Title
Semantic-guided Visual Analysis and Synthesis with Spatio-temporal Models

Author
Tsai, Yi-Hsuan

Publication Date
2017

UNIVERSITY OF CALIFORNIA, MERCED

Semantic-guided Visual Analysis and Synthesis with Spatio-temporal Models

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering and Computer Science

by

Yi-Hsuan Tsai

Committee in charge:

Professor Ming-Hsuan Yang, Chair
Professor Shawn Newsam
Doctor Onur Hamsici

2017

The dissertation of Yi-Hsuan Tsai is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Professor Shawn Newsam

Doctor Onur Hamsici

Professor Ming-Hsuan Yang Chair

University of California, Merced

2017

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

VITA

2009	B. S. in Electronics Engineering, National Chiao-Tung University, Hsinchu, Taiwan
2012	M. S. in Electrical Engineering and Computer Science, University of Michigan, Ann Arbor
2017	Ph. D. in Electrical Engineering and Computer Science, University of California, Merced

PUBLICATIONS

Yi-Hsuan Tsai, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, Xin Lu, Ming-Hsuan Yang, *Deep Image Harmonization*, In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

Guangyu Zhong*, Yi-Hsuan Tsai* (* indicates equal contribution), Ming-Hsuan Yang, *Weakly-supervised Video Scene Co-parsing*, In Asian Conference on Computer Vision (ACCV), 2016.

Yi-Hsuan Tsai*, Guangyu Zhong* (* indicates equal contribution), Ming-Hsuan Yang, *Semantic Co-segmentation in Videos*, In European Conference on Computer Vision (ECCV), 2016.

Yi-Hsuan Tsai, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, Ming-Hsuan Yang, *Sky is Not the Limit: Semantic-Aware Sky Replacement*, ACM Trans. Graph. (SIGGRAPH), 2016.

Yi-Hsuan Tsai, Ming-Hsuan Yang, Michael J. Black, *Video Segmentation via Object Flow*, In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

Yi-Hsuan Tsai, Onur Hamsici, Ming-Hsuan Yang, *Adaptive Region Pooling for Object Detection*, In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

Jimei Yang, Yi-Hsuan Tsai, Ming-Hsuan Yang, *Exemplar Cut*, In IEEE International Conference on Computer Vision (ICCV), 2013.

Yi-Hsuan Tsai, Jimei Yang, Ming-Hsuan Yang, *Decomposed Learning for Joint Object Segmentation and Categorization*, In British Machine Vision Conference (BMVC), 2013.

ABSTRACT OF THE DISSERTATION

Semantic-guided Visual Analysis and Synthesis with Spatio-temporal Models

by

Yi-Hsuan Tsai
Doctor of Philosophy in Electrical Engineering and Computer Science

University of California Merced, 2017

Professor Ming-Hsuan Yang, Chair

Visual analysis is concerned with problems to identify object status or scene layout in images or videos. There are numerous concepts that are of great interest for visual analysis and understanding in the computer vision and machine learning communities. For instance, researchers have been working on developing algorithms to recognize, detect and segment objects/scenes in images. To understand such content, numerous challenges make these problems significantly challenging in the real world scenario, since objects or scenes usually appear in different conditions such as viewpoints, scales, and background noise, and even may deform with different shapes, parts or poses.

In addition to images, video understanding has drawn much attention in various research areas due to the ease of obtaining video data and the importance of video applications, such as virtual reality, autonomous driving and video surveillance. Different from images, videos contain richer information in the temporal domain, thereby it also produces difficulties and requires larger computational powers to fully exploit video content. In this thesis, we propose to construct optimization frameworks for video object tracking and segmentation tasks. First, we utilize a spatial-temporal model to jointly optimize video object segmentation and optical flow estimation, and show that both results can be improved in the proposed framework. Second, we introduce a co-segmentation algorithm to further

understand object semantics by considering relations between objects among a collection of videos. As a result, our proposed algorithms achieve state-of-the-art performance in video object segmentation.

With such visual understanding in images and videos, the following question would be how to use them in real world applications. In this thesis, we focus on the visual synthesis problem, where it is a task for people to create or edit contents in the original data. For instance, numerous image editing problems have been studied widely, such as inpainting, harmonization and colorization. For these tasks, as the human can easily discover unrealistic artifacts after the original data is edited, one important challenge is to create realistic contents. To tackle this challenge, we propose to extract semantics by utilizing visual analysis as the guidance to improve the realism of synthesized outputs. With such guidance, we show that our visual synthesis systems produce visually pleasing and realistic results on sky replacement and object/scene composition tasks.

Chapter 1

Introduction

1.1 Overview

Visual analysis is one of the fundamental problems in computer vision. Given an image, a human can easily identify and recognize object status or scene layout in a few seconds. To execute this ability in computers, researchers have been actively working on this area due to its theoretical and practical interests. My main research also lies in visual understanding, in which I have developed algorithms to solve the following tasks in images: 1) Which category does the object or scene belong to (object and scene categorization [100])? 2) Whether and where do the objects appear (object detection [97])? 3) Which pixels belong to the objects or scenes (object segmentation and scene parsing [118])? With these abilities to understand visual content, it is a critical step toward other related problems, such as 3D understanding, sequence analysis and image editing.

Image analysis vs. video analysis. We have discussed the cases for understanding image content. However, in the real world scenario, objects and scenes usually appear in dynamic conditions and have intensive interactions with the surrounding environment. For instance, a person riding a motorcycle may move fast in a dynamic environment, in which it may disappear several times in a video due to occlusions, causing severe challenges to keep

tracking on this object. In such cases, more complex models are required to consider the richer information in the temporal domain and handling the dynamic environment.

Another fact is that the amount of video data is significantly larger than the image data, which may contain lots of noise. Therefore, to fully exploit the information contained in videos, developing algorithms to extract useful knowledge or discover relations between videos becomes an important step before further utilizing the video data for other usages. In this thesis, we will introduce solutions to handle these two challenges in video analysis as described above.

Visual analysis vs. visual synthesis. Recently, visual synthesis has drawn much attention due to its intelligent ability to create interesting content. Numerous tasks such as inpainting [67], image colorization [121] and harmonization [127] achieve impressive results in real data. However, maintaining the realism of edited results still remains a challenging problem as humans are really sensitive to generated artifacts.

Another challenge is that usually there is no single solution when creating content in the original data. For instance, we may insert a dog in an image and adjust the appearance of this dog in order to fit background colors. In such cases, there could be multiple ways to produce output results that may look realistic. Thus, how do we know which one looks more realistic so that the users may prefer more? Before addressing this question, we must remember that in visual analysis, algorithms are designed to understand image content, while visual synthesis aims to generate or edit image content. Therefore, is there a link between these two tasks? If they are related to each other, how can we utilize such relations and help both tasks? In this thesis, we will provide a few examples to illustrate this concept and demonstrate the usefulness of using this knowledge.

1.2 Solutions

Visual Analysis. As discussed in Section 1.1, we have introduced the problem of visual analysis and its challenges. Recently, numerous deep learning frameworks [44, 81, 33]

have shown significant improvement in object recognition tasks. However, these methods only explore learning deep networks based on the image data, while the video data may provide more knowledge to better understand visual content. In this thesis, we still apply image-based deep networks, but in the meanwhile, we focus more on how to exploit the video data and achieve improved performance in video object segmentation. Specifically, we would like to answer two questions as below.

- How to construct models that consider information in the temporal domain?
- What are the relations that can be utilized between different videos?

To address the first question, video object tracking [25, 56, 107], video segmentation [55, 66, 37] and motion estimation [2, 9, 83] have been extensively studied. However, the joint problem of these tasks has been rarely explored. In this thesis, we propose to formulate a joint optimization for video object segmentation and optical flow estimation [101], and show that both tasks can help each other efficiently and effectively.

Once we are able to exploit the information contained in a single video, the ensuing question is how to deal with multiple videos [123]. Since different videos may share information that can benefit each other, we develop a video co-segmentation algorithm to consider semantic relations between videos [102]. With such semantic guidance, we demonstrate that our segmentation results are better than the one that only considers a single video.

Visual Synthesis. We have described the problem of visual synthesis in Section 1.1. Based on the understanding of visual content, we aim to utilize visual analysis and help visual synthesis tasks, where these problems all share a common knowledge: *semantics*. In this thesis, we would like to answer two questions.

- How can semantics help visual synthesis tasks?
- Could we learn a joint model for visual analysis and synthesis?

We will briefly introduce two of our works in Section 1.3, in which we show that using semantics can help visual synthesis and produce visually pleasing results [99, 98]. In addition, a deep neural network is proposed to jointly learn semantic segmentation and image harmonization [98], where semantics provide rich cues to improve the realism of edited outputs.

1.3 Organization

In Chapter 3, we propose an algorithm for video object segmentation. To tackle such problems, optical flow can be used to propagate an object segmentation over time but, unfortunately, flow is often inaccurate, particularly around object boundaries. One of our observations is that, considering the segmentation of the object is known, optical flow within the same object should be smooth but flow across the boundary needs not be smooth. Hence, we formulate a principled, multiscale, spatio-temporal objective function [101] for joint estimation of video object segmentation and optical flow. We call the process *object flow* and demonstrate the effectiveness of jointly optimizing optical flow and video segmentation using an iterative scheme. Experiments on several benchmark datasets show that the proposed algorithm performs favorably against the other state-of-the-art methods.

In Chapter 4, we further consider to segment objects and understand their semantics in videos. Since each video only contains limited information, we propose to utilize a collection of videos that link to each other, which we refer to as *semantic co-segmentation*. Within the proposed co-segmentation framework, we aim to find the common representation for each semantic category and exploit relations between objects. To achieve this, we first generate multiple object-like tracklets across the video, where each tracklet maintains temporally connected segments and is associated with a predicted category. To exploit rich information from other videos, we co-select tracklets that belong to true objects by solving a submodular function, where this function accounts for object appearance, shape and motion, and hence facilitates the co-segmentation process.

In Chapter 5, we study the first problem of visual synthesis. We propose an automatic background replacement algorithm that can generate realistic, artifact-free images with diverse styles of skies. The key idea of our algorithm is to utilize visual semantics to guide the entire process including sky segmentation, search and replacement. First, we train a deep convolutional neural network for semantic scene parsing, which is used as visual prior to segment sky regions in a coarse-to-fine manner. Second, in order to find proper skies for replacement, we propose a data-driven sky search scheme based on semantic layout of the input image. Finally, to re-compose the stylized sky with the original foreground naturally, an appearance transfer method is developed to match statistics locally and semantically. We show that the proposed algorithm can automatically generate a set of visually pleasing results.

In Chapter 6, we develop an image harmonization method that adopts semantic scene parsing. In such tasks, in order to generate realistic results for composite images, the appearances of foreground and background need to be adjusted to make them compatible. Previous approaches focus on learning statistical relationships between hand-crafted appearance features of the foreground and background, which is unreliable especially when the contents in the two layers are vastly different. In contrast, we propose an end-to-end deep convolutional neural network that captures both the context and semantic information of the composite images during harmonization. Experiments on the synthesized dataset and real composite images show that the proposed network outperforms previous state-of-the-art methods.

Finally, we conclude this thesis and discuss future work in Chapter 7.

Chapter 2

Literature Review

In this Chapter, we review the literature related to the research work in terms of visual analysis and synthesis.

2.1 Visual Analysis

For visual analysis, we first discuss problems related to video understanding, including video object tracking, video segmentation and motion estimation. To further consider the case that utilizes multiple videos, we introduce the work for object co-segmentation and co-localization.

Video Segmentation via Graph-based Models. One approach to segment objects in videos is to propagate foreground labels [1, 37, 96, 104, 110] between frames based on graphical models. Graphical models for video segmentation typically use unary terms that are determined by the foreground appearance, motions or locations, and pairwise terms that encode spatial and temporal smoothnesses. These methods typically use optical flow to maintain temporal links, but they are likely to fail when the flow is inaccurate. In addition, graphical models can be used for refining segments [55, 66]. Lee *et al.* [55] use ranked object proposals and select key segments for shape matching. However, it is

computationally expensive to generate proposals and they are likely to contain foreground and background pixels [56].

Motion Estimation with Layered Models. Video segmentation and motion estimation are closely related. Layered models [11, 38, 41, 94, 105, 125] jointly optimize for segmentation and optical flow. These models can be extended to the temporal domain with more than two frames [84]. Early methods focus only on motion information but more recent formulations combine image and motion information in segmenting the scene into layers [85, 112]. Most layered methods use complicated and computationally expensive inference, thereby limiting their applications.

Object Segmentation in Weakly-Supervised Videos. Weakly-supervised methods have attracted attention due to their effectiveness for facilitating the segmentation process with known video-level object categories. Several learning-based approaches are proposed to collect semantic samples for training segment classifiers [30, 90] or performing label transfer [59], and then identify the target object in videos. However, these methods rely on training instances and may generate inaccurate segmentation results. Zhang et al. [122] propose to segment semantic objects via detection without the need of training. In this method, object detections and proposals are integrated within an optimization framework to refine the final tracklets for segmentation. However, they require additional computational costs or information such as object proposals and video-level annotations.

Video Object Co-segmentation. Recently, co-segmentation methods are developed to segment common objects in images [42, 77, 103] and videos [16, 23, 28, 78, 120]. Most co-segmentation schemes assume that all the input videos contain at least one common target object [16, 23, 28, 78], which is rarely true in real world scenarios. With a less strict assumption in [120], objects with unknown number of categories can be segmented from a collection of videos by tracking and matching object proposals. However, another assumption underlying the above-mentioned methods is that usually common objects have

almost identical appearances. Therefore, these methods are not able to segment objects with large variations in appearances without knowing some priors, e.g., number of object instances and number of object categories.

Object Discovery and Co-localization. Object discovery and co-localization methods are developed in a way similar to object co-segmentation, and these methods assume that input images or videos contain object instances from the same category. Recent image-based approaches [14, 17, 76, 89] are proposed to overcome the problem of large amounts of intra-class variations and inter-class diversity. Several video-based methods are extended to account for temporal information. In [106], superpixel-level labels are propagated across frames via a boosting algorithm. However, this approach requires supervision from a few frame-level annotations. Kwak et al. [45] propose a video object discovery method by matching correspondences across videos and tracking object regions across frames. For the above-mentioned schemes, mostly they have an assumption on objects appearing in videos, which may not be true in the real world scenario.

2.2 Visual Synthesis

The focus of synthesis problems in this thesis is to create or edit realistic composites. This task entails a combination of high-quality semantic matching and appearance transfer. In this section, we discuss existing methods closely related to these modules within the context of rendering images with composites. In addition, recent learning-based frameworks for image editing within this scope are discussed.

Appearance Matching for Compositing. Creating realistic composites requires a good match between both the content and the appearance of the images being merged. When the images being merged are already specified, existing techniques use color and tone matching to ensure that the generated results have consistent appearance. Color and tone matching can be carried out by transferring global statistics [73, 70] or using correspon-

dences between local regions [88, 29]. Gradient domain schemes have been developed to address inconsistencies on boundaries [68, 92]. In addition, the problems with textural inconsistency between images can be alleviated by matching multi-scale statistics [87] or patch-level adjustments [21].

While these methods directly match the appearance of images being composited, another class of techniques learns the transformations from external datasets. Xue et al. [116] learn color and tone transformations such that the appearance of the composited foreground regions is adjusted properly with respect to the background. Color and tone transformations have also been exploited to hallucinate changes in time of day [80] and other high-level transient attributes [46]. On the other hand, data-driven techniques have been proposed to restore degraded photographs [20] and improve the realism of computer generated images [40]. Lalonde and Efros [48] learn color statistics from a set of natural images to predict the realism of photos, and use them to adjust foregrounds to improve the chromatic compatibility. Recently, Zhu et al. [127] extend this work with a convolutional neural network (CNN) to learn a model for predicting and improving the realism of composites. Moreover, a CNN based method [117] that utilizes semantic features is developed to learn appearance adjustment, in which pairs of input and output styles are required for training the CNN.

Semantic Search for Compositing. Appearance matching methods perform well when the content of the images being considered are consistent, and numerous search techniques have been developed to determine compatible images. Hays et al. [31] present an image completion method by searching a large database for images with compatible layout measured by the GIST descriptor [95] and appearance. To composite images, Lalonde et al. [50] search for objects that are consistent with the input photograph in terms of camera orientation, lighting, resolution, etc. While the above-mentioned techniques consider generic objects, Bitouk et al. [4] propose a method that specifically replaces faces with compatible pose, appearance and lighting. Note that all these approaches rely on handcrafted features to find compatible images without learning the semantics to enhance the

discriminative ability of features.

Learning-based Image Editing. Recently, neural network based methods for image editing tasks such as image colorization [35, 51, 121], inpainting [67] and filtering [58], have drawn much attention due to their efficiency and impressive results. Similar to autoencoders [3], these methods adopt an unsupervised learning scheme that learns feature representations of the input image, where raw data is used for supervision. Without using semantics as the guidance, these image editing pipelines may suffer from missing semantic information in the finer level during reconstruction, and such semantics are important cues for understanding image contents.

