Image Co-segmentation Using Maximum Common Subgraph Matching and Region Co-growing

Avik Hati(✉), Subhasis Chaudhuri, and Rajbabu Velmurugan

Electrical Engineering Department,
Indian Institute of Technology Bombay, Mumbai, India
{avik,sc,rajbabu}@ee.iitb.ac.in

Abstract. We propose a computationally efficient graph based image co-segmentation algorithm where we extract objects with similar features from an image pair or a set of images. First we build a region adjacency graph (RAG) for each image by representing image superpixels as nodes. Then we compute the maximum common subgraph (MCS) between the RAGs using the minimum vertex cover of a product graph obtained from the RAG. Next using MCS outputs as the seeds, we iteratively co-grow the matched regions obtained from the MCS in each of the constituent images by using a weighted measure of inter-image feature similarities among the already matched regions and their neighbors that have not been matched yet. Upon convergence, we obtain the co-segmented objects. The MCS based algorithm allows multiple, similar objects to be co-segmented and the region co-growing stage helps to extract different sized, similar objects. Superiority of the proposed method is demonstrated by processing images containing different sized objects and multiple objects.

Keywords: Maximum common subgraph · Region co-growing

1 Introduction

Co-segmentation is the problem of segmenting objects with similar features from more than one image (see Fig. 1) or from multiple frames in a video. The objects of common interest in multiple images are detected as co-segmented objects [1], [2], [3]. Image foreground segmentation without supervision is a difficult problem. If an additional image containing a similar foreground is provided, both images can be segmented simultaneously with a higher accuracy using co-segmentation. Co-segmentation can also be used to detect objects of common

Partial financial supports from Bharti Centre for Communication in IIT Bombay and K.N Bajaj Chair Professorship are gratefully acknowledged.

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-46466-4_44) contains supplementary material, which is available to authorized users.

B. Leibe et al. (Eds.): ECCV 2016, Part VI, LNCS 9910, pp. 736–752, 2016.
DOI: 10.1007/978-3-319-46466-4_44

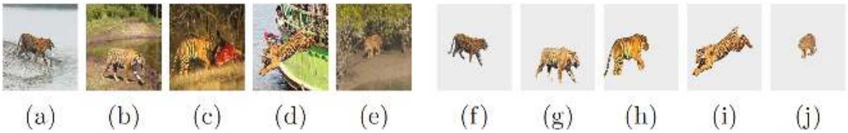


Fig. 1. Illustration of the co-segmentation problem. (a–e) Images retrieved by a child from the internet, when asked to provide pictures of a tiger, and (f–j) common object quite apparent from the given set of images (Color figure online)

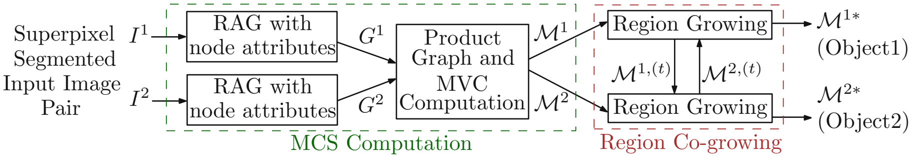


Fig. 2. Block diagram of the proposed co-segmentation algorithm. Input image pair I^1 and I^2 is represented as region adjacency graphs (RAGs) G^1 and G^2 that are used to obtain the maximum common subgraph (MCS) that gives the initial matched regions $\mathcal{M}^1$ and $\mathcal{M}^2$ in I^1 and I^2. These are iteratively (index-(t)) co-grown to obtain the final matched regions $\mathcal{M}^{1*}$ and $\mathcal{M}^{2*}$. In order to grow the region $\mathcal{M}^1$ in I^1, the region $\mathcal{M}^2$ is needed to find the match and similarly $\mathcal{M}^2$ requires $\mathcal{M}^1$ to grow (Color figure online)

interest in a set of crowd sourced images. A related topic in this area is image co-saliency. Co-saliency measures the saliency of co-occurring objects in multiple images. Image segmentation using co-segmentation is, in principle, different from object segmentation using co-saliency as the segmented object need not be the salient object in both images.

The co-segmentation methods in [1], [4], [5], [6] incorporate the foreground similarity of an image pair in their Markov Random Field model based optimization problem. Rother *et al.* [1] first introduced co-segmentation of an image pair using histogram matching through graph cuts. Mukherjee *et al.* [4] used a similar method by replacing l_1-norm in the cost function of [1] by l_2-norm. But the optimization problem in both methods is computationally intensive. Hochbaum and Singh [5] rewarded foreground histogram consistency, instead of minimizing foreground histogram difference [1], [4] to simplify the optimization. They also use prior information about foreground and background colors. The methods in [1], [4], [5], [6] perform well only for exactly same object on different background.

There has been some work on simplifying the co-segmentation problem by including user interaction for segmentation [7], [8]. Recent works focus on co-segmenting more than two images as it has more applications. Joulin *et al.* [9] formulated co-segmentation as a two-class clustering problem using a discriminative clustering method. They extended this work for multiple classes in [10] by incorporating spectral clustering. As their kernel matrix is defined for all possible pixel pairs of all images, the complexity goes up rapidly with the number of images. Mukherjee *et al.* [11] proposed a scale invariant co-segmentation method. Vicente *et al.* [12] used proposal object segmentations to train a random forest regressor for co-segmentation. This method relies heavily on the accuracy of individual segmentation outputs as it is assumed that one segment contains the complete object. Kim *et al.* [13] used anisotropic diffusion to optimize the number and location of image segments. As all the images are segmented into an equal number of clusters, over-segmentation may become an issue in a set of different types of images. Furthermore, this method cannot co-segment heterogeneous objects. An improvement to this has been proposed in [14] using supervision. The graph based method in [15] includes high-level information like object detection, which is also a complex problem. Lee *et al.* [16] proposed a multiple random walk based image co-segmentation method. Tao *et al.* [17] proposed a co-segmentation method based on shape conformability. But this method cannot handle shape variations caused by viewpoint and posture changes. The co-segmentation methods in [3], [18], [19] use saliency to initialize their methods.

Recently co-saliency based methods [20], [21], [22], [23], [24], [25], [26], [27] have also been used for co-segmentation. These methods detect common, salient objects by combining (i) individual image saliency outputs and (ii) pixel or superpixel feature distances among the images. Liu *et al.* [26] used hierarchical segmentation and Tan *et al.* [25] used a bipartite graph to compute feature similarity. Cao *et al.* [24] combined outputs of multiple saliency detection methods. Objects with high saliency value may not necessarily have common features while considering a set of images, hence these saliency guided methods do not always detect similar objects across images correctly. Also a good saliency detection method introduces additional complexity to the co-segmentation algorithm. Our solution to the co-segmentation problem is independent of saliency or any prior knowledge or pre-processing.

