Joint Recovery of Dense Correspondence and Cosegmentation in Two Images

Tatsunori Taniai
The University of Tokyo

Sudipta N. Sinha
Microsoft Research

Yoichi Sato
The University of Tokyo

Abstract

We propose a new technique to jointly recover cosegmentation and dense per-pixel correspondence in two images. Our method parameterizes the correspondence field using piecewise similarity transformations and recovers a mapping between the estimated common "foreground" regions in the two images allowing them to be precisely aligned. Our formulation is based on a hierarchical Markov random field model with segmentation and transformation labels. The hierarchical structure uses nested image regions to constrain inference across multiple scales. Unlike prior hierarchical methods which assume that the structure is given, our proposed iterative technique dynamically recovers the structure along with the labeling. This joint inference is performed in an energy minimization framework using iterated graph cuts. We evaluate our method on a new dataset of 400 image pairs with manually obtained ground truth, where it outperforms state-of-the-art methods designed specifically for either cosegmentation or correspondence estimation.

1. Introduction

Recovering dense per-pixel correspondence between image regions in two or more images is a central problem in computer vision. While correspondence estimation for images of the same scene (stereo, optic flow, etc.) is well studied, there has been growing interest in the case where the images portray semantically similar but different scenes or depict semantically related but different object instances [36]. Due to the variability in appearance, shape and pose of distinct object instances, camera viewpoint, scene lighting and backgrounds in the images, the task is quite challenging in the unsupervised setting. Yet, correspondence estimation enables fine-grained image alignment crucial in tasks such as non-parametric scene parsing and label transfer [36], 3D shape recovery [51], image editing [19] and unsupervised visual object discovery [11, 42, 45, 50].

In parallel to advances in correspondence estimation, there has also been rapid progress in image cosegmentation [17, 26, 41, 46] methods that automatically segment similar "foreground" areas in two or more images. These methods often require the foregrounds depicting common objects to have similar region statistics. Most cosegmentation methods do not explicitly recover dense pixel correspondence and alignment in the region labeled foreground. On the other hand, correspondence estimation methods [36, 29, 53, 23] align all the pixels without explicitly inferring which pixels in the two images actually have valid correspondence. Thus, recovering cosegmentation along with a dense alignment of the common foregrounds can be viewed as a holistic approach to solving both tasks.

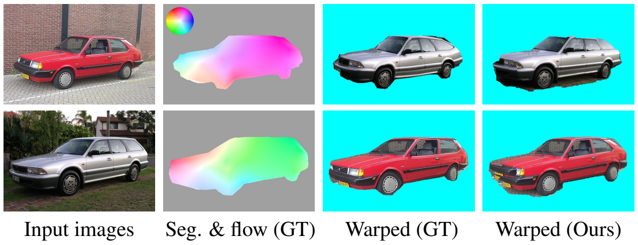


Figure 1. Joint recovery of dense correspondence and cosegmentation where foregrounds are segmented and aligned. We show our results and corresponding ground truth (GT) from our new dataset.

In this paper, we present insight into how image cosegmentation and correspondence (or flow) estimation can be tackled within a unified framework by framing it as a labeling problem (Figure 1). We show that jointly solving the two tasks in this way can improve performance on both of them. This paper deals with the case where only two input images are given. The setting is unsupervised and we do not assume a priori information about the objects or the scene.

Our contributions are three folds. First, we propose a new hierarchical Markov random field (MRF) model for joint cosegmentation and correspondence recovery. The hierarchy is defined over nested image regions in the reference image and the nodes representing these regions take segmentation and flow labels. In our method, the hierarchy itself is inferred in conjunction with the labeling and is crucial for achieving robustness to dissimilar appearance of different object instances. Precomputed hierarchical structures [29, 23, 27] are unsuitable for our task because pixels inferred as background must be excluded from matching.

Second, we propose a new optimization technique for the joint inference of the graph structure and labeling. Performing exact inference jointly on the whole hierarchical structure is intractable. In the proposed approach, layers of the hierarchy are incrementally estimated with the labeling in an energy minimization framework using iterated graph cuts [8, 31] (alpha expansion moves).

Finally, we release a new dataset with 400 image pairs for which we provide ground truth cosegmentation masks and flow maps. The original images and some of the segmentation masks are taken from existing datasets [35, 42, 20]. The remaining segmentation masks were obtained using interactive image segmentation. The flow maps were obtained by selecting sparse keypoint correspondence with our interactive annotation tool and applying natural neighbor interpolation [44] on the sparse data. Poor flow maps were discarded by visually inspecting the flow-induced image warping result. The ground truth flow maps makes it possible to directly evaluate dense image alignment. Even SIFT Flow [36] and other correspondence estimation methods [29, 54] are evaluated indirectly on tasks such as segmentation transfer and scene parsing using datasets that lack ground truth pixel correspondence.

The paper is organized as follows. We describe related work in Section 2 and our proposed model in Section 3. Section 4 presents our optimization method whereas implementation details and the images features used are described in Section 5. Finally, in Section 6, we report experimental evaluation and comparisons with existing approaches.

2. Related Work

We are not aware of any existing method that explicitly solves both cosegmentation and dense correspondence recovery together. However, the motivation behind our work is similar to that behind some recent cosegmentation methods [13, 17, 43]. We review those and other broadly related works on cosegmentation and correspondence estimation.

Cosegmentation. Rubio *et al*. [43] formulate cosegmentation in terms of region matching. However, the matches are computed independently using graph matching [16] and then exploited in their cosegmentation algorithm. Faktor and Irani [17] describe a model where common foregrounds in multiple images can be composed from interchangeable image regions. Although region matching is a key element of their method, it is primarily used to estimate unary potentials (foreground/background likelihoods) for a standard image segmentation method. While, Dai *et al*. [13] propose to cosegment images by matching foregrounds through a codebook of deformable shape templates, it involves learning a codebook requiring external background images. While a notion of correspondence implicitly exists in all these works, none of them explicitly compute dense correspondence maps between the cosegmented regions, which is an important distinction to our work.

Cosegmentation methods originally proposed by Rother et al. [41] have been applied in broader settings [5, 24, 25, 32, 28, 52] and also on large sets of Internet images [42, 11]. Interesting convex formulations have also been proposed for a variant of cosegmentation – the object co-localization task [26, 46], which aims to find a bounding box around related objects in multiple images.

Correspondence Estimation. SIFT Flow [36] generalizes optic flow to images of different scenes and estimates complete flow maps with 2D translations at every pixel. Their energy function uses local matching costs based on dense SIFT features, and smoothness terms encoding standard pairwise potentials. SIFT Flow uses loopy belief propagation (BP) [18] for inference in a coarse-to-fine pipeline but other inference techniques [19, 53] have also been explored. HaCohen et al. [19] propose an extension of PatchMatch [3, 4] that handles images of identical scenes with large motions. However, their method is often unable to handle different scenes as it lacks regularization on correspondence fields. As another extension, DAISY filter flow (DFF) [53] proposes to use efficient cost-volume filtering [38] for enforcing smoothness, instead of adding explicit regularization. Deformable spatial pyramid (DSP) matching [29] and its generalization [23] propose hierarchical regularization using a regular grid-cell pyramid for flow estimation. Correspondence maps are parameterized using similarity transformations in [23] similarly to our work. Images with scale differences are handled by [39, 21, 23]. Cosegmentation has been used to guide sparse feature correspondence recovery [10]. However, such methods do not aim to accurately segment common regions.

Hierarchical Models. To exploit multi-scale image cues or to add flexible regularization, hierarchical conditional random fields (CRFs) have been proposed for single image segmentation [22], image matching [49], stereo correspondence [33], and much recently for optic flow [34, 27] and more general correspondence estimation [29, 23]. These methods use precomputed hierarchical structures such obtained by an external hierarchical oversegmentation method [2], or spatial pyramids as used in DSP [29, 23].

Optimization Techniques. Discrete optimization is commonplace in stereo but often problematic in general dense correspondence estimation because of the large label spaces involved. For this issue, SIFT Flow [36] performs hierarchical BP [18] on the image pyramid from coarse to fine levels using limited translation ranges. Recently, inspired by randomization search and label propagation schemes of PatchMatch [3, 4, 7], optimization methods using BP [6] or graph cuts [48, 47] have been proposed for efficient inference in pairwise MRFs with large label spaces. However, they are not directly applicable to our hierarchical model. We extend graph cut techniques [48, 47] for our inference task where we recover both the graph structure as well as the labeling.

3. Proposed Model

Given two images $\mathbf{I}$ and $\mathbf{I}'$ our goal is to find dense correspondence and cosegmentation of a common object shown in the two images. The reference image $\mathbf{I}$ is represented by a set of superpixel nodes $i \in V$ where $\Omega_i \subseteq \Omega$ denotes a superpixel region in the image domain $\Omega \subset \mathbb{Z}^2$.

In the reference image, we seek a labeling involving geometric transformation $\mathbf{T}_i \in \mathcal{T}$ and a foreground alpha-matte value $\alpha_i \in [0,1]$ for each superpixel $i \in V$. We formulate this as a mapping function $f_i = f(i) : V \to \{\mathcal{T} \times [0,1]\}$ that assigns each node a pair of labels $f_i = (\mathbf{T}_i, \alpha_i)$. Here, α_i is continuous during inference and binarized at the final step[1]. $\mathbf{T}_i$ denotes a similarity transform parameterized using a quadruplet (t_u, t_v, s, r). Slightly abusing the notation, we express the warped pixel location of $\mathbf{p}'$ in the other image as follows.

$$\mathbf{p}' = \mathbf{T}_i(\mathbf{p}) = sR_r(\mathbf{p} - \mathbf{c}_i) + \mathbf{c}_i + \mathbf{t}. \tag{1}$$

Here, $\mathbf{c}_i$ is the centroid of pixels in region Ω_i, and centering at this point, $\mathbf{p}$ is rotated by the 2D rotation matrix R_r of angle r and scaled by s, and then translated by $\mathbf{t} = (t_u, t_v)$.

In following sections we present the proposed model, by first defining a standard 2D MRF model in Section 3.1 and later generalizing it to a hierarchical model in Section 3.2. We discuss the allowed hierarchical structure in Section 3.3.

3.1. Single Layer Model

Let $L = (V, E)$ be a graphical representation of the image $\mathbf{I}$, where nodes $i \in V$ and edges $(i,j) \in E$ represent superpixels and spatial neighbors, respectively. Given this graph, our single layer model is defined as a standard 2D MRF model:

$$\begin{aligned}\mathcal{E}_{\text{mrf}}(f|L) =& \lambda_{\text{flo}} \sum_{i\in V} \mathcal{E}^i_{\text{flo}}(f_i) + \lambda_{\text{seg}} \sum_{i\in V} \mathcal{E}^i_{\text{seg}}(f_i) \\ &+ \sum_{(s,t)\in E} w_{st}\, \mathcal{E}^{st}_{\text{reg}}(f_s, f_t),\end{aligned} \tag{2}$$

which consists of the flow data term, cosegmentation data term and the spatial regularization term described below.