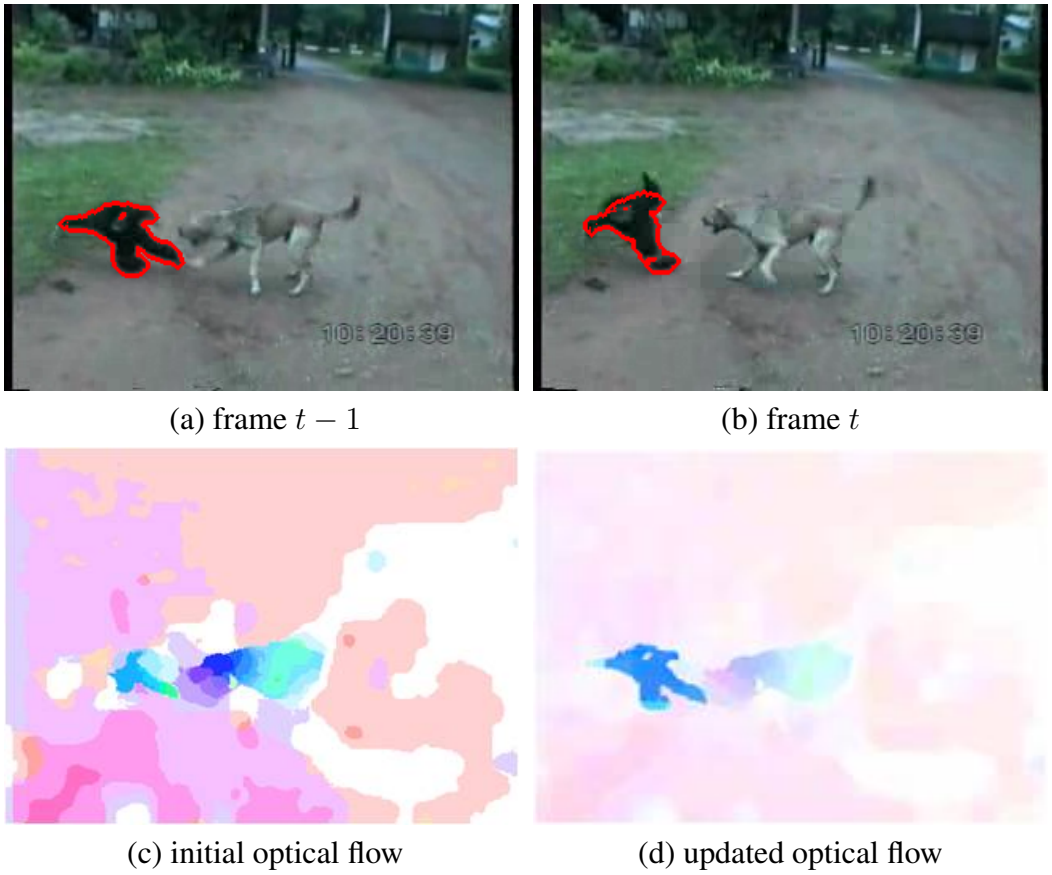
Chapter 3

Video Segmentation via Object Flow

3.1 Introduction

Our goal is to segment video sequences, classifying each pixel as corresponding to a foreground object or the background in every frame. Critical to solving this task is the integration of information over time to maintain a consistent segmentation across the entire video. Numerous methods have been proposed to enforce temporal consistency in videos by tracking pixels, superpixels or object proposals [8, 18, 25, 56, 74, 96, 107, 110]. Another line of research formulates this problem with a graphical model and propagates the foreground regions throughout an image sequence [1, 24, 37, 96, 104]. In addition, several algorithms [55, 56, 63, 66, 119] focus on object-level segmentations that favor temporal consistency. Such object-level methods, however, may not be accurate on the pixel level, generating inaccurate object boundaries.

Optical flow estimation has been extensively studied [2, 9, 83] and it is widely used for video segmentation and related problems [12, 27, 65, 113]. For instance, graph-based video segmentation methods [37, 96] use optical flow in the formulation of pairwise potentials that ensure frame-to-frame segmentation consistency. However, estimated optical flow may contain significant errors, particularly due to large displacements or occlu-



(a) frame $t-1$ (b) frame t

(c) initial optical flow (d) updated optical flow

Figure 3.1: Object flow. (a)-(b) two consecutive frames. (c) optical flow computed by [83] from frame $t-1$ to t. (d) optical flow that is updated using the segmentation marked by the red contour. The motions within the object are more consistent and the motion boundaries are more precise compared with the initial flow.

sions [9, 15, 82, 109]. To compute accurate optical flow fields, it is common to segment images or extract edges to preserve motion details around object boundaries [5, 114, 115, 128]. However, most methods do not consider both flow estimation and video segmentation together. In contrast, we estimate object segmentation and optical flow synergistically such that the combination improves both. Figure 3.1 summarizes the main ideas of this work. If the segmentation of the object is known, optical flow within the same object

should be smooth but flow across the boundary need not be smooth. Similarly if an object moves differently from the background, then the motion boundary will correspond to the object boundary. Hence, accurate optical flow facilitates detecting precise object boundaries and vice versa.

This notion, of course, is not entirely new, but few methods have tried to integrate video segmentation and flow estimation. Specifically, in this work, we address the above problems by considering object segmentation and optical flow simultaneously, and propose an efficient algorithm, which we refer as *object flow*. For the segmentation model, we construct a multi-level graphical model that consists of pixels and superpixels, where each of these play different roles for segmentation. On the superpixel level, each superpixel is likely to contain pixels from the foreground and background as the object boundary may not be clear. On the pixel level, each pixel is less informative although it can be used for more accurate estimation of motion and segmentation. With the combination of these two levels, the details around the object boundary can be better identified by exploiting both statistics contained in superpixels and details on the pixel level. Furthermore, we generate superpixels by utilizing supervoxels [27, 113] between two frames to exploit temporal information in addition to the use of optical flow. After obtaining the segmentation results, we apply the foreground and background information to re-estimate optical flow (Figure 3.1), and then iteratively use the updated optical flow to re-segment the object region.

We evaluate the proposed object flow algorithm on the SegTrack v2 [56] and Youtube-Objects [71] datasets. We work in the standard paradigm of tracking that assumes an initialization of the object segmentation in the first frame, which could come from simple user input [75] . We quantitatively compare our segmentation accuracy to other state-of-the-art results and show improvements to the estimated optical flow. With the updated optical flow using the segmentation, we show that the iterative procedure improves both segmentation and optical flow results in terms of visual quality and accuracy.

The contributions of this work are as follows. First, we propose a multi-level spatial-

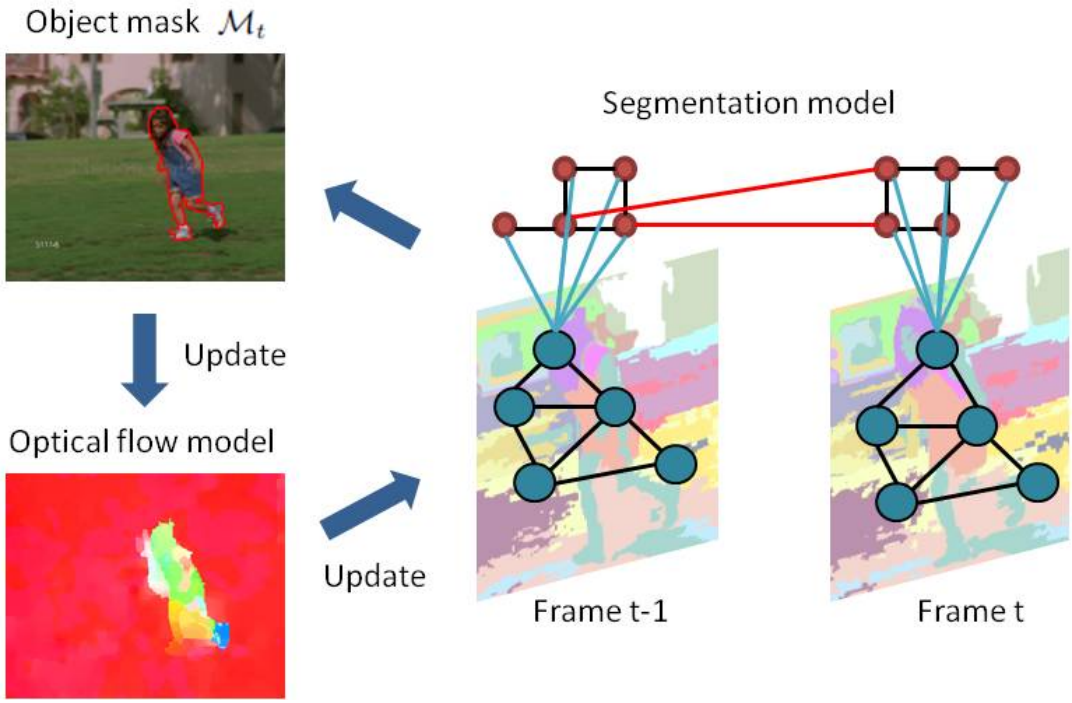


Figure 3.2: Overview of the proposed model. For segmentation, we consider a multi-level spatial-temporal model. Red circles denote pixels, which belong to the superpixel marked by the turquoise circles. The black and the red lines denote the spatial and temporal relationships, respectively. The relationships between the pixels and the superpixel are denoted by the turquoise lines. After obtaining the object mask, $\mathcal{M}_t$, we use this mask to re-estimate the optical flow, and update both models iteratively.

temporal graphical model for video object segmentation and demonstrate that it performs better than single-level models. Second, we show that the segmentation results can be used to refine the optical flow, and vice versa, in the proposed object flow algorithm. Third, we demonstrate that our joint model of segmentation and optical flow can be efficiently computed by iterative optimization.

3.2 Video Object Segmentation

In this section, we first explain how we construct the object segmentation model. Given the initialization in the first frame, we aim to propagate the foreground label throughout the entire video. Note that, in contrast to unsupervised methods [55, 56], that rely on motion and object proposals and process the entire video offline in batch mode, the proposed algorithm is able to track and segment objects for online applications. Before assigning each pixel a label, we search the possible locations for the object in each frame to reduce the background noise. A multi-level spatial-temporal graphical model is then applied to the estimated object regions. In this stage, unary and pairwise terms for superpixels are introduced by the supervoxel to better ensure temporal consistency. Figure 3.2 shows the proposed multi-level segmentation model.

Object Location. Instead of using the segmentation model on the entire image, we first estimate the object location to reduce the computational load and the effect of background noise. We design a scoring function for each pixel based on color and location:

$$S_t(x_i^t) = A_t(x_i^t) + L_t(x_i^t, \mathcal{M}_{t-1}), \tag{3.1}$$

where A_t is the color score on the pixel x_i^t computed by a Gaussian Mixture Model (GMM), and L_t is the location score measured by the Euclidean distance transform of the binary object mask $\mathcal{M}_{t-1}$ in the previous frame.

Since we do not know the exact object location or shape in the current frame, we assume that the object does not move abruptly. Therefore, we generate the rough object mask in the current frame t using the segmentation mask $\mathcal{M}_{t-1}$ in the previous frame translated by the average optical flow vector within the mask. We then use and expand this coarse mask on a local search region that is s times the size of the object mask $\mathcal{M}_{t-1}$ (s is from 2 to 3 depending on the object size in this work). This mask ensures that most of the pixels within the object are covered. After obtaining the local search region, we use the distance transform on this expanded mask to compute location scores. Similarly, we also

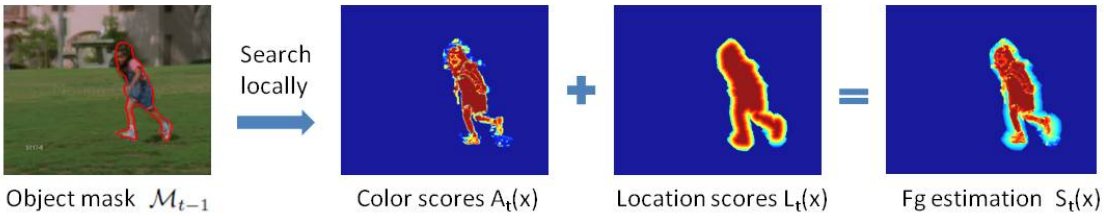


Figure 3.3: Estimated object location R_t. Given the object mask $\mathcal{M}_{t-1}$ marked as the red contour, we search for a local region in the current frame t and compute the foreground scores based on color and location. The estimated foreground region is used for label assignment.

consider color scores based on this local region. Figure 3.3 illustrates one example of the combination of two scores. A threshold is then applied to select the object location R_t for further determining label assignment.

Graphical Model. After the object region for label assignment is selected, we utilize a spatial-temporal graphical model to assign each pixel with a foreground or background label. We define an energy function in a Conditional Random Field (CRF) form for the pixel $x_i^t \in X$ with label $\in \{0, 1\}$:

$$E_{pix}(X) = \bar{U}_t(X, \mathcal{M}_{t-1}) + \gamma_1^s \sum_{(i,j,t)\in\mathcal{E}_t} \bar{V}_t(x_i^t, x_j^t) + \gamma_1^t \sum_{(i,j,t)\in\mathcal{E}_t} \bar{W}_t(x_i^{t-1}, x_j^t), \tag{3.2}$$

where $\bar{U}_t$ is the unary potential for the cost to be foreground or background, and $\bar{V}_t$ and $\bar{W}_t$ are pairwise potentials for spatial and temporal smoothnesses with weights γ_1^s and γ_1^t, respectively. Both of the pairwise terms are defined as in [66]. Note that we only consider $\mathcal{E}_t$ within the region R_t generated in the object location estimation step.

For the unary term in (3.2), we consider appearance and location energies defined by $\bar{U}_t(X, \mathcal{M}_{t-1})=$

$$\alpha_1 \sum_{(i,t)\in R_t} \bar{\Phi}_a(x_i^t) + \beta_1 \sum_{(i,t)\in R_t} \bar{\Phi}_l(x_i^t, \mathcal{M}_{t-1}), \tag{3.3}$$

pixel: 75.8 superpixel: 82.9 multi-level: 85.5

Figure 3.4: Segmentation results at different levels with overlap ratios with respect to the ground truth mask. On the pixel or superpixel level, both results are not complete. The results on the pixel level miss part of the leg, while the results on the superpixel level include part of the bike. The multi-level segmentation approach exploits results from both levels for higher accuracy.

where $\bar{\Phi}_a$ is the appearance term, and $\bar{\Phi}_l$ is the location term defined similar to the one in (3.1). The difference is that for the nodes in the previous frame, we can simply compute the distance transform of the object mask $\mathcal{M}_{t-1}$. For the appearance term, we construct the color GMM in the first frame, and an online SVM model with CNN features [62] is updated every frame. The weight α_1 consists of α_1^{col} and α_1^{cnn} for the color and CNN features, respectively. By minimizing (3.2), we obtain labels within R_t and thus the object mask $\mathcal{M}_t$, and then continue to propagate to the next frame.

Multi-level Model. Although the model based on pixels can achieve fine-grained segmentation, pixels are usually sensitive to noise when optical flow is not accurately estimated. On the other hand, an alternative way is to use larger segments such as superpixels that contain more information by considering every pixel in the neighborhood (i.e., spatial support). However, superpixels may not contain the entire object or may have imprecise object boundaries due to occlusion or motion blur (See Figure 3.4). Therefore, we construct a multi-level graphical model including pixels and superpixels to ensure boundary as well as temporal consistency.

In this model, the energy terms for both pixels and superpixels have unary and pairwise

potentials, and a pairwise term is used for the connection where pixels belong to a superpixel (See Figure 3.2). In addition, since optical flow may not be estimated correctly due to large displacement, we use supervoxels [27] between two frames to generate coherent superpixels and enhance temporal consistency.

The multi-level model is formulated by

$$E_{seg} = \lambda_1 E_{pix}(X) + \lambda_2 E_{sup}(Y) + \lambda_3 E_{pair}(X, Y), \tag{3.4}$$

where $E_{pix}(X)$ is the model based on pixels in (3.2); $E_{sup}(Y)$ is the energy function based on superpixels $y_m^t \in Y$; $E_{pair}(X, Y)$ is the pairwise term between pixels and superpixels; and λ_i is the weight. We define $E_{sup}(Y)$ as:

$$E_{sup}(Y) = \hat{U}_t(Y) + \gamma_2 \sum_{(m,n,t)\in\mathcal{E}_t} \hat{V}_t(y_m^t, y_n^t), \tag{3.5}$$

where $\hat{U}_t$ is the unary potential for labeling a superpixel as foreground or background, and $\hat{V}_t$ is the spatial smoothness within the region R_t. Note that it is not necessary to model the temporal smoothness in (3.5) since we design a term for the supervoxel and optical flow in the unary term (explained in detail below). The unary term $\hat{U}_t$ is defined in a way similar to (3.3):

$$\hat{U}_t(Y) = \alpha_2 \sum_{(m,t)\in R_t} \hat{\Phi}_a(y_m^t) + \beta_2 \sum_{(m,t)\in R_t} \hat{\Phi}_l(y_m^t), \tag{3.6}$$

where $\hat{\Phi}_a$ is the color term defined as the mean color likelihood over all pixels within the superpixel, and a location term $\hat{\Phi}_l$ measures the consistency between the optical flow and the supervoxels. The location term is defined as:

$$\hat{\Phi}_l(y) = flow(y) \times obj(y), \tag{3.7}$$

where $flow(y)$ is defined by the percentage of pixels in y that are successfully transferred to the next time instant. A successful transfer means that a pixel x_i^{t-1} transfers from a

superpixel y_m^{t-1} to a superpixel y_n^t via optical flow, and y_m^{t-1} and y_n^t belong to the same supervoxel. In addition, $obj(y)$ computes the percentage of pixels within the superpixel belonging to the segmentation mask $\mathcal{M}$.

The value of the first term in (3.7) is high if the supervoxel and optical flow mostly agree with each other. The second term basically measures the likelihood that a superpixel is part of the object. Note that, to compute $obj(y)$ for superpixel nodes in the current frame t, since the object location is unknown, we use the approximate object mask as described in the step for estimating the object location.

