Random Walks based Multi-Image Segmentation: Quasiconvexity Results and GPU-based Solutions

Maxwell D. Collins[1] Jia Xu[1] Leo Grady[2] Vikas Singh[1]
[1]University of Wisconsin-Madison Madison, WI
[2]Siemens Corporate Research Princeton, NJ
{mcollins,jiaxu}@cs.wisc.edu, leo.grady@siemens.com, vsingh@biostat.wisc.edu

Abstract

We recast the Cosegmentation *problem using Random Walker (RW) segmentation as the core segmentation algorithm, rather than the traditional MRF approach adopted in the literature so far. Our formulation is similar to previous approaches in the sense that it also permits Cosegmentation constraints (which impose consistency between the extracted objects from $\geq$ 2 images) using a nonparametric model. However, several previous nonparametric cosegmentation methods have the serious limitation that they require adding one auxiliary node (or variable) for every pair of pixels that are similar (which effectively limits such methods to describing only those objects that have high entropy appearance models). In contrast, our proposed model completely eliminates this restrictive dependence – the resulting improvements are quite significant. Our model further allows an optimization scheme exploiting quasiconvexity for model-based segmentation with no dependence on the scale of the segmented foreground. Finally, we show that the optimization can be expressed in terms of linear algebra operations on sparse matrices which are easily mapped to GPU architecture. We provide a highly specialized CUDA library for Cosegmentation exploiting this special structure, and report experimental results showing these advantages.*

1. Introduction

The problem of Cosegmentation [1], has attracted much attention from the community in the last few years [2–6]. The basic goal in Cosegmentation is to segment a common salient foreground object from two or more images, as shown in Fig. 1. Here, consistency between the (extracted) object regions is accomplished by imposing a *global* constraint which penalizes variations between the objects' respective histograms or appearance models. The idea has been adopted for obtaining concise measures of image similarity [1], discovering common appearance patterns in image sets [7], medical imaging [8], and building 3D models from community photo collections [2, 9]. Motivated by the spectrum of these applications, some recent papers have sought to better understand the optimization-specific aspects of this problem – in particular, issues such as sub-modularity [1], linear programming relaxations [4], dual-decomposition based strategies [5], network flow constructions [8], and maximum-margin inspired interpretations [10]. Most of these works provide good insights on cosegmentation, but *only* in the context of the traditional Markov Random Field (MRF) based segmentation objective (referred to as *graph-cuts* [11]). This may be partly because the first work on Cosegmentation [1] presented means for including global constraints within segmentation but was designed *specifically* for the MRF function. The present paper complements this body of research, and provides an end-to-end analysis of the Cosegmentation problem in terms of the Random Walker segmentation function [12] – these results show that in many cases, well-known advantages of the Random Walker model extend nicely to the Cosegmentation setting as well.

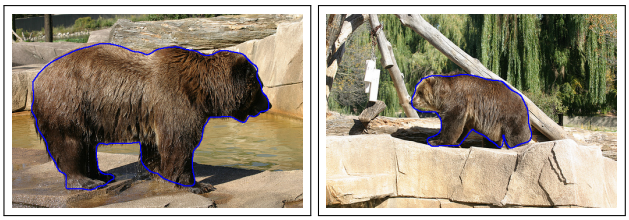
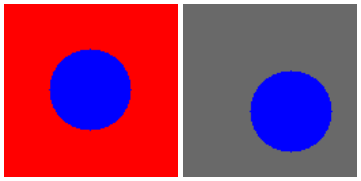


Figure 1. A representative application for cosegmentation.

A Toy example. An important aspect of our formulation is that it is possible to employ a *nonparametric* appearance model for *arbitrary* distributions but *without* incurring rather substantial additional computational costs. When



there are significant regions of homogeneity in the foreground (inline figure), we clearly want a distribution which has a corresponding peak. In Fig. 1, the distribution should capture the fact that the bear is "brown and furry", and *not* try to differentiate one patch of fur from another across multiple images. To illustrate why this point is relevant, let us analyze the overhead of some existing methods for cosegmentation by considering a simple toy example (see inline image pair) where we should identify the common blue circle (in distinct backgrounds). Several approaches for cosegmentation with a nonparametric model require that an auxiliary node (or variable) be introduced into the graph whenever two pixels share the *same* bin [4, 8] (i.e., the two pixels are similar). While the segmentation aspect for the blue circles by itself is easy, the cost of introducing an auxiliary node for perceptually similar pixels can grow *very* quickly – counting just the blue foreground pixels for a 196×196 image pair, one must introduce 42 million additional variables, and the associated cost is infeasible even for moderately sized images. As a result, these previous models are limited to feasibly cosegmenting only those image pairs that have a relatively high entropy distribution (i.e., each bin is shared by only a few pixels). Our formulation has *no such limitation*, since auxiliary nodes are *not* needed to perform the optimization. Consequently, it is possible to perform cosegmentation in general settings where the target foreground is summarized with an arbitrary appearance model (color, texture), but with no associated restriction on its entropy. Further, several nonparametric cosegmentation models (except [10]) are somewhat limited to segmenting foregrounds which are at the same scale within each image. Our method compares histograms *independent* of scale, and the objective is shown to be *quasiconvex*. This is leveraged to develop an optimization scheme which can solve this *nonconvex* problem with optimality guarantees. Finally, all optimization steps in the core method can be expressed as sparse linear algebra operations, which makes GPU implementations possible.