Flow Data Term. $\mathcal{E}^i_{\text{flo}}$ measures similarity between corresponding regions in the image pair. We define it as

$$\mathcal{E}^i_{\text{flo}}(f_i) = \sum_{\mathbf{p}\in\Omega_i} \Big[\alpha_i \rho(\mathbf{p}, \mathbf{p}') + \bar{\alpha}_i \lambda_{\text{occ}}\Big], \tag{3}$$

where $\bar{\alpha}_i = 1 - \alpha_i$ and λ_{occ} is a constant penalty for background pixels to avoid trivial solutions where all pixels are labeled background. The $\rho(\mathbf{p}, \mathbf{p}')$ robustly measures visual dissimilarity between $\mathbf{p}$ and its correspondence $\mathbf{p}'$ as

$$\rho(\mathbf{p}, \mathbf{p}') = \min\{\|\mathbf{D}(\mathbf{p}) - \mathbf{D}'(\mathbf{p}'))\|_2^2, \tau_{\text{D}}\}, \tag{4}$$

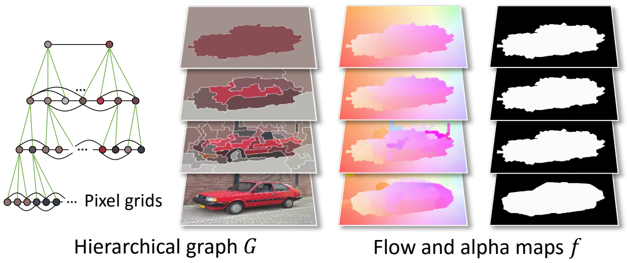


Figure 2. Hierarchical model. Each layer (2D MRF) estimates a dense flow and alpha map f, which is regularized by higher-level estimates and the final estimates are obtained at the bottom layer.

where truncation using the threshold τ_{D} adds robustness. $\mathbf{D}(\mathbf{p})$ is a local feature descriptor extracted at $\mathbf{p}$ in the image $\mathbf{I}$, and $\mathbf{D}'(\mathbf{p}')$ is extracted in $\mathbf{I}'$[2]. We use a variant of the HOG descriptor [14]. See Section 5.1 for the details.

Cosegmentation Data Term. The foreground and background likelihoods for each node are defined as follows.

$$\mathcal{E}^i_{\text{seg}}(f_i) = -\sum_{\mathbf{p}\in\Omega_i} \Big[\alpha_i \ln P(\mathbf{I}_\mathbf{p}|\boldsymbol{\theta}^F) + \bar{\alpha}_i \ln P(\mathbf{I}_\mathbf{p}|\boldsymbol{\theta}^B)\Big]. \tag{5}$$

Here, $P(\cdot|\boldsymbol{\theta})$ is likelihood given a foreground or background color model $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ of the image $\mathbf{I}$, which is implemented as 64^3 bins of RGB color histograms. The color models are estimated during initialization (Section 5.3).

Spatial Regularization Term. The term $\mathcal{E}^{st}_{\text{reg}}$ encourages flow and alpha values of neighboring nodes to be similar.

$$\begin{aligned}\mathcal{E}^{st}_{\text{reg}}(f_s, f_t) =& \lambda_{\text{st1}} \min\{\alpha_s, \alpha_t\} \sum_{\mathbf{p}\in B_{st}} \psi^{st}(\mathbf{p})/|B_{st}| \\ &+ \lambda_{\text{st2}}|\alpha_s - \alpha_t|.\end{aligned} \tag{6}$$

Here, $B_{st} = \partial\Omega_s \cap \partial\Omega_t$ is the set of pixels on the boundary of two adjoining regions Ω_s and Ω_t, and $\psi^{st}(\mathbf{p})$ penalizes flow discontinuities at these pixels. It is defined as

$$\psi^{st}(\mathbf{p}) = \min\{\|\mathbf{T}_s(\mathbf{p}) - \mathbf{T}_t(\mathbf{p})\|_2, \tau_{\text{st}}\}. \tag{7}$$

If α were binary, then the first term in Eq. (6) would enforce flow smoothness when two adjoining regions are labeled foreground ($\alpha_s = \alpha_t = 1$), and the second term would give a constant penalty λ_{st2} only when $\alpha_s \neq \alpha_t$. However, Eq. (6) generalizes this idea to continuous valued α.

3.2. Hierarchical Model

Now we introduce the notion of a layered graph and generalize the single layer model to a full hierarchical model. As illustrated in Figure 2, our hierarchical graph $G = (V, E)$ consists of multiple layered subgraphs $\{L_0, L_1, \cdots, L_H\}$. Each layer $L_l = (V_l, E_l)$ represents a

[1] To avoid degenerate flow solutions, we set α_i always larger than 0.1 during inference. See supplementary for a detailed explanation.

[2] As suggested in [23, 53], $\mathbf{D}'(\mathbf{p}')$ can be more accurately computed by using the scale s and rotation r of the similarity transformation $\mathbf{T}_i$.

superpixel graph of the image. In addition to spatial edges within each layer E_l, our hierarchy G contains parent-child edges $(p, c) \in E_l^{\text{pc}}$ that connect parent nodes $p \in V_l$ to their children nodes $c \in V_{l-1}$ (green edges in Figure 2).

Using a layered graph G and the model $\mathcal{E}_{\text{mrf}}(f|L)$ defined in Eq. (2), we define our hierarchical model as

$$\mathcal{E}(f, G) = \sum_{l=0}^{H} \Big[\mathcal{E}_{\text{mrf}}(f|L_l) + \mathcal{E}_{\text{reg}}^l(f|G) + \mathcal{E}_{\text{gra}}^l(V_l)\Big]. \quad (8)$$

Here, we treat the hierarchical graph G as a variable that is dynamically estimated together with f. Our construction is fundamentally different from prior work [23, 27, 29], where the hierarchical structure is computed before flow inference.

Multi-layer Regularization Term. Similar to the spatial regularization term in Eq. (6), the term $\mathcal{E}_{\text{reg}}^l$ enforces smoothness between parent child pairs of V_l and V_{l-1} as

$$\mathcal{E}_{\text{reg}}^l(f|G) = \sum_{(p,c) \in E_l^{\text{pc}}} w_{pc}\, \mathcal{E}_{\text{reg}}^{pc}(f_p, f_c), \quad (9)$$

where $\mathcal{E}_{\text{reg}}^{pc}$ is defined using Eq. (7) and c's centroid $\mathbf{c}_c$ as

$$\mathcal{E}_{\text{reg}}^{pc}(f_p, f_c) = \lambda_{\text{pc1}} \min\{\alpha_p, \alpha_c\} \psi^{pc}(\mathbf{c}_c) + \lambda_{\text{pc2}} |\alpha_p - \alpha_c|. \quad (10)$$

Graph Validity Term. The term $\mathcal{E}_{\text{gra}}^l$ measures validity of the layer structure V_l as

$$\mathcal{E}_{\text{gra}}^l(V_l) = \lambda_{\text{nod}} \beta^l |V_l| - \lambda_{\text{col}} \sum_{i \in V_l} \sum_{\mathbf{p} \in \Omega_i} \ln P(\mathbf{I_p}|\boldsymbol{\theta}^i). \quad (11)$$

The first term reduces nodes in the higher layers. We set $\beta = 2$ to reduce the node count approximately by half at each layer. The second term enforces color consistencies within each region Ω_i. $\boldsymbol{\theta}^i$ represents the RGB color histogram of the region Ω_i. Our definition of Eq. (11) is motivated by work in multi-region segmentation [15].

3.3. Hierarchical Structure

Here we describe the form of hierarchical graphs allowed in our method. The nodes $i \in V_l$ in each layer divide the image domain Ω into $|V_l|$ connected regions $\Omega_i \subseteq \Omega$. Our hierarchical superpixels have a nested (or tree) structure, *i.e.*, a superpixel (parent) in a layer V_l consists of the union of superpixels (children) in its sublayer V_{l-1}. The lowest layer V_0 named the *pixel layer* is special because each node $i \in V_0$ represents a pixel $\mathbf{p}_i \in \Omega$. The finest region layer V_1 has about 500 nodes which are set to SLIC superpixels [1].

For parent-child edges $(p, c) \in E_l^{\text{pc}}$ $(l = 1, \cdots, H)$, the edge weights w_{pc} are assigned to the area of child regions

$$w_{pc} = |\Omega_c|. \quad (12)$$

At the two lowest layers $(l = 0, 1)$, edges $(s, t) \in E_l$ between adjoining nodes are assigned edge weights w_{st} as

$$w_{st} = e^{-\|\mathbf{I}_s - \mathbf{I}_t\|_2^2/\kappa}, \quad (13)$$

where $\mathbf{I}_i \in \mathbb{R}^3$ is the mean color of the region Ω_i. Following [40], we set κ to the expected value of $2\|\mathbf{I}_s - \mathbf{I}_t\|_2^2$ over $(s, t) \in E_l$. For the upper layers $(l \geq 2)$, the edge weights w_{st} are set to the sum of the children's edge weights as

$$w_{st} = \sum w_{s't'}, \quad (14)$$

where $(s', t') \in E_{l-1}$ are children of s and t, respectively.

4. Optimization

Optimizing $\mathcal{E}(f, G)$ in Eq. (8) has two main difficulties. 1) The joint inference of f and G is intractable due to the dependency of f on G. 2) The label space of f resides in a 5-dimensional continuous domain and the number of candidate labels is essentially infinite. To practically address these issues, we propose two-pass bottom-up and top-down optimizing procedures that approximately optimize the energy. In the bottom-up phase, we construct a hierarchical structure G by incrementally adding layers from lower to higher levels, while simultaneously estimating the labeling f. In the top-down phase, we refine the labeling f while keeping the structure G fixed. The optimization procedure is summarized in Algorithm 1. Next we discuss the details.

4.1. Bottom-Up Hierarchy Construction

To formally describe our bottom-up procedure, we denote $G^k = (V^k, E^k)$ as a hierarchy consisting of $k+1$ layered subgraphs $\{L_0, \cdots, L_k\}$ where $L_l = (V_l, E_l)$. We also define it sequentially, *i.e.*, G^k and G^{k+1} share the same structure for the bottom $k+1$ layers.

At a high level, our bottom-up procedure is presented as a sequence of subtasks, where given a current solution $\{f, G^k\}$ we estimate $\{f, G^{k+1}\}$ as illustrated in Figures 3 (a) and (d), respectively. We estimate $\{f, G^{k+1}\}$ as approximate minimizers of $\mathcal{E}(f, G^{k+1})$ in Eq. (8). Here, $\mathcal{E}(f, G^{k+1})$ given G^k can be separated into two parts

$$\mathcal{E}(f, G^{k+1}) = \mathcal{E}(f|G^k) + \mathcal{E}_{\text{top}}(f, L_{k+1}), \quad (15)$$

where $\mathcal{E}(f|G^k)$ is energy involved in the *known* graph G^k while $\mathcal{E}_{\text{top}}(f, L_{k+1})$ refers to the *unknown* top layer L_{k+1}.

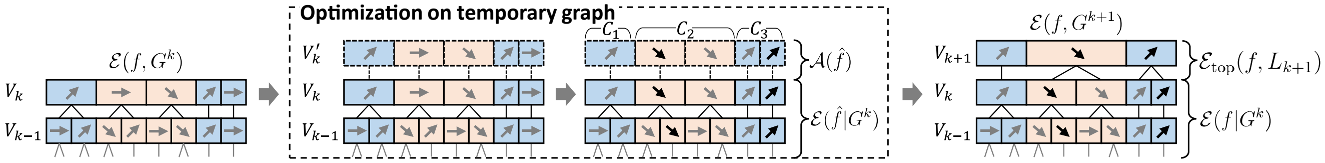
$$\mathcal{E}_{\text{top}}(f, L_{k+1}) = \mathcal{E}_{\text{mrf}}(f|L_{k+1}) + \mathcal{E}_{\text{reg}}^{k+1}(f|G^{k+1}) + \mathcal{E}_{\text{gra}}^{k+1}(V_{k+1}). \quad (16)$$

Jointly inferring G^{k+1} and its labeling f is difficult. Therefore, we assume a known *temporary graph* $\hat{G}^{k+1}$ for unknown G^{k+1}, and we replace this joint problem by a simpler labeling problem $\hat{f}$ on $\hat{G}^{k+1}$.

$$\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1}) = \mathcal{E}(\hat{f}|G^k) + \mathcal{A}(\hat{f}). \quad (17)$$

Here, $\mathcal{E}(\hat{f}|G^k)$ is equivalent to $\mathcal{E}(f|G^k)$ in Eq. (15), and $\mathcal{A}$ is an approximation of the top layer energy $\mathcal{E}_{\text{top}}$.