$E_{pair}(X, Y)$ models the relationship between superpixels and pixels, encouraging pixels inside the superpixel to have the same label. This pairwise term is defined by

$$E_{pair}(x_i^t, y_m^t) = \begin{cases} 1 - |(p(x_i^t) - p(y_m^t))| & \text{if } l_x \neq l_y \\ 0 & \text{else,} \end{cases}$$

where p is the foreground color probability computed by a color GMM, and l_x and l_y are the labels for the pixel and superpixel. This energy computes the penalty of assigning different labels to pixel x and superpixel y. The subtraction of probabilities indicates how similar x and y are, and the absolute value is from 0 to 1. That is, if the color of the pixel is similar to the mean color of the superpixel, it is likely to have the same label and should have a higher penalty if assigned to a different label.

Overall, to propagate foreground labels, we estimate the object location guided by the optical flow, and utilize a multi-level model to assign the label to each pixel. On the pixel level, optical flow is used to maintain temporal smoothness, whereas for the superpixel, the model measures the consistency between supervoxels and optical flow, and propagates the location information to the next frame.

3.3 Object Flow

In the previous section, we address video segmentation by utilizing a multi-level spatial-temporal graphical model with the use of optical flow and supervoxels. The ensuing ques-

tion is how to use the segmentation results to help the estimation of optical flow and vice versa? Since flow vectors within the same object are likely to be similar, we propose to estimate them on the object level. The updated optical flow can then be used again to improve object segmentation. The problem is formulated jointly as follows.

Optical Flow with Segmentation. We minimize the classical robust optical flow objective function [83],

$$\begin{aligned} E(\mathbf{u}, \mathbf{v}; R) = \sum_{i,j \in R} \{ & \rho_D(I_{t-1}(i,j) - I_t(i + u_{i,j}, j + v_{i,j})) \\ & + \lambda[\rho_S(u_{i,j} - u_{i+1,j}) + \rho_S(u_{i,j} - u_{i,j+1}) \\ & + \rho_S(v_{i,j} - v_{i+1,j}) + \rho_S(v_{i,j} - v_{i,j+1})]\}, \end{aligned} \tag{3.8}$$

where $\mathbf{u}$ and $\mathbf{v}$ are the horizontal and vertical components of the optical flow from image I_{t-1} to I_t, and ρ_D and ρ_S are robust data and spatial penalty functions.

We further consider the object mask $\mathcal{M}$ obtained from the segmentation step. One can consider this as a binary assignment of pixels to layers in a layered flow model. Here we use it to decompose the flow problem into two separate estimations,

$$\begin{aligned} & E_{flow}(\mathbf{u}_{fg}, \mathbf{v}_{fg}, \mathbf{u}_{bg}, \mathbf{v}_{bg}) = \\ & \quad E(\mathbf{u}_{fg}, \mathbf{v}_{fg}; fg) + E(\mathbf{u}_{bg}, \mathbf{v}_{bg}; bg), \end{aligned} \tag{3.9}$$

where fg and bg are local regions that are slightly larger than the foreground and background regions. This step ensures that the optical flow estimation is less affected by the background noise but still takes partial background regions into account. The final optical flow can be merged by applying the segmentation mask. That is, $\mathbf{u} = \mathcal{M} \cdot \mathbf{u}_{fg} + (1 - \mathcal{M}) \cdot \mathbf{u}_{bg}$, which are obtained from $E(\mathbf{u}_{fg}, \mathbf{v}_{fg}; fg)$ and $E(\mathbf{u}_{bg}, \mathbf{v}_{bg}; bg)$, respectively.

Joint Formulation. To formulate the joint problem for segmentation and optical flow, we combine (3.4) and (3.9) as:

$$\min_{\mathcal{M}, \mathbf{u}, \mathbf{v}} E_{total} = E_{seg} + E_{flow}. \tag{3.10}$$

Algorithm 1: Object Flow

```
Initialize: u, v by minimizing (3.8);
while not converged do
    u^p ← u, v^p ← v;
    Segmentation;
        Compute energy terms for (3.4) using u^p, v^p;
        minimize (3.4) by graph cuts and obtain M;
        M^p ← M;
    Optical Flow;
    if large displacement then
            minimize (3.9) using M^p and obtain u, v;
            u^p ← u, v^p ← v;
    end
    M ← M^p, u ← u^p, v ← v^p;
end
```

Note that in E_{seg}, we use optical flow in (3.2) and (3.5), and the estimated object location. In E_{flow}, we use the segmentation mask obtained by E_{seg} to decide the foreground and background local regions.

We optimize (3.10) by iteratively updating the two models once the segmentation or optical flow energy converges. First, we initialize and fix the optical flow to minimize E_{seg} using graph cuts [6]. We then optimize the optical flow by fixing the segmentation mask $\mathcal{M}$, and minimizing E_{flow} using the Classic+NL method [83].

Optimization Details. The main steps of the optimization procedure for (3.10) are summarized in Algorithm 1. We make a few assumptions to expedite the process. First, we find that for many frames, optical flow can be obtained accurately without the need to re-estimate. For instance, this is true when the object is stationary or moves slightly. In

addition, we observe that if the consistency between supervoxels and optical flow is low, it is a good indication that the object moves by a large displacement. Thus, we design a strategy that only re-estimates the optical flow if the value of $flow(y)$ in (3.7) is less than a threshold. This speeds up the process significantly while maintaining good accuracy.

Second, since our goal is to find a stable object mask $\mathcal{M}$, instead of using the energy E_{seg} to decide the convergence status during the update of the segmentation model, we measure the difference of object mask solutions $\mathcal{M}$. If the overlap ratio of $\mathcal{M}$ is larger than a value (e.g. 95%), it should be a stable solution. In our experiments, the entire optimization process for (3.10) converges within five iterations, and converges in two iterations for most frames from our experiments.

3.4 Experimental Results

Implementation Details. To evaluate the proposed algorithm, we first construct the foreground and background color GMMs in the RGB space from the first frame, and set K to 5 for each GMM. For learning the online SVM model, we extract hierarchical CNN features [62] combining the first 5 convolutional layers from a pre-trained VGG net [81] into 1472 dimensional vectors. We use the method [27] to generate supervoxels and convert them to superpixels in each frame. For parameters in the graphical model, we use $\alpha_1^{col} = 1, \alpha_1^{cnn} = 3, \beta_1 = 2, \gamma_1^s = 3$ and $\gamma_1^t = 0.2$ on the pixel level. On the superpixel level, parameters are set as $\alpha_2 = 1, \beta_2 = 1$ and $\gamma_2 = 2$. For (3.4), we set $\lambda_1 = 1, \lambda_2 = 15$ and $\lambda_3 = 5$. Since one superpixel contains numerous pixels, we use larger weight for λ_2 on the superpixel level as otherwise the superpixel energy is easily absorbed to have the same label as the pixels (a similar issue holds for λ_3). All these parameters are fixed in the experiments.

SegTrack v2 Dataset. We first evaluate the proposed algorithm on the SegTrack v2 dataset [56] which consists of 14 videos with 24 objects and 947 annotated frames. The dataset includes different challenging sequences with large appearance change, occlusion,

Table 3.1: Segmentation results on the SegTrack v2 dataset with the overlap ratio. Note that "-" indicates that the method fails to track an object and is excluded in measuring accuracy.

Sequence/Object	[110]	[56]	[55]	[27]	[25]	[107]	Ours
Online?	√				√	√	√
Unsupervised?		√	√	√			
Girl	83.7	**89.2**	87.7	31.9	53.6	52.4	87.9
Birdfall	**77.5**	62.5	49.0	57.4	56.0	32.5	57.4
Parachute	94.4	93.4	**96.3**	69.1	85.6	69.9	94.5
Cheetah-Deer	**63.1**	37.3	44.5	18.8	46.1	33.1	33.8
Cheetah-Cheetah	35.3	40.9	11.7	24.4	47.4	14.0	**70.4**
Monkeydog-Monkey	**82.2**	71.3	74.3	68.3	61.0	22.1	54.4
Monkeydog-Dog	21.1	18.9	4.9	18.8	18.9	10.2	**53.3**
Penguin-#1	92.7	51.5	12.6	72.0	54.5	20.8	**93.9**
Penguin-#2	**91.8**	76.5	11.3	80.7	67.0	20.8	87.1
Penguin-#3	**91.9**	75.2	11.3	75.2	7.6	10.3	89.3
Penguin-#4	**90.3**	57.8	7.7	80.6	54.3	13.0	88.6
Penguin-#5	76.3	66.7	4.2	62.7	29.6	18.9	**80.9**
Penguin-#6	**88.7**	50.2	8.5	75.5	2.1	32.3	85.6
Drifting-#1	67.3	74.8	63.7	55.2	62.6	43.5	**84.3**
Drifting-#2	**63.7**	60.6	30.1	27.2	21.8	11.6	39.0
Hummingbird-#1	58.3	54.4	46.3	13.7	11.8	28.8	**69.0**
Hummingbird-#2	50.7	72.3	**74.0**	25.2	-	45.9	72.9
BMX-Person	**88.9**	85.4	87.4	39.2	2.0	27.9	88.0
BMX-Bike	5.7	24.9	**38.6**	32.5	-	6.0	7.0
Frog	61.9	72.3	0.0	67.1	14.5	45.2	**81.4**
Worm	76.5	82.8	84.4	34.7	36.8	27.4	**89.6**
Soldier	81.1	83.8	66.6	66.5	70.7	43.0	**86.4**
Monkey	86.0	84.8	79.0	61.9	73.1	61.7	**88.6**
Bird of Paradise	93.0	94.0	92.2	86.8	5.1	44.3	**95.2**
Mean per Object	71.8	65.9	45.3	51.8	40.1	30.7	**74.1**
Mean per Sequence	72.2	71.2	57.3	50.8	41.0	37.0	**75.3**

motion blur, complex deformation and interaction between objects. For videos containing multiple objects, since instance-level annotations are provided, each object can be segmented in turn, treating each as a problem of segmenting that object from the background. We first present segmentation results and demonstrate the effectiveness of the multi-level model.

Table 3.1 shows segmentation accuracy of the proposed algorithm and the state-of-the-art methods including tracking and graph based approaches [25, 27, 55, 56, 107, 110]. Note that our model can generate labeled results on both pixel and superpixel levels, and we present the pixel level fine-grained segmentation results. We use the intersection-over-union (overlap) ratio for evaluation as it has been shown that the pixel error metric used in the SegTrack dataset is sensitive to object size [56] and is less informative.

Overall, the proposed algorithm achieves favorable results in most sequences especially for non-rigid objects (*Hummingbird, Worm, Soldier*). These sequences usually contain large deformation due to fast motions or complex cluttered backgrounds. The superpixel-based tracking methods [107, 110] do not perform well on these sequences since superpixels may not preserve object boundaries well. The Hough-based tracking method [25] only uses pixels, which may result in noisy temporal links. In the proposed spatial-temporal multi-level model, we consider both pixels and superpixels in the tracking and segmentation to maintain object boundaries as well as temporal consistency.

For the *Penguin* and *Frog* sequences, the object appearance is similar to the background with slow motions. The off-line methods [55, 56] that generate object proposals from all frames, are likely to have large segmentation errors due to wrong association or incomplete regions that contain foreground and background pixels. In contrast, the proposed algorithm performs well in these sequences with objects surrounded by other objects or background with similar appearance. In the location term (3.7), our model considers consistency between supervoxels and optical flow, and this helps maintain temporal consistency. We present qualitative segmentation results in Figure 3.5 and show more results in the supplementary material.

Table 3.2: Segmentation results using multi-level and single-level models on the SegTrack v2 dataset with the overlap ratio. Note that there are results on both pixel and superpixel levels using the multi-level model.

Methods	Pixel multi-level	Pixel only	Superpixel multi-level	Superpixel only
Mean per Object	**74.1**	69.6	65.6	50.3

To evaluate the proposed multi-level model that integrates both levels, we compare to our model using only pixels or superpixels in Table 3.2. The information contained on these two levels complement each other as the one based on the superpixel level maintains local region consistency, while the one based on the pixel level refines incomplete object contours (See Figure 3.4). Specifically, the location term in (3.7) enhances temporal information such that the model can handle cases including fast movements and background noise in sequences. In addition, the proposed multi-level model with superpixels performs better than that using only superpixels, which demonstrates that the refinement with the pixel level information is critical for obtaining good performance especially in sequences that contain unclear object boundaries. We also note that our single-level models already perform comparably to the state-of-the-art methods.

Youtube-Objects Dataset. The Youtube-Objects dataset [71] contains 1407 videos with 10 object categories, and the length of the sequences is up to 400 frames. We evaluate the proposed algorithm in a subset of 126 videos with more than 20000 frames, where the pixel-wise annotations in every 10 frames are provided by Jain and Grauman [37]. Table 3.3 shows the segmentation results of the proposed algorithm and other state-of-the-art methods[1]. For tracking-based or foreground propagation algorithms [25, 37, 64, 65, 104], ground truth annotations in the first frame are used as initializations to propagate

[1]We evaluate the code of [110] released by the authors. However, the algorithm requires different parameter settings for challenging sequences. We discuss and report results in the supplementary material.

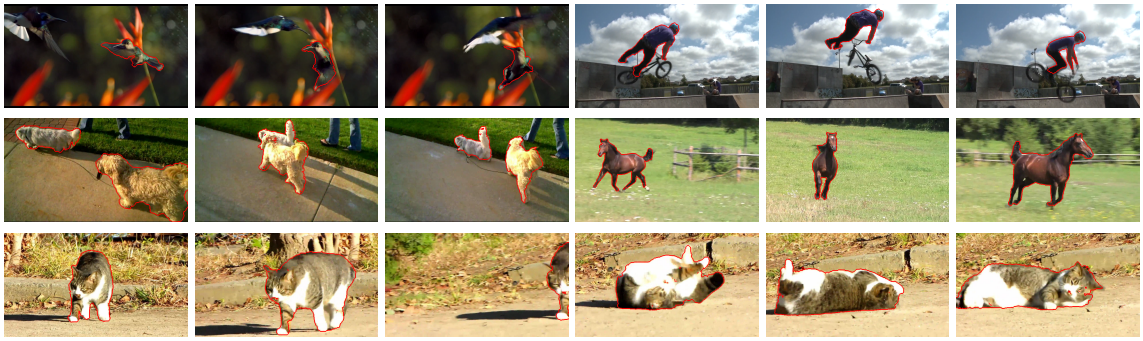


Figure 3.5: Example results for segmentation on the SegTrack v2 (first row) and Youtube-Objects (second and third rows) datasets. The output on the pixel level of our multi-level model is indicated as the red contour. The results show that our method is able to track and segment (multiple) objects under challenges such as occlusions, fast movements, deformed shapes and cluttered backgrounds. Best viewed in color with enlarged images.

Table 3.3: Segmentation results on the Youtube-Objects dataset with the overlap ratio.

Category	[64]	[37]	[104]	[25]	[66]	[65]	Ours
aeroplane	89.0	86.3	79.9	73.6	70.9	13.7	**89.9**
bird	81.6	81.0	78.4	56.1	70.6	12.2	**84.2**
boat	**74.2**	68.6	60.1	57.8	42.5	10.8	74.0
car	70.9	69.4	64.4	33.9	65.2	23.7	**80.9**
cat	67.7	58.9	50.4	30.5	52.1	18.6	**68.3**
cow	79.1	68.6	65.7	41.8	44.5	16.3	**79.8**
dog	70.3	61.8	54.2	36.8	65.3	18.0	**76.6**
horse	67.8	54.0	50.8	44.3	53.5	11.5	**72.6**
motorbike	61.5	60.9	58.3	48.9	44.2	10.6	**73.7**
train	**78.2**	66.3	62.4	39.2	29.6	19.6	76.3
Mean	74.0	67.6	62.5	46.3	53.8	15.5	**77.6**

segmentation masks. For multiple objects in videos, the proposed algorithm is able to propagate multiple object segmentations at the same time. Note that there are no instance-

level annotations provided.

Overall, the proposed algorithm performs well in terms of overlap ratio, especially in 8 out of 10 categories. Compared to optical flow based methods [64, 104], the proposed algorithm performs well on fast moving objects such as *car* and *motorbike* as the optical flow errors are reduced. Although the recent method [37] utilizes long-term supervoxels to enhance the temporal connection, the segmentation results contain noisy object boundaries as only superpixels are used. In contrast, the proposed algorithm considers visual information at multiple levels and delineates boundaries well especially on non-rigid objects (*dog, horse, cow*). We show qualitative results in Figure 3.5.

Optical Flow. We demonstrate the effectiveness of iteratively optimizing two models for updated optical flow and segmentation results (See Algorithm 1) on the SegTrack v2 dataset. Here, we only consider sequences with large displacements in which the flow is re-estimated. First, we measure the quality of updated optical flow. As the optical flow ground truth is not available, we warp the object segmentation ground truth from frame t to frame $t-1$ using optical flow with bicubic interpolation. We then compute the overlap ratio between the warped and ground truth masks. Since we focus on optical flow of the object, this metric measures consistency for flow directions and whether they connect to the same objects between two frames.

Table 3.4 shows results compared to two other optical flow methods [9, 83] and the layered model [85]. The updated results improve the optical flow estimation [83] consistently in all the sequences, especially for fast moving objects (*Girl, Drifting, BMX*). This validates our approach since the method [83] is used to compute the initial flow. Figure 3.6 and 3.7 illustrate the optical flow results. Compared to the other three methods, the updated optical flow exhibits clearer object boundaries. It shows the importance of computing optical flow on local object regions. In contrast, the results from [9] are usually oversmoothed around the object boundaries, and the layered model [85] generates incomplete flows inside objects.