In this paper, we propose a novel foreground co-segmentation algorithm using an efficient graph matching based approach. We set up the problem as a maximum common subgraph (MCS) computation problem. We find a solution to MCS of two RAGs obtained from an image pair and then perform region co-growing to obtain the complete co-segmented objects.

In a standard MCS problem, node attributes are matched exactly for a pair of graphs. But in natural images, there can be some changes in features (e.g. color, texture, size) of similar objects or regions. So in our approach, node attributes do not need to match exactly. This necessitates selecting a threshold for node matching. The MCS algorithm matching allows multiple similar objects to be co-segmented. Region co-growing allows objects of different size to be detected. We show that an efficient use of the MCS algorithm followed by region co-growing can co-segment high resolution images without increasing computations.

We present the co-segmentation algorithm initially for two images in Sects. 2 and 3. We extend it for multiple images in Sect. 4. We show comparative results in Sect. 5 and conclude in Sect. 6.

2 Co-segmentation for Two Images

In the co-segmentation problem for two images, we are interested in finding the objects of interest that are present in both the images and have similar features. The flow of the proposed co-segmentation algorithm is shown in Fig. 2. First we segment each image (Fig. 3(a),(e)) into superpixels using SLIC method [28] and represent each image as a graph by representing the superpixels as nodes. Superpixel segmentation allows a coarse level description of the image through a limited number (N) of nodes of the graph. An increase in N increases the computation in graph matching drastically. So, we use superpixels instead of pixels as nodes. Moreover, each superpixel contains pixels from a single object and is homogeneous in feature and helps in retaining the shape of an object boundary. As an image is a group of connected components, we build a region adjacency graph (RAG) for each image where every spatially contiguous superpixel (node) pair is connected by an edge.

2.1 Image Representation as Attributed RAGs

We build two RAGs $G^1 = (\mathcal{V}^1, E^1)$ and $G^2 = (\mathcal{V}^2, E^2)$ corresponding to images I^1 and I^2, respectively (see Fig. 4(a) for illustration of MCS matching problem). Here $\mathcal{V}^i = \{v_k^i\}$ and $E^i = \{e_{kl}^i\}$ for $i = 1, 2$ denote the set of nodes and edges, respectively. In each graph G^i, an edge exists between a pair of nodes (superpixels) if they are spatial neighbors of each other. One can assign several features to each node. We use two features: (i) CIE Lab mean color and (ii) rotation invariant histogram of oriented gradient (HoG) of the pixels within the corresponding superpixel. The use of HoG feature is motivated by the fact that multiple superpixels can have similar mean color in spite of being completely different in color, and HoG features are useful to capture the image texture. To co-segment similar objects with different orientation, we use rotation invariant HoG of each superpixel. If an image is rotated, the gradient direction of every pixel is also changed by the same angle. Hence, the histogram (of gradients of a superpixel) will be shifted as a function of the rotation angle. In order to achieve rotation invariance, we circularly shift the computed HoG with respect to the location of the maximum histogram value. We compute the feature similarity ($\mathcal{S}_f(\cdot)$) between nodes v_k^1 in G^1 and v_l^2 in G^2 as a weighted sum of the corresponding color and HoG feature similarities denoted as $\mathcal{S}_c(\cdot)$ and $\mathcal{S}_h(\cdot)$, respectively as

$$\mathcal{S}_f\left(v_k^1, v_l^2\right) = 0.5\,\mathcal{S}_c\left(v_k^1, v_l^2\right) + 0.5\,\mathcal{S}_h\left(v_k^1, v_l^2\right). \tag{1}$$

Here a normalized Euclidean distance measure is used for computing the feature distance. Normalization is done with respect to the maximum pairwise distance between all nodes. The similarity measure $\mathcal{S}_f(\cdot)$ is defined as the additive inverse of the computed normalized distance. We then obtain the MCS between the two RAGs to obtain the common objects as explained next.

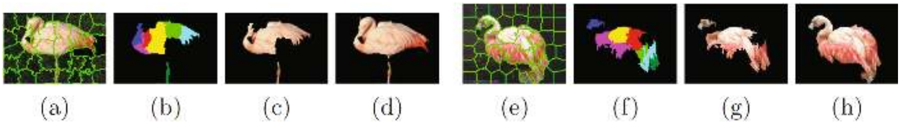


Fig. 3. Illustration of co-segmentation of an image pair. ((a),(e)) Input images and their SLIC segmentation. ((b),(f)) The matched nodes i.e., superpixels across images (shown in same color) obtained from the MCS computation and the corresponding ((c),(g)) object regions in the images. ((d),(h)) Co-segmented objects obtained after performing region co-growing on the initially matched regions in ((c),(g)) (Color figure online)

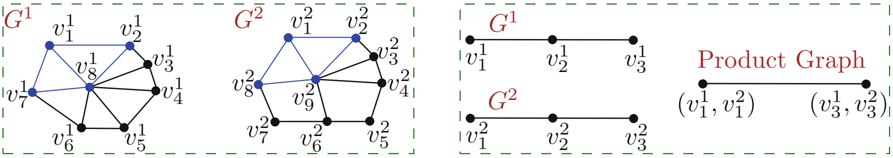
2.2 MCS Computation from RAGs

To solve the co-segmentation problem of an image pair, we need superpixel correspondences from one image to the other and match the superpixels within the objects of similar features across images. The computational complexity is $O((|G^1| + |G^2|)^3)$ assuming a minimum cost many-to-many matching algorithm [29]. Without any prior information about the objects, this matching becomes exhaustive and may result in many disconnected segments as matched regions. Each of these segments may be a group of superpixels or even a single superpixel and such matching may not be meaningful. To obtain a meaningful match, wherein the connectivity among the superpixels are maintained, we use a graph based approach to jointly segment the complete objects from an image pair. Thus our objective is to obtain the maximum common subgraph (MCS) that represents the co-segmented objects. The MCS corresponds to the common subgraphs M^1 in G^1 and M^2 in G^2. It may be noted that, in general, $M^1 \neq M^2$ as the common object in both the images need not undergo identical superpixel segmentation, and hence many-to-one matching must be permitted, unlike in a standard MCS finding algorithm. The computation time depends on the number of nodes in the graph, and this explains why we use the superpixel segmentation first as it cuts down the number of nodes drastically. Further to reduce the complication arising from many-to-one node matching, we assume that upto a maximum of p nodes in one image may match to a single node in the other image, based on a similarity measure. Following the work of Madry [30], it is possible to show that the computation complexity reduces to $O((p(|G^1| + |G^2|))^{10/7})$ when the matching is restricted to a maximum of p nodes only.

