

# Interactive Image Segmentation Using Constrained Dominant Sets

Eyasu Zemene and Marcello Pelillo

Ca' Foscari University of Venice, Italy  
`{eyasu.zemene, pelillo}@unive.it`

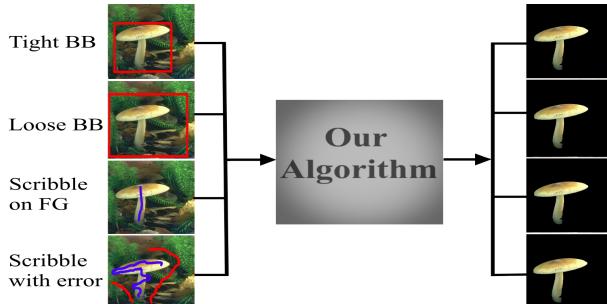
**Abstract.** We propose a new approach to interactive image segmentation based on some properties of a family of quadratic optimization problems related to dominant sets, a well-known graph-theoretic notion of a cluster which generalizes the concept of a maximal clique to edge-weighted graphs. In particular, we show that by properly controlling a regularization parameter which determines the structure and the scale of the underlying problem, we are in a position to extract groups of dominant-set clusters which are constrained to contain user-selected elements. The resulting algorithm can deal naturally with any type of input modality, including scribbles, sloppy contours, and bounding boxes, and is able to robustly handle noisy annotations on the part of the user. Experiments on standard benchmark datasets show the effectiveness of our approach as compared to state-of-the-art algorithms on a variety of natural images under several input conditions.

**Keywords:** Interactive segmentation, dominant sets, quadratic optimization.

## 1 Introduction

User-assisted image segmentation has recently attracted considerable attention within the computer vision community, especially because of its potential applications in a variety of different problems such as image and video editing, medical image analysis, etc. [1,2,3,4,5,6,7,8]. Given an input image and some information provided by a user, usually in the form of a scribble or of a bounding box, the goal is to provide as output a foreground object in such a way as to best reflect the user's intent. By exploiting high-level, semantic knowledge on the part of the user, which is typically difficult to formalize, we are therefore able to effectively solve segmentation problems which would be otherwise too complex to be tackled using fully automatic segmentation algorithms.

Existing algorithms fall into two broad categories, depending on whether the user annotation is given in terms of a scribble or of a bounding box, and supporters of the two approaches have both good reasons to prefer one modality against the other. For example, Wu et al. [3] claim that bounding boxes are the most natural and economical form in terms of the amount of user interaction, and develop a multiple instance learning algorithm that extracts an arbitrary object located inside a tight bounding box at unknown location. Yu et al. [9] also support the bounding-box approach, though their algorithm is different from others in that it does not need bounding boxes tightly enclosing the object of interest, whose production of course increases the annotation burden. They provide an algorithm, based on a Markov Random Field (MRF) energy function, that can handle



**Fig. 1:** **Left:** An input image with different user annotations. Tight bounding box (Tight BB), loose bounding box (Loose BB), a scribble made (only) on the foreground object (Scribbles on FG), scribbles with errors. **Right:** Results of the proposed algorithm.

input bounding box that only loosely covers the foreground object. Xian et al. [10] propose a method which avoids the limitations of existing bounding box methods - region of interest (ROI) based methods, though they need much less user interaction, their performance is sensitive to initial ROI. On the other hand, several researchers, arguing that boundary-based interactive segmentation such as intelligent scissors [8] requires the user to trace the whole boundary of the object, which is usually a time-consuming and tedious process, support scribble-based segmentation. Bai et al. [11], for example, propose a model based on ratio energy function which can be optimized using an iterated graph cut algorithm, which tolerates errors in the user input.

In general, the input modality in an interactive segmentation algorithm affects both its accuracy and its ease of use. Existing methods work typically on a single modality and they focus on how to use that input most effectively. However, as noted recently by Jain and Grauman [12], sticking to one annotation form leads to a suboptimal tradeoff between human and machine effort, and they tried to estimate how much user input is required to sufficiently segment a novel input.

In this paper, we propose a novel approach to interactive image segmentation which can deal naturally with any type of input modality and is able to robustly handle noisy annotations on the part of the user. Figure 1 shows an example of how our system works in the presence of different input annotations. Our approach is based on some properties of a parameterized family of quadratic optimization problems related to dominant-set clusters, a well-known generalization of the notion of maximal cliques to edge-weighted graph which have proven to be extremely effective in a variety of computer vision problems, including (automatic) image and video segmentation [13,14]. In particular, we show that by properly controlling a regularization parameter which determines the structure and the scale of the underlying problem, we are in a position to extract groups of dominant-set clusters which are constrained to contain user-selected elements. We provide bounds that allow us to control this process, which are based on the spectral properties of certain submatrices of the original affinity matrix.

The resulting algorithm has a number of interesting features which distinguishes it from existing approaches. Specifically: 1) it is able to deal in a flexible manner with

both scribble-based and boundary-based input modalities (such as sloppy contours and bounding boxes); 2) in the case of noiseless scribble inputs, it asks the user to provide *only* foreground pixels; 3) it turns out to be *robust* in the presence of input noise, allowing the user to draw, e.g., imperfect scribbles (including background pixels) or loose bounding boxes. Experimental results on standard benchmark datasets show the effectiveness of our approach as compared to state-of-the-art algorithms on a wide variety of natural images under several input conditions.

## 2 Dominant sets and quadratic optimization

In this section we review the basic definitions and properties of dominant sets, as introduced in [13,14]. In the dominant set framework, the data to be clustered are represented as an undirected edge-weighted graph with no self-loops  $G = (V, E, w)$ , where  $V = \{1, \dots, n\}$  is the vertex set,  $E \subseteq V \times V$  is the edge set, and  $w : E \rightarrow R_+^*$  is the (positive) weight function. Vertices in  $G$  correspond to data points, edges represent neighborhood relationships, and edge-weights reflect similarity between pairs of linked vertices. As customary, we represent the graph  $G$  with the corresponding weighted adjacency (or similarity) matrix, which is the  $n \times n$  nonnegative, symmetric matrix  $A = (a_{ij})$  defined as  $a_{ij} = w(i, j)$ , if  $(i, j) \in E$ , and  $a_{ij} = 0$  otherwise. Since in  $G$  there are no self-loops, note that all entries on the main diagonal of  $A$  are zero.