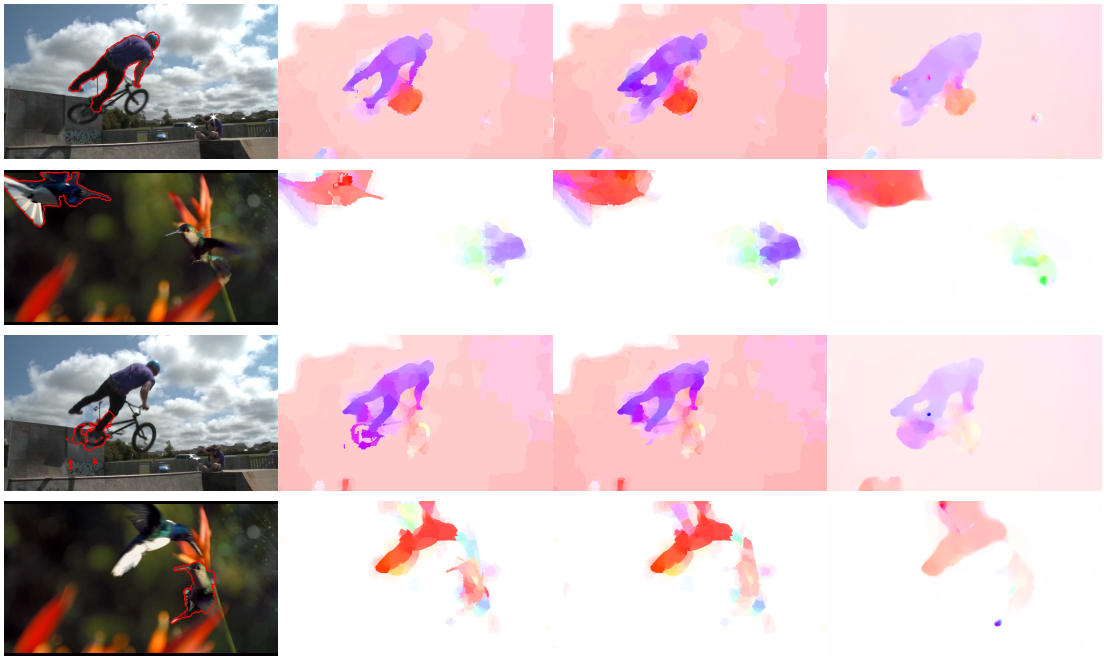
In addition, we use the normalized interpolation error (NE) as described in [2] for

Table 3.4: Intersection-over-union ratio for warped images by interpolation and updated optical flow on the SegTrack v2 dataset. The last row shows the average of normalized interpolation error. The performance is evaluated on frames with sigificant motion.

Sequence/Object	Ours	Sun [83]	Brox [9]	Sun [85]
Girl	**64.6**	56.1	63.2	**64.6**
Parachute	**86.8**	84.9	83.2	83.3
MonkeyDog-Monkey	**70.8**	70.0	67.4	69.0
MonkeyDog-Dog	69.0	69.0	68.9	**75.0**
Drifting-#1	89.5	86.6	**91.3**	82.5
Drifting-#2	87.3	82.5	**87.7**	79.8
Hummingbird-#1	**55.2**	52.2	52.6	51.3
Hummingbird-#2	**71.1**	70.6	68.7	65.5
Worm	73.3	71.1	69.8	**91.1**
Monkey	77.4	76.3	**80.9**	70.5
Soldier	**84.5**	83.5	80.8	82.7
Bird of Paradise	**94.4**	87.9	88.3	89.7
BMX-Person	**80.0**	77.4	75.0	72.2
BMX-Bike	**38.4**	33.9	37.5	38.3
Mean	**74.5**	71.6	72.5	72.5
Average NE	0.36	0.38	**0.32**	0.37

evaluation. Similarly, the ground truth of the colored object is warped by the optical flow from frame t to $t-1$. The average NE of the updated optical flow is better than [83], but slightly worse than [9]. This can be attributed to the fact that the oversmoothed optical flow in [9] usually generates more complete warped images after interpolation; this is favored by the NE metric.

Second, by using the updated optical flow, we re-estimate object segmentations and measure overlap ratios. and the average overlap ratio is increased from 72.9% to 75.1% in sequences that rely on the optical flow. The improvement varies in different sequences

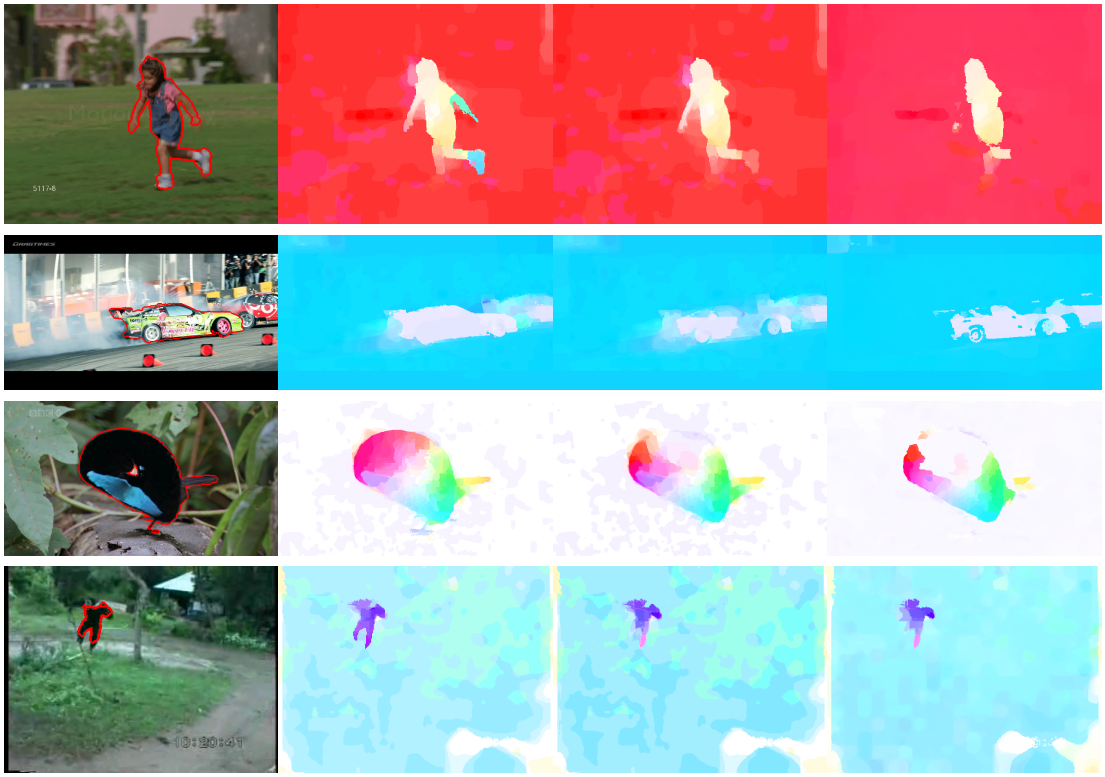


Segmentation Ours Sun [83] Brox [9]

Figure 3.6: Results for updated optical flow on the SegTrack v2 dataset. We present our updated optical flow compared to the initial flow [83] and Brox [9]. Our results contain object boundaries guided by the segmented object marked with the red contour. Note that in the same sequence with multiple objects, the updated optical flow varies depending on the segmentation. Best viewed in color with enlarged images.

since the segmentation model also takes other cues such as appearance and location into account. For instance, the improvement of overlap ratio is limited in the *Bird of Paradise* and *Parachute* sequences since the objects move steadily. On the other hand, for objects with noisy cluttered backgrounds (*Drifting-#2*) or with similar appearance to the background regions (*Worm*), the overlap ratios are improved by 2.4% and 2.9% respectively.

Runtime Performance. Our MATLAB implementation of object flow takes 3 to 20 seconds per frame on the SegTrack v2 dataset depending on the object size, and on average



Segmentation Ours Sun [83] Sun [85]

Figure 3.7: Results for updated optical flow on the SegTrack v2 dataset. We present our updated optical flow compared to the initial flow [83] and Sun [85]. Our results contain object boundaries guided by the segmented object marked with the red contour. Note that in the same sequence with multiple objects, the updated optical flow varies depending on the segmentation. Best viewed in color with enlarged images.

it takes 12.2 seconds per frame. In contrast, the state-of-the-art method [110] takes 59.6 seconds per frame on average. Note that all the timings are measured on the same computer with 3.60 GHz Intel i7 CPU and 32 GB memory, and exclude the time to compute optical flow as each method uses different algorithms. We use the MATLAB implementation of [83] to generate optical flow fields (around 30 seconds per frame) and it could be replaced by faster algorithms [86, 112].

3.5 Summary

In this work, we present a novel algorithm for joint optimization of segmentation and optical flow in videos, and show that the problem can be efficiently solved. For segmentation, a multi-level model containing pixels and superpixels is utilized to track objects. We show that both levels complement each other and maintain object boundaries and temporal consistency throughout the video. Using the segmentation, we modify the optical flow estimation to be performed within local foreground and background regions, resulting in more accurate optical flow, particularly around object boundaries. We show that our method performs favorably against state-of-the-art methods on two datasets, and both the segmentation and optical flow results are improved by iteratively updating both models.

Chapter 4

Semantic Co-segmentation in Videos

4.1 Introduction

Objects may appear at any location in various shapes and appearances with different visual semantics across videos. Given a set of videos, localizing and segmenting all the objects is a challenging task, especially when the visual categories are unknown. In this work, we propose an algorithm to segment objects and understand visual semantics from a video collection, which we refer to as *semantic co-segmentation*. Within the proposed co-segmentation framework, we aim to find the common representation for each semantic category and exploit relations between objects. For instance, dogs from different videos may share more commonalities and have stronger relations between each other than objects with other semantics (see Figure 4.1).

Numerous algorithms have been proposed for video object co-segmentation [16, 23, 78, 120]. However, most existing methods [16, 23, 78] assume that at least one common object appears all the time in two or more videos, which limits the applicability in real world scenarios. In this work, we propose an algorithm to segment semantic objects from a collection of videos containing various categories despite large variations in appearances, shapes, poses and sizes.

We exploit semantic information to facilitate co-segmentation to associate objects of the same category from different videos. Visual semantics has been used as prior information for object segmentation in weakly labeled videos [90, 106, 122]. In semantic video object segmentation, an object detector or a segmentation algorithm is first applied to localize objects according to the video label. However, for videos without any semantic label, an object detector may find noisy segments that do not belong to any semantic object (i.e., due to the trade-off between recall and precision). In this work, we propose an algorithm to associate semantic representations between objects in different videos and help the object co-segmentation process, where non-object detections can be removed.

Toward this end, we first extract semantic objects in each video. Compared with methods that use region proposals [120, 122] to localize objects, we develop a proposal-free tracking-based approach that generates multiple tracklets of regions (segments) across the video. Each tracklet maintains temporal connections and contains a predicted category that is initialized by an image-based semantic segmentation algorithm. After collecting tracklets from all videos, we link the relations between tracklets for each object category by formulating a submodular optimization problem, which maximizes the similarities between object regions (segments). With this formulation, prominent objects in each video can be discovered and segmented based on similarities of regions.

We first conduct experiments on the Youtube-Objects dataset [71] in a weakly-supervised manner. Then we evaluate the proposed method in a more generalized setting without knowing any semantic information as a prior. Both results show that our algorithm performs favorably against the state-of-the-art methods. In addition, we compare our method to the other video object co-segmentation approaches on the MOViCS [16] and Safari [120] datasets. Experimental results on three datasets show that the proposed algorithm performs favorably in terms of visual quality and accuracy.

The contributions of this work are summarized as follows. First, we propose a semantic co-segmentation method that considers relations between objects from a collection of videos, where object categories can be unknown. Second, a proposal-free tracking-

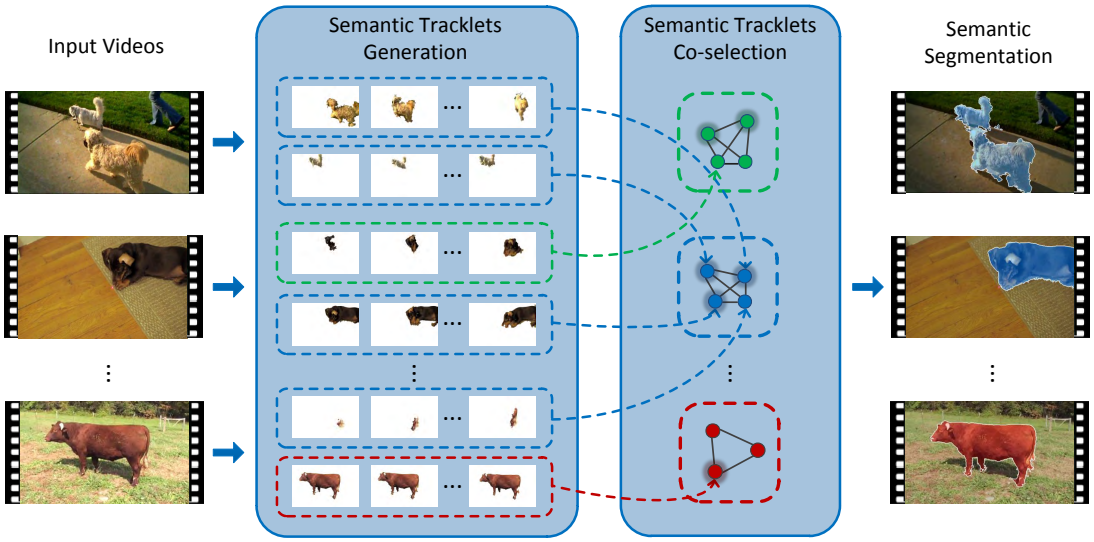


Figure 4.1: Overview of the proposed algorithm. Given a collection of videos without providing category labels, we aim to segment semantic objects. First, a set of tracklets is generated for each video, and each tracklet is associated with a predicted category illustrated in different colors (e.g., blue represents the dog and red represents the cow). Then a graph that connects tracklets as nodes from all videos is constructed for each object category. We formulate it as the submodular optimization problem to co-select tracklets that belong to true objects (depicted as glowing nodes), and produce final semantic segmentation results.

based method is developed to segment object-like tracklets while maintaining temporal consistency in videos. Third, a submodular function is formulated to carry out semantic co-segmentation from tracklets in all videos.

4.2 Proposed Algorithm

4.2.1 Overview

Given a set of videos with unknown object categories, our goal is to discover and segment prominent objects, as well as assign each object a semantic label. To achieve this, we first utilize a fully convolutional network (FCN) [61] trained on the PASCAL VOC 2011 dataset [22] to segment objects in each frame, where each segment has a predicted category. To reduce noisy segments in each video, we cluster segments and eliminate clusters containing noisy segments through the video. Among the selected clusters with object segments, we randomly choose a few of them as initializations and apply a spatial-temporal graph-based tracking algorithm to generate tracklets. Each tracklet maintains coherent appearances of an object region (segment) in the spatial and temporal domains.

However, tracklets may still contain only object parts or noisy background clutters, and the available visual information is limited within each video. We construct a graph that connects tracklets within the same category from all videos as nodes, and utilize a submodular function to define the corresponding relations based on their appearances, shapes and motions. After maximizing this submodular function, tracklets are ranked according to their mutual similarities, and hence prominent objects can be discovered in each video. Figure 4.1 shows the overview of the proposed algorithm.

4.2.2 Semantic Tracklet Generation

Video object segmentation methods usually utilize object proposals in each frame to detect where instances may appear [23, 55, 56, 120]. One challenge is to associate thousands of proposals from different objects while maintaining temporal connections for each of them across all sequences. Here, we propose to utilize a semantic segmentation algorithm (e.g., FCN) to generate object segments as initializations, and then construct a spatial-temporal graphical model to track each object segment and form tracklets. The

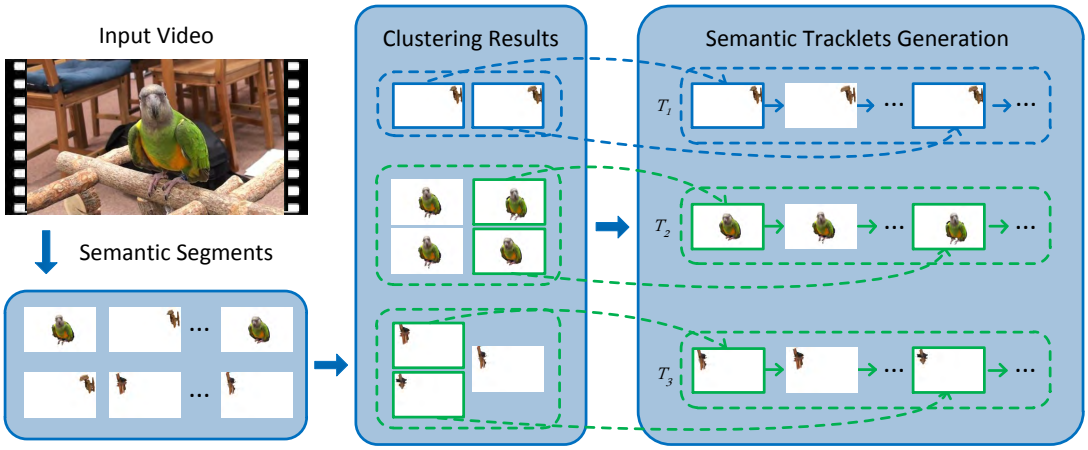


Figure 4.2: Illustration of the proposed method for semantic tracklet generation. Given an input video, we first utilize the FCN algorithm to produce semantic segments in each frame. We then cluster all segments within each object category into different groups, where each color denotes one category (e.g., two green groups for birds and one blue group for dogs). Within each group, we randomly select a few segments as multiple initializations (depicted as rectangular boxes with solid color lines) and utilize a tracking-based approach to generate semantic tracklets T_i. Note that we only show the forward tracklets in this figure (similar process when generating backward tracklets).

procedure to generate tracklets is illustrated in Figure 4.2.