Related Work. The initial model for Cosegmentation in [1] provided means for including global constraints to enforce consistency among the two foreground histograms in *addition* to the MRF segmentation terms for each image. The objective function incorporating these ideas was expressed as follows

$$E_{\text{coseg}} = \text{MRF}_{\text{image 1}} + \text{MRF}_{\text{image 2}} + \lambda E^{\text{global}}(h_1, h_2), \quad (1)$$

where $E^{\text{global}}(\cdot,\cdot)$ was assumed to penalize the ℓ_1 variation between each h_1 and h_2 (the two foreground histograms obtained *post* segmentation). The appearance model for the histogram was assumed to be generative (i.e., Gaussian), and a novel scheme called trust region graph cuts was presented for optimizing the resulting form of (1). Subsequently, [4] argued in favor of using an ℓ_2^2-distance for $E^{\text{global}}(\cdot,\cdot)$ whereas [8] developed a reward based model. Batra et al. [2] suggested exploiting user interaction if available for cosegmentation (again, using MRF terms) and [3, 10] have adopted a clustering based approach to the problem. Recently, [5] compared several existing MRF-based models, and presented a new posterior-maximizing scheme which was solved using dual decomposition. Other recent ideas include modulating the graph-cuts objective *a priori* by finding similar patterns across images [3], generating multiple segmentations of each image and identifying which segmentation pair is similar [13], and identifying salient regions followed by a filtering step to leave out distinct regions *not* shared across images [14]. One reason for these varied strategies for the problem is that when a histogram difference term is added to the segmentation, the resultant objective is no longer *submodular* (therefore, not easy to optimize). So, the focus has been on improved approximate algorithms for different choices of E^{global}.

A commonality among these existing works has been the preference for MRF (i.e., graph-cuts) based terms for segmentation. Part of the reason is that combinatorial methods such as graph-cuts are extensively used in vision, and are known to be efficient. On the other hand, graph-partitioning methods such as Random Walker [12] also work well for image segmentation and are widely used. Our formalization here suggests that it is also well suited for the Cosegmentation problem and offers efficiency benefits (e.g., issues identified in the blue circles example) in the nonparametric setting.

The **contributions** of this paper are: **(1)** We derive a cosegmentation model with the Random Walker segmentation at its core. The model finally reduces to a Box-QP problem (convex program with box constraints). Based on this structure, we propose a specialized (and efficient) gradient projection based procedure which finds a global real-valued optimum of the model (which preserves many advantages of Random Walker [12]). **(2)** Our model allows for a nonparametric representation of the foregrounds (e.g., using distributions over texture words), but one which permits *any* distribution of features without incurring additional computational costs. This provides a substantial advantage over the existing nonparametric cosegmentation approaches which are limited only to regions described by a high entropy model (i.e., object features must be spread evenly across bins). **(3)** We extend this model to a scale-independent penalty. This paper presents a novel optimization method for a class of objectives based on quasiconvex functions. We prove correctness, and demonstrate it for model-based image segmentation. These theoretical results are of independent interest. **(4)** Our optimization consists of linear algebra operations (BLAS Level 1, 2) on sparse matrices. Consequently, the algorithm is easy to implement on a GPU architecture. We give a specialized open-source CUDA library for Cosegmentation.

2. Random Walker and its properties

The Random Walker segmentation algorithm has been studied extensively in the computer vision literature. Essentially, the method simulates a random walk from each pixel in the image to a set of user specified seed points where the walk is biased by image intensity gradients. The eventual assignment of pixels to foreground or background is determined by whether the pixel-specific walk reaches a foreground (or background) seed first. Observe a subtle yet important property of how the Random Walker model is specified, and what the solution actually denotes. Because of direct analogues in circuit theory and physics, the formalization, even in its original form, seeks a solution in reals (not integers). What is eventually solved is therefore *not* a relaxation because the variables have a clear probabilistic meaning. As a result, thresholding these probabilities at 0.5 is statistically sound; conceptually, this is different from solving a binary linear program in reals and recovering a $\{0,1\}$ solution by rounding. In practice, Random Walker is optimized by recasting segmentation as the solution to a combinatorial Dirichlet problem. Random Walker derived segmentations offer some benefits with respect to boundary length regularization, number of seeds, metrication errors, and shrinking bias [15].