For a non-empty subset  $S \subseteq V$ ,  $i \in S$ , and  $j \notin S$ , define

$$\phi_S(i, j) = a_{ij} - \frac{1}{|S|} \sum_{k \in S} a_{ik} \quad (1)$$

Next, to each vertex  $i \in S$  we assign a weight defined (recursively) as follows:

$$w_S(i) = \begin{cases} 1, & \text{if } |S| = 1, \\ \sum_{j \in S \setminus \{i\}} \phi_{S \setminus \{i\}}(j, i) w_{S \setminus \{i\}}(j), & \text{otherwise.} \end{cases} \quad (2)$$

As explained in [13,14], a positive  $w_S(i)$  indicates that adding  $i$  into its neighbors in  $S$  will increase the internal coherence of the set, whereas in the presence of a negative value we expect the overall coherence to be decreased. Finally, the total weight of  $S$  can be simply defined as

$$W(S) = \sum_{i \in S} w_S(i). \quad (3)$$

A non-empty subset of vertices  $S \subseteq V$  such that  $W(T) > 0$  for any non-empty  $T \subseteq S$ , is said to be a *dominant set* if:

1.  $w_S(i) > 0$ , for all  $i \in S$ ,
2.  $w_{S \cup \{i\}}(i) < 0$ , for all  $i \notin S$ .

It is evident from the definition that a dominant set satisfies the two basic properties of a cluster: internal coherence and external incoherence. Condition 1 indicates that a dominant set is internally coherent, while condition 2 implies that this coherence will

be destroyed by the addition of any vertex from outside. In other words, a dominant set is a maximally coherent data set.

Now, consider the following linearly-constrained quadratic optimization problem:

$$\begin{aligned} & \text{maximize } f(\mathbf{x}) = \mathbf{x}' A \mathbf{x} \\ & \text{subject to } \mathbf{x} \in \Delta \end{aligned} \quad (4)$$

where a prime denotes transposition and

$$\Delta = \left\{ \mathbf{x} \in R^n : \sum_{i=1}^n x_i = 1, \text{ and } x_i \geq 0 \text{ for all } i = 1 \dots n \right\}$$

is the standard simplex of  $R^n$ . In [13,14] a connection is established between dominant sets and the local solutions of (4). In particular, it is shown that if  $S$  is a dominant set then its “weighted characteristics vector,” which is the vector of  $\Delta$  defined as,

$$x_i = \begin{cases} \frac{w_S(i)}{W(S)}, & \text{if } i \in S, \\ 0, & \text{otherwise} \end{cases}$$

is a strict local solution of (4). Conversely, under mild conditions, it turns out that if  $\mathbf{x}$  is a (strict) local solution of program (4) then its “support”

$$\sigma(\mathbf{x}) = \{i \in V : x_i > 0\}$$

is a dominant set. By virtue of this result, we can find a dominant set by first localizing a solution of program (4) with an appropriate continuous optimization technique, and then picking up the support set of the solution found. A generalization of these ideas to hypergraphs has recently been developed in [15].

A simple and effective optimization algorithm to extract a dominant set from a graph is given by the so-called *replicator dynamics*, developed and studied in evolutionary game theory, which are defined as follows:

$$x_i^{(t+1)} = x_i^{(t)} \frac{(A\mathbf{x}^{(t)})_i}{(\mathbf{x}^{(t)})' A (\mathbf{x}^{(t)})} \quad (5)$$

for  $i = 1, \dots, n$ .

### 3 Constrained dominant sets

Let  $G = (V, E, w)$  be an edge-weighted graph with  $n$  vertices and let  $A$  denote as usual its (weighted) adjacency matrix. Given a subset of vertices  $S \subseteq V$  and a parameter  $\alpha > 0$ , define the following parameterized family of quadratic programs:

$$\begin{aligned} & \text{maximize } f_S^\alpha(\mathbf{x}) = \mathbf{x}' (A - \alpha \hat{I}_S) \mathbf{x} \\ & \text{subject to } \mathbf{x} \in \Delta \end{aligned} \quad (6)$$

where  $\hat{I}_S$  is the  $n \times n$  diagonal matrix whose diagonal elements are set to 1 in correspondence to the vertices contained in  $V \setminus S$  and to zero otherwise, and the 0's represent

null square matrices of appropriate dimensions. In other words, assuming for simplicity that  $S$  contains, say, the first  $k$  vertices of  $V$ , we have:

$$\hat{I}_S = \begin{pmatrix} 0 & 0 \\ 0 & I_{n-k} \end{pmatrix}$$

where  $I_{n-k}$  denotes the  $(n-k) \times (n-k)$  principal submatrix of the  $n \times n$  identity matrix  $I$  indexed by the elements of  $V \setminus S$ . Accordingly, the function  $f_S^\alpha$  can also be written as follows:

$$f_S^\alpha(\mathbf{x}) = \mathbf{x}' A \mathbf{x} - \alpha \mathbf{x}'_S \mathbf{x}_S$$

$\mathbf{x}_S$  being the  $(n-k)$ -dimensional vector obtained from  $\mathbf{x}$  by dropping all the components in  $S$ . Basically, the function  $f_S^\alpha$  is obtained from  $f$  by inserting in the affinity matrix  $A$  the value of the parameter  $\alpha$  in the main diagonal positions corresponding to the elements of  $V \setminus S$ .

Notice that this differs markedly, and indeed generalizes, the formulation proposed in [16] for obtaining a hierarchical clustering in that here, only a subset of elements in the main diagonal is allowed to take the  $\alpha$  parameter, the other ones being set to zero. We note in fact that the original (non-regularized) dominant-set formulation (4) [14] as well as its regularized counterpart described in [16] can be considered as degenerate version of ours, corresponding to the cases  $S = V$  and  $S = \emptyset$ , respectively. It is precisely this increased flexibility which allows us to use this idea for finding groups of “constrained” dominant-set clusters.