Selecting Objects Segments via Clustering. We first apply the FCN algorithm to extract object segments in each frame of one video. To reduce noisy segments that are not likely to be any object, a simple yet effective clustering method is utilized to select object-like segments through each video. Since the number of object instances is unknown, we apply the mean shift clustering method on all the segments within each object category based on color histograms in the RGB space. Then we select the N largest clusters (i.e., top 80% of the largest ones) while removing the others.

The object segments in selected clusters are considered as initializations for tracking. We randomly choose a few segments from each cluster, while ensuring the selected seg-

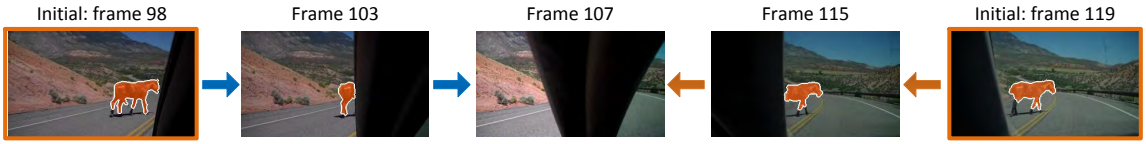


Figure 4.3: An example to track the object under heavy occlusions based on the proposed bi-directional approach with multiple initializations, where initialized segments are denoted as colored rectangular boxes.

ments are within a certain time frame (e.g., at least 20 frames apart between two selected segments) to increase the diversity. However, these initializations may not contain the entire object region or include background clutters. To refine each initialized segment, we learn an online SVM model based on color histograms (as used in the clustering stage), and re-estimate the foreground region using an iterative scheme (e.g., one iteration is sufficient in this work) as in the GrabCut method [75].

Tracking Object Segments. Based on multiple initializations from the previous step, we aim to track segments and generate consistent tracklets (as illustrated in Figure 4.2). The tracking scheme can better localize objects that may be missed by detection algorithms in a single frame, while maintaining temporal connections between object segments. Since selected segments within the same cluster share similar appearances, we track multiple segments in both forward and backward directions, and group them into two tracklets. Hence, we obtain $2N$ tracklets for each cluster. We note that the bi-directional approach facilitates tracking segments under heavy occlusions (see Figure 4.3 for an example). Further note that each initialized segment only tracks a small number of frames until reaching the next initialization, as most tracking methods perform well within a number of frames.

Considering the case of forward tracking from frame $t-1$ to t, the goal is to assign each pixel $x_i^t \in X$ with a foreground or background label $\in \{0, 1\}$. We define an energy

function in a Conditional Random Field (CRF):

$$E(X) = U_t(X) + \gamma^s \sum_{(i,j,t)\in\mathcal{N}_t} V_t(x_i^t, x_j^t) + \gamma^t \sum_{(i,j,t)\in\mathcal{N}_t} W_t(x_i^{t-1}, x_j^t), \tag{4.1}$$

where U_t is the unary potential to be foreground or background, and V_t and W_t are pairwise potentials for spatial and temporal smoothnesses with weights γ^s and γ^t, respectively. The pairwise terms are defined in a way similar to those in [66]. To reduce the computational load and the effect of background noise, we only segment the object within an estimated object location R_t, obtained as in [101]. Note that we also define $\mathcal{N}_t$ as the neighboring set within this region. For the unary term in (4.1), we compute appearance and location energies defined by:

$$U_t(X) = \alpha \sum_{(i,t)\in R_t} \Phi_a(x_i^t) + \beta \sum_{(i,t)\in R_t} \Phi_l(x_i^t), \tag{4.2}$$

where Φ_a is the appearance term, and Φ_l is the location term. For the appearance term, we learn a SVM model based on color histograms (as used in the clustering stage) from the first frame, and an online SVM model with CNN features [62] updated every frame. The weight α consists of α^{col} and α^{cnn} for the color and CNN features, respectively. By minimizing (4.1) using the graph cut method [6], we obtain labels and thus the object mask within R_t, and continue to track segments in the next frame.

4.2.3 Semantic Tracklet Co-selection via Submodular Function

For each video, we generate a set of tracklets where each one is assigned to an object category from the FCN method. However, these tracklets are usually noisy (false negatives) and may not belong to any true object (false positives). In addition, objects within the same category usually share more similarities. To better select object-like tracklets, we collect all those within the same category from all videos to help each other. That is achieved by constructing a graph where the tracklets are nodes, and formulating it as a

submodular optimization problem which aims to find a subset that shares more similarities. Once tracklets are selected in each video, we rank different semantic objects based on the submodular energies and find prominent objects.

Graph Construction on Tracklets. We first collect tracklets from all videos where each one is associated with an object category from a set of M categories $\mathcal{L} = \{1, 2, \cdots, M\}$. For each category $l \in \mathcal{L}$, we can find a tracklet set $\mathcal{O}$, and construct a graph $G = (\mathcal{V}, \mathcal{E})$ containing tracklets from all videos (with the same category l), where each node $v \in \mathcal{V}$ is a tracklet and the edges $e \in \mathcal{E}$ model the pairwise relations. For each G, we aim to discover an object-like tracklet set $\mathcal{A}$ of $\mathcal{O}$ by iteratively selecting elements of $\mathcal{O}$ into $\mathcal{A}$.

Submodular Function. Our submodular objective function is designed to find tracklets that meet two criteria: 1) sharing more similarities, 2) maintaining high quality object-like segments. To achieve this, we model the submodular function with a facility location term [53, 126] to compute similarities, and a unary term that measures how likely the tracklet belongs to the true object. We first introduce the facility location term defined as:

$$\mathcal{F}(\mathcal{A}) = \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{V}} w_{ij} - \sum_{i \in \mathcal{A}} \phi_i, \tag{4.3}$$

where w_{ij} is the pairwise relation between a potential facility v_i and a node v_j, and ϕ_i is the cost to open a facility fixed to a constant ϵ. In (4.3), we define w_{ij} as the similarity $S(v_i, v_j)$ to encourage the model to find a similar facility v_i to v_j such that the final selected tracklets share more similarities.

To compute the similarity between two tracklets, we represent each tracklet by a feature vector F_i, and compute their inner product, $S(v_i, v_j) = \langle F_i, F_j \rangle$, as the similarity. For each tracklet, we extract CNN features (same as mentioned in (4.2)) in each frame and utilize an average pooling method to compute a feature vector that represents each object. Then F_i is computed by averaging feature vectors from all the frames to represent each tracklet. Note that F_i represents appearance of the tracklet in semantics that is learned from CNN, and hence tracklets within the same category are likely to have higher mutual similarities.

However, with only the facility location term, it is not effective in removing all the noisy tracklets in the selected subset $\mathcal{A}$. Hence we propose to include a unary term in the submodular function that can measure the quality of tracklets while preserving the submodularity. The proposed unary term is defined as:

$$\mathcal{U}(\mathcal{A}) = \lambda_o \sum_{i \in \mathcal{A}} \Phi_o(i) + \lambda_m \sum_{i \in \mathcal{A}} \Phi_m(i) + \lambda_s \sum_{i \in \mathcal{A}} \Phi_s(i), \tag{4.4}$$

where $\Phi_o(i)$ measures how likely v_i belongs to the true object (objectness score), and $\Phi_m(i)$ and $\Phi_s(i)$ evaluate the quality of v_i based on the consistency of motions and shapes.

First, we compute $\Phi_o(i) = p_o(i)$ by utilizing probabilities from the FCN output layer according to its category, where $p_o(i)$ is the average probability on all the pixels in v_i. For motion consistency, we use a method similar to [119] and compute motion scores around segment boundaries based on the average gradient magnitude of optical flow estimations [112]. Then we compute $\Phi_m(i)$ by averaging all the motion scores obtained for every two frames. The shape consistency is also considered by computing the intersection-over-union (overlap) ratio between two object segments in adjacent frames. We then compute the variance $\nu_s(i)$ of these overlap ratios, and define $\Phi_s(i) = 1 - \nu_s(i)$, which reflects that larger variance has lower consistency.

Optimization for Tracklet Co-selection. We aim to formulate a submodular function such that tracklets in the selected set $\mathcal{A}$ share more similarities and maintain object-like as well as consistent segments. We combine the facility location term (4.3) and the unary term (4.4) with a weight δ into an objective function, and the submodularity is preserved by linearly combining two non-negative terms:

$$\begin{aligned} \max_{\mathcal{A}} \mathcal{C}(\mathcal{A}) = \max_{\mathcal{A}} \mathcal{F}(\mathcal{A}) + \delta \mathcal{U}(\mathcal{A}), \\ \text{s.t. } \mathcal{A} \subseteq \mathcal{O} \subseteq \mathcal{V},\ \mathcal{N}_{\mathcal{A}} \leq \mathcal{N}, \\ \mathcal{H}(\mathcal{A}^i) \geq 0, \\ \mathcal{H}(\mathcal{A}^i) \geq \rho \cdot \mathcal{H}(\mathcal{A}^{i-1}), \end{aligned} \tag{4.5}$$

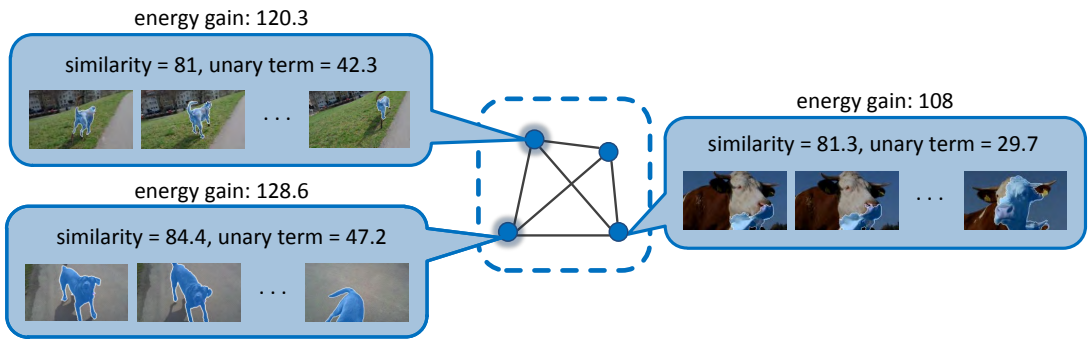


Figure 4.4: Illustration of the proposed submodular function for tracklet co-selection. We show three tracklets within the dog category, where the left two tracklets are selected as true objects (denoted as glowing nodes). For each tracklet, we show energy gain, unary term and summed pairwise energy (similarity) in the facility location term. While all three tracklets have high similarity scores, the right tracklet (false positive) has lower energy gain due to low unary term resulting from inconsistent motions and shapes, and hence it is not selected as the object.

where $\mathcal{N}_{\mathcal{A}}$ is the number of open facilities, and $\mathcal{H}(\mathcal{A}^i)$ is the energy gain at iterations i during iterative optimization, which is defined as: $\mathcal{C}(\mathcal{A}^i) - \mathcal{C}(\mathcal{A}^{i-1})$. We adopt a greedy algorithm to optimize (4.5) in a way similar to [126]. We start from an empty set of $\mathcal{A}$ and iteratively add an element $a \in \mathcal{V} \backslash \mathcal{A}$ to $\mathcal{A}$ that provides the largest energy gain. The iterative process stops when one of the following conditions is satisfied. First, the number of selected nodes is reached, i.e., $\mathcal{N}_{\mathcal{A}} > \mathcal{N}$. Second, the energy gain is negative, i.e., $\mathcal{H}(\mathcal{A}^i) < 0$. Third, the ratio of increased energy gain is below a threshold, i.e., $\mathcal{H}(\mathcal{A}^i) < \rho \cdot \mathcal{H}(\mathcal{A}^{i-1})$, when $i \geq 2$. We show the main steps of the tracklet co-selection algorithm for each category l in Algorithm 2 and Figure 4.4 illustrates the effectiveness of the proposed submodular function.

After optimizing (4.5) for each graph G within one category, we select a set of tracklets $\mathcal{T}_l$ for each category l. Considering each video, we can obtain a few tracklets from different sets of $\mathcal{T}_l$, where l can be any category among $\mathcal{L}$. In each video, we then compute

Algorithm 2: Tracklet Co-selection for Each Category

Input: $G = (\mathcal{V}, \mathcal{E}), \mathcal{N}, \rho$;
Initialization: $\mathcal{A}^0 \leftarrow \emptyset$, $\mathcal{O}^0 \leftarrow \mathcal{V}$, $i \leftarrow 1$;
repeat
  $a^* = \arg\max_{\{\mathcal{A}^i \in \mathcal{V}\}} \mathcal{H}(\mathcal{A}^i)$, where $\mathcal{A}^i = \mathcal{A}^{i-1} \cup a$;
  $\mathcal{A}^i \leftarrow \mathcal{A}^{i-1} \cup a^*$, $\mathcal{O}^i \leftarrow \mathcal{O}^{i-1} - a^*$, $i = i + 1$;
until $\mathcal{N}_{\mathcal{A}} > \mathcal{N}$ *or* $\mathcal{H}(\mathcal{A}^i) < 0$ *or* $\mathcal{H}(\mathcal{A}^i) < \rho \cdot \mathcal{H}(\mathcal{A}^{i-1})$ *when* $i \geq 2$;
Output: $\mathcal{A} \leftarrow \mathcal{A}^i$, $\mathcal{O} \leftarrow \mathcal{O}^i$;

the normalized energy gain for each obtained tracklet and re-rank all of them. This is, a normalized gain for a tracklet with category l added at iteration i during optimization is computed as $\mathcal{G}_l^i = \frac{\mathcal{H}(\mathcal{A}^i)}{\mathcal{C}(\mathcal{A}^1)}$, where $\mathcal{C}(\mathcal{A}^1)$ is the energy as the normalization term after adding the first tracklet. Based on the re-ranked results, a threshold (i.e., 0.85 in this work) is applied to all $\mathcal{G}_l^i$ for selecting a set of semantic tracklets that represent prominent objects. To obtain final semantic segmentation results, since object segments from different tracklets may overlap with each other, we choose the one with larger $\mathcal{G}_l^i$ in overlapped regions.

4.3 Experimental Results

We evaluate the proposed co-segmentation algorithm against the state-of-the-art methods on numerous benchmark datasets.

4.3.1 Experimental Settings

For tracklet generation, we learn an online SVM model with CNN features combining the first three convolutional layers [61] (i.e., 448 dimensional vectors). For parameters in the graphical model (4.1) and (4.2), we use $\alpha^{col} = 1, \alpha^{cnn} = 1, \beta = 0.5, \gamma^s = 3.5$ and $\gamma^t = 1$. In the submodular function, we set ϵ as 3 in the facility location term of (4.3), and

use $\lambda_o = \lambda_m = \lambda_s = 1$ in the unary term of (4.4). During submodular optimization, we use $\delta = 20$ in (4.5), and set $\mathcal{N} = 10$ and $\rho = 0.8$ to determine stopping conditions. All these parameters are fixed in the experiments for fair evaluation.

4.3.2 Youtube-Objects Dataset

The Youtube-Objects dataset [71] contains 10 object categories, and the length of each sequence is up to 400 frames. We evaluate the proposed algorithm in a subset of 126 videos with more than 20000 frames, where the pixel-wise annotations in every 10 frames are provided by [37]. Note that, different from previous video co-segmentation datasets [16, 120], appearances and shapes of objects from the same category in this dataset are significantly different.

We first conduct experiments in a weakly supervised manner, where a semantic label is given for each video. Next, we evaluate our algorithm in a way that object categories are unknown in videos. Table 4.1 shows segmentation results of the proposed method and other state-of-the-art approaches. We use the intersection-over-union (overlap) ratio to evaluate all the methods.

Weakly Labeled Videos. For the video labeled with a semantic category, we use FCN segments belonging to its video-level category as initializations, such that tracklets generated in each video (as described in Section 4.2.2) are all associated with the same category. We compare our approach with other supervised tracking-based [25, 65] or weakly supervised [122][1] methods. Table 4.1 shows that the proposed method with weak supervision performs favorably in terms of overlap ratio, especially in 7 out of 10 categories.

In general, our method performs well on non-rigid objects (*bird*, *cat*, *dog*, *horse*) and fast moving objects (*car*, *train*). As the appearances and shapes of these objects vary significantly, it is challenging to segment these objects from all videos accurately. Although

[1] [122] evaluates the method on their annotated images, and we obtain their results on the same annotation set [37] directly from the authors.

Table 4.1: Segmentation results on the Youtube-Objects dataset with the overlap ratio.

Category	[25]	[122]	Ours	[65]	[66]	Baseline (FCN)	Ours
Supervised?	Y	weakly	weakly	N	N	N	N
aeroplane	**73.6**	72.4	69.3	13.7	**70.9**	60.8	69.3
bird	56.1	66.6	**76.1**	12.2	70.6	69.7	**76.0**
boat	**57.8**	43.0	57.2	10.8	42.5	44.7	**53.5**
car	33.9	58.9	**70.4**	23.7	65.2	60.3	**70.4**
cat	30.5	36.4	**67.7**	18.6	52.1	53.9	**66.8**
cow	41.8	58.2	**59.7**	16.3	44.5	**52.8**	49.0
dog	36.8	48.7	**64.2**	18.0	**65.3**	52.8	47.5
horse	44.3	49.6	**57.1**	11.5	53.5	42.4	**55.7**
motorbike	**48.9**	41.4	44.1	10.6	44.2	**47.3**	39.5
train	39.2	49.3	**57.9**	19.6	29.6	**54.7**	53.4
Mean	46.3	52.4	**62.3**	15.5	53.8	53.9	**58.1**

the recent method [122] utilizes object detectors and generates proposals to localize objects in each frame, it is less effective for videos with large appearance and shape variations as the generated proposals are usually noisy and less consistent across videos. In contrast, the proposed tracking-based algorithm is able to capture detailed appearance and shape changes, and hence generate tracklets consistently for segmentation.