In following three sections, we detail lines 4–10 of Algorithm 1 and explain how we derive $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$, optimize it, and obtain the desired solution $\{f, G^{k+1}\}$ from $\{\hat{f}, \hat{G}^{k+1}\}$.



(a) Graph G^k and its labeling f (b) Temporary graph $\hat{G}^{k+1}$ (c) $\hat{G}^{k+1}$ and optimized labeling $\hat{f}$ (d) Graph G^{k+1} with new layer

Figure 3. **Bottom-up Graph Construction (one step)**. Each rectangular cell in the illustration represents a node $i \in V_l$ and a set of contiguous cells represents a graph layer V_l. The arrows and colors denote flow and alpha labels f_i (red: foreground, blue: background). (a) Graph G^k and its labeling f. (b) By duplicating the top layer V_k of G^k, we create a temporary graph $\hat{G}^{k+1}$ as an approximation of G^{k+1}. (c) We optimize the labeling $\hat{f}$ on $\hat{G}^{k+1}$. The number of unique labels in V'_k is reduced by *label costs* [15] to induce region merging. (d) V'_k is converted into a new layer V_{k+1}, by merging nodes of V'_k assigned the same label that form connected components in $\hat{G}^{k+1}$.

Energy Approximation using Temporary Graphs

We now briefly explain the conversion from $\mathcal{E}(f, G^{k+1})$ in Eq. (15) to $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ in Eq. (17). For detailed derivations, please refer to the supplementary material.

To relax the joint inference of $\mathcal{E}(f, G^{k+1})$, we create a temporary graph $\hat{G}^{k+1}$ as an approximation of G^{k+1}, by duplicating the top layer of G^k as $L'_k = (V'_k, E'_k) \leftarrow (V_k, E_k)$ (line 4 of Algorithm 1). We illustrate G^k and $\hat{G}^{k+1}$ in Figures 3 (a) and (b), respectively. Here, a labeling $\hat{f}$ on $\hat{G}^{k+1}$ (or V'_k) can equivalently express all possible f on G^{k+1} (or V_{k+1}), because V'_k is the finest form of any possible V_{k+1}. The labeling $\hat{f}$ is copied from f at line 5.

Substituting $G^{k+1} \leftarrow \hat{G}^{k+1}$ into $\mathcal{E}(f, G^{k+1})$, we derive its approximation $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ in Eq. (17) with following $\mathcal{A}$.

$$\mathcal{A}(\hat{f}) = \mathcal{E}_{\text{mrf}}(\hat{f}|L'_k) + \mathcal{E}^{k+1}_{\text{reg}}(\hat{f}|\hat{G}^{k+1}) + \mathcal{E}^{k+1}_{\text{gra}}(\hat{f}|V'_k). \quad (18)$$

The conversion from $\mathcal{E}_{\text{top}}$ in Eq. (16) to this $\mathcal{A}$ is provably exact except for only terms $\mathcal{E}^{k+1}_{\text{gra}}$ and $\mathcal{E}^{st}_{\text{reg}}$ in $\mathcal{E}_{\text{mrf}}$ of Eq. (2). Here, conversion of $\mathcal{E}^{k+1}_{\text{gra}}$ is tricky because we need to convert variables from V_{k+1} to a labeling $\hat{f}$ on V'_k. We observe that the region of each node $i \in V_{k+1}$ should optimally 1) be a connected component, 2) assigned a single label unique from neighbors, and 3) be the union of regions in V'_k. Thus, we can treat nodes $i \in V_{k+1}$ as connected components C_i of nodes in V'_k assigned the same label, *i.e.*,

$$V_{k+1} \equiv \left\{ C_i \middle| \begin{array}{l} \text{nodes } {}^\forall c \in C_i \text{ in } V'_k \text{ are assigned} \\ \text{the same label } \hat{f}_c \text{ and connected.} \end{array} \right\}. \quad (19)$$

This property allows us to rewrite $\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1})$ in Eq. (11) as a function of $\hat{f}$. To further make inference tractable, we relax the connectivity of $|V_{k+1}|$ and treat $|V_{k+1}|$ as *label costs* [15] of $\hat{f}$, *i.e.*, the number of unique labels $\hat{f}_i$ in V'_k without considering their spatial connections. In this manner, the formulation of Eq. (11) becomes the same as that of multi-region segmentation [15]. Following their model fitting approach based on alpha expansion moves [9], we treat the label costs and the likelihood terms of Eq. (11) as pairwise submodular terms and unary terms, respectively. See more discussions in the supplementary.

```
Algorithm 1: TWO-PASS OPTIMIZATION PROCESS
   input  : Two images $\mathbf{I}, \mathbf{I}'$
   output : Hierarchical graph $G$ and flow-alpha map $f$
 1 Initialize the graph: $G \leftarrow G^1$
 2 Initialize the labeling $f$ and color models $\boldsymbol{\theta}^F, \boldsymbol{\theta}^B$ (Sec. 5.3)
 3 for $k = 1, 2, \cdots$ do ◇ bottom-up graph construction ◇
 4     Create temporary $\hat{G}^{k+1}$ by duplicating $V_k$ of $G^k$ ;
 5     Initialize temporary $\hat{f}$ by copying labels from $f$ ;
 6     Perform local expansion moves ($\hat{\mathcal{E}}$, $\hat{f}$, $\hat{G}^{k+1}$, $V'_k$)
           $\hat{f} \leftarrow \operatorname{argmin} \hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ ;
 7     Create $G^{k+1}$ by merging nodes of $V'_k$ in $\hat{G}^{k+1}$ ;
 8     if rejection criterion is met then break;
 9     Update solution $\{f, G\} \leftarrow \{\hat{f}, G^{k+1}\}$ ;
10     if any stopping criteria is met then break;
11 end
12 for $k = H, \cdots, 1$ do ◇ top-down label refinement ◇
13     Perform local expansion moves ($\mathcal{E}$, $f$, $G$, $V_k$)
           $f \leftarrow \operatorname{argmin} \mathcal{E}(f|G)$ with $f_i$ fixed for $\forall i \in V_{l>k}$ ;
14 end
```

```
Algorithm 2: LOCAL EXPANSION MOVES [48, 47]
 arguments: (model $\mathcal{E}$, labeling $f$, graph $G$, target layer $V_T$)
 for each target node $i \in V_T$ do
     Make neighborhood: $N_i \leftarrow \{i\text{'s neighbors}\} \cup \{i\}$ ;
     Make expansion region: $R_i \leftarrow \{N_i\text{'s descendants}\} \cup N_i$ ;
     for each candidate proposer do
         Generate a candidate label $\boldsymbol{\ell} = (\mathbf{T}, \alpha)$ ;
         Apply a local expansion move using min-cut:
           $f \leftarrow \operatorname{argmin} \mathcal{E}(f|G)$ with $f_j \in \{f_j, \boldsymbol{\ell}\}$ for
           $j \in R_i$
     end
 end
 return $f$ ;
```

Optimization of Approximation Energy

In Figure 3 (c) and at line 6 of Algorithm 1, we minimize the approximation energy $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ of Eq. (17) with known $\hat{G}^{k+1}$. To efficiently infer the continuous 5dof labels in $\hat{f}$, we use the local expansion move method of [48, 47].

In its general form, the local expansion move algorithm repeatedly solves the following binary labeling problem for each target node $i \in V$ visited in sequence.

$$f^{(t+1)} = \operatorname{argmin} \mathcal{E}(f | f_j \in \{f_j^{(t)}, \boldsymbol{\ell}\} \text{ for } j \in R_i). \quad (20)$$

Here, $R_i \subset V$ is a set of local nodes around the target node i (named *expansion region*), and this operation tries to improve the labels of the local nodes $j \in R_i$ by assigning them either their current label $f_j^{(t)}$ or a candidate label $\boldsymbol{\ell}$. We use graph cuts [8] to solve this binary problem.

Our version of local expansion moves is summarized in Algorithm 2. During the bottom-up process, we randomly visit all nodes in the top layer V'_k (*i.e.*, target layer V_T) at line 1, and update the labeling of local nodes. In order to apply the local expansion move algorithm for our hierarchical MRF model, we extended it in two ways. First, the expansion region R_i is extended from i's neighbors (N_i at line 2) to include all their descendants (line 3). Second, when generating candidate labels $\boldsymbol{\ell}$ for the target node i (line 5), we use four types of candidate proposers listed below.

- *Expansion proposer* generates a label by copying the current label as $\boldsymbol{\ell} \leftarrow f_i$. This tries to propagate the current label f_i to nearby nodes in R_i as explained in [48].
- *Cross-view proposer* refers to the current labeling f' of the other image, and uses a label $f'_{i'}$ that gives warping to the target node region Ω_i as a candidate $\boldsymbol{\ell}$, using inverse warp of $f'_{i'}$. This is similar to *view propagation* in [7, 6].
- *Merging proposer* generates labels $\boldsymbol{\ell} \leftarrow w_i f_i + w_j f_j$ as weighted sums of i's current label f_i and its neighbors' labels f_j, $j \in N_i$. The weights $w_i, w_j \in [0, 1]$ are proportional to their region sizes $|\Omega_i|, |\Omega_j|$. This is a new extension for promoting better region merging.
- *Perturbation proposer* generates labels $\boldsymbol{\ell} \leftarrow f_i + \Delta$ by randomly perturbing the current label f_i. Similarly to [48, 7], we iterate between lines 5 and 6 several times while reducing the perturbation size $|\Delta|$ by half.

Incremental Layer Construction

In Figure 3 (d) and at line 7 of Algorithm 1, we create a new graph G^{k+1} by merging nodes of V'_k in $\hat{G}^{k+1}$. Here, V_{k+1} is created from V'_k and $\hat{f}$ using the variable conversion of Eq. (19). After merging regions, we check the number of new foreground regions at the top layer. If it is zero (line 8), we reject the new solution and stop the graph construction process. Otherwise, we adopt the new solution $\{\hat{f}, G^{k+1}\}$ as $\{f, G\}$ (line 9). Later we check the foreground count again and if it is one or not reduced from the previous iteration (line 10), we stop the graph construction process.

4.2. Top-Down Labeling Refinement

After the bottom-up phase, we further refine the labeling f during the top-down phase shown at lines 12–14 of Algorithm 1. Since G is held fixed during this step, we can directly optimize $\mathcal{E}(f|G)$ using local expansion moves without requiring the energy conversion described in Sec. 4.1. During this phase, we visit layers V_k in G in top-down order (from $k = H$ to $k = 1$) and apply local expansion moves with V_k as a target layer V_T. Here, the labeling f for the higher layers V_l ($l > k$) does not change, because the expansion regions R_i only contain nodes in layers V_k and below.

5. Implementation Details

We now discuss initialization steps and features used in our method. See the supplementary material for details.