We now derive the Karush-Kuhn-Tucker (KKT) conditions for program (6), namely the first-order necessary conditions for local optimality (see, e.g., [17]). For a point  $\mathbf{x} \in \Delta$  to be a KKT-point there should exist  $n$  nonnegative real constants  $\mu_1, \dots, \mu_n$  and an additional real number  $\lambda$  such that

$$[(A - \alpha \hat{I}_S) \mathbf{x}]_i - \lambda + \mu_i = 0$$

for all  $i = 1 \dots n$ , and

$$\sum_{i=1}^n x_i \mu_i = 0.$$

Since both the  $x_i$ 's and the  $\mu_i$ 's are nonnegative, the latter condition is equivalent to saying that  $i \in \sigma(\mathbf{x})$  implies  $\mu_i = 0$ , from which we obtain:

$$[(A - \alpha \hat{I}_S) \mathbf{x}]_i \begin{cases} = \lambda, & \text{if } i \in \sigma(\mathbf{x}) \\ \leq \lambda, & \text{if } i \notin \sigma(\mathbf{x}) \end{cases}$$

for some constant  $\lambda$ . Noting that  $\lambda = \mathbf{x}' A \mathbf{x} - \alpha \mathbf{x}'_S \mathbf{x}_S$  and recalling the definition of  $\hat{I}_S$ , the KKT conditions can be explicitly rewritten as:

$$\begin{cases} (Ax)_i - \alpha x_i = \mathbf{x}' A \mathbf{x} - \alpha \mathbf{x}'_S \mathbf{x}_S, & \text{if } i \in \sigma(\mathbf{x}) \text{ and } i \notin S \\ (Ax)_i = \mathbf{x}' A \mathbf{x} - \alpha \mathbf{x}'_S \mathbf{x}_S, & \text{if } i \in \sigma(\mathbf{x}) \text{ and } i \in S \\ (Ax)_i \leq \mathbf{x}' A \mathbf{x} - \alpha \mathbf{x}'_S \mathbf{x}_S, & \text{if } i \notin \sigma(\mathbf{x}) \end{cases} \quad (7)$$

We are now in a position to discuss the main results which motivate the algorithm presented in this paper. Note that, in the sequel, given a subset of vertices  $S \subseteq V$ , the face of  $\Delta$  corresponding to  $S$  is given by:  $\Delta_S = \{\mathbf{x} \in \Delta : \sigma(\mathbf{x}) \subseteq S\}$ .

**Proposition 1.** Let  $S \subseteq V$ , with  $S \neq \emptyset$ . Define

$$\gamma_S = \max_{\mathbf{x} \in \Delta_{V \setminus S}} \min_{i \in S} \frac{\mathbf{x}' A \mathbf{x} - (A \mathbf{x})_i}{\mathbf{x}' \mathbf{x}} \quad (8)$$

and let  $\alpha > \gamma_S$ . If  $\mathbf{x}$  is a local maximizer of  $f_S^\alpha$  in  $\Delta$ , then  $\sigma(\mathbf{x}) \cap S \neq \emptyset$ .

*Proof.* Let  $\mathbf{x}$  be a local maximizer of  $f_S^\alpha$  in  $\Delta$ , and suppose by contradiction that no element of  $\sigma(\mathbf{x})$  belongs to  $S$  or, in other words, that  $\mathbf{x} \in \Delta_{V \setminus S}$ . By letting

$$i = \arg \min_{j \in S} \frac{\mathbf{x}' A \mathbf{x} - (A \mathbf{x})_j}{\mathbf{x}' \mathbf{x}}$$

and observing that  $\sigma(\mathbf{x}) \subseteq V \setminus S$  implies  $\mathbf{x}' \mathbf{x} = \mathbf{x}'_S \mathbf{x}_S$ , we have:

$$\alpha > \gamma_S \geq \frac{\mathbf{x}' A \mathbf{x} - (A \mathbf{x})_i}{\mathbf{x}' \mathbf{x}} = \frac{\mathbf{x}' A \mathbf{x} - (A \mathbf{x})_i}{\mathbf{x}'_S \mathbf{x}_S}.$$

Hence,  $(A \mathbf{x})_i > \mathbf{x}' A \mathbf{x} - \alpha \mathbf{x}'_S \mathbf{x}_S$  for  $i \notin \sigma(\mathbf{x})$ , but this violates the KKT conditions (7), thereby proving the proposition.  $\square$

The following proposition provides an easy-to-compute upper bound for  $\gamma_S$ .

**Proposition 2.** Let  $S \subseteq V$ , with  $S \neq \emptyset$ . Then,

$$\gamma_S \leq \lambda_{\max}(A_{V \setminus S}) \quad (9)$$

where  $\lambda_{\max}(A_{V \setminus S})$  is the largest eigenvalue of the principal submatrix of  $A$  indexed by the elements of  $V \setminus S$ .

*Proof.* Let  $\mathbf{x}$  be a point in  $\Delta_{V \setminus S}$  which attains the maximum  $\gamma_S$  as defined in (8). Using the Rayleigh-Ritz theorem [18] and the fact that  $\sigma(\mathbf{x}) \subseteq V \setminus S$ , we obtain:

$$\lambda_{\max}(A_{V \setminus S}) \geq \frac{\mathbf{x}'_S A_{V \setminus S} \mathbf{x}_S}{\mathbf{x}'_S \mathbf{x}_S} = \frac{\mathbf{x}' A \mathbf{x}}{\mathbf{x}' \mathbf{x}}.$$

Now, define  $\gamma_S(\mathbf{x}) = \max\{(A \mathbf{x})_i : i \in S\}$ . Since  $A$  is nonnegative so is  $\gamma_S(\mathbf{x})$ , and recalling the definition of  $\gamma_S$  we get:

$$\frac{\mathbf{x}' A \mathbf{x}}{\mathbf{x}' \mathbf{x}} \geq \frac{\mathbf{x}' A \mathbf{x} - \gamma_S(\mathbf{x})}{\mathbf{x}' \mathbf{x}} = \gamma_S$$

which concludes the proof.  $\square$

The two previous propositions provide us with a simple technique to determine dominant-set clusters containing user-selected vertices. Indeed, if  $S$  is the set of vertices selected by the user, by setting

$$\alpha > \lambda_{\max}(A_{V \setminus S}) \quad (10)$$

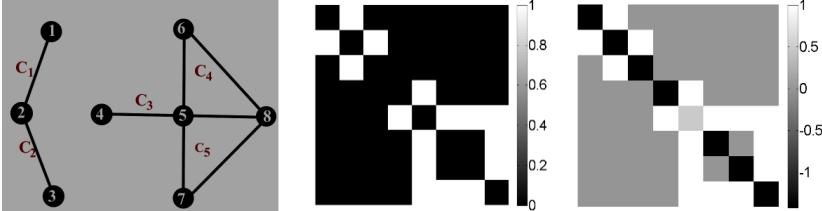


Fig. 2: An example graph (left), corresponding affinity matrix (middle), and scaled affinity matrix built considering vertex 5 as a user constraint (right). Notation  $C_i$  refers to the  $i^{th}$  maximal clique.