Semantic Co-segmentation. In addition to weakly supervised settings, the proposed algorithm can segment objects and discover the corresponding object categories without any supervision. Table 4.1 shows our segmentation results compared with the state-of-the-art unsupervised method [66]. The proposed algorithm generates more accurate segmentation results in most categories with significant improvement (e.g., more than 10% gain in *boat, cat* and *train*). It demonstrates the effectiveness of our co-segmentation scheme that links relations between semantic objects from all videos, which is not addressed in [66].

To evaluate the effectiveness of the proposed tracking-based algorithm for tracklet generation, we establish a baseline method which directly groups FCN segments from every frame into a tracklet for each category (i.e., without using tracking). We then use the same submodular function for tracklet co-selection (Section 4.2.3). Compared to this

Figure 4.5: Example results for semantic co-segmentation on the Youtube-Objects dataset (without knowing object categories). The colors overlapping on the objects indicate different semantic labels. The results show that our method is able to track and segment (multiple) objects under challenges such as occlusions, fast movements, deformed shapes, scale changes and cluttered backgrounds. Best viewed in color with enlarged images.

baseline method, the proposed algorithm performs well on most categories, especially for deformable objects such as *bird*, *cat* and *horse*, as consistent tracklets can be extracted. However, the proposed algorithm does not perform well in some videos (*cow*, *motorbike*) as some segments are not initialized well, which causes inaccurate tracking results in these videos.

Compared to the proposed algorithm with weakly supervised setting, the results on categories such as *aeroplane*, *bird* and *car* have identical and high overlap ratios. It shows that without providing video-level labels, our co-segmentation approach can reduce noisy segments that are generated from other false categories, and hence retain high accuracies as with weakly supervised setting. Moreover, it is worth noticing that the proposed algorithm without supervision, already performs favorably against the state-of-the-art method that requires weak supervision [122].

Different from other methods [66, 122], the proposed algorithm can segment objects as well as discover object categories (labels). We evaluate the classification accuracy for predicting object categories based on ranked tracklets, and the average precision (AP) is 85.3 on average over all categories. The results show that with the proposed submodular function and re-ranking in each video, false positives can be reduced, and hence prominent objects are discovered. We show qualitative results in Figure 4.5.

4.3.3 MOViCS Dataset

The MOViCS dataset [16], which contains 4 sets with 11 sequences, is used for evaluation on multi-class video co-segmentation. For each set, at least one common object appears in all videos, while the number of object categories is unknown. The proposed algorithm is evaluated against three state-of-the-art methods including image co-segmentation (ICS) [42], video co-segmentation (VCS) [16] and RMWC [120]. We use the unsupervised method [66] as a baseline and produce segments in each frame as initializations for tracklet generation (Section 4.2.2). In addition, since categories are not known for different segments at this stage, one graph including tracklets from all videos is constructed for co-selecting tracklets in each video.

Based on the evaluation metric in [16], Table 4.2 shows that the proposed algorithm performs well in all the video sets, especially in the *tiger* set. As the variations of objects in some videos are large, other approaches are less effective in segmenting objects in these

Table 4.2: Segmentation results on the MOViCS dataset with the overlap ratio.

Video Set	ICS [42]	RMWC [120]	VCS [16]	Baseline [66]	Ours
chicken & turtle	8.0	86.0	65.0	73.6	**87.7**
zebra & lion	23.0	58.8	48.0	45.9	**71.3**
giraffe & elephant	7.0	52.8	52.0	36.5	**59.0**
tiger	30.0	33.6	30.0	44.1	**70.9**
Mean	17.0	57.8	48.8	50.0	**72.2**

Input RMWC [120] Baseline [66] Our results

Figure 4.6: Example results for object co-segmentation on the MOViCS dataset. Segmentation outputs are indicated as colored contours, where each color represents an instance. Compared to the state-of-the-art approach [120] and the baseline method [66] that often produce noisy segments or missing objects, our method obtains better segmentation results. Best viewed in color.

videos. In contrast, our method works for objects with various appearances in different videos by utilizing the submodular optimization that accounts for appearances, shapes and motions together to co-select tracklets containing common objects. We show qualitative comparisons to other methods in Figure 4.6.

4.3.4 Safari Dataset

In addition to co-segmentation in videos where each set contains at least one common object, our method is able to segment objects given a collection of sequences without any prior knowledge. The Safari dataset [120] contains 9 videos with 5 object categories, where each video may contain one or two object categories. To evaluate the proposed

Table 4.3: Segmentation results on the Safari dataset with the overlap ratio.

Object	RMWC [120]	VCS [16]	Baseline [66]	Ours
buffalo	86.9	68.6	90.0	**91.3**
elephant	35.3	26.6	73.8	**74.9**
giraffe	2.4	2.4	9.8	**15.8**
lion	**31.7**	30.2	19.0	21.9
sheep	36.3	4.8	32.3	**65.8**
Mean	38.5	26.5	45.0	**54.0**

algorithm, we input these 9 videos together and segment common objects. Note that, we use [66] as the baseline method for single video object segmentation. Then we initialize these segments to generate tracklets and construct a graph for tracklet co-selection.

Table 4.3 shows the results by the proposed algorithm and two state-of-the-art methods. In 4 out of 5 categories, our method achieves better results over the other methods. The VCS [16] method is not effective for the general setting when videos contain unknown types of object categories, and hence generates less accurate results. The RMWC method [120] relies on object proposals and does not generate consistent tracklets across videos when more than one object category is involved. In our proposed algorithm, we utilize a tracking-based method to generate consistent tracklets, and segment objects via submodular optimization in multiple videos without any assumption on the commonality of objects in the videos. We show some example results in Figure 4.7.

4.4 Summary

In this work, we present a novel algorithm to segment objects and understand their visual semantics from a collection of videos. To exploit semantic information, we first assign a category for each discovered segment in videos via the FCN method. A tracking-based approach is presented to generate consistent tracklets across videos. We then link the relations between videos by constructing graphs which contain tracklets from different videos.

Figure 4.7: Example results for object co-segmentation on the Safari dataset. Segmentation outputs are indicated as colored contours, where each color represents an instance. Compared to the state-of-the-art approach [120] (second row) and the baseline method [66] (first row) that often produce noisy segments, false positives or missing objects, our method obtains better segmentation results. Best viewed in color.

Without any assumption of objects appearing in videos, we formulate a submodular optimization problem and co-select tracklets, which accounts for their appearances, shapes and motions. This step considers other sequences and reduces noisy tracklets that can not be filtered out within a single video. As a result, prominent objects are discovered and segmented in videos. Extensive experimental results on the Youtube-Objects, MOViCS and Safari datasets show that our method performs favorably against the state-of-the-art approaches in terms of visual quality and accuracy.

Chapter 5

Automatic Semantic-aware Sky Replacement

5.1 Introduction

Skies are one of the most common backgrounds in photos. However, we have no control over the weather or lighting conditions at the moment of photography. As a result, numerous interesting and valuable photos have uninteresting or poorly exposed sky regions. Professional photographers fix this problem using sophisticated tools by manually delineating the sky regions precisely, testing different skies for compatibility, and finally adjusting the foreground to match the new composited sky. This requires time and expertise that is beyond the abilities of novice users. In this work, we propose a fully automatic sky replacement tool that can take an input image and generate a diverse set of realistically edited photos with interesting skies and different styles (see Figure 5.1). This can expedite the editing process for professionals, and allow casual users with minimal expertise to explore interesting results.

To achieve this goal, we address three challenging questions in this work. Can we accurately segment the sky region from the image? How do we find interesting sky images

Input photograph Automatic sky replacement results with different styles

Figure 5.1: Given an input photograph, the proposed algorithm automatically generates a set of images with stylized skies. The proposed algorithm exploits visual semantics for sky editing, in which scene parsing is first performed on the input image, and such semantic information is utilized for the subsequent steps including sky segmentation, search and replacement.

that are compatible with the input image? Finally, can we match the appearance of the input and new sky images to create realistic composites? In this work, we show that a deep semantic understanding of images is critical to all these tasks. For example, sky appearance varies widely among images, and without an understanding of scene layout, it can be indistinguishable from non-sky regions (e.g., reflections in water). Similarly, it is important that we search photos whose semantic layout and content match the input image. This ensures that the perspective of the composited sky is consistent with the input image. It also allows us to adjust the foreground appearance after sky replacement and improve the realism of the result. For instance, when adjusting the color of water under a new sky, it is better to transfer appearance from a water region appearing with that sky than from an arbitrary region with other content.

In this work, we propose a semantic-aware approach for sky editing, in which visual semantics are extracted from an input image, and instilled into the subsequent steps: sky segmentation, search and replacement. The overall framework of our approach is illustrated in Figure 5.2. We first learn a deep Fully Convolutional Neural Network (FCN) [61] to parse an input image and generate a dense pixel-wise prediction of semantic labels such as sky, tree, building, mountain and water. By understanding the global layout of the scene,

the proposed algorithm can robustly localize the sky regions in spite of different colors, shapes, sizes and attributes. The proposed online classifier is then trained to model the sky appearance and used to generate a refined sky segmentation mask with clean boundaries across the sky and non-sky regions. Experiments show that the generated sky segmentation results by the proposed algorithm achieve high intersection-over-union (IOU) ratio against the ground truth masks, thereby making it effective at identifying background regions without manual refinement.

To select interesting and stylized sky exemplars for replacement, we construct a database with 415 high aesthetic quality images covering a wide range of sky appearance and scene categories. We use the FCN trained for scene parsing to construct feature vectors that represent the semantic content and layout of those exemplars, which enables us to retrieve images that are semantically similar to an input photo yet have diverse styles of sky regions. Once the sky images are selected, the new sky regions are automatically aligned and composited into the input photo. The color, saturation and luminance of the non-sky region are then adaptively adjusted to ensure that the composite photos are visually realistic. Since the content of the reference images are similar to the input image, and the semantic regions have already been segmented on both the input and reference images, we leverage this information and propose a novel semantic-aware approach to transfer the appearance of the reference exemplars to the input and create results with stylized sky backgrounds.

We demonstrate that the proposed algorithm is able to render photorealistic and pleasing images through extensive user studies. In particular, we show that our approach performs favorably against existing methods for scene compositing in terms of realism, diversity, and interestingness. The contributions of this work for automatic semantic-aware sky replacement are summarized as follows:

1. We propose a fully automatic sky replacement algorithm that is driven by scene semantics.

2. We develop an accurate coarse-to-fine sky segmentation method using a deep neural

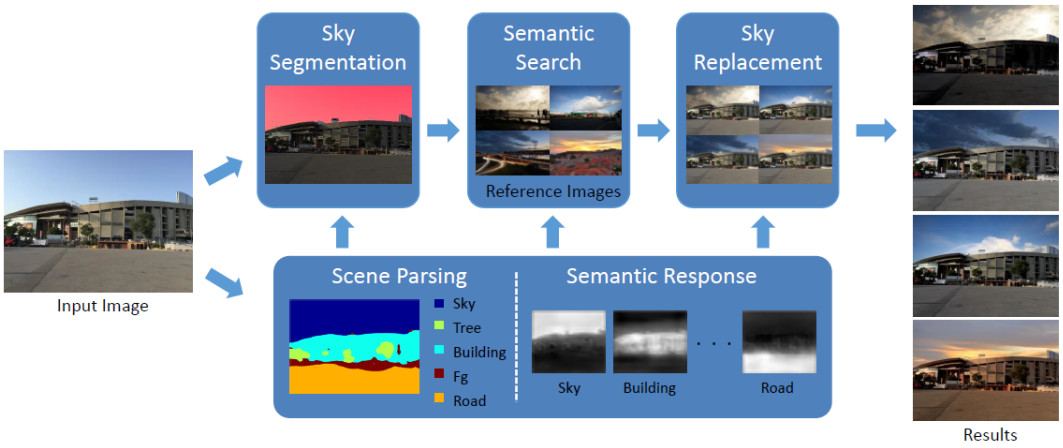


Figure 5.2: Overview of the proposed algorithm. Given an input image, we first utilize the FCN to obtain scene parsing results and semantic response for each category. A coarse-to-fine strategy is adopted to segment sky regions (illustrated as the red mask). To find reference images for sky replacement, we develop a method to search images with similar semantic layout. After re-composing images with the found skies, we transfer visual semantics to match foreground statistics between the input image and the reference image. Finally, a set of composite images with different stylized skies are generated automatically.

network and online classifier learning (Section 5.3).

3. We show a semantic search scheme to determine a set of high-quality images with diverse sky appearance and similar semantic content for replacement (Section 5.4).

4. We present a semantic-aware appearance transfer approach for replacing sky backgrounds to render images that are aesthetically pleasing and photorealistic (Section 5.5).

5.2 Algorithmic Overview

Given an input image I, we aim to automatically generate a set of results with the same foreground as in I but different sky backgrounds. To achieve this, we decompose the task into three sub-tasks: (1) sky segmentation for separating the sky region I^{sky} and the foreground region I^{fg} in image I; (2) sky search for retrieving a set of reference images matching the input image semantics; (3) sky replacement for replacing the original sky region I^{sky} with a new sky R^{sky}, given a reference image R. Meanwhile, the color statistics of the foreground region I^{fg} are automatically adjusted to ensure the composed images are visually consistent and realistic.

The main idea of our work is to exploit visual semantics to help improve the image quality in all three tasks. Toward this, we train a Fully Convolutional Neural Network (FCN) [61] for scene parsing, which is a state-of-the-art end-to-end model for semantic segmentation (see Figure 5.2 for an example). To learn an accurate FCN for scene parsing, we randomly select 15000 outdoor images form the LMSun dataset [93] as training samples. As the labels in the LMSun dataset include many small objects or labels that do not appear often in our daily photos, we manually choose the common labels that cover most scene categories, and merge other object labels to one category as the foreground object. Figure 5.3 shows the list of 11 pre-defined labels for the image editing task considered in this work. In this work, the semantic deep neural network is trained in a way similar to [61]. We quantitatively evaluate on 1045 randomly selected images form the LMSun dataset, and compute the pixel accuracy and intersection-over-union (IOU) ratio. Figure 5.3 shows that semantic segmentation with the defined labels can be accurately computed by the trained neural network. This model can robustly localize arbitrary skies and effectively distinguish the true sky from those most confusing regions such as reflections in the water and mountains in the hazy background. Based on the coarse scene parsing results, we delineate more accurate sky regions with clear segmentation boundaries by online refinement (Section 5.3).

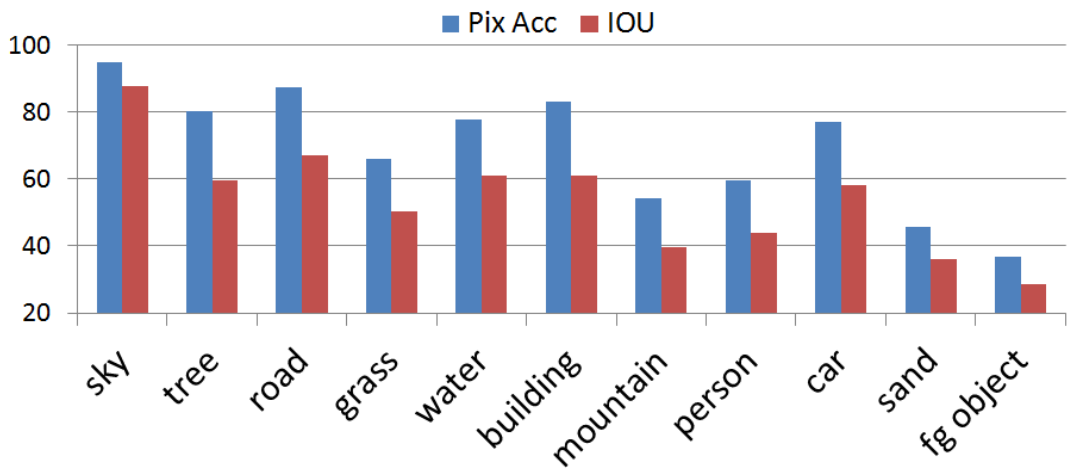


Figure 5.3: Pixel accuracy and intersection over union ratio of the scene parsing results for each category on the LMSun dataset.

The FCN output provides visual presentations describing the scene layout, which facilitates the subsequent sky search and replacement steps. In particular, we normalize the semantic response maps generated by the FCN forward propagation over all the categories to obtain a probability map $F_i = \{f_i^1, f_i^2, \ldots, f_i^n\}$, where f_i^n indicates the likelihood of pixel i belonging to category n. We construct a semantic layout descriptor based on the probability map for the input image as well as for all the images in the sky database, which is used to retrieve a set of images from the database that have similar content and layout to the input image. The skies in those images are therefore likely compatible with the input foreground region, and are proper for replacement. Moreover, since we are only using semantic information for retrieval instead of traditional appearance features, we can find images with different sky appearance to ensure the diversity of our results. The algorithmic details of the proposed semantic search scheme are described in Section 5.4.