To find the MCS, we build two product graphs H^{12} and H^{21} (ideally known as vertex product graph) from the RAGs G^1 and G^2 based on their inter-image (superpixel) feature similarities (see Eq. (1)). A node in a product graph [31] is denoted as a 2-tuple (v_k^1, v_l^2) with $v_k^1 \in G^1$ and $v_l^2 \in G^2$. We call it a product node to differentiate it from single image nodes. As motivated in Sect. 1, node features do not need to match exactly for natural images. So, we select a threshold t_G $(0 \leq t_G \leq 1)$ for matching. For a fixed $v_k^1 \in \mathcal{V}^1$, let $\mathcal{U}_k^2$ be the ordered list of nodes $\{v_l^2\}$ in $\mathcal{V}^2$ such that $\{\mathcal{S}_f\left(v_k^1, v_l^2\right)\}_{\forall l}$ are in descending order of magnitude. We define the set of product nodes $\mathcal{H}^{12}$ of the product graph H^{12} as

$$\mathcal{H}^{12} = \bigcup_{\forall k} \left\{ \left(v_k^1, u_l \in \mathcal{U}_k^2 \right)_{l=1,2,\ldots p} | \mathcal{S}_f \left(v_k^1, u_l \right) > t_G \right\} \quad (2)$$

Similarly, we compute $\mathcal{H}^{21}$ by keeping $\mathcal{V}^2$ as reference. It is interesting to note that allowing one node in one graph to match to p nodes in the other graph leads to $\mathcal{H}^{12} \neq \mathcal{H}^{21}$, resulting in $M^1 \neq M^2$ (i.e. not commutative) as noted earlier. A large value of t_G restricts the matching to only a few candidate superpixels, and yet allowing certain amount of inter-image variations in the common objects. A small value of p ensures a fast computation during subgraph matching, still allowing the soft matches to be recovered during the region co-growing phase in Sect. 2.3. Thus the product graph size as well as the possibility of spurious matching reduces. For example, the size of the product graph for many-to-many matching is $O(|G^1||G^2|)$, and the choice of p in the matching process reduces the size to $O(p(|G^1| + |G^2|))$, while the additional use of the threshold t_G makes it $O(\alpha p(|G^1| + |G^2|))$ with $0 < \alpha \ll 1$. This reduces the computation drastically.



(a) RAGs G^1,G^2 (b) RAGs G^1,G^2, product graph H^{12}

Fig. 4. Illustration of maximum common subgraph computation. (a) Two RAGs G^1 and G^2 are obtained from images I^1 and I^2, respectively. The set of nodes $\mathcal{M}^1$, $\mathcal{M}^2$ and edges in the maximum common subgraphs M^1 and M^2 of G^1 and G^2, respectively, are highlighted (*in blue*). (b) Illustration of requirement of condition **C.2** of edge assignment in Sect. 2.2. Let the product nodes in the product graph obtained from the RAGs G^1, G^2 be $\left(v_1^1, v_1^2\right)$ and $\left(v_3^1, v_3^2\right)$ due to the constraint defined in Eq. (2). They are connected by an edge according to condition **C.2** although condition **C.1** is not satisfied. It is easy to derive that the nodes in the MCS are $\left(v_1^1, v_1^2\right)$ and $\left(v_3^1, v_3^2\right)$. This shows that multiple disconnected but similar objects can be co-segmented (Color figure online)

In H^{12}, we add an edge between two product nodes $\left(v_{k_1}^1, v_{l_1}^2\right)$ and $\left(v_{k_2}^1, v_{l_2}^2\right)$ with $k_1 \neq k_2 \wedge l_1 \neq l_2$ if
C.1. $e_{k_1 k_2}^1 \in G^1 \wedge e_{l_1 l_2}^2 \in G^2$, or **C.2.** $e_{k_1 k_2}^1 \notin G^1 \wedge e_{l_1 l_2}^2 \notin G^2$,
where $\wedge$ stands for the logical AND operation. As edges in the product graph H^{12} represent matching, the edges in its complement graph H_C^{12} and the product nodes which they are incident on represent non-matching. These product nodes are essentially the minimum vertex cover (MVC) of H_C^{12}. The MVC of a graph is the smallest set of vertices required to cover all the edges in that graph [32]. So, the set of product nodes ($\mathcal{M}^{12}$) other than this MVC represents the *left* matched product nodes, known as the maximal clique of H^{12} in the literature [31] (i.e. the reference graph G^1 being matched to G^2). Similarly, we obtain the *right* matched product nodes $\mathcal{M}^{21}$ from H^{21} (i.e. the reference graph G^2 being matched to G^1).

Let $\mathcal{M} \triangleq \mathcal{M}^{12} \cup \mathcal{M}^{21}$, and $\mathcal{M}^1 \subseteq \mathcal{V}^1$ and $\mathcal{M}^2 \subseteq \mathcal{V}^2$ be the set of nodes (see Fig. 3(b),(f)) in the corresponding common subgraphs M^1 in G^1 and M^2 in G^2, respectively, with

$$\mathcal{M}^1 = \{v_k^1 | (v_k^1, v_l^2) \in \mathcal{M}\} \quad \text{and} \quad \mathcal{M}^2 = \{v_l^2 | (v_k^1, v_l^2) \in \mathcal{M}\}, \tag{3}$$

and they correspond to the matched regions (nodes) in I^1 and I^2, respectively (see Fig. 3(c),(g)). Note M^1 and M^2 are induced subgraphs. Here $|\mathcal{M}^1|$ and $|\mathcal{M}^2|$ need not be equal due to reasons mentioned earlier. The maximum common subgraphs for the example graphs G^1 and G^2, respectively, are highlighted in Fig. 4(a). Condition **C.1** alone cannot perform co-segmentation of multiple objects, if present, that are not connected to each other. The addition of condition **C.2** helps to achieve this. We illustrate this using an example. In Fig. 4(b), let the disconnected nodes v_1^1 and v_3^1 in G^1 be similar to the disconnected nodes v_1^2 and v_3^2 in G^2, respectively. Here use of condition **C.1** alone will co-segment either (i) v_1^1 and v_1^2 or (ii) v_3^1 and v_3^2, but not both. But using both conditions, we will be able to co-segment both (i) v_1^1 and v_1^2 and (ii) v_3^1 and v_3^2 which is the correct result. In the case of product nodes $\left(v_k^1, v_{l_1}^2\right)$ and $\left(v_k^1, v_{l_2}^2\right)$ (i.e. $k_1 = k_2$), we add an edge if $e_{l_1 l_2}^2$ exists.

As we have obtained an MCS with the constraints on the choice of similarity threshold t_G and the maximal many-to-one matching parameter p, $\mathcal{M}^1$ and $\mathcal{M}^2$ may not contain all the nodes within the co-segmented objects. So, we iteratively grow these matched regions in both images simultaneously based on neighborhood feature similarities across both images till convergence to obtain the complete co-segmented objects as explained next.

2.3 Region Co-growing

In the MCS algorithm of Sect. 2.2, our goal is to keep the product graph size small to reduce the computation even if the subgraphs obtained at the MCS output do not cover the complete objects. We do so by using a relatively large value of t_G and a small value of p. If two superpixels do match, it is expected to find matching of superpixels in their neighborhoods when the object is partially recovered. We can perform region co-growing on the regions $\mathcal{M}^1$ and $\mathcal{M}^2$ obtained from the MCS matching algorithm using them as seeds to obtain the complete objects. So, even if an image pair contains common objects of different size (and number of superpixels), they are completely detected after region co-growing. Moreover, obtaining an MCS with a small number of nodes followed by region co-growing is computationally less intensive than solving for a large product graph.

