ullman. Unsupervised classification and part localization by consistency amplification. In *ECCV*, 2008. 2
- [9] Y. J. Lee and K. Grauman. Shape discovery from unlabeled image collections. In *CVPR*, 2009. 2, 3
- [10] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik. Using contours to detect and localize junctions in natural images. In *CVPR*, 2008. 3
- [11] L. Mukherjee, V. Singh, and C. Dyer. Half-integrality based algorithms for cosegmentation of images. In *CVPR*, 2009. 2
- [12] M. H. Nguyen, L. Torresani, F. de la Torre, and C. Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *ICCV*, 2009. 2
- [13] C. Rother, V. Kolmogorov, T. Minka, and A. Blake. Cosegmentation of image pairs by histogram matching-incorporating a global constraint into MRFs. In *CVPR*, 2004. 2
- [14] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *CVPR*, 2007. 1, 2, 4, 5
- [15] J. Winn and N. Jojic. LOCUS: Learning object classes with unsupervised segmentation. In *ICCV*, 2005. 2
- [16] Y. N. Wu, Z. Si, H. Gong, and S.-C. Zhu. Learning active basis model for object detection and recognition. *IJCV*, 2009. 2
- [17] S. X. Yu and J. Shi. Understanding popout through repulsion. In *CVPR*, 2001. 6
- [18] L. Zhu, C. Lin, H. Huang, Y. Chen, and A. Yuille. Unsupervised structure learning: hierarchical recursive composition, suspicious coincidence and competitive exclusion. In *ECCV*, 2008. 2, 3