we are guaranteed that all local solutions of (6) will have a support that necessarily contains elements of  $S$ . As customary, we can use replicator dynamics or more sophisticated algorithms to find them. Note that this does not necessarily imply that the (support of the) solution found corresponds to a dominant-set cluster of the original affinity matrix  $A$ , as adding the parameter  $-\alpha$  on a portion of the main diagonal intrinsically changes the scale of the underlying problem. However, we have obtained extensive empirical evidence which supports a conjecture which turns out to be very useful for our interactive image segmentation application.

To illustrate the idea, let us consider the case where edge-weights are binary, which basically means that the input graph is unweighted. In this case, it is known that dominant sets correspond to maximal cliques [14]. Let  $G = (V, E)$  be our unweighted graph and let  $S$  be a subset of its vertices. For the sake of simplicity, we distinguish three different situations of increasing generality.

**Case 1.** The set  $S$  is a singleton, say  $S = \{u\}$ . In this case, we know from Proposition 2 that all solutions  $\mathbf{x}$  of  $f_\alpha^S$  over  $\Delta$  will have a support which contains  $u$ , that is  $u \in \sigma(\mathbf{x})$ . Indeed, we conjecture that there will be a unique local (and hence global) solution here whose support coincides with the *union* of all maximal cliques of  $G$  which contain vertex  $u$ .

**Case 2.** The set  $S$  is a clique, not necessarily maximal. In this case, Proposition 2 predicts that all solutions  $\mathbf{x}$  of (6) will contain at least one vertex from  $S$ . Here, we claim that indeed the support of local solutions is the union of the maximal cliques that contain  $S$ .

**Case 3.** The set  $S$  is not a clique, but it can be decomposed as a collection of (possibly overlapping) maximal cliques  $C_1, C_2, \dots, C_k$  (maximal with respect to the subgraph induced by  $S$ ). In this case, we claim that if  $\mathbf{x}$  is a local solution, then its support can be obtained by taking the union of all maximal cliques of  $G$  containing one of the cliques  $C_i$  in  $S$ .

To make our discussion clearer, consider the graph shown in Fig. 2. In order to test whether our claims hold, we used as the set  $S$  different combinations of vertices, and enumerated all local solutions of (6) by multi-start replicator dynamics. Some results are shown below, where on the left-hand side we indicate the set  $S$ , while on the right hand-side we show the supports provided as output by the different runs of the algorithm.

1.  $S = \{2\} \Rightarrow \sigma(\mathbf{x}) = \{1, 2, 3\}$
2.  $S = \{5\} \Rightarrow \sigma(\mathbf{x}) = \{4, 5, 6, 7, 8\}$
3.  $S = \{4, 5\} \Rightarrow \sigma(\mathbf{x}) = \{4, 5\}$
4.  $S = \{5, 8\} \Rightarrow \sigma(\mathbf{x}) = \{5, 6, 7, 8\}$
5.  $S = \{1, 4\} \Rightarrow \sigma(\mathbf{x}_1) = \{1, 2\}, \sigma(\mathbf{x}_2) = \{4, 5\}$
6.  $S = \{2, 5, 8\} \Rightarrow \sigma(\mathbf{x}_1) = \{1, 2, 3\}, \sigma(\mathbf{x}_2) = \{5, 6, 7, 8\}$

The previous observations can be summarized in the following general statement which does comprise all three cases. Let  $S = C_1 \cup C_2 \cup \dots \cup C_k$  ( $k \geq 1$ ) be a subset of vertices of  $G$ , consisting of a collection of cliques  $C_i$  ( $i = 1 \dots k$ ). Suppose that condition (10) holds, and let  $\mathbf{x}$  be a local solution of (6). Then,  $\sigma(\mathbf{x})$  consists of the union of all maximal cliques containing some clique  $C_i$  of  $S$ .

We conjecture that the previous claim carries over to edge-weighted graphs, where the notion of a maximal clique is replaced by that of a dominant set. In the supplementary material we report the results of an extensive experimentation we have conducted on standard DIMACS graphs which provide support to our claim. This is going to play a key role in our applications of these ideas to interactive image segmentation.

## 4 Application to interactive image segmentation

In this section we apply our model to the interactive image segmentation problem. As input modalities we consider scribbles as well as boundary-based approaches (in particular, bounding boxes) and, in both cases, we show how the system is robust under input perturbations, namely imperfect scribbles or loose bounding boxes.

In this application the vertices of the underlying graph  $G$  represent the pixels of the input image (or superpixels, as discussed below), and the edge-weights reflect the similarity between them. As for the set  $S$ , its content depends on whether we are using scribbles or bounding boxes as the user annotation modality. In particular, in the case of scribbles,  $S$  represents precisely those pixels that have been manually selected by the user. In the case of boundary-based annotation instead, it is taken to contain only the pixels comprising the box boundary, which are supposed to represent the background scene. Accordingly, the union of the extracted dominant sets, say  $\mathcal{L}$  dominant sets are extracted which contain the set  $S$ , as described in the previous section and below,  $\mathbf{UDS} = \mathcal{D}_1 \cup \mathcal{D}_2 \dots \cup \mathcal{D}_{\mathcal{L}}$ , represents either the foreground object or the background scene depending on the input modality. For scribble-based approach the extracted set,  $\mathbf{UDS}$ , represent the segmentation result, while in the boundary-based approach we provide as output the complement of the extracted set, namely  $\mathbf{V} \setminus \mathbf{UDS}$ .

Figure 3 shows the pipeline of our system. Many segmentation tasks reduce their complexity by using superpixels (a.k.a. over-segments) as a preprocessing step [3,9,19,20,21]. While [3] used SLIC superpixels [22,9] used a recent superpixel algorithm [23] which considers not only the color/feature information but also boundary smoothness among the superpixels. In this work, we used the over-segments obtained from Ultrametric Contour Map (UCM) which is constructed from Oriented Watershed Transform (OWT) using globalized probability of boundary (gPb) signal as an input [24].