5.1. Local HOG Features

The images are first resized so that their larger dimension becomes 512 pixels. A Gaussian pyramid is then built for each image (we use 1 octave and 1 sub-octave). From each pyramid layer, we densely extract local histogram of gradient (HOG) feature descriptors [14]. These features are extracted at every pixel on the image grid from patches of size 27 × 27 pixels. Our HOG descriptors are 96-dimensional. We use a 3×3 cell grid for each patch and 16 equally spaced bins for the oriented gradient histograms. Each gradient histogram thus has 16 bins for signed gradients and 8 bins for unsigned gradients. The histograms for each contiguous 2 × 2 block of the 3 × 3 cell grid are aggregated to form a 24-dimensional vector. These are then L2-normalized followed by element-wise truncation (using a threshold of 0.5). Four such vectors are concatenated to form the final 96-dimensional HOG descriptor. These HOG features are used to compute the flow data terms $\mathcal{E}^i_{\text{flo}}$ described earlier. They are also used to construct bag of visual words (BoW) histogram features required during the initialization stage.

5.2. BoW Histogram Features

Each HOG descriptor is vector-quantized using a K-means codebook of size 256. Next, BoW histograms are computed from several overlapping image patches of size 64 × 64 pixels. These patches are sampled every 4 pixels (both horizontally and vertically) in the image. We use integral images (one per visual word) to speed up the BoW histogram computation. All the visual words are aggregated into a histogram. This is repeated for 2×2 sub-regions. The five BoW histograms are then L2-normalized followed by element-wise square root normalization[3]. The 256-dimensional histograms are concatenated to form 1280-dimensional BoW histogram features.

5.3. Initialization

During initialization, initial flow candidates and foreground/background color models for each image are com-

[3] This is equivalent to using a Hellinger kernel instead of the Euclidean distance to measure the similarity of two feature vectors.

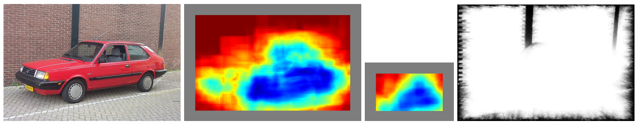


Figure 4. Min/Max ratios of BoW feature matching distances in local windows (middle). Low ratios (blue) are likely to suggest foreground. Geodesic distances from the image boundary (right) are used to add background clues (black regions).

puted as follows. First, dense matching is done using the BoW features at three levels in the image pyramid. The Euclidean distances between each pixel feature in the first image and features for all pixels within a search window in the second image are computed. Fortunately this is quite fast due to the sparsity of the BoW features. The best match is stored as a flow candidate. The ratio of the Euclidean distances of the best and worst match is computed. We use this heuristic to predict the probability of a true match, motivated by the ratio test [37] (see Figure 4).

Areas with high and low match probabilities are likely to be the "foreground" and "background" respectively. By thresholding the ratio values, we create foreground/background soft seeds and initial segmentations as input to GrabCut [40] and learn color models $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ for each image. Geodesic distance from the image boundary is used as an additional unary background likelihood term (Figure 4, right). See the supplementary material for details.

5.4. Efficient Implementation

Three key ideas allow our optimization method to be efficiently implemented. First, unary terms $\mathcal{E}^i_{\text{flo}}(f_i)$ and $\mathcal{E}^i_{\text{seg}}(f_i)$ in Eq. (2) can be efficiently computed using the tree structure of G. Specifically, the unary cost $\mathcal{E}^p(\boldsymbol{\ell})$ of a node $p \in V_l$ is computed as the sum $\sum_c \mathcal{E}^c(\boldsymbol{\ell})$ over its children $c \in V_{l-1}$, if their labels $\boldsymbol{\ell}$ are the same. This constant-label property is satisfied during local expansion moves because the candidate label $\boldsymbol{\ell}$ is the same for all nodes in an expansion region R_i. Thus, at line 6 of Algorithm 2, we compute the unary costs $\mathcal{E}^j(\boldsymbol{\ell})$ for $j \in R_i$ by sequentially summing them up from bottom to top layer nodes. Second, we exclude the pixel layer L_0 / V_0 from the graph G during the main iterations. We add it to G just before the last refinement step in the top-down phase ($k = 1$ at line 13 of Algorithm 1). Finally, we use efficient graph cuts [8] at line 6 of Algorithm 2, instead of QPBO [30]. This is possible because our energy is submodular under (local) expansion moves [48, 47]. The proofs are in the supplementary.

6. Experiments

We evaluate our method for flow and segmentation accuracy and compare it to existing methods on our new dataset.

Dataset. Our dataset comprises of 400 image pairs divided into three groups – **FG3DCar** contains 195 image pairs of vehicles from [35]. **JODS** contains 81 image pairs of airplanes, horses, and cars from [42]. **PASCAL** contains 124 image pairs of bicycles, motorbikes, buses, cars, trains from [20]. See Figure 6 for some examples from each group.

Flow accuracy. We evaluate flow accuracy by the percentage of pixels in the true foreground region that have an error measure below a certain threshold. Here, we compute the absolute flow endpoint error (*i.e.*, the Euclidean distance between estimated and true flow vectors) in a normalized scale where the larger dimensions of images are 100 pixels.

Segmentation accuracy. We use the standard *intersection-over-union* ratio metric for segmentation accuracy. As existing flow estimation methods do not recover common foreground regions, we compute them by post-processing the estimated flow maps. Specifically, given the two flow maps, we do a left-right consistency check with a suitable threshold and treat pixels that pass this test as foreground.

Settings. We strictly fixed all the parameters throughout the experiments as follows. For the data and graph term parameters, we set $\{\lambda_{\text{flo}}, \lambda_{\text{occ}}, \tau_{\text{D}}, \lambda_{\text{seg}}, \lambda_{\text{nod}}, \lambda_{\text{col}}\} \leftarrow \{0.25, 2.4, 6.5, 0.8, 125, 1\}$. For regularization parameters $\{\lambda_{\text{st1}}, \lambda_{\text{st2}}, \tau_{\text{st}}, \lambda_{\text{pc1}}, \lambda_{\text{pc2}}, \tau_{\text{pc}}\}$ associated with the pixel layer (edges E_0 and E_1^{pc}) we use $\{0.5, 20, 20, 0.005, 10, 200\}$, and for the other edges we use $\{0.1, 4, 20, 0.04, 8, 200\}$. See the supplementary material for our strategy of tuning parameters. Our method is implemented using C++ and run by a single thread on a Core i7 CPU of 3.5 GHz.

6.1. Comparison with Existing Approaches

For correspondence, we compare our method with SIFT Flow [36], DSP [29] and DFF [53][4]. We also evaluate our method using only the single layer model without hierarchy, which can be done by skipping the bottom-up construction step in Algorithm 1. This single layer method can be seen as a variant of [48]. For cosegmentation, we compare our method with Joulin *et al.* [24][5] and Faktor and Irani [17] based only on segmentation accuracies[6]. We summarize average accuracy scores for each subset in the upper part of Table 1, where flow accuracy is evaluated using a threshold of 5 pixels. The plots in Figure 5 show average flow accuracies with varying thresholds. As shown here, our method achieves the best performance on all three groups at all thresholds. Our average flow accuracies for FG3DCar, JODS and PASCAL, respectively, are up to 45%, 19% and 34% higher than SIFT Flow (best existing method). Su-

[4] We omit results of HaCohen *et al.* [19] for its low performance on our dataset. It could not find any correspondence for many image pairs.

[5] For Joulin *et al.* [24] that cannot identify the "foreground" label from $\{0, 1\}$, we refer to ground truth and choose for each image pair either 0 or 1 to maximize the scores. Results of their extension method [25] are omitted since we could not observe improvemnts over [24] in our settings.

[6] We omit results of Dai *et al.* [13] as it did not work for many image pairs. The method seems to fail in finding matches with learned templates.

Table 1. Benchmark results. FAcc is flow accuracy rate for an error threshold of 5 pixels in a normalized scale. SAcc is segmentation accuracy by intersection-over-union ratios. SAcc scores ($\star$) of optic flow mothods are computed by post-processing using left right consistency check.

Optic flow / cosegment. Methods	FG3DCar		JODS		PASCAL	
	FAcc	SAcc	FAcc	SAcc	FAcc	SAcc
Ours	**0.829**	**0.756**	**0.595**	0.504	**0.483**	**0.649**
Our single layer ([48])	0.727	**0.757**	0.473	0.499	0.414	0.616
SIFT Flow [36]	0.634	(0.420)	0.522	(0.241)	0.453	(0.407)
DSP [29]	0.487	(0.285)	0.465	(0.219)	0.382	(0.336)
DFF [53]	0.493	(0.326)	0.303	(0.207)	0.224	(0.207)
Faktor and Irani [17]	–	0.689	–	**0.544**	–	0.500
Joulin *et al.* [24]	–	0.461	–	0.320	–	0.400

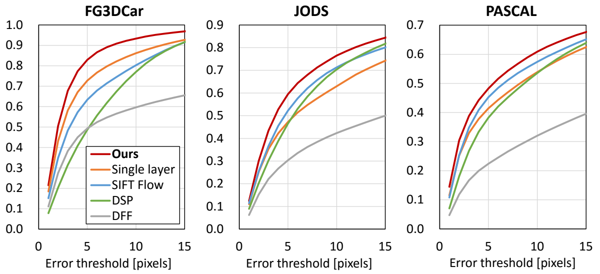


Figure 5. Average flow accuracies evaluated by endpoint errors with varying thresholds. Ours always shows best scores. DFF [53] is not robust due to lack of explicit regularization.

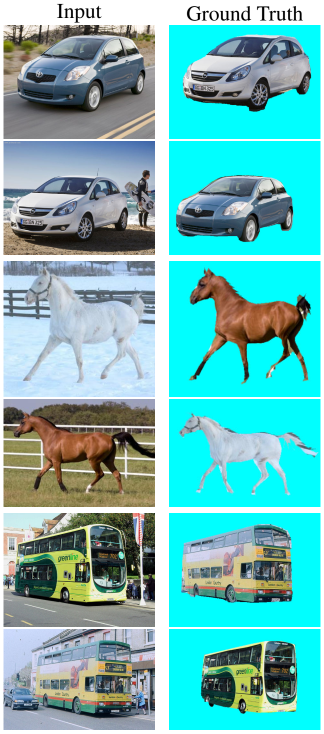


Figure 6. Dataset.

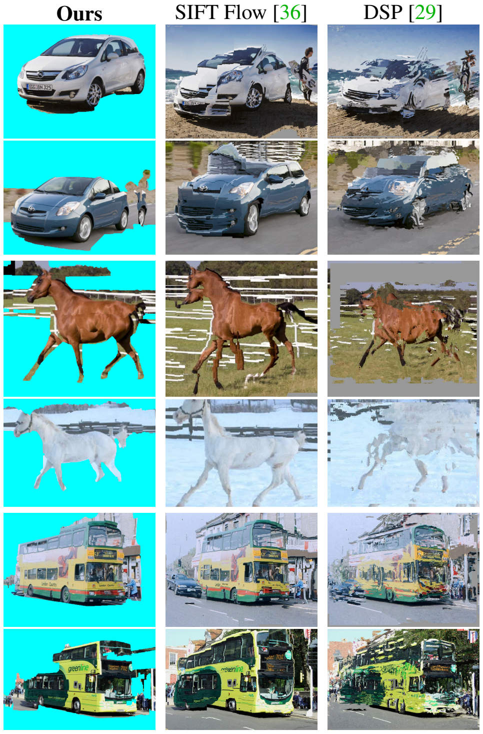


Figure 7. Correspondence results.