As we are interested in co-segmentation, we jointly and iteratively grow $\mathcal{M}^1$ and $\mathcal{M}^2$. Here our objective is to find nodes, in the *neighborhood of already matched regions (nodes)* in one image, having high feature similarity to the *already matched regions (nodes)* in the other image. We use these neighborhood nodes for region growing. Let $\mathcal{N}_{\mathcal{M}^i}$ denote the set of neighbors of $\mathcal{M}^i$, with $\mathcal{N}_{\mathcal{M}^i} = \bigcup_{v \in \mathcal{M}^i} \{u \in \mathtt{N}(v)\}$ for $i = 1, 2$, where $\mathtt{N}(\cdot)$ denotes neighborhood. In every iteration-t, we append a certain set of neighbors $\mathcal{N}_{s_2}^{(t)} \subseteq \mathcal{N}_{\mathcal{M}^2}^{(t)}$ to $\mathcal{M}^{2,(t)}$ if they

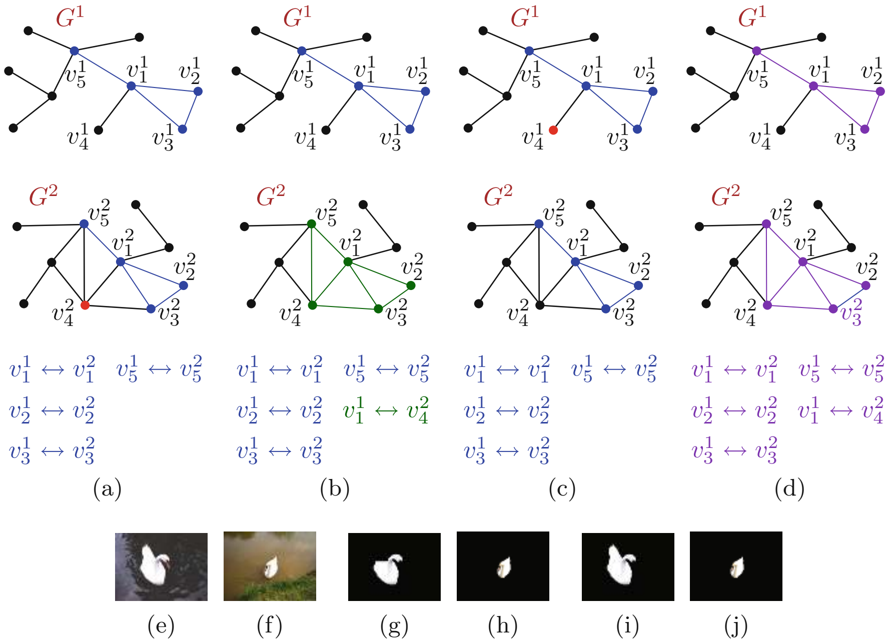

G^1
v_5^1
v_1^1
v_2^1
v_4^1
v_3^1
G^2
v_5^2
v_1^2
v_2^2
v_4^2
v_3^2
$v_1^1 \leftrightarrow v_1^2$ $v_5^1 \leftrightarrow v_5^2$
$v_2^1 \leftrightarrow v_2^2$ $v_1^1 \leftrightarrow v_4^2$
$v_3^1 \leftrightarrow v_3^2$
(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)
(i)
(j)

Fig. 5. Illustration of region co-growing. (a) The set of nodes $\mathcal{M}^1$ and $\mathcal{M}^2$ at the MCS outputs of the graphs G^1 and G^2, with v_1^1, v_2^1, v_3^1, v_5^1 match v_1^2, v_2^2, v_3^2, v_5^2, respectively. The nodes in MCSs are $\mathcal{M}^{1,(t)}$ and $\mathcal{M}^{2,(t)}$ (*blue*) at $t = 1$. To grow $\mathcal{M}^{2,(t)}$, we compare feature similarities of each node, e.g. v_4^2 (*red*), in the neighborhood of $\mathcal{M}^{2,(t)}$ to all the nodes in $\mathcal{M}^{1,(t)}$. (b) $\mathcal{M}^{2,(t+1)}$ (*green*) has been obtained by growing $\mathcal{M}^{2,(t)}$ and v_4^2 has been included in the set due to high feature similarity with v_1^1. (c) To grow $\mathcal{M}^{1,(t)}$, we compare feature similarities of each node, e.g. v_4^1 (*red*), in the neighborhood of $\mathcal{M}^{1,(t)}$ to all the nodes in $\mathcal{M}^{2,(t)}$. (d) The set of matched nodes (*purple*) after iteration-1 of region growing, assuming no match has been found for v_4^1. Effectiveness of region co-growing is illustrated in (e–j). ((e),(f)) Input images. ((g),(h)) The object regions in the images obtained from the MCS algorithm. As the two objects are of different size, the larger object (of image (e)) is not completely detected. ((i),(j)) Co-segmented objects are completely obtained after performing region growing on the initially matched regions (Color figure online)

have high inter-image feature similarity to the nodes in $\mathcal{M}^{1,(t)}$. Similarly, we append a certain set of neighbors ($\mathcal{N}_{s^1}^{(t)} \subseteq \mathcal{N}_{\mathcal{M}^1}^{(t)}$) of $\mathcal{M}^{1,(t)}$ to it. The matched region sets are updated as shown in the program.

We compute the weighted feature similarity $\mathcal{S}'_f(v_k^1, v_l^2)$ between a node v_k^1 in $\mathcal{M}^{1,(t)}$ and a node v_l^2 in $\mathcal{N}_{\mathcal{M}^2}^{(t)}$ as a function of (i) their feature similarity $\mathcal{S}_f(v_k^1, v_l^2)$ of Eq. (1) and (ii) the average feature similarity between their neighbors ($\mathcal{N}_{v_k^1}$ and $\mathcal{N}_{v_l^2}$) that are already in the set of matched regions i.e., in $\mathcal{M}^{1,(t)}$ and $\mathcal{M}^{2,(t)}$, respectively. Thus the similarity measure for region growing has an additional measure of neighborhood similarity compared to the measure used for graph matching in Sect. 2.2. We illustrate the proposed region co-growing method using Fig. 5. In Fig. 5(a), to compute the weighted feature similarity $\mathcal{S}'_f(v_1^1, v_4^2)$ between a matched node $v_1^1 \in \mathcal{M}^1$ and an unmatched neighboring node $v_4^2 \in \mathcal{N}_{\mathcal{M}^2}$ while growing $\mathcal{M}^2$, we consider their feature similarity $\mathcal{S}_f(v_1^1, v_4^2)$ and the feature similarity between the respective (matched) neighboring node pairs ($v_3^1 \in \mathcal{M}^1 \cap \mathcal{N}_{v_1^1}$, $v_3^2 \in \mathcal{M}^2 \cap \mathcal{N}_{v_4^2}$) and ($v_5^1 \in \mathcal{M}^1 \cap \mathcal{N}_{v_1^1}$, $v_5^2 \in \mathcal{M}^2 \cap \mathcal{N}_{v_4^2}$). We ignore the neighboring nodes $v_2^1 \in \mathcal{M}^1$ and $v_1^2 \in \mathcal{M}^2$ assuming they have not been matched to each other. Similarly while growing $\mathcal{M}^1$, we compute the weighted feature similarity between $v_4^1 \in \mathcal{N}_{\mathcal{M}^1}$ and the nodes in $\mathcal{M}^2$ (see Fig. 5(c)).