3. Random Walker for Cosegmentation

We begin our presentation by rewriting the Random Walker algorithm for a single image as a quadratic minimization problem (also see [16]). As is common, we assume a 4-connected neighborhood over the image, weighted according to a Gaussian function of normalized Euclidean distances between pixel intensities, $w_{ij} = \exp\left(-\beta\|p_i - p_j\|\right)$. The Laplacian L of the graph is then

$$L_{ij} = \begin{cases} \sum_k w_{ik} & \text{if } i = j \\ -w_{ij} & \text{if } i \neq j \text{ and } (i,j) \in \text{neighborhood graph} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The Laplacian is diagonally dominant and so $L \succeq 0$; we can derive the following convex quadratic program,

$$\min_x \quad x^T L x \quad \text{subject to} \quad x^{(s)} = m^{(s)}, \tag{3}$$

where $x^{(s)}$ are the values for certain seed pixels, and $m^{(s)}$ is the known value of those seeds (i.e., foreground or background). Each component of the solution x^* will then be a pixel's probability of being assigned to the foreground. To output a $\{0,1\}$ segmentation, we may threshold x^* at $\frac{1}{2}$, to obtain a hard $x \in \{0,1\}^n$ segmentation which matches the solution from [12].

Pre-processing. Cosegmentation methods [8] use a preprocessing procedure to determine inter- (and intra-) image pixel similarity. This is generated by tesselating the RGB color space (i.e., pixel distribution) into clusters or by using SIFT (or color pattern models, edge-profiles, textures etc) based correspondence methods, see [17]. We can derive a matrix H such that

$$H_i^{kj} = \begin{cases} 1 & \text{if pixel } j \text{ is in histogram bin } k \text{ in image } i \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Here, pixels are assigned to the same bin if they are similar. With an appropriate H, the global term E^{global} from (1) requires that at the level of individual histogram bins k, the algorithm assign approximately the same number of pixels to each foreground region (the objective incurs a penalty proportional to this difference). This ensures that the appearance models of the two foregrounds based on the features of interest are similar, and has been used very successfully in object recognition [18]. Observe that this difference only serves as a *regularizer* for the main segmentation task, and does not drive it on its own. This is relevant because as with any global measure, such models (and the measurement of their variations) may not be perfect. But existing literature suggests that when derived from good context-based features [18], such appearance model based differences provide a meaningful global bias for Cosegmentation [2].

Cosegmentation for 2+ images. Given a segmentation for n pixels, $x \in \{0,1\}^n$, one may use the H matrix from (4) to write the histogram of only the foreground pixels as $h = Hx$. The expression gives the form of constraints needed for Cosegmentation. Let $L_1, \cdots, L_m$ be the Laplacian matrices of graphs constructed using each of the images, and $H_1, \cdots, H_m$ be the histogram assignment matrices from (4), with the property that $H_i^{kj} = H_{i'}^{kj'} = 1$ for pixels j and j' if and only if j and j' are similar. Here, if one uses SIFT matches, then the matrix entry may reflect the confidence of the match. Now, we seek to segment the two images simultaneously, under the constraint that their histograms match. For this purpose, it suffices to consider the following optimization model

$$\min_{x_i, h_i, \bar{h}} \quad \sum_i x_i^T L_i x_i + \lambda \|h_i - \bar{h}\|_2^2$$
$$\text{s.t.} \quad x_i \in [0,1]^{n_i}, \;\; x_i^{(s)} = m_i^{(s)}, \;\; H_i x_i = h_i, \;\; i = 1...m. \tag{5}$$

The second term in the objective above corresponds to $E^{\text{global}}(h_1, h_2)$ in (1), and the last constraint sets up the foreground histograms, h_i using H_i and (bin k in H_1 corresponds to bin k in $H_2, \cdots, H_m$ which makes a direct comparison between histograms possible). Instead of comparing the histograms to each other, we compare them to a common *global* histogram $\bar{h}$ which at the optimum will be the centroid of the h_i's. The resulting inter-image matching penalty will then be the trace of the covariance between foreground histograms across the image set. This model additionally extends to multiple labels (i.e., multiple objects) by adding additional columns to the optimization variables identically to [12]. The resulting problem can

be easily decomposed into separate segmentations for each object class. Existing cosegmentation methods, on the other hand, mostly tackle figure-ground labeling.

Each Laplacian matrix L_i is postive semidefinite, so along with the histogram distances the objective function is convex. Further, the feasible region is the intersection of bound constraints and linear equalities. We directly have:

Theorem 3.1. *For $\lambda \geq 0$, (5) is a convex problem.*

3.1. Deriving an equivalent Box-QP

The model in (5) can already be solved using widely available convex programming methods, and provides the desired solution to the Cosegmentation problem using the Random Walker segmentation function. Next, we will derive an equivalent model (but with a much nicer structure) that will allow the design of specialized solvers and thereby lead to far more efficient algorithms.