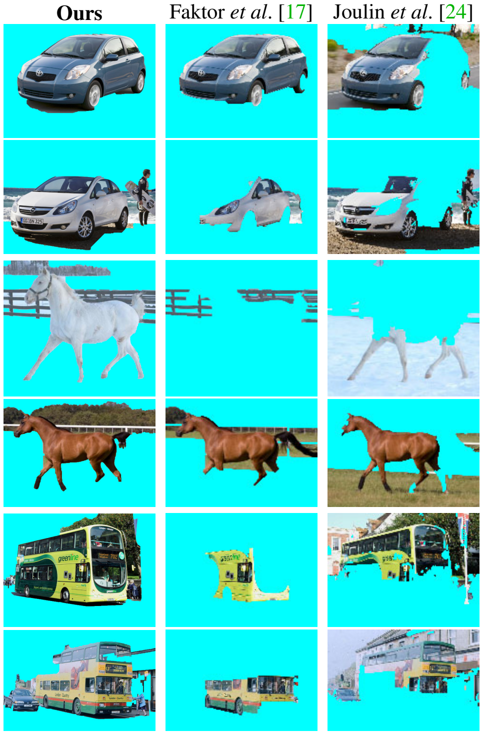


Figure 8. Cosegmentation results.

perior results to our single layer method shows the effectiveness of our hierarchical model and inference. DFF [53] cannot handle large appearance differences of objects due to lack of explicit regularization. We show qualitative comparisons with SIFT Flow [36] and DSP [29] in Figure 7.

We report average segmentation scores in the lower part of Table 1. Figure 8 shows qualitative comparisons with Faktor and Irani [17] and Joulin *et al.* [24]. Although our model for segmentation is quite simple compared to other methods, our method is competitive or has higher accuracy due to joint inference of foreground correspondence.

Running time of our method is about 7 minutes for obtaining a pair of flow-alpha maps of 512 × 384 pixels, including 1 minute for the feature extraction and initialization, 3 minutes for the final refinement step with the pixel layer.

7. Conclusion

We have presented a joint method for cosegmentation and dense correspondence estimation in two images. Our method uses a hierarchical MRF model and jointly infers the hierarchy as well as segmentation and correspondence using iterated graph cuts. Our method outperforms a number of methods designed specifically either for correspondence recovery [36, 29, 19, 53] or cosegmentation [24, 25, 17, 13]. We provide a new dataset for quantitative evaluation. Enforcing left-right consistencies on flow and segmentation maps for two images, or by using multiple images [12, 55, 25] are promising avenues for future work.

Acknowledgments. We thank Richard Szeliski, Pushmeet Kohli and Tianfan Xue for valuable feedback. T. Taniai was partially supported by JSPS KAKENHI Grant Number 14J09001 and a Microsoft Research Asia PhD fellowship.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Sustrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 34(11):2274–2282, 2012.

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 33(5):898–916, 2011.

[3] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. on Graph.*, 28(3):24:1–24:11, 2009.

[4] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized patchmatch correspondence algorithm. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 29–43, 2010.

[5] D. Batra, C. M. Univerity, A. Kowdle, D. Parikh, J. Luo, and T. Chen. icoseg: Interactive co-segmentation with intelligent scribble guidance. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[6] F. Besse, C. Rother, A. W. Fitzgibbon, and J. Kautz. PMBP: patchmatch belief propagation for correspondence field estimation. *Int'l Journal of Computer Vision*, 110(1):2–13, 2014.

[7] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In *Proc. of British Machine Vision Conf. (BMVC)*, pages 14.1–14.11, 2011.

[8] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 26(9):1124–1137, 2004.

[9] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 23(11):1222–1239, 2001.

[10] J. Cech, J. Matas, and M. Perdoch. Efficient sequential correspondence selection by cosegmentation. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 32(9):1568–1581, 2010.

[11] X. Chen, A. Shrivastava, and A. Gupta. Enriching visual knowledge bases via object discovery and segmentation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2035–2042, 2014.

[12] M. Cho, S. Kwak, C. Schmid, and J. Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1201–1210, 2015.

[13] J. Dai, Y. N. Wu, J. Zhou, and S.-C. Zhu. Cosegmentation and cosketch by unsupervised learning. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 1305–1312, 2013.

[14] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.

[15] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *Int'l Journal of Computer Vision*, 96(1):1–27, 2012.

[16] O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 33(12):2383–2395, 2011.

[17] A. Faktor and M. Irani. Co-segmentation by composition. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 1297–1304, 2013.

[18] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *Int'l Journal of Computer Vision*, 70(1):41–54, 2006.

[19] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *ACM Trans. on Graph.*, 30(4):70:1–70:10, July 2011.

[20] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, 2011.

[21] T. Hassner, V. Mayzels, and L. Zelnik-Manor. On sifts and their scales. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[22] X. He, R. S. Zemel, and M. Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 695–702, 2004.

[23] J. Hur, H. Lim, C. Park, and S. C. Ahn. Generalized deformable spatial pyramid: Geometry-preserving dense correspondence estimation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1392–1400, 2015.

[24] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[25] A. Joulin, F. Bach, and J. Ponce. Multi-class cosegmentation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[26] A. Joulin, K. Tang, and L. Fei-Fei. Efficient image and video co-localization with frank-wolfe algorithm. In *Proc. of European Conf. on Computer Vision (ECCV)*, 2014.

[27] R. Kennedy and C. J. Taylor. Hierarchically-constrained optical flow. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[28] G. Kim, E. P. Xing, L. Fei-Fei, and T. Kanade. Distributed cosegmentation via submodular optimization on anisotropic diffusion. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 169–176. IEEE, 2011.

[29] J. Kim, C. Liu, F. Sha, and K. Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2307–2314, 2013.

[30] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts-a review. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 29(7):1274–1279, 2007.

[31] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 26(2):147–159, 2004.

[32] A. Kowdle, S. N. Sinha, and R. Szeliski. Multiple view object cosegmentation using appearance and stereo cues. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 789–803. Springer Berlin Heidelberg, 2012.

[33] C. Lei, J. Selzer, and Y.-H. Yang. Region-tree based stereo using dynamic programming optimization. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*,

volume 2, pages 2378–2385, 2006.

[34] C. Lei and Y.-H. Yang. Optical flow estimation on coarse-to-fine region-trees using discrete optimization. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 1562–1569. IEEE, 2009.

[35] Y.-L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *Proc. of European Conf. on Computer Vision (ECCV)*, 2014.

[36] C. Liu, J. Yuen, and A. Torralba. SIFT Flow: Dense Correspondence across Scenes and Its Applications. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 33(5):978–994, 2011.

[37] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int'l Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[38] J. Lu, H. Yang, D. Min, and M. N. Do. Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1854–1861, 2013.

[39] W. Qiu, X. Wang, X. Bai, Z. Tu, et al. Scale-space sift flow. In *2014 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1112–1119. IEEE, 2014.

[40] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Trans. on Graph.*, 23(3):309–314, 2004.

[41] C. Rother, T. P. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 993–1000, 2006.

[42] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1939–1946, 2013.

[43] J. C. Rubio, J. Serrat, A. Lopez, and N. Paragios. Unsupervised co-segmentation through region matching. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 749–756, 2012.

[44] R. Sibson. A Brief Description of Natural Neighbour Interpolation. In *Interpreting multivariate data*, chapter 2, pages 21–36. John Wiley & Sons, 1981.

[45] J. Sivic, B. C. Russell, A. Efros, A. Zisserman, W. T. Freeman, et al. Discovering objects and their location in images. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 370–377, 2005.

[46] K. Tang, A. Joulin, L.-J. Li, and L. Fei-Fei. Co-localization in real-world images. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[47] T. Taniai, Y. Matsushita, and T. Naemura. Graph cut based continuous stereo matching using locally shared labels. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1613–1620, 2014.

[48] T. Taniai, Y. Matsushita, Y. Sato, and T. Naemura. Continuous Stereo Matching Using Local Expansion Moves. arXiv:1603.08328, http://arxiv.org/abs/1603.08328, 2016.

[49] S. Todorovic and N. Ahuja. Region-based hierarchical image matching. *Int'l Journal of Computer Vision*, 78(1):47–66, 2008.

[50] T. Tuytelaars, C. H. Lampert, M. B. Blaschko, and W. Buntine. Unsupervised object discovery: A comparison. *Int'l Journal of Computer Vision*, 88(2):284–302, 2010.

[51] S. Vicente, J. Carreira, L. Agapito, and J. Batista. Reconstructing PASCAL VOC. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 41–48, 2014.

[52] S. Vicente, C. Rother, and V. Kolmogorov. Object cosegmentation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2217–2224, 2011.

[53] H. Yang, W. Lin, and J. Lu. DAISY filter flow: A generalized discrete approach to dense correspondences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3406–3413, 2014.

[54] C. Zhang, C. Shen, and T. Shen. Unsupervised feature learning for dense correspondences across scenes. *Int'l Journal of Computer Vision*, pages 1–18, 2015.

[55] T. Zhou, Y. J. Lee, S. X. Yu, and A. A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1191–1200, 2015.

Joint Recovery of Dense Correspondence and Cosegmentation in Two Images — Supplementary Material —

Tatsunori Taniai
The University of Tokyo

Sudipta N. Sinha
Microsoft Research

Yoichi Sato
The University of Tokyo

In the supplementary material we present derivations and proofs associated with the proposed technique that were omitted from the main paper due to lack of space. Some additional notes and implementation details are also provided. We will be referring to certain equations and figures in the main paper. Please note that the new equations and figures provided in the supplementary material have numbers with the letter A as prefix to distinguish them from those in the main paper. We also provide additional qualitative comparisons as a supplementary video on our project website.

A. Continuous Alpha Map Formulation

Here, we explain why in our method, the per-pixel segmentation labels must be continuous alpha-matte values $\alpha \in [0,1]$ rather than binary values $\{0,1\}$. If α were binary, the flows $\mathbf{T}_i$ at nodes labeled background ($\alpha_i = 0$) would be under-constrained, because the flow data term $\mathcal{E}^i_{\text{flo}}$ in Eq. (3) at such nodes would always be a constant λ_{occ} regardless of the values of $\mathbf{T}_i$. This would be problematic, because if true foreground nodes are incorrectly labeled background in early stages of our inference process, it would be harder to recover their true flow labels in later iterations. To avoid this issue, we require α to be a continuous value that is larger than a small positive value (0.1 in our implementation). By doing this we will have meaningful flow labels $\mathbf{T}_i$ even at nodes labeled (incorrectly) background, because those flow labels still slightly affect matching energies of $\mathcal{E}^i_{\text{flo}}$.

B. Energy Approximation

Next, we present the derivations of our approximation energy described in Section 4.1.