If a node in G^i has less number of *already matched neighbors*, it is more likely to be part of the background in I^i. So, less importance should be given to it even if it has relatively high feature similarities with the nodes within the object in I^j. In Fig. 5(a), the unmatched node $v_4^2 \in \mathcal{N}_{\mathcal{M}^2}$ has three matched neighboring nodes v_1^2, v_3^2 and v_5^2, whereas in Fig. 5(c), the unmatched node $v_4^1 \in \mathcal{N}_{\mathcal{M}^1}$ has one matched neighboring node v_1^1. The weighted similarity measure $\mathcal{S}'_f(v_k^1, v_l^2)$ is computed as

$$\mathcal{S}'_f(v_k^1, v_l^2) = \omega_{\mathcal{N}} \mathcal{S}_f\left(v_k^1, v_l^2\right) + (1 - \omega_{\mathcal{N}})\left(1 - (Q)^{|U_1||U_2|}\right)\ , \text{ where} \tag{4}$$

$$Q = \frac{1}{|U_1||U_2|} \sum_{u_1 \in U_1, u_2 \in U_2} (1 - \mathcal{S}_f\left(u_1, u_2\right)) \mathbb{1}\left(u_1, u_2\right) . \tag{5}$$

Here $\omega_{\mathcal{N}}$ is an appropriately chosen weight, $U_1 = \mathcal{N}_{v_k^1} \cap \mathcal{M}^{1,(t)}, U_2 = \mathcal{N}_{v_l^2} \cap \mathcal{M}^{2,(t)}$, $|\cdot|$ indicates cardinality and the indicator function $\mathbb{1}\left(u_1, u_2\right) = 1$ if the MCS matching algorithm yields a match between nodes u_1 and u_2 and $\mathbb{1}\left(u_1, u_2\right) = 0$ otherwise. The first term in Eq. (4) is the feature similarity, as defined earlier in Eq. (1), between the two nodes in consideration, v_k^1 and v_l^2. The second term is a measure of inter-image feature similarity among neighbors of v_k^1 and v_l^2. As desired, this value increases as the number of neighbors that have *already been matched* increases.

```
Region co-growing program
   begin
     repeat until convergence
       $\mathcal{N}_{s_1}^{(t)} := \bigcup_{v_l^2 \in \mathcal{M}^{2,(t)}} \{v_k^1 \in \mathcal{N}_{\mathcal{M}^1}^{(t)} | \mathcal{S}_f{}'(v_k^1, v_l^2) > t_G\}$;
       $\mathcal{N}_{s_2}^{(t)} := \bigcup_{v_k^1 \in \mathcal{M}^{1,(t)}} \{v_l^2 \in \mathcal{N}_{\mathcal{M}^2}^{(t)} | \mathcal{S}_f{}'(v_k^1, v_l^2) > t_G\}$;
         Region growing in $G^1$: $\mathcal{M}^{1,(t+1)} := \mathcal{M}^{1,(t)} \cup \mathcal{N}_{s_1}^{(t)}$;
         Region growing in $G^2$: $\mathcal{M}^{2,(t+1)} := \mathcal{M}^{2,(t)} \cup \mathcal{N}_{s_2}^{(t)}$;
end.
```

The region co-growing algorithm converges when $\mathcal{M}^{1,(t)} = \mathcal{M}^{1,(t-1)}$, $\mathcal{M}^{2,(t)} = \mathcal{M}^{2,(t-1)}$. We use $\mathcal{M}^{1*} \triangleq \mathcal{M}^{1,(t)}$ and $\mathcal{M}^{2*} \triangleq \mathcal{M}^{2,(t)}$ (see Fig. 2) to extract common objects completely from I^1 and I^2, respectively (also see Fig. 3(d),(h)). The example in Fig. 5(e–j) shows that region growing helps to completely detect common objects of different size. The larger object has been partially detected from MCS (Fig. 5(g)) and is fully recovered after region co-growing (Fig. 5(i)).

Relevance Feedback. The weight $\omega_{\mathcal{N}}$ in Eq. (4) is used to compute the similarity $\mathcal{S}'_f(v_1, v_2)$ between a pair of nodes from G^1 and G^2 during region co-growing from two constituent similarity measures. Instead of using heuristics, we use relevance feedback [33] to quantify the importance of the neighborhood information and find $\omega_{\mathcal{N}}$. It has been used by Rui *et al.* [34], among many others, to find optimal weights while combining different features for various applications.

2.4 Common Background Elimination

In the co-segmentation problem, we are interested in common foreground segmentation and not in common background segmentation. If an image pair contains background regions with similar features such as the sky or water body, the co-segmentation algorithm, as described so far, will also include the background regions as part of the co-segmented objects. Moreover, inclusion of similar background nodes will unnecessarily increase the size of product graph. We use the method of Zhu *et al.* [35] to obtain an estimate of the probability of a superpixel belonging to the background to eliminate it while building the product graphs and region co-growing. This method is briefly described next.

As we normally capture images keeping the objects of interest at the center of the image, the superpixels at the image boundary are more likely to be part of the background. In addition to the boundary superpixels, some superpixels not at the image boundary will also belong to the background and they have features similar to the boundary superpixels ($\mathcal{B}$). The boundary connectivity $\mathcal{C}_{\mathcal{B}}(v_i)$ of a superpixel v_i is defined as the fraction of its cumulative similarity to all superpixels in the image present at the image boundary. The probability of a superpixel v_i belonging to the background is given by [35] $\mathcal{P}_{\mathcal{B}}(v_i) = 1 - \exp\left(-\frac{1}{2}(\mathcal{C}_{\mathcal{B}}(v_i))^2\right)$. We compute this probability for all superpixels in images I^1 and I^2 independently, and discard the superpixels that have $\mathcal{P}_{\mathcal{B}}(v_i) > t_{\mathcal{B}}$, where $t_{\mathcal{B}}$ is a threshold, while constructing the graphs G^1 and G^2.