Once a reference image is selected, its sky region will be used to replace the original one. As the appearance of the sky is dramatically changed, the color statistics of the foreground need to be adjusted accordingly to make the image visually realistic. However,

global transfer techniques are susceptible to differences in the foreground and can create non-realistic results. In contrast, we have semantic segmentation of the scene that we use to drive a local transfer technique that produces more realistic results. The results show that our transfer method can generate realistic and visually pleasing results compared to other methods (Section 5.5). Figure 5.2 shows the main steps of our method for replacing sky backgrounds based on image semantics.

5.3 Sky Segmentation

Based on the scene parsing results by the trained FCN, we first introduce an accurate sky segmentation algorithm to facilitate sky replacement and reduce visual artifacts. As discussed in Section 5.2, the scene parsing model trained by FCN can robustly localize the sky regions with various appearance and scene layout. In particular, it can achieve 94% pixel accuracy on our evaluation set. Nevertheless, since the image resolution of the FCN output is low, the resulting segmentation masks are coarse with missing details around the boundaries, which cannot be directly used for replacement (see Figure 5.5).

Sky Segment Refinement via Online Models. In order to generate accurate sky segmentation masks, we use the FCN results to bootstrap online classifiers that learn image-specific color and texture models. We formulate a two-class CRF problem for refinement by considering neighboring pixels x_i and x_j with the energy $E(X)$,

$$\begin{aligned} E(X) = \lambda_1 \sum_i U_c(x_i) + \lambda_2 \sum_i U_t(x_i) \\ + \lambda_3 \sum_i U_f(x_i) + \lambda_4 \sum_{(i,j)\in\mathcal{E}} V(x_i, x_j), \end{aligned} \tag{5.1}$$

where U_c and U_t are color and texture unary potentials for the cost to be the sky or non-sky labels, which are obtained from the learned online classifier, and U_f is a location term that accounts for the FCN output. In addition, V is the pairwise potential for smoothness in a

set $\mathcal{E}$ of adjacent pixels, and $\lambda's$ are the weights for each term. Here we use equal weights for three unary terms ($\lambda_1 = \lambda_2 = \lambda_3 = 1$), and a higher weight ($\lambda_4 = 100$) for the pairwise term to ensure the boundary smoothness.

To learn online sky/non-sky classifiers and model unary potentials, we first use the sky response map generated by the FCN output layer as sky/non-sky priors. More specifically, if a pixel has higher response (e.g., larger than a threshold) on the sky label, we use this pixel as the positive training sample to learn the sky classifier, and vice versa. We consider two cues for learning online models. First, we use the Gaussian Mixture Models (GMMs) on RGB channels to model the appearance of sky regions, and obtain U_c by computing the negative logarithm of GMM probability outputs.

However, when the color of sky regions are similar to foreground ones, the CRF model with only the chromatic cues may generate noisy segmentation results. Hence, we also model the texture of sky regions by computing histogram of gradients on superpixels, and learn a Support Vector Machine (SVM) classifier. Similarly, U_t is obtained by calculating the negative logarithm of the SVM outputs that are converted to probabilities through a sigmoid mapping function. Note that the texture model is based on superpixels, and we assign the energy to each pixel according to its parent region and then perform pixel-wise energy minimization.

In addition, we use the sky response map from the FCN to guide the sky location for each pixel x_i, where $U_f(x_i)$ is equal to the negative logarithm of response f_i^{sky}. For the pairwise term V, we use the magnitude of gradient between two adjacent pixels to ensure the boundary smoothness. In order to minimize (5.1), we use the efficient graph cut algorithm [6] to obtain the final pixel-wise sky/non-sky label assignments. However, the sky segments often contain fine details (e.g., small sky patches among tree regions) for the problem considered in this work. As such, we assign soft labels around the boundary with alpha mattes [79], and obtain final composition results that are visually appealing after replacing the sky.

Sky Segmentation Results. We quantitatively evaluate the sky segmentation results on

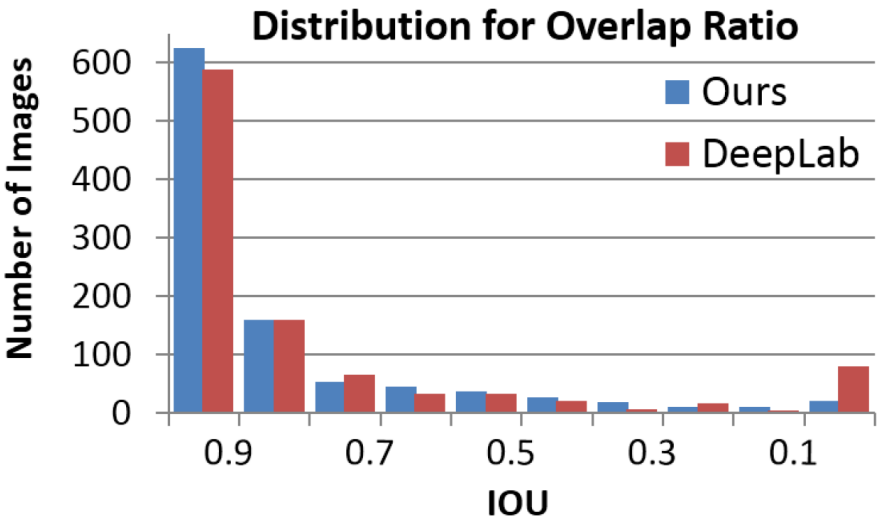
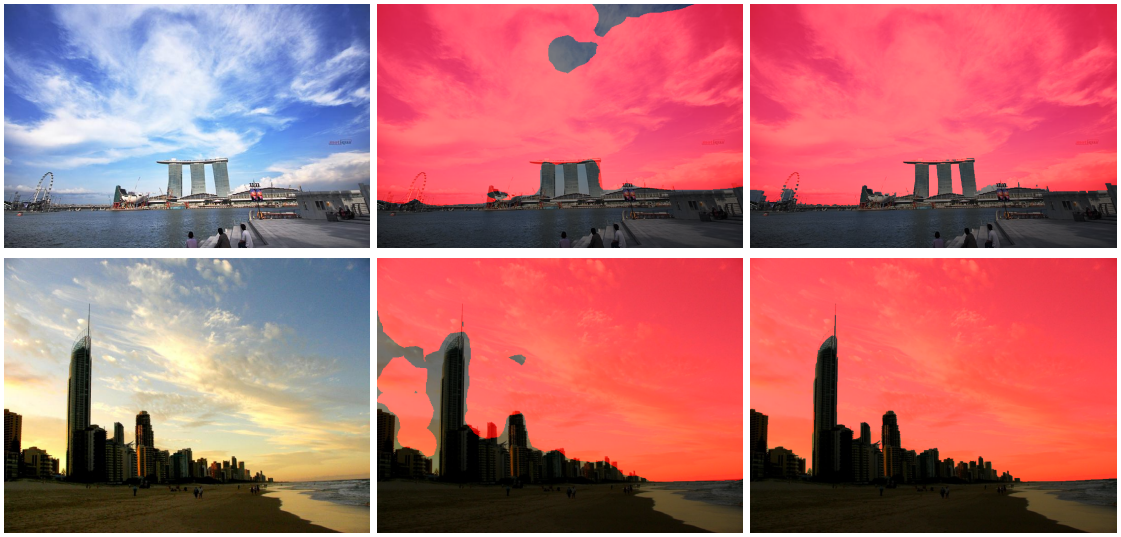


Figure 5.4: Distribution of images in terms of IOU ratio compared to the DeepLab method [13]. Most sky segmentation results by the proposed algorithm have over 90% IOU.

the LMSun dataset. We use the same training and test sets as for scene parsing and model learning. The average IOU is improved from 87.6% to 88.7% after refining the sky segmentation. In addition, we compute the boundary precision-recall (BPR) [24] to measure the quality of boundaries. For refined sky segmentation results, we obtain the BPR as 0.839 with online models, which significantly outperform the FCN with the BPR of 0.639. These refinements are important since inaccurate boundaries will result in obvious artifacts during the sky replacement step. We also compare our refined results to those generated by the state-of-the-art segmentation method with dense CRF [13], whose 87.9% average IOU is lower than that of our method. Figure 5.4 further presents the distribution of results in terms of IOU ratio and shows that we have more results over 90% IOU, which is usually considered visually pleasing without much need of manual refinements. Figure 5.5 shows two examples in which the proposed coarse-to-fine method generates accurate sky segmentations, thereby facilitating better composition around the boundaries.



(a) Input image (b) FCN result (c) Fine segmentation

Figure 5.5: Sample sky segmentation results. Given an input image, the FCN generates results that localize the sky well but contain inaccurate boundaries and noisy segments. The proposed online model refines segmentations that are complete and accurate, especially around the boundaries (best viewed in color with enlarged images).

5.4 Sky Search

In this section, we introduce the proposed algorithm that searches exemplar images for sky replacement, which is of critical importance for generating visually pleasing results. As discussed in Section 5.2, a sky region from a reference image with similar content to the input image is more suitable for replacement. Existing methods use global descriptors such as GIST features to search for similar images [31, 60]. However, the GIST descriptors only describe the global scene layout without important semantic information, and are more effective when the reference images are holistically similar to the inputs, thereby limiting its use for generating composite images with diverse styles. We propose a novel semantic search approach to find a reference image R that is similar to the input image I in both the semantic content and spatial layout, yet with diverse sky appearance. In the

following, we first illustrate how we compute the semantic scene layout descriptors, and then introduce a sky selection scheme.

Semantic Layout Descriptors. As described in Section 5.2, we can obtain a response map from the FCN output layer for each category. We first normalize all the response maps to the range from 0 to 1. Given the response $F_i = \{f_i^1, f_i^2, \ldots, f_i^n\}$ for each pixel i with n categories (i.e., $n = 11$ in this work), its label histogram can be computed as an average pooling process: $H = [h^1; h^2; \ldots; h^n]$, where $h^j = \frac{1}{m}\sum_{i=1}^{m} f_i^j$ and m is the number of pixels. This label histogram indicates the semantic distribution of the region of interest. To obtain a structural version of this descriptor, we use the spatial pyramid pooling method as described in [52]. First, we divide the image into three by three grids, and extract histograms H_s for grid s (i.e., $s = 9$ here). Second, a global histogram from the entire image is extracted. After concatenating all the histograms, a final descriptor is constructed as: $\mathbf{H} = [H_1; H_2; \ldots; H_{s+1}]$, where the dimension of $\mathbf{H}$ to describe each image is $n * (s + 1)$.

These descriptors capture both spatial information and semantic cues. Note that [117] develops a contextual feature descriptor based on a semantic label map generated by traditional scene parsing [93] and object detection [108] approaches. Their descriptors adopt a fine-grained pooling scheme at multiple scales around each pixel, due to the need of enabling CNN training for local photo adjustment. In contrast, we aim to better capture the overall semantic layout of the image and allow certain degree of layout variations during image search. Therefore, our semantic layout descriptor is designed to be a flexible feature representation with spatial pyramid pooling over the entire response maps, and is constructed on the semantic distribution of all the possible labels, instead of on the final semantic label map as in [117]. As demonstrated in Section 5.6, these descriptors are effective for finding relevant reference sky images for both replacement and appearance transfer. Figure 5.8 shows the composite results based on retrieved reference sky images by the proposed semantic search method and descriptors, the GIST based approach and random selection. The results indicate that our method can generate more realistic images

with diverse styles.

Selection of Sky Images. In addition to semantic matching, the retrieved images need to be further pruned by a few properties, including aspect ratio and resolution, to ensure that the replaced sky can align well in the target image.

Although the sky regions are less sensitive to scale changes, we consider the aspect ratio and resolution to ensure that the reference images are not significantly deformed or distorted for alignment. We compute the aspect ratio $P_a = \frac{\text{width}}{\text{height}}$ and resolution $P_s =$ width$*$height for each sky region. Then a metric [97] comparing I^{sky} and R^{sky} is computed as $Q = \frac{\min(P^I, P^R)}{\max(P^I, P^R)}$, where P^I and P^R are properties for original and reference skies, and Q can be defined for aspect ratio or resolution (i.e., Q_a or Q_s). Note that each measurement is between 0 and 1 and a threshold (i.e., 0.5) is applied to determine whether the sky can be used for replacement or not. If any of the above conditions is not satisfied, we evaluate the next retrieved sky image for replacement.

Diversity of Sky Images. One of our goals is to automatically replace the skies with diverse stylized backgrounds. In order to ensure diversity in the retrieved images, we select sky images based on the inner product of color histograms between reference skies. In addition, this step also enables the flexibility of our system. For instance, if a user prefers strong diversity, the system rejects sky images with a high color similarity in comparison to the ones already selected. On the other hand, if the user prefers a certain sky style, we set a color similarity close to the preferred sky.

5.5 Sky Replacement

Given a selected sky region R^{sky}, we align and place it in the input image I for replacement. We first extract the maximum rectangular sky region within R^{sky}, and re-scale this extracted sky rectangle to the size of the minimum rectangle that covers all the sky regions of the input image. Since the necessary scale and aspect ratio changes have been

addressed in the search step as described in Section 5.4, the selected new sky region would not have significant distortion and its main interesting region would be retained.

After compositing the new sky R^{sky} into the input image, color adjustment of the foreground region I^{fg} is required to make it compatible with R^{sky}. As a reference image shares similar semantic content with the input image, we propose a semantic-aware transfer method to adjust foreground appearance.

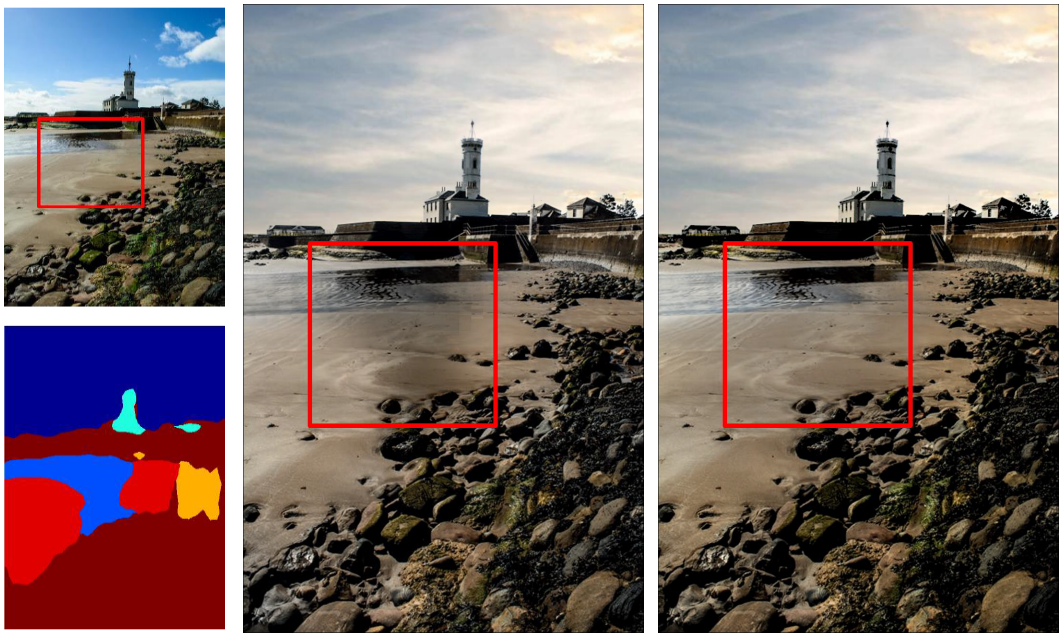
5.5.1 Semantic-aware Transfer

Transferring color statistics from one image to another is a common technique in image editing. Existing approaches usually perform transfer over the entire region without taking visual semantics into account [73, 91] and generate less realistic results when the image content is not well matched. If we take an image pair with beach and sea images for example, without knowing the content, the color from the blue sea may be transferred to a white sand beach, thereby rendering unnatural bluish sand regions. A few local transfer approaches have been developed to directly transfer one region to another [111, 46]. However, due to large appearance variation between different local regions, the resulting images usually contain artifacts around boundaries, which are difficult to be removed by post-processing (e.g., bilateral filtering).

In this work, we exploit visual semantics based on our scene parsing results for transferring color tones. More importantly, we propose a simple yet effective method that uses soft mapping between semantic regions to generate results with smooth boundaries (see Figure 5.6 for comparison). Suppose the total number of semantic labels existing in the scene parsing map of the input image is n_r, we formulate the transfer process for each pixel x as:

$$T(x) = \sum_{n=1}^{n_r} W_n(x) \cdot T_n(x), \tag{5.2}$$

where $W_n(x)$ is the likelihood value in the normalized FCN response map on pixel x for semantic category n, and $\sum_n W_n(x) = 1$. For each semantic label n in the scene parsing



(a) Input (b) Direct local transfer (c) Our soft mapping

Figure 5.6: (a) Input image (before replacing the sky) and its scene parsing result. (b) After applying the local transfer functions and filtering [46], there are artifacts around boundaries. (c) The proposed method generates smooth result before applying any filter.

results of the input image, we compute a category-specific color/luminance transfer function T_n (defined in Section 5.5.2) using the regions associated with this label in the input and reference image. Intuitively, local transferring on a pixel x can be interpreted as a soft interpolation according to its semantic responses. The more likely pixel x belongs to label n, the more it would rely on transfer function T_n. This soft mapping based method can largely mitigate the errors in scene parsing (Figure 5.6), and generate realistic results with smooth boundary transitions, as shown in Figure 5.10.

Implementation Details. In practice, there may be some small noisy responses from irrelevant labels for each pixel x. As such, we only retain the labels with the top few responses for interpolation, and re-normalize the weights with unity sum. We also remove

the foreground object label when generating W_n, as we have merged objects of different categories into this label during the scene parsing, and the appearance variation within this label is too large to have any semantic consistency. After applying transfer functions, we use the original image as guidance, and apply the guided filter [32] to make it better aligned with image edges.