We first derive the energy function $\mathcal{E}(f, G^{k+1})$ in the form of Eq. (15). In order to simplify the energy formulation in Eq. (8), we denote energies involved in each layer as

$$\mathcal{E}^l_{\text{lay}}(f,G) = \mathcal{E}_{\text{mrf}}(f|L_l) + \mathcal{E}^l_{\text{reg}}(f|G) + \mathcal{E}^l_{\text{gra}}(V_l) \tag{A1}$$

and rewrite the energy function $\mathcal{E}(f, G^{k+1})$ as the sum of layer energies

$$\begin{aligned}\mathcal{E}(f,G^{k+1}) &= \sum_{l=0}^{k+1} \mathcal{E}^l_{\text{lay}}(f,G^{k+1}) && \text{(A2)}\\ &= \sum_{l=0}^{k} \mathcal{E}^l_{\text{lay}}(f,G^{k}) + \mathcal{E}^{k+1}_{\text{lay}}(f,G^{k+1}) && \text{(A3)}\\ &= \underbrace{\mathcal{E}(f,G^k)}_{\mathcal{E}(f|G^k)} + \underbrace{\mathcal{E}^{k+1}_{\text{lay}}(f,G^{k+1})}_{\mathcal{E}_{\text{top}}(f,L^{k+1})}. && \text{(A4)}\end{aligned}$$

Assuming that G^k is known from the previous iteration, we denote $\mathcal{E}(f, G^k)$ as $\mathcal{E}(f|G^k)$, and $\mathcal{E}^{k+1}_{\text{lay}}(f, G^{k+1})$ as $\mathcal{E}_{\text{top}}(f, L^{k+1})$

to obtain Eq. (15).

To approximate the above $\mathcal{E}(f, G^{k+1})$, we create a temporary graph $\hat{G}^{k+1}$ as an approximation of G^{k+1}, by duplicating the top layer of G^k as $L'_k = (V'_k, E'_k) \leftarrow (V_k, E_k)$. We further define a labeling $\hat{f}$ on this temporary graph $\hat{G}^{k+1}$. Since f and $\hat{f}$ are defined on the different graphs (G^{k+1} and $\hat{G}^{k+1}$) or different top layers (V_{k+1} and V'_k), we cannot simply assume $f = \hat{f}$. However, V'_k representing a superpixel segmentation is the finest form of any possible V_{k+1} due to the tree structure of G. Therefore, we can always define $\hat{f}$ so that f and $\hat{f}$ are equivalent $f \equiv \hat{f}$, *i.e.*, the pixelwise labeling included by both f and $\hat{f}$ are identical.

Using $\hat{f}$ and $\hat{G}^{k+1}$, our approximation function $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ for $\mathcal{E}(f, G^{k+1})$ in the form of Eq. (17) is obtained by substituting $G^{k+1} \leftarrow \hat{G}^{k+1}$ and $f \leftarrow \hat{f}$ into $\mathcal{E}(f, G^{k+1})$.

$$\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1}) = \mathcal{E}(\hat{f}, \hat{G}^{k+1}) \tag{A5}$$

$$= \underbrace{\mathcal{E}(\hat{f}, G^k)}_{\mathcal{E}(\hat{f}|G^k)} + \underbrace{\mathcal{E}^{k+1}_{\text{lay}}(\hat{f}, \hat{G}^{k+1})}_{\mathcal{A}(\hat{f})}. \tag{A6}$$

Here, because G^{k+1} and $\hat{G}^{k+1}$ share the same structure except for the top layers, the energies $\mathcal{E}(\cdot|G^k)$ involved in the bottom hierarchy G^k are equivalent between Eqs. (A4) and (A6). To discuss how $\mathcal{A}(\hat{f})$ approximates $\mathcal{E}^{k+1}_{\text{top}}(f, L^{k+1})$, we write it as

$$\mathcal{A}(\hat{f}) = \lambda_{\text{flo}} \sum_{i \in V'_k} \mathcal{E}^i_{\text{flo}}(\hat{f}_i) + \lambda_{\text{seg}} \sum_{i \in V'_k} \mathcal{E}^i_{\text{seg}}(\hat{f}_i) + \sum_{(s,t) \in E'_k} w_{st}\, \mathcal{E}^{st}_{\text{reg}}(\hat{f}_s, \hat{f}_t) + \sum_{(p,c) \in E^{\text{pc}\prime}_{k+1}} w_{pc}\, \mathcal{E}^{pc}_{\text{reg}}(f_p, f_c) + \mathcal{E}^{k+1}_{\text{gra}}(\hat{f}|V'_k). \tag{A7}$$

Here, the conversion for the three terms $\mathcal{E}^i_{\text{flo}}$, $\mathcal{E}^i_{\text{seg}}$ and $\mathcal{E}^{pc}_{\text{reg}}$ is exact, *i.e.*, those terms in $\mathcal{E}_{\text{top}}$ and corresponding terms in $\mathcal{A}$ yield the same energies as long as $f \equiv \hat{f}$. Next, we explain why these three conversions are exact and why the conversion for the two remaining terms are approximate.

Exact Conversion of Flow and Cosegmentation Data Terms

Exactness for the unary terms $\mathcal{E}^i_{\text{flo}}$ and $\mathcal{E}^i_{\text{seg}}$ in Eqs. (3) and (5) is shown in the same way. Notice that nodes in V_{k+1} are always obtained by merging nodes of V'_k, by following the rule of Eq. (19). Therefore, we can assume the domain Ω_i of each node $i \in V_{k+1}$ is the union of the domains of a connected component C_i of nodes in V'_k.

$$\Omega_i = \bigcup_{i' \in C_i} \Omega_{i'} \tag{A8}$$

Furthermore, from $f \equiv \hat{f}$ it holds that $f_i = \hat{f}_{i'}$ for $i \in V_{k+1}$ and $i' \in C_i$. Using these properties, a unary term $\mathcal{E}^i$ in $\mathcal{E}_{\text{top}}$ can be exactly converted to the form in $\mathcal{A}$ as follows. (Changes from previous equations are colored by blue).

$$\sum_{i \in V_{k+1}} \mathcal{E}^i(f_i) = \sum_{i \in V_{k+1}} \sum_{\mathbf{p} \in \Omega_i} \phi_{\mathbf{p}}(f_i) \tag{A9}$$

$$= \sum_{i \in V_{k+1}} \sum_{i' \in C_i} \sum_{\mathbf{p} \in \Omega_{i'}} \phi_{\mathbf{p}}(f_i) \tag{A10}$$

$$= \sum_{i \in V_{k+1}} \sum_{i' \in C_i} \sum_{\mathbf{p} \in \Omega_{i'}} \phi_{\mathbf{p}}(\hat{f}_{i'}) \tag{A11}$$

$$= \sum_{i' \in V'_k} \sum_{\mathbf{p} \in \Omega_{i'}} \phi_{\mathbf{p}}(\hat{f}_{i'}) \tag{A12}$$

$$= \sum_{i \in V'_k} \mathcal{E}^i(\hat{f}_i) \tag{A13}$$

Exact Conversion of Multi-layer Regularization Term

We perform a similar derivation for the multi-layer regularization term $\mathcal{E}_{\text{reg}}^{pc}$ in Eq. (10). From Figures 3 (c) and (d), we can see that each of the parent-child edges $(p,c) \in E_{k+1}^{\text{pc}}$ in the top layer of G^{k+1} has a corresponding edge $(p',c) \in E_{k+1}^{\text{pc}'}$ in $\hat{G}^{k+1}$ that has the same child c. Furthermore, for each one of those edges, $\mathbf{T}_p(\mathbf{p}) = \mathbf{T}_{p'}(\mathbf{p})$ and $\alpha_p = \alpha_{p'}$, since $f \equiv \hat{f}$. Therefore, we can exactly convert $\mathcal{E}_{\text{reg}}^{pc}$ in $\mathcal{E}_{\text{top}}$ to the form in $\mathcal{A}$ as follows.

$$\begin{aligned}\sum_{(p,c)\in E_{k+1}^{\text{pc}}} w_{pc}\, \mathcal{E}_{\text{reg}}^{pc}(f_p, f_c) &= \sum_{(p,c)\in E_{k+1}^{\text{pc}}} w_{pc} \left[\lambda_{\text{pc1}} \min\{\alpha_p, \alpha_c\} \psi^{pc}(\mathbf{c}_c) + \lambda_{\text{pc2}} |\alpha_p - \alpha_c|\right] && \text{(A14)}\\ &= \sum_{(p,c)\in E_{k+1}^{\text{pc}}} |\Omega_c| \left[\lambda_{\text{pc1}} \min\{\alpha_p, \alpha_c\} \min\{\|\mathbf{T}_p(\mathbf{c}_c) - \mathbf{T}_c(\mathbf{c}_c)\|_2, \tau_{\text{pc}}\} + \lambda_{\text{pc2}} |\alpha_p - \alpha_c|\right] && \text{(A15)}\\ &= \sum_{(p',c)\in E_{k+1}^{\text{pc}'}} |\Omega_c| \left[\lambda_{\text{pc1}} \min\{\alpha_{p'}, \alpha_c\} \min\{\|\mathbf{T}_{p'}(\mathbf{c}_c) - \mathbf{T}_c(\mathbf{c}_c)\|_2, \tau_{\text{pc}}\} + \lambda_{\text{pc2}} |\alpha_{p'} - \alpha_c|\right] && \text{(A16)}\\ &= \sum_{(p,c)\in E_{k+1}^{\text{pc}'}} w_{pc}\, \mathcal{E}_{\text{reg}}^{pc}(\hat{f}_p, \hat{f}_c) && \text{(A17)}\end{aligned}$$

Approximate Conversion of Spatial Regularization Term

For the spatial regularization term $\mathcal{E}_{\text{reg}}^{st}$ in Eq. (6), we split it into two parts.

$$\sum_{(s,t)\in E'_k} w_{st}\, \mathcal{E}_{\text{reg}}^{st}(\hat{f}_s, \hat{f}_t) = \lambda_{\text{st1}} \sum_{(s,t)\in E'_k} w_{st} \min\{\alpha_s, \alpha_t\} \sum_{\mathbf{p}\in B_{st}} \psi^{st}(\mathbf{p})/|B_{st}| + \lambda_{\text{st2}} \sum_{(s,t)\in E'_k} w_{st}\, |\alpha_s - \alpha_t|. \tag{A18}$$

Here, the first and second parts evaluate flow and segmentation smoothness, respectively. We can show exact conversion for the segmentation smoothness part. To show this, we classify the edges of E'_k in $\hat{G}^{k+1}$ into two types: Type A) edges $(s',t') \in A$ across two different components $s' \in C_s$ and $t' \in C_t$. Type B) edges $(s'',t'') \in B$ within the same component $s'', t'' \in C_i$. Notice that $\mathcal{E}_{\text{reg}}^{st}(f_s, f_t) = 0$ for Type A edges, because $f_s = f_t$ holds in the same component. We now derive exact conversion for the segmentation smoothness part as follows.

$$\begin{aligned}\sum_{(s,t)\in E_{k+1}} w_{st}\, |\alpha_s - \alpha_t| &= \sum_{(s,t)\in E_{k+1}} \left[\sum_{(s',t')\in A_{st}} w_{s't'}\right] |\alpha_s - \alpha_t| && \text{(A19)}\\ &= \sum_{(s,t)\in E_{k+1}} \sum_{(s',t')\in A_{st}} w_{s't'}\, |\alpha_{s'} - \alpha_{t'}| && \text{(A20)}\\ &= \sum_{(s',t')\in A} w_{s't'}\, |\alpha_{s'} - \alpha_{t'}| && \text{(A21)}\\ &= \sum_{(s',t')\in A} w_{s't'}\, |\alpha_{s'} - \alpha_{t'}| + \sum_{(s'',t'')\in B} w_{s''t''}\, |\alpha_{s''} - \alpha_{t''}| && \text{(A22)}\\ &= \sum_{(s,t)\in E'_k} w_{st}\, |\alpha_s - \alpha_t| && \text{(A23)}\end{aligned}$$

Here, $w_{st} = \sum w_{s't'}$ in Eq. (A19) is from Eq. (14), but the definition of (s',t') can be equivalently replaced as Type A edges $(s',t') \in A_{st}$ where $s' \in C_s$ and $t' \in C_t$. Equation (A20) is from $f \equiv \hat{f}$, where it holds that $\alpha_i = \alpha'_i$ for $i \in V_{k+1}$ and $i' \in C_i$.