3 Pyramidal Image Co-segmentation

With an increase in image size for a well textured scene, the number of superpixels increases and the graph size becomes larger. To maintain the computational efficiency of the proposed co-segmentation algorithm for high resolution images, we use a pyramidal representation of images. We compute the maximum common subgraph at the coarsest level as it contains the least number of superpixels (nodes). This reduces the computation of the MCS matching algorithm. Then we perform region co-growing at every finer level in the pyramid. This avoids any localization error that might have occurred if both MCS computation and

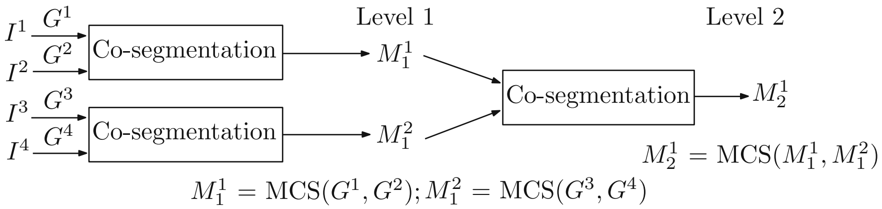


Fig. 6. Illustration of image co-segmentation method for the case of four images. Input images I^1–I^4 are represented as graphs G^1–G^4. Co-segmentation of I^1 and I^2 yields MCS M_1^1. Co-segmentation of I^3 and I^4 yields MCS M_1^2. Co-segmentation of M_1^1 and M_1^2 yields MCS M_2^1 that represents the co-segmented objects in images I^1–I^4. Here, M_l^j denotes the j-th subgraph at level l

region co-growing are performed at the coarsest level and that output is resized to the input image size.

Let the input image pair I^1 and I^2 be successively downsampled (by 2) P times with I_P^1 and I_P^2 being the coarsest level image pairs, and let $I_1^1 = I^1$ and $I_1^2 = I^2$. We segment the set of downsampled images into superpixels of same sizes. So, I_P^1 and I_P^2 contain the least number of superpixels. Let $\mathcal{M}_P^1$ and $\mathcal{M}_P^2$ be the set of matched superpixels in I_P^1 and I_P^2 obtained from the MCS matching algorithm. To find the matched superpixels in I_{P-1}^i, we map every superpixel in $\mathcal{M}_P^i$ to certain superpixels in I_{P-1}^i based on the co-ordinates of the pixels inside the superpixels. A superpixel $v \in \mathcal{M}_P^i$ is mapped to a superpixel $u \in I_{P-1}^i$ if u has the highest overlap with the twice-scaled co-ordinates of pixels of v among all superpixels in $\mathcal{M}_P^i$. Then we perform region co-growing on the mapped superpixels in I_{P-1}^1 and I_{P-1}^2 as discussed in Sect. 2.3 and obtain the matched superpixel sets $\mathcal{M}_{P-1}^1$ in I_{P-1}^1 and $\mathcal{M}_{P-1}^2$ in I_{P-1}^2. We repeat this process for subsequent levels and obtain the final matched superpixel sets $\mathcal{M}_1^1$ and $\mathcal{M}_1^2$ that constitute the co-segmented objects in I_1^1 and I_1^2, respectively.

4 Co-segmentation of Multiple Images

Here we extend the proposed co-segmentation method to multiple images, instead of finding matches over just an image pair. This is more relevant in analyzing crowd sourced images in an event or a touristic location. If we try to obtain the MCS of K number of images simultaneously, the size of the product graph grows drastically $\left(O\left(p^{K-1}|G^1|^{K-1}\right)\right.$, assuming same cardinality of every graph for simplicity) making the proposed algorithm incomputatble. We propose a different scheme to convert this into an algorithm dealing with $K-1$ separate product graph pairs of size $O(\alpha p(|G^1|+|G^2|))$ using a hierarchical scheme involving pair-wise co-segmentation over a binary tree structured organization of the constituent images (see Fig. 6).

To co-segment a set of K images I^1, I^2, ..., I^K, we perform $L = \lceil \log_2 K \rceil$ levels of co-segmentation. Let G^1, G^2, ..., G^K denote the graphs of the respective input images and M_l^j denotes the j-th subgraph at level l. We independently compute the co-segmentation outputs of image pairs (I^1,I^2), (I^3,I^4), ...,

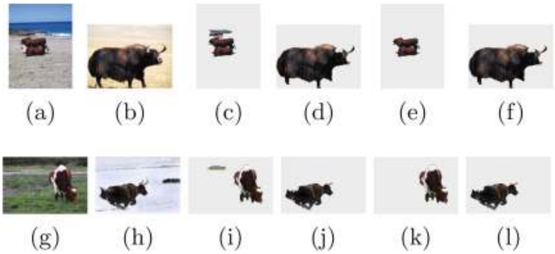


Fig. 7. Illustration of image co-segmentation from four images. Co-segmentation of the image pair in ((a),(g)) yields outputs ((c),(i)), and the pair in ((b),(h)) yields outputs ((d),(j)). These outputs are co-segmented to obtain the final outputs ((e),(f),(k),(l)). Notice how small background regions present in ((c),(i)) have been removed in ((e),(k)) after the second round of co-segmentation (Color figure online)

(I^{K-1}, I^K). Let M_1^1, M_1^2, $\ldots$, $M_1^{K/2}$ be the resulting subgraphs for pairwise co-segmentation at level $l = 1$ in Fig. 6. Then we again compute MCS of each pair (M_1^1, M_1^2), (M_1^3, M_1^4), $\ldots$, $(M_1^{K/2-1}, M_1^{K/2})$ and then obtain the corresponding co-segmentation map M_2^1, M_2^2, $\ldots$ for $l = 2$. We repeat this process until we obtain the final co-segmentation map M_L^1 at level $l = L$. Figure 6 shows the block diagram when considering co-segmentation for four images ($L = 2$).

The advantage with this approach is that the computational complexity greatly reduces after the first level of operation as $|M_l^j| \ll |G^i|$ in any level l and the graph size reduces at every subsequent level. We need to perform co-segmentation at most $K - 1$ times for K input images i.e., the complexity increases linearly with the number of images to be co-segmented. Also if at any level any MCS is null, we can stop the algorithm and conclude that there is no common object over the image set. It may be noted that due to non-commutativity, as the MCS output of two graphs at any level corresponds to two different matched regions, we may choose any of them for the next level of co-segmentation. Figure 7 shows an example of co-segmentation for four images. Co-segmentation outputs M^1, M^2 in (c),(i) and M^3, M^4 in (d),(j) at level $l = 1$ are obtained from input image pairs I^1, I^2 in (a),(g) and I^3, I^4 in (b),(h), respectively. Final co-segmented objects are in (e),(k),(f),(l).