Consider the left hand side of the equality constraint on each h_i, substituted into the objective function with a penalty $\|h_1 - h_2\|$. Further, let us choose bounds to limit x to the unit box as well as suitably enforce the seed constraints. This process gives a quadratic problem of the form

$$\begin{aligned} \min_{x_1,x_2} \quad & \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} L_1 + \lambda H_1^T H_1 & -\lambda H_1^T H_2 \\ -\lambda H_2^T H_1 & L_2 + \lambda H_2^T H_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{s.t.} \quad & l_i \leq x_i \leq u_i \quad x_i \text{ is of size } [0,1]^{n_i} \quad i = 1, 2, \end{aligned} \tag{6}$$

where the 2-tuple (l_i, u_i) is $(1, 1)$ for foreground seeds, $(0, 0)$ for background seeds, and $(0, 1)$ otherwise. For $m > 2$ images we optimize over $x_1, \cdots, x_m, \bar{h}$ with quadratic objective matrix

$$\begin{bmatrix} L_1 + \lambda H_1^T H_1 & & & -\lambda H_1 \\ & \ddots & & \vdots \\ & & L_m + \lambda H_m^T H_m & -\lambda H_m \\ -\lambda H_1^T & \cdots & -\lambda H_m^T & \lambda m I \end{bmatrix}$$

It can be verified that (6) is *equivalent* to (5). The difference is that it is now expressed as a bound-constrained quadratic problem (or *box-QP* due to the box constraints). Like (5), the model in (6) also permits general purpose convex programming methods. However, we can design means to exploit its special structure since the model is *nearly* an unconstrained quadratic problem.

4. Scale-Free Cosegmentation

A limitation of previous cosegmentation methods is their sensitivity to the scale of the target object, since histogram-based priors are dependent on scale. For example, if an otherwise identical object appears in the second image such that it occupies twice as many pixels as in the first image, then $h_2 = 2h_1$. Consequently, $||h_2 - h_1|| > 0$, meaning that the larger scale is penalized in traditional formulations. We show here how our formulation may be made

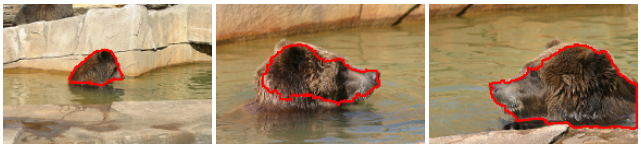


Figure 2. Segmentation using the model of section 4 on a set of images from the iCoseg dataset with differences in scale. $\bar{h}$ from an image in the same set was applied as a prior.

scale-invariant. Formally, our goal is to modify the cosegmentation term to satisfy

$$E(sh_i, \bar{h}) = E(h_i, \bar{h}) \quad \forall s \in \mathbb{R}_+. \tag{7}$$

This property may be satisfied by a *normalization* step,

$$E(h_i, \bar{h}) = \left\| \frac{h_i}{\|h_i\|} - \frac{\bar{h}}{\|\bar{h}\|} \right\|^2 = 2 - 2\frac{h_i^T \bar{h}}{\|h_i\|\|\bar{h}\|}. \tag{8}$$

Substituting these normalized histograms in (5) leads to a function that cannot be efficiently optimized. However, in the Random Walker setting, we *can* optimize this function when the model histogram $\bar{h}$ is fixed. The resulting problem is related to model-based segmentation, where we are imposing a known histogram distribution in segmenting images. Given normalized prior $\bar{h}$, the resulting energy is

$$\hat{g}_{\bar{h}}(h) = -\frac{\bar{h}^T h}{\|h\|}. \tag{9}$$

In order to proceed with the minimization of our scale-invariant energy, we must first establish some properties of (9). Proofs appear in the extended version of this paper.

Theorem 4.1. *$\hat{g}_{\bar{h}}$ is quasiconvex. [19]*

Corollary 4.2. *The scale-free energy on x, $g_{\bar{h}}(x) = \hat{g}_{\bar{h}}(Hx)$ is quasiconvex for histogram assignment matrix H.*

Theorem 4.3. *$g_{\bar{h}}(x)$ is Lipschitz-smooth when $\|Hx\| > 0$.*

The next section exploits these properties of $g_{\bar{h}}$ to solve the segmentation problem using this penalty.

4.1. Nonconvex Sum Minimization

For the following, we consider the setting of minimizing $h = f + g$, such that f is convex, g is quasiconvex and both are bounded below. Note that under these conditions, h is not necessarily quasiconvex and may have multiple local minima. Nonetheless, our method proposed below can optimize our segmentation objective with $f(x) = x^T L x$ and g as defined in Corollary 4.2. Let $x_f^* = \operatorname{argmin}_x f(x)$, similarily define x_g^* and x_h^*.

Theorem 4.4. *Define*

$$\mathcal{P}(\alpha) = \begin{pmatrix} \operatorname{argmin}_{x \in X} & f(x) \\ \text{s.t.} & g(x) \leq \alpha \end{pmatrix} \tag{10}$$

For x any solution to $\mathcal{P}(g(x_h^*))$, $h(x) = h(x_h^*)$.