In contrast, the conversion of the flow smoothness part in Eq. (A18) is not always exact. However, the pixel locations $\mathbf{p}$ where the flow difference penalties $\psi^{st}(\mathbf{p})$ actually occur are the same in $\mathcal{E}_{\text{top}}$ and $\mathcal{A}$. Furthermore, the total costs of the flow smoothness part are equally bounded by $\sum_{(s,t)\in E_{k+1}} \lambda_{\text{st1}} w_{st} \tau_{\text{st}}$ in both $\mathcal{E}_{\text{top}}$ and $\mathcal{A}$. Thus, Eq. (A18) is a good approximation for the spatial regularization term.

Approximate Conversion of Graph Validity Term

To derive an approximation $\mathcal{E}^{k+1}_{\text{gra}}(\hat{f}|V'_k)$ for the graph validity term $\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1})$ in Eq. (11), we need to deal with two issues. 1) We need to convert variables from the node structure V_{k+1} in $\mathcal{E}_{\text{top}}$ to the labeling $\hat{f}$ on V'_k in $\mathcal{A}$. 2) The approximation function must be pairwise submodular energies for allowing graph cut based optimization.

For the first issue, we apply the variable conversion of Eq. (19) and regard V_{k+1} as a function $V_{k+1}(\hat{f})$ that represents a set of connected components C_i of nodes in V'_k assigned the same label. Thus, $\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1})$ is converted to a function of $\hat{f}$ as follows.

$$\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1}) = \lambda_{\text{nod}}\beta^{k+1}|V_{k+1}| - \lambda_{\text{col}} \sum_{i\in V_{k+1}} \sum_{\mathbf{p}\in\Omega_i} \ln P(\mathbf{I_p}|\boldsymbol{\theta}^i) \tag{A24}$$

$$= \lambda_{\text{nod}}\beta^{k+1}|V_{k+1}(\hat{f})| - \lambda_{\text{col}} \sum_{i\in V_{k+1}} \sum_{\mathbf{p}\in\Omega_i} \ln P(\mathbf{I_p}|\boldsymbol{\theta}^{C_i}) \tag{A25}$$

$$= \lambda_{\text{nod}}\beta^{k+1}|V_{k+1}(\hat{f})| - \lambda_{\text{col}} \sum_{i'\in V'_k} \sum_{\mathbf{p}\in\Omega_{i'}} \ln P(\mathbf{I_p}|\boldsymbol{\theta}^{C_i}) \tag{A26}$$

Here, $|V_{k+1}(\hat{f})|$ is the count of the components defined by the labeling $\hat{f}$, and $\boldsymbol{\theta}^{C_i}$ is the color distribution within the region of a component C_i that $i' \in V'_k$ belongs to. The fact that the computation of both $|V_{k+1}(\hat{f})|$ and $\boldsymbol{\theta}^{C_i}$ involves regional (higher-order) information of $\hat{f}$ raises the second issue.

To deal with the second issue of higher-order terms, we relax the connectivity of $|V_{k+1}(\hat{f})|$ and treat it as the count of unique labels $\hat{f}_i$ in $i \in V'_k$ without considering their spatial connections.

$$|V_{k+1}(\hat{f})| \simeq \sum_{L\in\{\text{all labels}\}} \delta_L(\hat{f}), \tag{A27}$$

where $\delta_L(\hat{f}) = 1$ if $\exists i \in V'_k : \hat{f}_i = L$; otherwise $\delta_L(\hat{f}) = 0$. In this manner, $|V_{k+1}(\hat{f})|$ becomes *label costs* [2] of $\hat{f}$, and the formulation of Eq. (A26) is the same as that of multi-region segmentation of [2]. In their model fitting approach, the label costs are optimized as pairwise submodular terms under alpha expansion moves with additional auxiliary variables. Our optimization approach using local expansion moves allows the same strategy. Furthermore, the distribution $\boldsymbol{\theta}^{C_i}$ is treated as a label $\boldsymbol{\theta}_i$ given by $\hat{f}_i$, rather than a value computed from C_i. Thus, the likelihood terms in Eq. (A26) are approximated as unary potentials as follows.

$$-\sum_{i'\in V'_k} \sum_{\mathbf{p}\in\Omega_{i'}} \ln P(\mathbf{I_p}|\boldsymbol{\theta}^{C_i}) \simeq \sum_{i\in V'_k} \mathcal{E}^i_{\text{gra}}(\hat{f}_i) \tag{A28}$$

where $\mathcal{E}^i_{\text{gra}}(\hat{f}_i)$ evaluates the given distribution label $\boldsymbol{\theta}_i$ included in $\hat{f}_i$ as

$$\mathcal{E}^i_{\text{gra}}(\hat{f}_i) = -\sum_{\mathbf{p}\in\Omega_i} \ln P(\mathbf{I_p}|\boldsymbol{\theta}_i). \tag{A29}$$

Note that the energy conversion is unnecessary for the graph terms in $\mathcal{E}(\hat{f}|G^k)$, because those terms are constant with the fixed G^k. Likewise, it is unnecessary in the whole process of the top-down labeling refinement phase.

Consequently, $\hat{f}$ becomes the following labeling on $\hat{G}^{k+1}$.

$$\hat{f}_i = \begin{cases} (\mathbf{T}_i, \alpha_i, \boldsymbol{\theta}_i) & \text{if } i \in V'_k \\ (\mathbf{T}_i, \alpha_i) & \text{if } i \in V_l \ (0 \le l \le k) \end{cases} . \tag{A30}$$

The distribution label $\boldsymbol{\theta}_i$ of $i \in V'_k$ is initialized as the color distribution of the region Ω_i. Except for the cross-view proposer, the proposal generation for distribution labels is essentially the same as that of other labels $(\mathbf{T}, \alpha)$. The expansion and perturbation proposers simply copy the current label $\boldsymbol{\theta}_i$ of the target node i as a candidate. The average proposer generates candidates as the weighted sum of two distributions $w_i\boldsymbol{\theta}_i + w_j\boldsymbol{\theta}_j$. The cross-view proposer generates a candidate as the distribution within the region Ω_i of the target node i.

C. Initiailzation of Color Models

Here, we explain the implementation details of the initialization of color models $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ omitted in Section 5.3.

Geodesic Distance

We first compute a geodesic distance map for each of the input images. At every pixel $\mathbf{p}$ we compute the shortest geodesic distance to any of the image boundary pixels $\mathbf{q} \in B$:

$$D(\mathbf{p}) = \min_{\mathbf{q} \in B} d(\mathbf{p}, \mathbf{q}), \tag{A31}$$

where $d(\mathbf{p}, \mathbf{q})$ is the geodesic distance between two pixels $\mathbf{p}$ and $\mathbf{q}$ define as

$$d(\mathbf{p}, \mathbf{q}) = \min_{s \in \mathcal{P}} \sum_{k=1}^{|s|-1} \|\mathbf{I}(\mathbf{s}(k+1)) - \mathbf{I}(\mathbf{s}(k))\|_2. \tag{A32}$$

Here, $\mathcal{P}$ is the set of all paths joining $\mathbf{p}$ and $\mathbf{q}$. The approximate computation of $D(\mathbf{p})$ is efficiently implemented using a linear-order algorithm of [11].

We further normalize the value range of the geodesic distance map by

$$\bar{D}(\mathbf{p}) = e^{-D(\mathbf{p})^2/\gamma}. \tag{A33}$$

The parameter γ is given as $\gamma = \eta\sigma^2$ where $\sigma = E[\|\mathbf{I}(\mathbf{p}) - \mathbf{I}(\mathbf{q})\|_2]$ is computed as the expectation over all spatial neighbors $(\mathbf{p}, \mathbf{q})$, and η is set to 20 in our implementation. The values of $1 - \bar{D}(\mathbf{p})$ are visualized in the right part of Figure 4.

Seeds and Initial Mask Creation for GrabCut

Secondly, we compute seeds and initial masks of foreground and background as input to GrabCut [8]. The seeds of foreground and background regions give constant unary likelihoods. The initial masks are used to initialize the color distributions used in GrabCut. We compute these regions using the ratio values and the geodesic distance as follows.

As explained in Section 5.3, we have three ratio values $\{r_1, r_2, r_3\}$ at each pixel computed from the three levels of the image pyramid. For each level, we normalize the ratio values to be in the range of $[0, 1]$ using the minimum and maximum ratio values. After the layerwise normalization, we integrate the three ratio values to obtain a single value as $r = r_1r_2r_3 + (1-r_1)r_2r_3 + r_1(1-r_2)r_3 + r_1r_2(1-r_3)$. We then create the foreground / background seeds and foreground / background masks as regions where $r < 0.05$, $r > 0.95$, $r < 0.70$ and $r > 0.85$, respectively. The regions of foreground seed and mask are further reduced if the geodesic distance is $\bar{D}(\mathbf{p}) > 0.5$.

In our implementation, the color likelihood terms of GrabCut are implemented by 64^3 bins of RGB color histograms with a weight coefficient of 1. The pixels in the foreground/background seeds are assigned a constant likelihood value of 10. Using the geodesic distance in Eq. (A33), we also add background likelihood values of $10\bar{D}(\mathbf{p})$. For efficiency, we use the superpixel nodes of V_1 during this step and reuse them again in our main algorithm. Finally, we obtain estimated color models $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ of an image after a few iterations of GrabCut. We perform this computation for each of the two images.

D. Submodularity

As discussed in [10, 9] the submodularity condition of local expansion move energies in Eq. (20) is the same as that of conventional alpha expansion moves [1]. To prove that our energy is submodular under expansion moves, we need to show that our pairwise regularization terms $\mathcal{E}^{sp}_{\text{reg}}$ and $\mathcal{E}^{pc}_{\text{reg}}$ in Eqs. (6) and (10) are submodular. To simplify discussions, we rewrite these terms as a pairwise function, as follows.

$$\phi(\mathbf{x},\mathbf{y}) = \min\{x,y\}\psi(\mathbf{x},\mathbf{y}) + \lambda|x-y|. \tag{A34}$$

Here, $\lambda \geq 0$ is a scalar weight, a bold $\mathbf{x}$ denotes a label vector of $(\mathbf{T},\alpha)$ while a non-bold x denotes its scalar alpha label $\alpha \in [0,1]$. The two terms $\mathcal{E}^{sp}_{\text{reg}}$ and $\mathcal{E}^{pc}_{\text{reg}}$ can be expressed in this form by properly defining $\psi(\mathbf{x},\mathbf{y})$. Using this notation we prove the following two lemmas.

Lemma 1 *If $\psi(,)$ satisfies the following three conditions for any $\mathbf{x},\mathbf{y},\mathbf{z}$*

$$0 \leq \psi(\mathbf{x},\mathbf{y}) \leq \tau, \tag{A35}$$

$$\psi(\mathbf{x},\mathbf{x}) = 0, \tag{A36}$$

$$\psi(\mathbf{x},\mathbf{y}) + \psi(\mathbf{z},\mathbf{z}) \leq \psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}), \tag{A37}$$

and if

$$\tau \leq 2\lambda, \tag{A38}$$

then $\phi(\mathbf{x},\mathbf{y})$ is submodular under expansion moves, i.e., *it satisfies the following submodularity condition of expansion moves [1, 6]:*

$$\phi(\mathbf{x},\mathbf{y}) + \phi(\mathbf{z},\mathbf{z}) \leq \phi(\mathbf{x},\mathbf{z}) + \phi(\mathbf{z},\mathbf{y}). \tag{A39}$$

Proof.