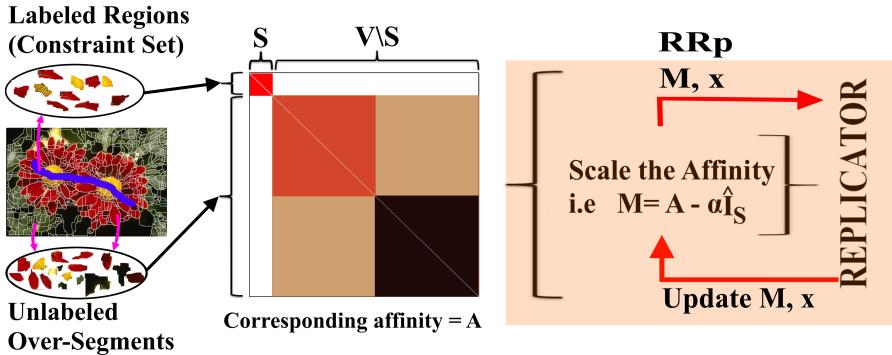


Fig. 3: Overview of our system. **Left:** Over-segmented image (output of the UCM-OWT algorithm [24]) with a user scribble (blue label). **Middle:** The corresponding affinity matrix, using each over-segments as a node, showing its two parts:  $S$ , the constraint set which contains the user labels, and  $V \setminus S$ , the part of the graph which takes the regularization parameter  $\alpha$ . **Right:** RRP, starts from the barycenter and extracts the first dominant set and update  $x$  and  $M$ , for the next extraction till all the dominant sets which contain the user labeled regions are extracted.

We then construct a graph  $G$  where the vertices represent over-segments and the similarity (edge-weight) between any two of them is obtained using a standard Gaussian kernel

$$A_{ij}^\sigma = \mathbb{1}_{i \neq j} \exp(-\|f_i - f_j\|^2 / 2\sigma^2)$$

where  $f_i$  is the feature vector of the  $i^{th}$  over-segment,  $\sigma$  is the free scale parameter, and  $\mathbb{1}_P = 1$  if  $P$  is true, 0 otherwise.

Given the affinity matrix  $A$  and the set  $S$  as described before, the system constructs the regularized matrix  $M = A - \alpha \hat{I}_S$ , with  $\alpha$  chosen as prescribed in (10). Then, the replicator dynamics (5) are run (starting them as customary from the simplex barycenter) until they converge to some solution vector  $x$ . We then take the support of  $x$ , remove the corresponding vertices from the graph and restart the replicator dynamics until all the elements of  $S$  are extracted.

#### 4.1 Experiments and results

As mentioned above, the vertices of our graph represents over-segments and edge weights (similarities) are built from the median of the color of all pixels in RGB, HSV, and  $L^*a^*b^*$  color spaces, and Leung-Malik (LM) Filter Bank [25]. The number of dimensions of feature vectors for each over-segment is then 57 (three for each of the RGB,  $L^*a^*b^*$ , and HSV color spaces, and 48 for LM Filter Bank).

In practice, the performance of graph-based algorithms that use Gaussian kernel, as we do, is sensitive to the selection of the scale parameter  $\sigma$ . In our experiments, we have reported three different results based on the way  $\sigma$  is chosen: 1) CDS\_Best\_Sigma, in this case the best parameter  $\sigma$  is selected on a per-image basis, which indeed can be thought of as the optimal result (or upper bound) of the framework. 2) CDS\_Single\_Sigma,

the best parameter in this case is selected on a per-database basis tuning  $\sigma$  in some fixed range, which in our case is between 0.05 and 0.2. 3) CDS\_Self.Tuning, the  $\sigma^2$  in the above equation is replaced, based on [26], by  $\sigma_i * \sigma_j$ , where  $\sigma_i = \text{mean}(K\text{NN}(f_i))$ , the mean of the K\_Nearest\_Neighbor of the sample  $f_i$ , K is fixed in all the experiment as 7.

**Datasets:** We conduct four different experiments on the well-known GrabCut dataset [1] which has been used as a benchmark in many computer vision tasks [27,2,28,29,3,9,30,31]. The dataset contains 50 images together with manually-labeled segmentation ground truth. The same bounding boxes as those in [2] is used as a baseline bounding box. We also evaluated our scribbled-based approach using the well known Berkeley dataset which contains 100 images.

**Metrics:** We evaluate the approach using different metrics: error rate, fraction of misclassified pixels within the bounding box, Jaccard index which is given by, following [32],  $J = \frac{|GT \cap O|}{|GT \cup O|}$ , where  $GT$  is the ground truth and  $O$  is the output. The third metric is the Dice Similarity Coefficient ( $DSC$ ), which measures the overlap between two segmented object volume, and is computed as  $DSC = \frac{2*|GT \cap O|}{|GT| + |O|}$ .

**Annotations:** In interactive image segmentation, users provide annotations which guides the segmentation. A user usually provides information in different forms such as scribbles and bounding boxes. The input modality affects both its accuracy and ease-of-use [12]. However, existing methods fix themselves to one input modality and focus on how to use that input information effectively. This leads to a suboptimal tradeoff in user and machine effort. Jain et al. [12] estimates how much user input is required to sufficiently segment a given image. In this work, as we have proposed an interactive framework, figure 1, which can take any type of input modalities, we will use four different types of annotations: bounding box, loose bounding box, scribbles - only on the object of interest -, and scribbles with error as of [11].

#### 4.1.1 Scribble based segmentation

Given labels on the foreground as constraint set, we built the graph and collect (iteratively) all unlabeled regions (nodes of the graph) by extracting dominant set(s) that contains the constraint set (user scribbles). We provided quantitative comparison against several recent state-of-the-art interactive image segmentation methods which uses scribbles as a form of human annotation: [7], Lazy Snapping [5], Geodesic Segmentation [4], Random Walker [33], Transduction [34], Geodesic Graph Cut [30], Constrained Random Walker [31]. Tables 1,2 and the plots in Figure 5 show the respective quantitative and the several qualitative segmentation results. Most of the results, reported on table 1, are reported by previous works [9,3,2,30,31]. We can see that the proposed framework outperforms all the other approaches.