Theorem 4.5. *$\mathcal{P}(\alpha)$ has no solutions for $\alpha < g(x_g^*)$, and x is a solution to $\mathcal{P}(g(x_f^*))$ iff x is a solution $\forall \alpha > g(x_f^*)$.*

Theorem 4.6. *$h \circ \mathcal{P}$ is one-sided Lipschitz.*

For more details on the one-sided Lipschitz condition see [20] and the extended version of this paper. With the result of Theorem 4.6, by simply sampling $h(\mathcal{P}(\alpha))$ densely enough, we can get an estimate of this function to arbitrary precision over the entire interval $\alpha \in [g(x_g^*), g(x_f^*)]$, using bounds similar to [21]. If we select a global minimum of this estimated function, we can derive a point with objective arbitrarily close to the true minimum of h.

5. Optimization

This section describes our strategy for solving (6) to optimality in a highly efficient manner. We use a projected gradient-based method which would additionally form the key component if we were to use any variation (*i.e.* augmented Langrangian) as stated explicitly in [22].

Identifying a Sparse Structure: Expressing our model as a purely bound-constrained problem as in (6) requires the formation of the $H_i^T H_i$ products which form *dense* $n \times n$ matrices. Consequently, our optimization method must be careful not to explicitly form these matrices. Fortunately, we observe that explicit formulation of these matrices may be avoided by gradient projection methods, which are a simple and efficient class of algorithms in which it is only necessary to be able to calculate matrix-vector products. Here, this product can be distributed over the *sparse* components,

$$(L + H_1^T H_1)x_1 = Lx_1 + H_1^T(H_1 x_1). \tag{11}$$

With this modification, we can solve our Box-QP in (6) by adapting the Gradient Projection/Conjugate Gradient (GPCG) algorithm of [23]. We describe this strategy next.

5.1. GPCG

GPCG solves quadratic problems with a rectilinear feasible region $\Omega = \{x : l \leq x \leq u\}$. The algorithm alternates between two main phases (GP and CG): these correspond to alternately estimating the active set at the minimum and finding the minimum while keeping the active set fixed.

A) Gradient Projection (GP). Gradient projection coupled with the projected line search allows fast searching of a wide range of Ω. As a result, we can rapidly estimate which face of the feasible region the optimum lies on.

A.1) Search Direction for GP: In this phase, we search along a *projected gradient* $\nabla_\Omega \mathcal{O}(x)$ which has been modified so the corresponding search direction $d = -\nabla_\Omega \mathcal{O}$ does not point outside the feasible region. Specifically, this search direction is constructed to satisfy $\exists\, \epsilon > 0$ such that $x + \epsilon d \in \Omega$. We then use a *projected line search* (described later) to arrive at a step length α and update $x \leftarrow x + \alpha d$.

A.2) Phase Switch for GP: We switch to the conjugate gradient phase if *either* of the following conditions are satisfied: **(a)** The *active set* $\mathcal{A} = \{i : x_i = l_i \text{ or } x_i = u_i\}$ remains unchanged between two iterations; note that the active set corresponds to the minimal face (w.r.t. inclusion) of Ω to which x belongs; **(b)** GP is making little progress.

B) Conjugate Gradient (CG). We search a given face of the feasible region of our model using the conjugate gradient phase described below.

B.1) Search Direction for CG: Given the active set, our algorithm calculates a search direction conjugate to the previous direction (under the projection on to the free variables). Note that this method of generating a search direction is the same as applying ordinary conjugate gradient descent to a restriction of the QP to the current minimal face.

B.2) Phase Switch for CG: If the projected gradient points *out* of the current face (or if the iterations are making little progress), we switch back to the gradient projection (GP) phase. Formally, this is true if $\exists i \in \mathcal{A}(x)$ and either

$$x_i = l_i \text{ and } \partial_i \mathcal{O}(x) < 0, \quad \text{or} \quad x_i = u_i \text{ and } \partial_i \mathcal{O}(x) > 0.$$

Note that these "phase switch" conditions will never be satisfied for the face which contains the global minimum for our model. Thus, when the gradient projection phase has found the correct active set, conjugate gradient iterations suffice to explore the final face.

C) Projected Line Search: The projected line search modifies the active set by an arbitrary number of elements, and helps our GPCG process. Given a starting point x and search direction d, the line search finds $\alpha > 0$ which produces a *sufficient decrease* under the Armijo rule of the function $\phi(\alpha) = \mathcal{O}(P[x + \alpha d])$, where P describes the projection function which maps its input to the closest point in Ω. This can be thought of as a "line search" along a 1-manifold which bends to stay within Ω (thus, not a ray). Rather than directly finding all the piecewise quadratic segments of $\phi(\alpha)$, we efficiently produce a sufficient decrease using an estimate of ϕ by sampling one point at a time (as in [23]). It can be verified that *all* operations above can be expressed as Level 1, 2 BLAS operations. This allows a highly parallel implementation, described next.