Notice that $\phi(\mathbf{z},\mathbf{z}) = 0$. Using this and assuming $x \geq y$ without loss of generality, Eq. (A39) can be expressed as

$$\min\{x,z\}\psi(\mathbf{x},\mathbf{z}) + \lambda|x-z| + \min\{z,y\}\psi(\mathbf{z},\mathbf{y}) + \lambda|z-y| - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) \geq 0. \tag{A40}$$

The proof for the above inequity is divided into the following three cases depending on z.

Case 1 where $x \geq y \geq z \geq 0$. We show in this case that

$$\begin{aligned}
\text{Eq. (A40, left)} &= z\psi(\mathbf{x},\mathbf{z}) + \lambda(x-z) + z\psi(\mathbf{z},\mathbf{y}) + \lambda(y-z) - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) && \text{(A41)}\\
&= z\Big[\psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y})\Big] - y\psi(\mathbf{x},\mathbf{y}) + 2\lambda(y-z) && \text{(A42)}\\
&\geq z\psi(\mathbf{x},\mathbf{y}) - y\psi(\mathbf{x},\mathbf{y}) + 2\lambda(y-z) && \text{(A43)}\\
&= (y-z)\Big[2\lambda - \psi(\mathbf{x},\mathbf{y})\Big] && \text{(A44)}\\
&\geq (y-z)\Big[2\lambda - \tau\Big] && \text{(A45)}\\
&\geq 0. && \text{(A46)}
\end{aligned}$$

Case 2 where $x \geq z \geq y \geq 0$. Similarly, we show that

$$\begin{aligned} \text{Eq. (A40, left)} &= z\psi(\mathbf{x},\mathbf{z}) + \lambda(x-z) + y\psi(\mathbf{z},\mathbf{y}) + \lambda(z-y) - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) && \text{(A47)} \\ &= z\psi(\mathbf{x},\mathbf{z}) + y\psi(\mathbf{z},\mathbf{y}) - y\psi(\mathbf{x},\mathbf{y}) && \text{(A48)} \\ &\geq y\Big[\psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}) - \psi(\mathbf{x},\mathbf{y})\Big] && \text{(A49)} \\ &\geq 0. && \text{(A50)} \end{aligned}$$

Case 3 where $z \geq x \geq y \geq 0$. Finally, we show that

$$\begin{aligned} \text{Eq. (A40, left)} &= x\psi(\mathbf{x},\mathbf{z}) + \lambda(z-x) + y\psi(\mathbf{z},\mathbf{y}) + \lambda(z-y) - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) && \text{(A51)} \\ &= x\psi(\mathbf{x},\mathbf{z}) + y\psi(\mathbf{z},\mathbf{y}) - y\psi(\mathbf{x},\mathbf{y}) + 2\lambda(z-x) && \text{(A52)} \\ &\geq y\Big[\psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}) - \psi(\mathbf{x},\mathbf{y})\Big] + 2\lambda(z-x) && \text{(A53)} \\ &\geq 0. && \text{(A54)} \end{aligned}$$

Lemma 2 *If $\psi(\mathbf{x},\mathbf{y})$ is given by a form of the truncated Euclidean distance as*

$$\psi(\mathbf{x},\mathbf{y}) = \min\{\|\mathbf{x}-\mathbf{y}\|_2, \tau\}, \tag{A55}$$

then $\psi(\mathbf{x},\mathbf{y})$ satisfies the aforementioned three conditions of Eqs. (A35) – (A37).

Proof.

The first and second conditions are obvious. We can also show the third condition as follows.

$$\begin{aligned} \psi(\mathbf{x},\mathbf{y}) + \psi(\mathbf{z},\mathbf{z}) &= \psi(\mathbf{x},\mathbf{y}) && \text{(A56)} \\ &= \min\{\|\mathbf{x}-\mathbf{y}\|_2, \tau\} && \text{(A57)} \\ &= \min\{\|(\mathbf{x}-\mathbf{z}) - (\mathbf{y}-\mathbf{z})\|_2, \tau\} && \text{(A58)} \\ &\leq \min\{\|\mathbf{x}-\mathbf{z}\|_2 + \|\mathbf{y}-\mathbf{z}\|_2, \tau\} && \text{(A59)} \\ &\leq \min\{\|\mathbf{x}-\mathbf{z}\|_2, \tau\} + \min\{\|\mathbf{y}-\mathbf{z}\|_2, \tau\} && \text{(A60)} \\ &= \psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}) && \text{(A61)} \end{aligned}$$

The above two lemmas directly derive the submodularity for the parent-children term $\mathcal{E}^{pc}_{\text{reg}}$ using substitutions $\lambda = \lambda_{\text{pc2}}$ and $\tau = \lambda_{\text{pc1}}\tau_{\text{pc}}$. By slightly modifying Eq. (A55) for the spatial term $\mathcal{E}^{st}_{\text{reg}}$, it can also be shown to be submodular where $\lambda = \lambda_{\text{st2}}$ and $\tau = \lambda_{\text{st1}}\tau_{\text{st}}$.

E. Tuning Hyper Parameters

We explain our strategy of tuning parameters. Since the graph term is independent of the labeling, we start with a simple energy function consisting of only the graph term. We set $\lambda_{\text{col}} = 1$ and tune λ_{nod} so that $|V_1| \simeq 2|V_2|$ in the obtained graph. We then use the single layer model and tune parameters of the flow (λ_{flo}, τ_D) and segmentation (λ_{seg}) data terms and spatial smoothness term (λ_{st1}, λ_{st2}, τ_{st}). While checking segmentation quality, we tune λ_{seg} at around λ_{col} and λ_{st2} at around 50 (the default setting in GrabCut [8]). The remaining flow-related parameters are tuned by checking flow quality. We finally use the hierarchical model and tune the parameters (λ_{pc1}, λ_{pc2}, τ_{pc}) of the multi-layer regularization.

Table A1. Benchmark results (**without flipped images**). FAcc is flow accuracy rate for an error threshold of 5 pixels in a normalized scale. SAcc is segmentation accuracy by intersection-over-union ratios. SAcc scores (⋆) of optic flow mothods are computed by post-processing using left right consistency check.

Optic flow / cosegment. Methods	FG3DCar		JODS		PASCAL	
	FAcc	SAcc	FAcc	SAcc	FAcc	SAcc
Ours	**0.837**	**0.756**	**0.665**	0.521	**0.754**	**0.659**
Our single layer ([10])	0.734	**0.757**	0.525	0.515	0.649	0.626
SIFT Flow [7]	0.640	(0.420)	0.582	(0.257)	0.695	(0.468)
DSP [5]	0.492	(0.285)	0.517	(0.227)	0.590	(0.364)
DFF [12]	0.498	(0.328)	0.330	(0.213)	0.333	(0.251)
Faktor and Irani [3]	–	0.688	–	**0.549**	–	0.486
Joulin *et al.* [4]	–	0.461	–	0.332	–	0.411

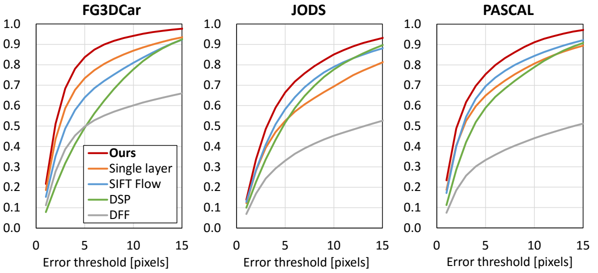


Figure A5. Average flow accuracies evaluated by endpoint errors with varying thresholds (**without flipped images**). Ours always shows best scores. Similarly to Figure 5, our method shows always best scores.

F. Dataset

In this section, we show more examples and report some statistics of our dataset. Our dataset comprises of 400 image pairs divided into three groups – **FG3DCar** contains 195 image pairs of vehicles. **JODS** contains 81 image pairs of airplanes, horses, and cars. **PASCAL** contains 124 image pairs of bicycles, motorbikes, buses, cars, trains. The charts in Figure A1 show the number of image pairs in each subcategory of JODS and PASCAL. Figures A2–A4 show examples of image pairs from FG3DCar, JODS and PASCAL, respectively. Notice that JODS and PASCAL contain some horizontally flipped image pairs, *i.e.*, one image requires a mirror reflection prior to alignment. The numbers of such flipped image pairs included in each group are follows. FG3DCar: 2 pairs (1 %). JODS: 9 pairs (11 %). PASCAL: 48 pairs (39 %).

G. Benchmark Scores without Flipped Images

As mentioned in the previous section, our dataset contains flipped image pairs. Since our method and others do not explicitly handle such image pairs, they fail to find correspondence for them. Therefore, we also evaluate accuracy scores similar to Table 1 and Figure 5 but excluding flipped image pairs from the evaluation. We show the average scores for the three groups in Table A1, and the plots of average flow accuracies with varying thresholds in Figure A5. We observe similar trends between scores with and without flipped image pairs.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 23(11):1222–1239, 2001.

[2] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *Int'l Journal of Computer Vision*, 96(1):1–27, 2012.

[3] A. Faktor and M. Irani. Co-segmentation by composition. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 1297–1304, 2013.

[4] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[5] J. Kim, C. Liu, F. Sha, and K. Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2307–2314, 2013.

[6] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 26(2):147–159, 2004.

[7] C. Liu, J. Yuen, and A. Torralba. SIFT Flow: Dense Correspondence across Scenes and Its Applications. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 33(5):978–994, 2011.

[8] C. Rother, V. Kolmogorov, and A. Blake. “grabcut”: Interactive foreground extraction using iterated graph cuts. *ACM Trans. on Graph.*, 23(3):309–314, 2004.

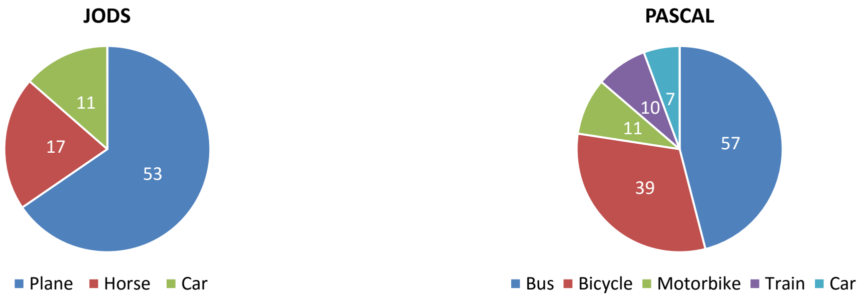


Figure A1. Subcategories of JODS and PASCAL.

Figure A2. Examples of FG3DCar

Figure A3. Examples of JODS

Figure A4. Examples of PASCAL

[9] T. Taniai, Y. Matsushita, and T. Naemura. Graph cut based continuous stereo matching using locally shared labels. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1613–1620, 2014.

[10] T. Taniai, Y. Matsushita, Y. Sato, and T. Naemura. Continuous Stereo Matching Using Local Expansion Moves. arXiv:1603.08328, http://arxiv.org/abs/1603.08328, 2016.

[11] P. J. Toivanen. New Geodesic Distance Transforms for Gray-scale Images. *Pattern Recogn. Lett.*, 17(5):437–450, 1996.

[12] H. Yang, W. Lin, and J. Lu. DAISY filter flow: A generalized discrete approach to dense correspondences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3406–3413, 2014.