**Error-tolerant Scribble Based Segmentation:** This is a family of scribble-based approach, proposed by Bai et. al [11], which tolerates imperfect input scribbles thereby avoiding the assumption of accurate scribbles. We have done experiments using synthetic scribbles and compared the algorithm against recently proposed methods specifically designed to segment and extract the object of interest tolerating the user input errors [11,38,39,40].

Methods	Error Rate
Graph Cut [7]	6.7
Lazy Snapping [5]	6.7
Geodesic Segmentation [4]	6.8
Random Walker [33]	5.4
Transduction [34]	5.4
Geodesic Graph Cut [30]	4.8
Constrained Random Walker [31]	4.1
CDS_Self Tuning (Ours)	<b>3.57</b>
CDS_Single Sigma (Ours)	<b>3.80</b>
CDS_Best Sigma (Ours)	2.72

Table 1: Error rates of different scribble-based approaches on the Grab-Cut dataset.

Methods	Jaccard Index
MILCut-Struct [3]	84
MILCut-Graph [3]	83
MILCut [3]	78
Graph Cut [1]	77
Binary Partition Trees [35]	71
Interactive Graph Cut [7]	64
Seeded Region Growing [36]	59
Simple Interactive O.E[37]	63
CDS_Self Tuning (Ours)	<b>93</b>
CDS_Single Sigma (Ours)	<b>93</b>
CDS_Best Sigma (Ours)	95

Table 2: Jaccard Index of different approaches – first 5 bounding-box-based – on Berkeley dataset.

Our framework is adapted to this problem as follows. We give, for the framework, the foreground scribbles as constraint set and check those scribbled regions which include background scribbled regions as their members in the extracted dominant set. Collecting all those dominant sets which are free from background scribbled regions generates the object of interest.

**Experiment using synthetic scribbles.** Here, a procedure similar to the one used in [40] and [11] has been followed. First, 50 foreground pixels and 50 background pixels are randomly selected based on ground truth (see Fig. 4). They are then assigned as foreground or background scribbles, respectively. Then an error-zone for each image is defined as background pixels that are less than a distance D from the foreground, in which D is defined as 5 %. We randomly select 0 to 50 pixels in the error zone and assign them as foreground scribbles to simulate different degrees of user input errors. We randomly select 0, 5, 10, 20, 30, 40, 50 erroneous sample pixels from error zone to simulate the error percentage of 0%, 10%, 20%, 40%, 60%, 80%, 100% in the user input. It can be observed from figure 4 that our approach is not affected by the increase in the percentage of scribbles from error region.

**4.1.2 Segmentation using bounding boxes** The goal here is to segment the object of interest out from the background based on a given bounding box. The corresponding over-segments which contain the box label are taken as constraint set which guides the segmentation. The union of the extracted set is then considered as background while the union of other over-segments represent the object of interest.

We provide quantitative comparison against several recent state-of-the-art interactive image segmentation methods which uses bounding box: LooseCut [9], GrabCut [1], OneCut [29], MILCut [3], pPBC and [28]. Table 3 and the pictures in Figure 5 show the respective error rates and the several qualitative segmentation results. Most of the results, reported on table 3, are reported by previous works [9,3,2,30,31].

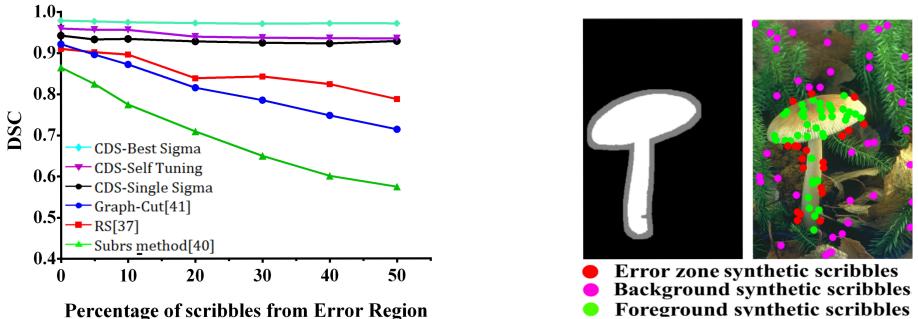


Fig. 4: **Left:** Performance of algorithms, on Grab-Cut dataset, for different percentage of synthetic scribbles from the error region. **Right:** Synthetic scribbles and error region

**Segmentation Using Loose Bounding Box:** This is a variant of the bounding box approach, proposed by Yu et.al [9], which avoids the dependency of algorithms on the tightness of the box enclosing the object of interest. The approach not only avoids the annotation burden but also allows the algorithm to use automatically detected bounding boxes which might not tightly encloses the foreground object. It has been shown, in [9], that the well-known GrabCut algorithm [1] fails when the looseness of the box is increased. Our framework, like [9], is able to extract the object of interest in both tight and loose boxes. Our algorithm is tested against a series of bounding boxes with increased looseness. The bounding boxes of [2] are used as boxes with 0% looseness. A looseness  $L$  (in percentage) means an increase in the area of the box against the baseline one. The looseness is increased, unless it reaches the image perimeter where the box is cropped, by dilating the box by a number of pixels, based on the percentage of the looseness, along the 4 directions: left, right, up, and down.

For the sake of comparison, we conduct the same experiments as in [9]: 41 images out of the 50 GrabCut dataset [1] are selected as the rest 9 images contain multiple objects while the ground truth is only annotated on a single object. As other objects, which are not marked as an object of interest in the ground truth, may be covered when the looseness of the box increases, images of multiple objects are not applicable for testing the loosely bounded boxes [9]. Table 3 summarizes the results of different approaches using bounding box at different level of looseness. As can be observed from the table, our approach performs well compared to the others when the level of looseness gets increased. When the looseness  $L = 0$ , [3] outperforms all, but it is clear, from their definition of tight bounding box, that it is highly dependent on the tightness of the bounding box. It even shrinks the initially given bounding box by 5% to ensure its tightness before the slices of the positive bag are collected. For looseness of  $L = 120$  we have similar result with LooseCut [9] which is specifically designed for this purpose. For other values of  $L$  our algorithm outperforms all the approaches.