D) GPU Implementation. Graphical Processing Units (GPUs) have gained attention from a wide range of the scientific computing community for their ability to efficiently address parallel problems. These architectures operate by running multiple instances of a piece of code simultaneously, operating on different parts of a dataset. While this approach is not well-suited to all algorithms, Level 1 and 2 BLAS operations used in our algorithm are known to fit well with this architecture and can therefore exhibit a significant speedup. The linear algebra operations comprising our GPCG algorithm for Cosegmentation may be easily parallelized using high-level languages as CUDA. In fact,

the CUSPARSE toolkit (used here), supports Level 2 and 3 BLAS operations on sparse matrices as well. Further, the control flow of our procedure relies only on the results of *accumulations*, so the standard bottleneck of transferring data between main and GPU memory is not a major factor, and entry-level hardware is sufficient.

6. Experiments

Summary: Our experimental evaluations included comparisons of our implementation (on a Nvidia Geforce 470) to another Cosegmentation method [8], and the Random Walker algorithm [12] (run independently on both images). We also performed experiments using the methods in [4] and [10], but due to the problem of solving a large LP and incorporation of foreground/background seeds respectively, results could not be obtained for the entire dataset described below. To assess *relative* advantages of the specialized GPCG procedure, we also compared it with a standalone implementation of (5) linked with a commercial QP solver (using multiple cores). We provide a qualitative and quantitative overview of the performance w.r.t. state of the art. Additional experiments demonstrate the efficacy of the multiple-image and scale-free segmentation models. The full set of segmentation results and code is available at
`http://pages.cs.wisc.edu/~mcollins/pubs/cvpr2012.html`

Datasets: In order to leverage all available test data we aggregated all images provided by the authors in iCoseg [2], Momi-coseg [3] and Pseudoflow-coseg [8], and further supplemented them with a few additional images from the web and [24]. In order to compare with algorithms that only handle image *pairs* we selected a dataset with 50 pairs of images (100 total). For the > 2 image case we used the iCoseg dataset from [2]. Since a number of image sets from (from [2]) share only semantically similar foreground objects and are unsuitable for cosegmentation with common appearance models, we selected 88 subsets comprising 389 of the 643 iCoseg images (also observed in [13]).

Winn Filters: Histograms were constructed using the 17-filter bank proposed in [18] as features. Pixels across both images were assigned to bins by clustering these responses, using nonparametric methods to estimate the number of clusters k. The first step produces local contextual *descriptors* of each image pixel. The clustering step finds those pixels which are similar under the given descriptor (similar *color* and *texture* will be in the same bin). We also incorporated SIFT-based features, but given that the above histograms already provided good results, this additional module was not utilized further. In Fig. 7 we perform correspondences based on *optical flow* in order to segment frames of a video sequence. Other correspondence determination methods can be used and no change to our algorithm is needed.

Low Entropy Histograms: When the number of bins is high (constructed histogram is flat), it is more likely that a non-trivial number of "matches" will be erroneous – especially because in cosegmentation the two images may have distinct backgrounds without a shared baseline. In our experiements we found that as the entropy increases, so does the JS divergence measure between the histograms for the true segmentations. Low entropy histograms, however, relate to smaller divergences, which impose the global Cosegmentation constraint more tightly.

Running time complexity: We now discuss what is the strongest advantage of this framework. We show an example in Fig. 3 of the running time of the proposed model relative to [8], as a function of *decreasing* entropy (number of bins). The plot suggests that for a realistic range, our implementation has a negligible computation time relative to [8] and the CPLEX-based option – in fact, our curve almost coincides with the x-axis (and even this cost was dominated primarily by the overhead from problem setup done on the CPU). For the most expensive data point shown in Fig. 3, the model from [8] generated 10^7 auxiliary nodes (about 12 GB of memory). Due to the utility of low entropy histograms, these experiments show a salient benefit of our framework and its immunity to the 'coarseness' of the appearance model. Over all 128×128 images, the wall-clock running time of our CUDA-based model was 10.609 ± 5.230 seconds (a significant improvement over both [8, 10]). The time for a Cplex driven method (*utilizing four cores* in parallel) was 17.982 ± 5.07 seconds, but this increases sharply with greater problem size.

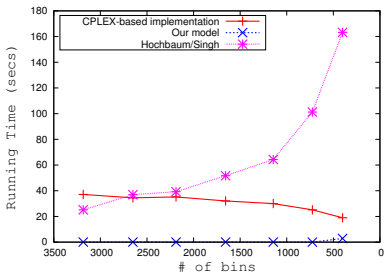


Figure 3. Variation in run-time with histogram granularity relative to a Cplex-based implementation and [8].

Performance w.r.t. pair methods: We evaluated the quality of segmentations ($0/1$ loss) on the 50 image pairs described above, relative to Pseudoflow-coseg [8], LP [4] (only partial), and discriminative clustering [10]. As in [12], a few seeds were placed to specify foreground and background regions, and given to all methods. Representative samples on the images from [2] using the method in [8, 10] are shown in Fig. 4. Averaged over the pair dataset, the segmentation accuracy of our method was $93.6 \pm 2.9\%$, where as the gross values for the algorithms from [8] and [10] were 89.1% and 84.1% respectively.