5 Experimental Results

Choice of parameters: For an $N_1 \times N_2$ image (in the coarsest level), we experimentally choose the number of superpixels to be $N = \min(100, 0.004 N_1 N_2)$. This limits the size of the graph to be under 100. The maximal many-to-one matching is limited to $p = 2$ as a trade-off between the size of the product graph and possible reduction in seeding the co-segmentation process before region co-growing. We have adaptively chosen the inter-image feature similarity threshold t_G in Eq. (2) to ensure that the size of the product graphs, H^{12} and H^{21}, is at most 40–50 due to computational restrictions. In Sect. 2.4, we have set the threshold

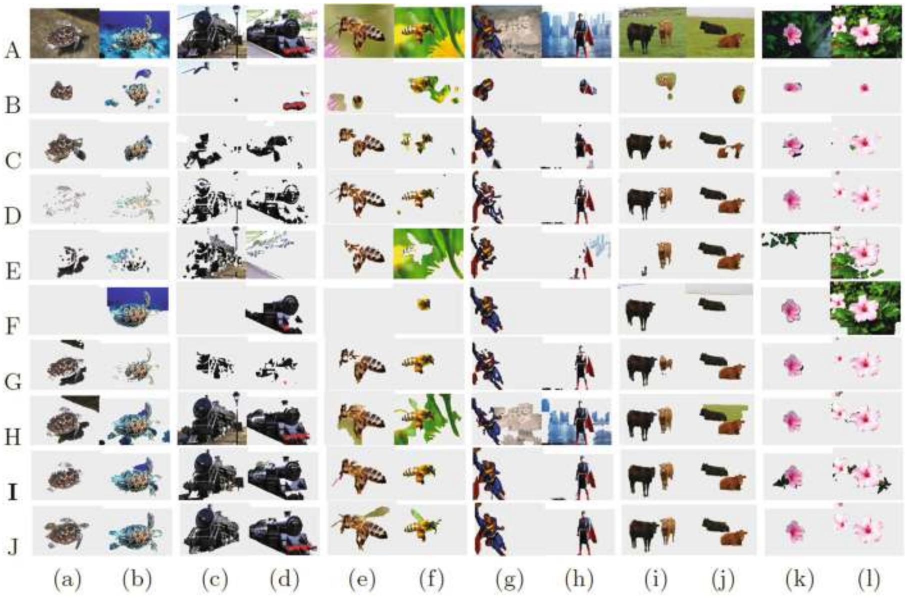


Fig. 8. Visual comparison of result of image co-segmentation. Co-segmentation outputs obtained from the image pairs in (a,b), (c,d), (e,f), (g,h), (i,j) and (k,l) of Row A using [22], [21], [23], [13], [3], [24], [16] and the proposed method (**PR**) are shown in Rows B–H and Row I, respectively. Ground-truth data are shown in Row J (Color figure online)

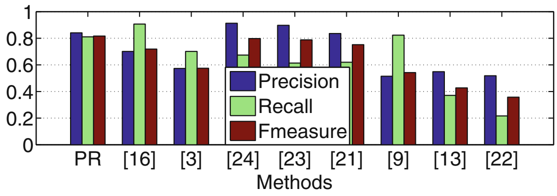


Fig. 9. Comparison of precision, recall and F-measure values of the proposed method (**PR**) with [16], [3], [24], [23], [21], [9], [13], [22] on the image pair dataset [21] (Color figure online)

for background probability as $t_{\mathcal{B}} = 0.75 \max(\{\mathcal{P}_{\mathcal{B}}(v_i), \forall v_i \in I\})$ to discard the possible background superpixels in the proposed co-segmentation algorithm.

Results: We have tested our algorithm with images selected from five datasets. Results for MSRC dataset [3] and the image pair dataset [21] are provided here. Results for the iCoseg dataset [7], the flower dataset [36] and the Weizmann horse dataset [37] are in the supplementary material. We first visually analyze results of some of the existing methods and compare with the results of the proposed method (PR) on images containing a single object (Fig. 8(a)–(h)) as well as

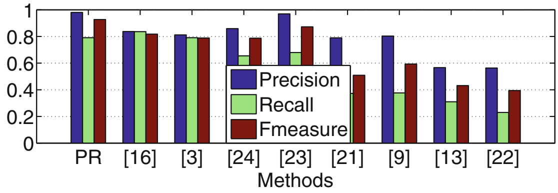


Fig. 10. Comparison of mean precision, recall and F-measure values of the proposed method (**PR**) with [16], [3], [24], [23], [21], [9], [13], [22] on images selected from 'cow', 'duck', 'dog', 'flower' and 'sheep' classes in the MSRC dataset [3] (Color figure online)

multiple objects (Fig. 8(i)–(l)). We show visual comparison of our result with the methods [22], [21], [23], [13], [3], [24], [16] in Fig. 8 (Rows B–H, respectively) and results demonstrate the superior performance of PR (Row I). For the input image pair in Fig. 8(i),(j), the methods in [22], [13], [3] detect only one of the two common objects (shown in Rows B, E, F). Most of the outputs of [22], [21], [23], [13] (shown in Rows B–E) contain discontiguous and spurious objects. Further, in most cases the common objects are either under-segmented or over-segmented. Although the method of Rubinstein *et al.* [3] yields contiguous objects, they very often fail to detect any object from both images (Row F in Fig. 8(a),(c),(e),(h)). However, the proposed method yields the entire object as a single entity with very little over or under-segmentation.

The quality of the proposed co-segmentation output is also quantitatively measured using precision, recall and F-measure, as used in earlier works e.g., [21]. These metrics are computed by comparing the segmentation output mask with the ground-truth provided in the database. Precision is the ratio of the number of correctly detected object pixels to the number of detected object pixels. This penalizes for classifying background pixels as object. Recall is the ratio of the number of correctly detected object pixels to the number of ground-truth pixels. This penalizes for not detecting all pixels of the object. F-measure is the weighted harmonic mean (we use weight = 0.3) of precision and recall. We compare these measures of the proposed method with those of methods in [16], [3], [24], [23], [21], [9], [13], [22] on the image pair dataset [21] (Fig. 9) and the MSRC dataset [3] (Fig. 10). Results show that the proposed method outperforms others. Moreover our precision and recall values are very close, as it should be, and yet being very high. This indicates that the proposed method reduces both false positives and false negatives. While the methods in [23], [24] (Fig. 9) also have high precision values, the recall rate is significantly inferior to the proposed method. Method [16] has a good recall measure, but the precision is quite low. In order to compare the computation time of the proposed method, we execute the competing methods also on the same system. Table 1 shows that the proposed method is significantly faster than existing methods [16], [3], and the advantage is more noticeable when the image size increases.

Table 1. Comparison of computation time (in seconds) of the proposed method (**PR**) with [16], [3], as the image pair size (86×128 and 98×128) increases by shown factors

Method	Increase in image size				
	1×1	2×2	$2^2 \times 2^2$	$2^3 \times 2^3$	$2^4 \times 2^4$
[16]	32.65	51.63	78.83	163.61	820.00
[3]	1.80	6.00	25.20	107.40	475.80
PR	1.54	2.08	2.94	5.69	13.90

6 Conclusions

In this paper, we have proposed a novel and computationally efficient image co-segmentation algorithm based on the concept of maximum common subgraph matching. Performing region co-growing on a small number of nodes (seeds) obtained as the MCS output and incorporating them in a pyramidal co-segmentation makes the proposed method computationally very efficient. The proposed method can handle variation in shape, size, orientation and texture in the common object among constituent images. It can also deal with presence of multiple common objects, unlike some existing methods.