**Complexity:** In practice, over-segmenting and extracting features may be treated as a pre-processing step which can be done before the segmentation process. Given the affinity matrix, a simple and effective optimization algorithm to extract the object of interest is given by the replicator dynamics 5. Its computational complexity per step is

Methods	$L = 0\%$	$L = 120\%$	$L = 240\%$	$L = 600\%$
GrabCut [1]	7.4	10.1	12.6	13.7
OneCut [29]	6.6	8.7	9.9	13.7
pPBC [28]	7.5	9.1	9.4	12.3
MilCut [3]	<b>3.6</b>	-	-	-
LooseCut [9]	7.9	<b>5.8</b>	6.9	6.8
CDS_Self Tuning (Ours)	7.54	6.78	<b>6.35</b>	7.17
CDS_Single Sigma (Ours)	7.48	5.9	<b>6.32</b>	<b>6.29</b>
CDS_Best Sigma (Ours)	6.0	4.4	4.2	4.9

Table 3: Error rates of different boundin-box approaches with different level of looseness as an input, on the Grab-Cut dataset.  $L = 0\%$  implies a baseline bounding box as those in [2]

$O(N^2)$ , with  $N$  being the total number of nodes of the graph. Infection-immunization dynamics [41] is a faster alternative which has an  $O(N)$  complexity for each step which allow convergence of the framework in fraction of second, with a code written in Matlab and run on a core i5 6 GB of memory. As for the pre-processing step, the original *gPb-owt-ucm* segmentation algorithm was very slow to be used as a practical tools. Catanzaro et al. [42] proposed a faster alternative, which reduce the runtime from 4 minutes to 1.8 seconds, reducing the computational complexity and using parallelization which allow *gPb* contour detector and *gPb-owt-ucm* segmentation algorithm practical tools. For the purpose of our experiment we have used the Matlab implementation which takes around four minutes to converge, but in practice it is possible to give for our framework as an input, the GPU implementation [42] which allows the convergence of the whole framework in around 4 seconds.

## 5 Conclusions

In this paper, we have developed an interactive image segmentation algorithm based on the idea of finding a collection of dominant-set clusters constrained to contain the elements of a user annotation. The approach is based on some properties of a family of quadratic optimization problems related to dominant sets which show that, by properly selecting a regularization parameter that controls the structure of the underlying function, we are able to “force” all solutions to contain the user-provided elements. The resulting algorithm is capable of dealing with both scribble-based and boundary-based annotation modes. Segmentation results of extensive experiments on natural images demonstrate that the approach compares favorably with state-of-the-art algorithms and turns out to be robust in the presence of loose bounding boxes and large amount of user input errors. Future work will focus on applying the framework on video sequences and other computer vision problems such as content-based image retrieval.

**Acknowledgments.** This work has been partly supported by Samsung Global Research Outreach Program.

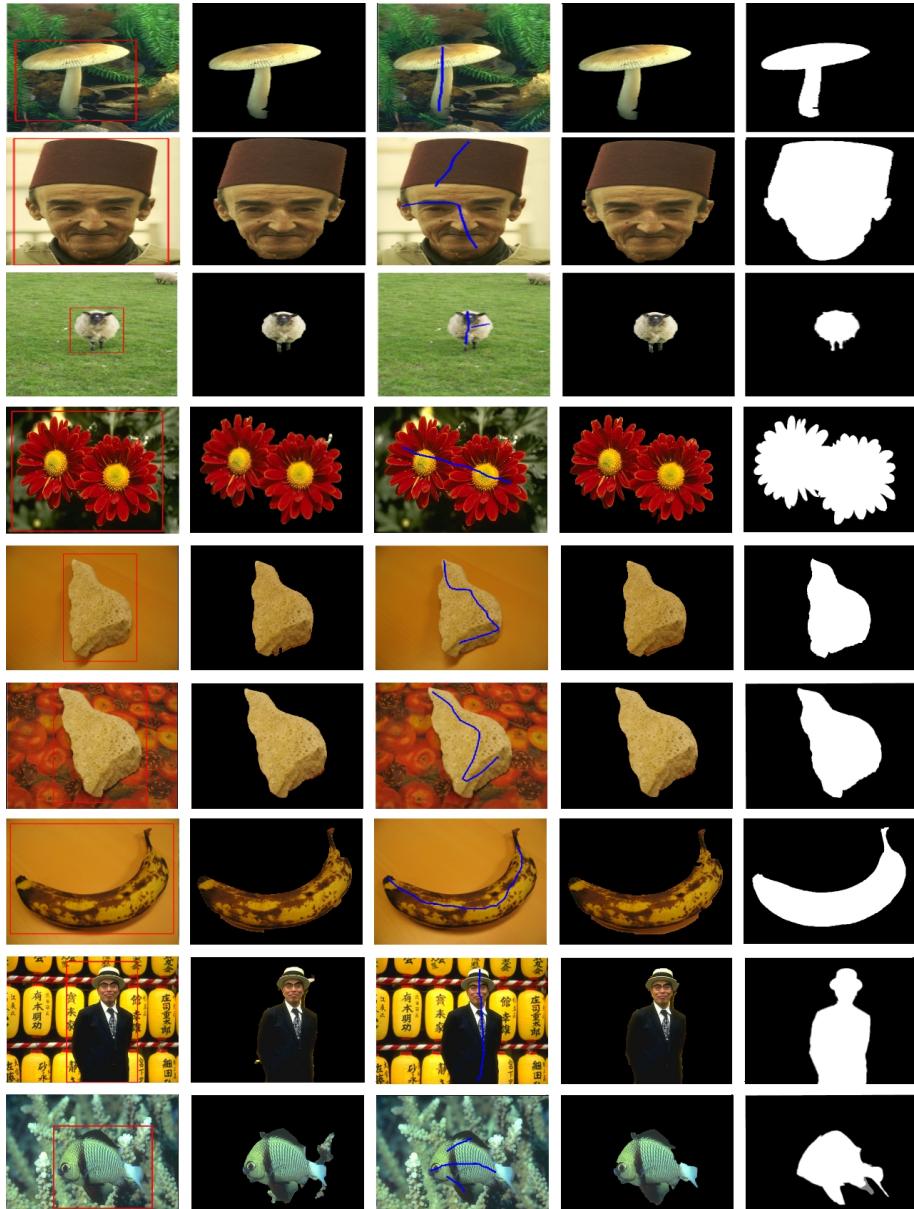


Fig. 5: Exemplar results of the algorithm tested on Grab-Cut dataset. **Left:** Original image with bounding boxes of [2]. **Middle left:** Result of the bounding box approach. **Middle:** Original image and scribbles (observe that the scribbles are only on the object of interest). **Middle right:** Results of the scribbled approach. **Right:** The ground truth.