Performance on 2+ images: Across the iCoseg dataset we achieved an accuracy of 93.7% with seeds provided by 5 different users. The algorithm of [10] achieves an accuracy of 82.2% across the dataset (excluding some for which the implementation provided by the authors did not complete). Representative image sets and accuracy comparisons appear

in Table 1 and Figure 5.

We note that these results must be interpreted with the caveat that different methods respond differently to seed locations and the level of discrimination offered by the underlying appearance model. Since most cosegmentation models (including this paper) share the same basic construction at their core (i.e., image segmentation with an appearance model constraint), variations in performance are in part due to the input histograms. The purpose of our experiments here is to show that at the very least one can expect similar (if not better) qualitative results with our model, but with more flexibility and significant computational advantages.

Comparisons to independent Random Walker runs: In Fig. 6, we present qualitative results from our algorithm, and from independent runs of Random Walker (both with up to two seeds per image). A trend was evident on all images – the probabilities from independent runs of Random Walker on the two images were diffuse and provide poorer boundary localization. This is due to the lack of global knowledge of the segmentation in the other image. Random walker based Cosegmentation is able to leverage this information, and provides better contrast and crisp boundaries for thresholding (a performance boost of up to 10%).

7. Conclusions

We present a new framework for the cosegmentation problem based on the Random Walker segmentation approach. While almost all other cosegmentation methods view the problem in the MRF (graph-cuts) setting, our algorithm translates many of the advantages of Random Walker to the task of simultaneously segmenting common foregrounds from related images. Significantly, our formulation completely eliminates a requirement in some cosegmentation methods that requires the overall image histogram to be approximately flat. Our model extends nicely to the multi-image setting using a penalty with statistical justification. A further extension allows model-based segmentation which is independent of the relative scales of the model and target foregrounds. We discuss its optimization specific properties, give a state of the art GPU based library, and show quantitative and qualitative performance of the method.

Acknowledgments This work is funded via NIH 5R21AG034315-02 and NSF 1116584. Partial support was provided by UW-ICTR through an NIH CTSA Award 1UL1RR025011 and NIH grant 5P50AG033514 to W-ADRC. Funding for M.D.C. was provided by NLM 5T15LM007359 via the CIBM Training Program. The authors would like to thank Petru M. Dinu for consultations on GPGPU development.

References

[1] C. Rother, V. Kolmogorov, T. Minka, and A. Blake. Cosegmentation of image pairs by histogram matching - Incorporating a global constraint in MRFs. In *CVPR*, 2006.

[2] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. Interactive co-segmentation with intelligent scribble guidance. In *CVPR*, 2010.

[3] W. Chu, C. Chen, and C. Chen. MOMI-cosegmentation: Simultaneous segmentation of multiple objects among multiple images. In *ACCV*, 2010.

[4] L. Mukherjee, V. Singh, and C. Dyer. Half-integrality based algorithms for cosegmentation of images. In *CVPR*, 2009.

[5] S. Vicente, V. Kolmogorov, and C. Rother. Cosegmentation revisited: Models & optimization. In *ECCV*, 2010.

[6] G. Kim, E.P, Xing, L. Fei-Fei, and T. Kanade. Distributed cosegmentation vis submodular optimization on anisotropic diffusion. In *ICCV*, 2011.

[7] Y. Lee and K. Grauman. CollectCut: Segmentation with top-down cues discovered in multi-object images. In *CVPR*, 2010.

[8] D.S. Hochbaum and V. Singh. An efficient algorithm for co-segmentation. In *ICCV*, 2009.

[9] A. Kowdle, D. Batra, W. C. Chen, and T. Chen. iModel: Interactive co-segmentation for object of interest 3D modeling. *ECCV Workshop*, 2010.

[10] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *CVPR*, 2010.

[11] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11), 2001.

[12] L. Grady. Random walks for image segmentation. *PAMI*, 28(11):1768–1783, 2006.

[13] S. Vicente, C. Rother, and V. Kolmogorov. Object cosegmentation. In *CVPR*, 2011.

[14] K. Chang, T. Liu, and S. Lai. From co-saliency to co-segmentation: An efficient and fully unsupervised energy minimization model. In *CVPR*, 2011.

[15] A. K. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *ICCV*, 2007.

[16] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *PAMI*, 30(2):228–242, 2008.

[17] J. Cui, Q. Yang, F. Wen, Q. Wu, C. Zhang, L. Van Gool, and X. Tang. Transductive object cutout. In *CVPR*, 2008.

[18] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV*, 2005.

[19] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming*. John Wiley & Sons, Inc., 2003.

[20] Reinhard Frank, Josef Schneid, and Christoph W. Ueberhuber. Stability properties of implicit Runge-Kutta methods. *SIAM J. on Numerical Analysis*, 22(3):497–514, 1985.

[21] D. Cremers, F. R. Schmidt, and F. Barthel. Shape priors in variational image segmentation: Convexity, Lipschitz continuity and globally optimal solutions. In *CVPR*, 2008.

[22] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, USA, 2nd edition, 2006.

[23] J.J. More and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM J. on Optimization*, 1(1):93–113, 1991.