References

1. Rother, C., Minka, T., Blake, A., Kolmogorov, V.: Cosegmentation of image pairs by histogram matching - incorporating a global constraint into MRFs. In: Proceedings of CVPR, vol. 1, pp. 993–1000 (2006)
2. Rubio, J.C., Serrat, J., López, A., Paragios, N.: Unsupervised co-segmentation through region matching. In: Proceedings of CVPR, pp. 749–756 (2012)
3. Rubinstein, M., Joulin, A., Kopf, J., Liu, C.: Unsupervised joint object discovery and segmentation in internet images. In: Proceedings of CVPR, pp. 1939–1946, June 2013
4. Mukherjee, L., Singh, V., Dyer, C.: Half-integrality based algorithms for cosegmentation of images. In: Proceedings of CVPR, pp. 2028–2035, June 2009
5. Hochbaum, D., Singh, V.: An efficient algorithm for co-segmentation. In: Proceedings of ICCV, pp. 269–276, September 2009
6. Vicente, S., Kolmogorov, V., Rother, C.: Cosegmentation revisited: models and optimization. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 465–479. Springer, Heidelberg (2010)
7. Batra, D., Kowdle, A., Parikh, D., Luo, J., Chen, T.: iCoseg: Interactive co-segmentation with intelligent scribble guidance. In: Proceedings of CVPR, pp. 3169–3176, June 2010
8. Collins, M., Xu, J., Grady, L., Singh, V.: Random walks based multi-image segmentation: Quasiconvexity results and GPU-based solutions. In: Proceedings of CVPR, pp. 1656–1663, June 2012
9. Joulin, A., Bach, F., Ponce, J.: Discriminative clustering for image co-segmentation. In: Proceedings of CVPR, pp. 1943–1950, June 2010

10. Joulin, A., Bach, F., Ponce, J.: Multi-class cosegmentation. In: Proceedings of CVPR, pp. 542–549, June 2012
11. Mukherjee, L., Singh, V., Peng, J.: Scale invariant cosegmentation for image groups. In: Proceedings of CVPR, pp. 1881–1888, June 2011
12. Vicente, S., Rother, C., Kolmogorov, V.: Object cosegmentation. In: Proceedings of CVPR, pp. 2217–2224, June 2011
13. Kim, G., Xing, E., Fei-Fei, L., Kanade, T.: Distributed cosegmentation via submodular optimization on anisotropic diffusion. In: Proceedings of ICCV, pp. 169–176, November 2011
14. Kim, G., Xing, E.: On multiple foreground cosegmentation. In: Proceedings of CVPR, pp. 837–844, June 2012
15. Meng, F., Li, H., Liu, G., Ngan, K.N.: Object co-segmentation based on shortest path algorithm and saliency model. IEEE Trans. Multimedia **14**(5), 1429–1441 (2012)
16. Lee, C., Jang, W.D., Sim, J.Y., Kim, C.S.: Multiple random walkers and their application to image cosegmentation. In: Proceedings of CVPR, pp. 3837–3845, June 2015
17. Tao, W., Li, K., Sun, K.: Sacoseg: Object cosegmentation by shape conformability. IEEE Trans. Image Process. **24**(3), 943–955 (2015)
18. Jerripothula, K., Cai, J., Meng, F., Yuan, J.: Automatic image co-segmentation using geometric mean saliency. In: Proceedings of ICIP, pp. 3277–3281, October 2014
19. Chen, M., Velasco-Forero, S., Tsang, I., Cham, T.J.: Objects co-segmentation: propagated from simpler images. In: Proceedings of ICASSP, pp. 1682–1686 (2015)
20. Chang, K.Y., Liu, T.L., Lai, S.H.: From co-saliency to co-segmentation: An efficient and fully unsupervised energy minimization model. In: Proceedings of CVPR, pp. 2129–2136, June 2011
21. Li, H., Ngan, K.N.: A co-saliency model of image pairs. IEEE Trans. Image Process. **20**(12), 3365–3375 (2011)
22. Chen, H.T.: Preattentive co-saliency detection. In: Proceedings of ICIP, pp. 1117–1120, September 2010
23. Fu, H., Cao, X., Tu, Z.: Cluster-based co-saliency detection. IEEE Trans. Image Process. **22**(10), 3766–3778 (2013)
24. Cao, X., Tao, Z., Zhang, B., Fu, H., Feng, W.: Self-adaptively weighted co-saliency detection via rank constraint. IEEE Trans. Image Process. **23**(9), 4175–4186 (2014)
25. Tan, Z., Wan, L., Feng, W., Pun, C.M.: Image co-saliency detection by propagating superpixel affinities. In: Proceedings of ICASSP, pp. 2114–2118, May 2013
26. Liu, Z., Zou, W., Li, L., Shen, L., Meur, O.: Co-saliency detection based on hierarchical segmentation. IEEE Signal Process. Lett. **21**(1), 88–92 (2014)
27. Li, Y., Fu, K., Liu, Z., Yang, J.: Efficient saliency-model-guided visual co-saliency detection. IEEE Signal Process. Lett. **22**(5), 588–592 (2015)
28. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
29. Colannino, J., Damian, M., Hurtado, F., Langerman, S., Meijer, H., Ramaswami, S., Souvaine, D., Toussaint, G.: Efficient many-to-many point matching in one dimension. Graphs Comb. **23**(1), 169–178 (2007)
30. Madry, A.: Navigating central path with electrical flows: from flows to matchings, and back. In: IEEE Annual Symposium on FOCS, pp. 253–262 (2013)
31. Koch, I.: Enumerating all connected maximal common subgraphs in two graphs. Theor. Comput. Sci. **250**(1), 1–30 (2001)

32. Cormen, T.H., Stein, C., Rivest, R.L., Leiserson, C.E.: Introduction to Algorithms, 2nd edn. McGraw-Hill Higher Education, New York (2001)
33. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
34. Rui, Y., Huang, T., Ortega, M., Mehrotra, S.: Relevance feedback: a power tool for interactive content-based image retrieval. IEEE Trans. Circ. Syst. Video Technol. **8**(5), 644–655 (1998)
35. Zhu, W., Liang, S., Wei, Y., Sun, J.: Saliency optimization from robust background detection. In: Proceedings of CVPR, pp. 2814–2821, June 2014
36. Nilsback, M.E., Zisserman, A.: A visual vocabulary for flower classification. In: Proceedings of CVPR, vol. 2, pp. 1447–1454 (2006)
37. Borenstein, E., Ullman, S.: Combined top-down/bottom-up segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **30**(12), 2109–2125 (2008)