## References

1. Rother, C., Kolmogorov, V., Blake, A.: “Grabcut”: Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.* **23**(3) (2004) 309–314
2. Lempitsky, V.S., Kohli, P., Rother, C., Sharp, T.: Image segmentation with a bounding box prior. In: *ICCV*. (2009) 277–284
3. Wu, J., Zhao, Y., Zhu, J., Luo, S., Tu, Z.: Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation. In: *CVPR*. (2014) 256–263
4. Bai, X., Sapiro, G.: Geodesic matting: A framework for fast interactive image and video segmentation and matting. *Int. J. Computer Vision* **82**(2) (2009) 113–132
5. Li, Y., Sun, J., Tang, C., Shum, H.: Lazy snapping. *ACM Trans. Graph.* **23**(3) (2004)
6. Protiere, A., Sapiro, G.: Interactive image segmentation via adaptive weighted distances. *IEEE Trans. Image Processing* **16**(4) (2007) 1046–1057
7. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: *ICCV*. (2001) 105–112
8. Mortensen, E.N., Barrett, W.A.: Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing* **60**(5) (1998) 349–384
9. Yu, H., Zhou, Y., Qian, H., Xian, M., Lin, Y., Guo, D., Zheng, K., Abdelfatah, K., Wang, S.: Loosecut: Interactive image segmentation with loosely bounded boxes. *CoRR abs/1507.03060* (2015)
10. Xian, M., Zhang, Y., Cheng, H.D., Xu, F., Ding, J.: Neutro-connectedness cut. *CoRR abs/1512.06285*
11. Bai, J., Wu, X.: Error-tolerant scribbles based interactive image segmentation. In: *CVPR*. (2014) 392–399
12. Jain, S.D., Grauman, K.: Predicting sufficient annotation strength for interactive foreground segmentation. In: *ICCV*. (2013) 1313–1320
13. Pavan, M., Pelillo, M.: A new graph-theoretic approach to clustering and segmentation. In: *CVPR*. (2003) 145–152
14. Pavan, M., Pelillo, M.: Dominant sets and pairwise clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(1) (2007) 167–172
15. Rota Bulò, S., Pelillo, M.: A game-theoretic approach to hypergraph clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(6) (2013) 1312–1327
16. Pavan, M., Pelillo, M.: Dominant sets and hierarchical clustering. In: *ICCV*. (2003) 362–369
17. Luenberger, D.G., Ye, Y.: *Linear and Nonlinear Programming*. Springer, New York (2008)
18. Horn, R.A., Johnson, C.R.: *Matrix Analysis*. Cambridge University Press, New York (1985)
19. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: *ICCV*. (2005) 654–661
20. Wang, J., Jia, Y., Hua, X., Zhang, C., Quan, L.: Normalized tree partitioning for image segmentation. In: *CVPR*. (2008)
21. Xiao, J., Quan, L.: Multiple view semantic segmentation for street view images. In: *ICCV*. (2009) 686–693
22. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(11) (2012) 2274–2282
23. Zhou, Y., Ju, L., Wang, S.: Multiscale superpixels and supervoxels based on hierarchical edge-weighted centroidal voronoi tessellation. In: *WACV*. (2015) 1076–1083
24. Arbelaez, P., Maire, M., Fowlkes, C.C., Malik, J.: Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **33** (2011) 898–916
25. Leung, T.K., Malik, J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Computer Vision* **43**(1) (2001) 29–44

26. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS. (2004) 1601–1608
27. Li, H., Meng, F., Ngan, K.N.: Co-salient object detection from multiple images. IEEE Trans. Multimedia **15**(8) (2013) 1896–1909
28. Tang, M., Ayed, I.B., Boykov, Y.: Pseudo-bound optimization for binary energies. In: ECCV. (2014) 691–707
29. Tang, M., Gorelick, L., Veksler, O., Boykov, Y.: Grabcut in one cut. In: ICCV. (2013) 1769–1776
30. Price, B.L., Morse, B.S., Cohen, S.: Geodesic graph cut for interactive image segmentation. In: CVPR. (2010) 3161–3168
31. Yang, W., Cai, J., Zheng, J., Luo, J.: User-friendly interactive image segmentation through unified combinatorial user inputs. IEEE Trans. Image Processing **19**(9) (2010) 2470–2479
32. McGuinness, K., O'Connor, N.E.: A comparative evaluation of interactive segmentation algorithms. Pattern Recognition **43**(2) (2010) 434–444
33. Grady, L.: Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11) (2006) 1768–1783
34. Duchenne, O., Audibert, J., Keriven, R., Ponce, J., Ségonne, F.: Segmentation by transduction. In: CVPR. (2008)
35. Salembier, P., Garrido, L.: Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. IEEE Trans. Image Processing **9**(4) (2000) 561–576
36. Adams, R., Bischof, L.: Seeded region growing. IEEE Trans. Pattern Anal. Mach. Intell. **16**(6) (1994) 641–647
37. Friedland, G., Jantz, K., Rojas, R.: SIOX: Simple interactive object extraction in still images. In: (ISM). (2005) 253–260
38. Liu, J., Sun, J., Shum, H.: Paint selection. ACM Trans. Graph. **28**(3) (2009)
39. Sener, O., Ugur, K., Alatan, A.A.: Error-tolerant interactive image segmentation using dynamic and iterated graph-cuts. In: IMMPD@ACM Multimedia. (2012) 9–16
40. Subr, K., Paris, S., Soler, C., Kautz, J.: Accurate binary image selection from inaccurate user input. Comput. Graph. Forum **32**(2) (2013) 41–50
41. Rota Bulò, S., Bomze, I.M.: Infection and immunization: A new class of evolutionary game dynamics. Games and Economic Behavior **71**(1) (2011) 193–211
42. Catanzaro, B.C., Su, B., Sundaram, N., Lee, Y., Murphy, M., Keutzer, K.: Efficient, high-quality image contour detection. In: ICCV. (2009) 2381–2388