[24] D. Tsai, M. Flagg, and J. M. Rehg. Motion coherent tracking with multi-label MRF optimization. In *BMVC*, 2010.

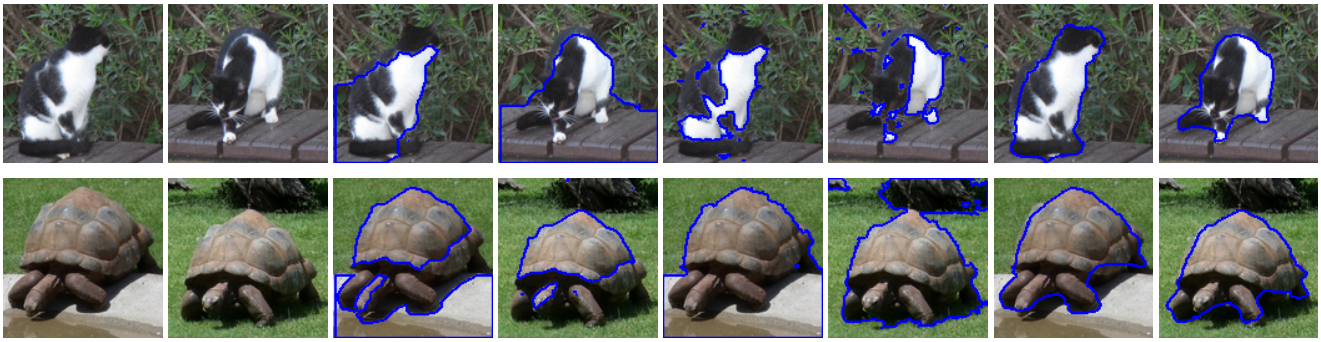


Figure 4. Comparison results on example images (columns 1,2) of the the Pseudoflow based method of [8] (columns 3,4) and the discriminative clustering approach of [10] (columns 5,6), with segmentation from RW-based cosegmentation (columns 7,8).

Figure 5. Interactive segmentation results using the multi-image model across five images from the "Kendo" set of iCoseg. As the number of images increases, *less* user input is required, as seen by the lack of such for the middle image.

	Our Method	[10]
Airshow (8/21)	99.7	80.5
Alaskan bear (5/18)	92.2	76.6
Christ (6/14)	95.6	97.3
Ferrari (4/10)	96.4	59.4
Goose (13/31)	97.3	97.5
Helicopter (9/11)	96.9	98.1
Kendo (10/31)	91.4	93.4
Lobster Kite (4/11)	97.0	88.5
Monk (4/17)	83.3	87.1
Soccer (5/36)	93.9	79.9
Speed Skater (9/13)	80.8	77.3
Liberty (11/41)	93.2	86.1

Table 1. Segmentation accuracy for some iCoseg image sets. Subsets were chosen which have similar appearance under a histogram model.

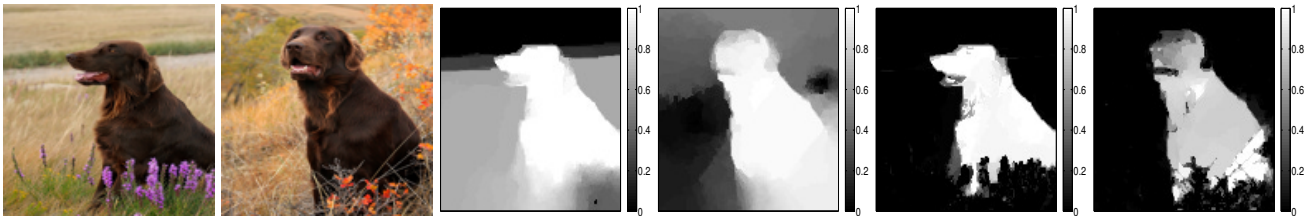


Figure 6. Columns (1–2): Input images; Columns (3–4) segmentation potentials from independent random walker runs on the two images; Columns (5–6) Segmentation potentials from Random Walker based cosegmentation. Note that the object boundaries have become significantly more pronounced because of the histogram constraint.

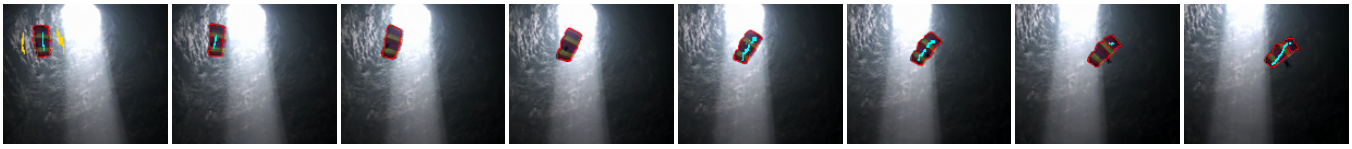


Figure 7. Segmentation using correspondences from optical flow on video sequence from [24]. Shows outline of segmented foreground in red, with foreground and background indications. Our algorithm achieves 99.3% accuracy.