5.5.2 Transfer Functions

In this section we describe the details of the category-specific transfer function T_n in (5.2) for each semantic label in the input image. For a semantic label n, even though the reference and input images are semantically similar, it is still possible that there are no regions assigned with label n in the reference image. Thus, we compute T_n based on whether there are regions associated with label n in both input and reference images or not.

Matched region. In the first case, suppose I_n and R_n are the regions associated with the label n in the input and reference images respectively, we then compute both luminance and chrominance transfer functions from R_n to I_n. For luminance, we shift the mean of luminance (L channel of the LAB color space) in I_n to the one in R_n. In addition, we observe that the foreground appearance should not change much when the new sky is similar to the original one, while the appearance may change drastically with a differently stylized sky image. Hence, we compute color differences between the original sky and the new one, and use it to regularize the shift of luminance mean. Specifically, suppose $c(I^{sky})$ and $c(R^{sky})$ are the means of color in I^{sky} and R^{sky} respectively, we have $\beta = \tanh(\left|c(I^{sky}) - c(R^{sky})\right|)$. Then we compute the new desired mean of luminance as: $\hat{L} = L(I_n) + \beta(L(R_n) - L(I_n))$, where $L(R_n)$ and $L(I_n)$ denote the means of luminance in R_n and I_n. It indicates that when the original and reference skies are significantly different, we aggressively adjust more luminance to match R_n, and when the skies are similar, we retain the luminance of I_n.

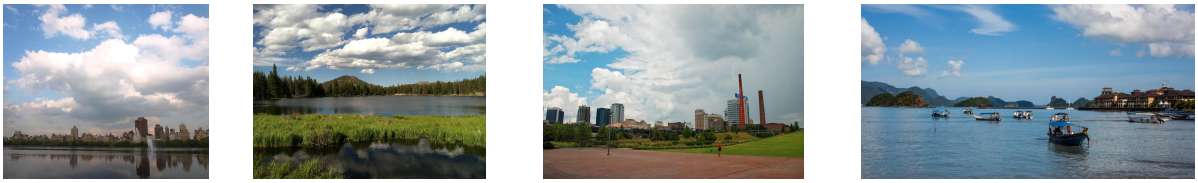
For chrominance transfer, we edit the channels in the LAB color space by matching the mean and covariance of I_n to R_n using the regularized matching method of [54], which performs robustly in practice.

Non-matched region. In the second case, when there is no matched region found in the reference image for label n, we resort to the entire foreground region. That is, we compute a transfer function from the entire reference foreground to the entire original foreground, and use it to represent T_n. We use the same method described in the first case for transferring luminance. For color adjustment, the visual results are more sensitive since no semantic matching is enforced between two foreground regions. Therefore, we transfer the color temperature (CCT) in the XYZ color space [116] rather than the chrominance, which is more conservative but robust to semantic inconsistencies.

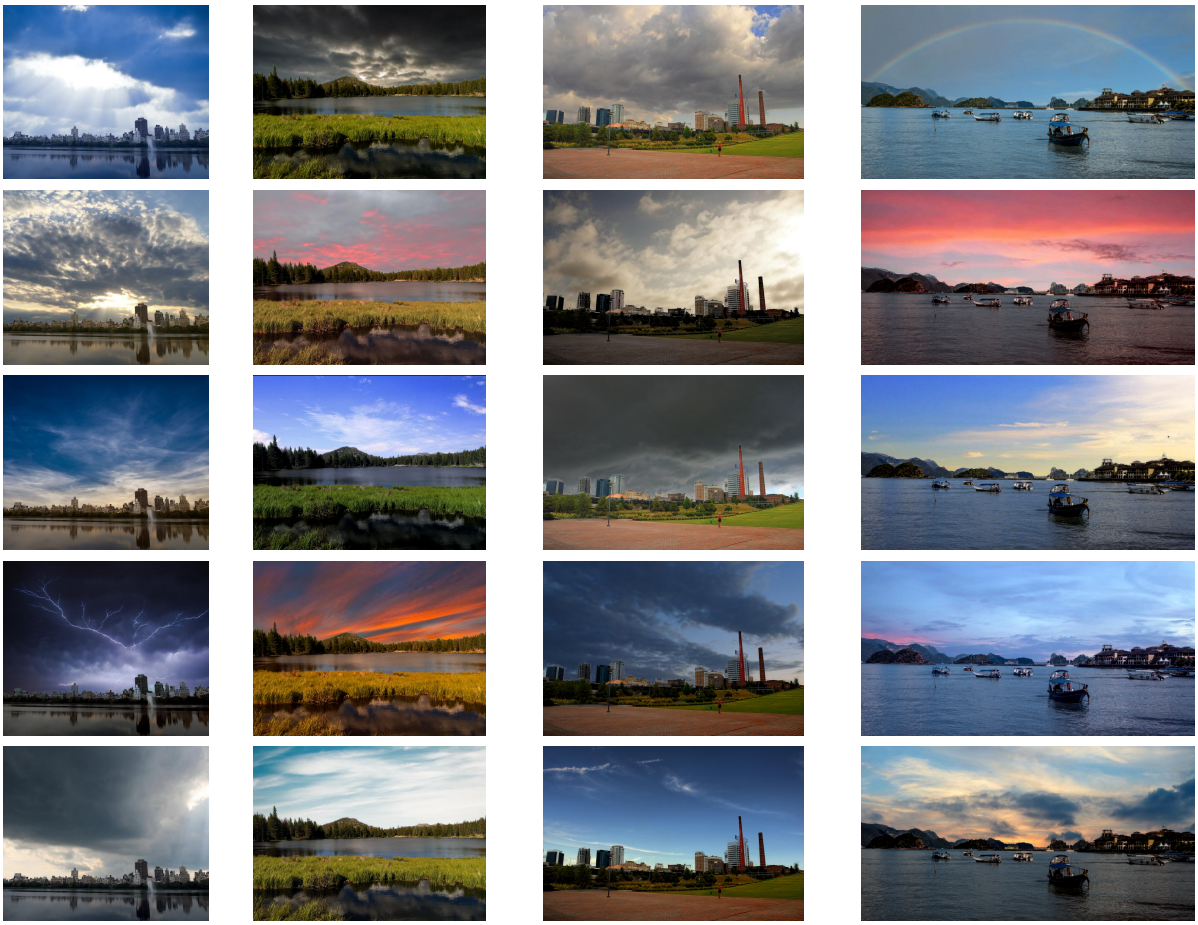
To further prevent generating artifacts due to inconsistent matching, we use a continuous transfer function for histogram matching in a way similar to the regularization used in [54].

5.6 Results and Analysis

We evaluate the proposed algorithm for rendering composite images with stylized sky regions using a large set from our own collection and Flickr. Figure 5.7 shows a subset of the generated composite images, and more results and comparisons are provided in the supplementary material. Experimental results demonstrate that our algorithm can handle input images with a variety of scenes and generate a diverse set of visually pleasing sky backgrounds. To quantitatively evaluate the quality of the proposed method, we randomly select 30 test images for user studies. We design three different tasks to evaluate different components of the proposed algorithm. These tasks include comparisons of the proposed sky search and transfer methods to baseline and existing approaches, as well as the comparison of realism of rendered images by the proposed method with respect to the input photographs.



(a) Input image



(b) Composite images by the proposed algorithm

Figure 5.7: Composite images with stylized sky backgrounds generated by the proposed algorithm. Given an input image (a), we show the top five results (b) with a set of composite images with diverse sky backgrounds.

Figure 5.8: An example of the comparison for different sky searching method. For each method, we search the top four skies to replace the input image (a), and use the same technique to transfer appearance. In (b), random selection may find arbitrary skies that are not proper to match statistics. The GIST based method (c) is able to find a diverse set of skies, but it may produce non-realistic results with artifacts due to the poor matching between images (e.g., the third and fourth results). Our semantic search approach (d) can handle the both issues, and produces a set of results with diverse skies that are visually pleasing.

Comparison of sky search methods. We first compare our semantic search method with random selection and the GIST based retrieval approach [31, 60]. Note that the same semantic-aware transfer method is used for appearance adjustment for all three methods. Qualitatively, the sky examples retrieved by random selection usually do not match the input images well. On the other hand, the method based on the GIST descriptors does not always find images with similar layout and foreground regions as visual semantics are not exploited. In contrast, the proposed method retrieves reference images with diverse styles.

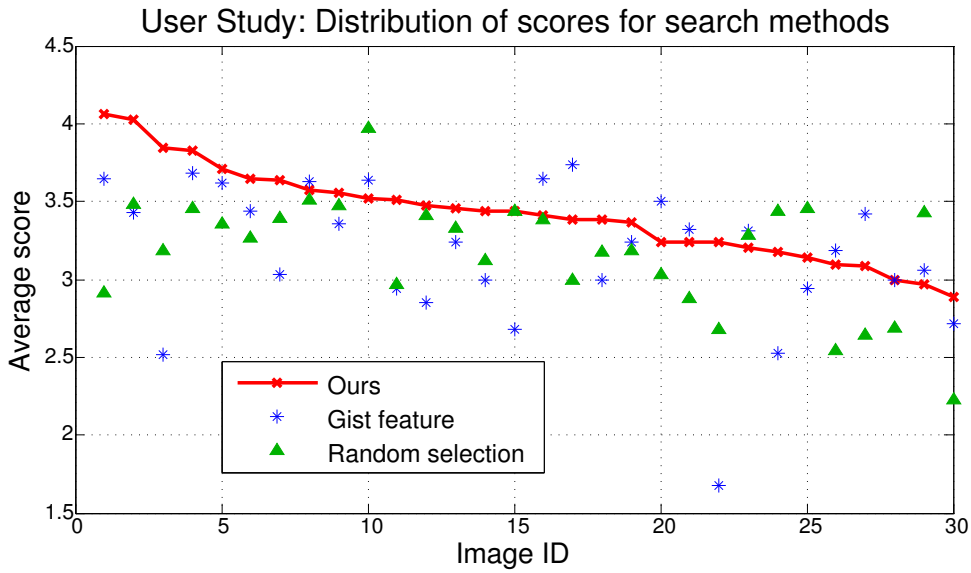


Figure 5.9: Average scores of three different methods for each image with x-axis sorted by our score. The proposed technique outperforms random selection in 80% and GIST in 63% of cases. Even in a few cases that our method does not perform well, the results are close to those by the other two schemes.

Figure 5.8 shows an example comparing those methods.

To better understand the performance of these methods, we perform user studies for quantitative evaluations. Participants were shown an input image and the top five results generated by the three methods. Each subject is asked to rate each set based on how interesting and realistic the images are, using 5-point Likert scale (1 being worst and 5 best). A set of 1028 scores from 39 subjects is tallied. The proposed method obtains the best average score of 3.42, while the average scores are 3.17 for the GIST based method and 3.18 for random selection. The scores of individual images in the evaluation set are shown in Figure 5.9, and our method almost always achieves scores larger than 3 while the other two often fail with low scores.

Comparison of sky transfer methods. Next, we compare the proposed semantic-aware

(a) Input image (b) Reference (c) [91] (d) w/o semantic (e) Ours

Figure 5.10: Rendered results by different sky transfer methods. Given the input image (a) and reference image (b), we show the results using the transfer method proposed in the SkyFinder method [91] (c), our method without using semantic matching (d) and our semantic transfer approach (e). For the global methods (c) and (d), the results are likely to contain clear artifacts (top row), over-colorized and unnatural foreground regions (middle and bottom row) due to transfer methods and reference images. In contrast, our method is robust to the reference images and can generate photorealistic results.

sky transfer method to the SkyFinder approach [91] which matches the mean and standard deviation in the LAB space, and a baseline transfer method without using semantic cues (i.e., directly match from R^{fg} to I^{fg}). The first two methods compute the transfer functions based on the entire foreground regions, which usually generate results with obvious artifacts or over-colorized and unrealistic foregrounds. In contrast, our method takes semantic cues into account and matches regions locally, while using a soft mapping strategy to reduce artifacts around boundaries. Figure 5.10 shows sample results generated by these three methods.

We conduct user studies to evaluate these methods using similar setups as the previous

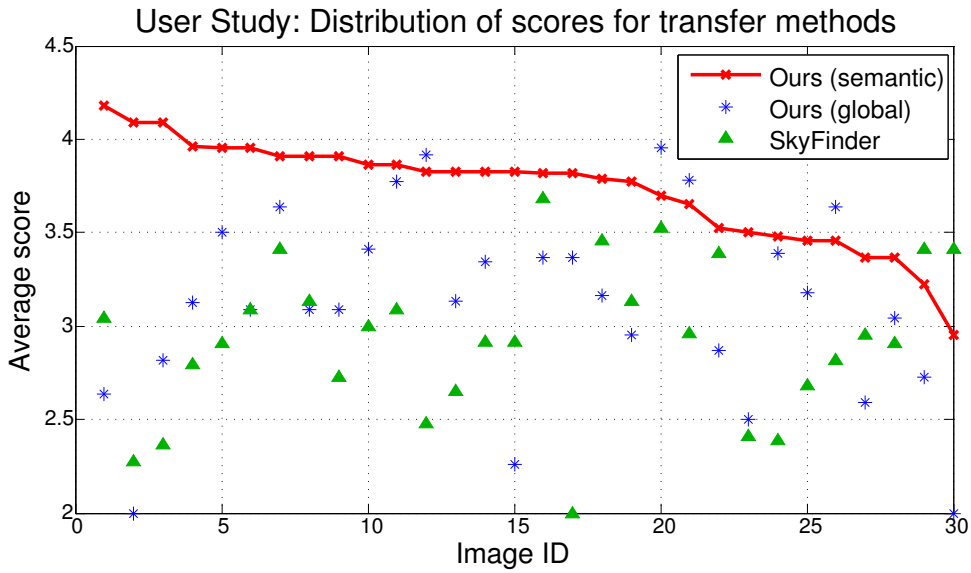


Figure 5.11: Average evaluation scores for each image, sorted by our score. Overall, the proposed technique performs better than other two methods in most cases.

experiments. Each subject is asked to rate each set based on how realistic the images are. From 27 subjects, we obtain 627 scores for evaluation on transfer methods. On average, the proposed semantic transfer approach achieves the best score of 3.73, while the average score is 3.11 for the baseline transfer method without using semantic information. In contrast, the users give an average score of 2.93 for the SkyFinder method. Average scores of individual test images are shown in Figure 5.11 where most of our results are rated above 3.5 and are significantly higher than the other two methods.

Comparison to real photographs. Our third user study is designed to evaluate the visual realism and interestingness of the rendered results compared to original (real) photographs. In this study, each participant evaluates a set of image pairs, where each one contains an original image and one of our rendered results that has the same foreground and a stylized sky background. We randomly choose one of our five results for comparisons to ensure that each user only sees an example test image at a time. Specifically, each user is shown the input and one rendered result side by side in randomized order, and is asked to select

the more realistic image.

A total of 40 subjects participate in this study and a total of 1054 results are tallied. Overall, 61.9% of the real images are favored over the rendered results by the proposed algorithm, 21.2% are rated equally realistic between the two, and interestingly in 16.9% of the test cases our results are considered more realistic than the original images. Since the image pairs are presented side by side, it is easy to find small artifacts in the edited results by directly comparing with the original images, yet still in 38.1% of cases, our results are rated no worse than the real photographs. It indicates the proposed algorithm is able to generate visually pleasing images despite the challenging test set.

In addition, we also ask the subjects to select the image that is more interesting. Among all the evaluated images, 51.2% prefer our results, and 16.1% consider both are equally interesting, while only 32.7% of the cases are in favor of the original images. These results indicate that our algorithm is able to generate more appealing images with interesting styles than the original ones in most cases. For all user studies, the p values are smaller than 0.001, showing the results are statistically significant.

Sky Replacement with Relevance Feedback. In addition to generating images with stylized backgrounds automatically, our system can also be used to guide a user to find preferred sky styles by combining semantic and visual search. Once a user selects a preferred style from our initial results, our system can generate more similar images for users to further explore in a fine-grained manner.

To achieve this, similar to the sky search method described in Section 5.4, we use our semantic search approach to first rank images in the database to ensure the consistency in image content. We then compute the color similarity between the preferred sky by a user and ranked database images, where this similarity can be set flexibly to control the diversity of retrieved sky backgrounds. Figure 5.12 illustrates two examples that similar skies are retrieved when a user preferred reference sky image is given. This relevance-feedback scheme facilitates users to find preferred stylized sky backgrounds, while ensuring the quality and realism of rendered images.

(a) Input image

(b) Preferred sky

(c) Results

Figure 5.12: Results of appearance-guided sky replacement. Given the input image (a) and one preferred sky style (b), our system is able to find other similar skies (c) in the database. We show two sets of sky replacement results in each row, where each set includes the result of preferred sky style and the other three results that have the similar sky appearance to the preferred one.

Runtime Performance. We measure the runtime of the proposed algorithm on a desktop computer with 3.4GHz Core Xeon CPU, and normalize all the images with the maximum width or height equal to 800 pixels. Implemented in MATLAB, it takes 12 seconds (0.1 seconds with a Titan X GPU and 12GB memory) for scene parsing and generating the FCN semantic responses, and 4 seconds to refine the segmentation results. In addition, it takes 0.5 seconds to retrieve a sky image, and 4 seconds to match a region (where there are usually 2 to 5 regions in an input image) with the C++ implementation. The runtime performance can be improved with high-performance programming languages and code optimization.

Limitation. While the proposed algorithm considers scene semantics in the sky replacement process, it does not take lighting conditions into account. As a result, it is less effective for images with strong directional lighting or high-level cues like shadow directions and reflections. Figure 5.13 shows one example where the proposed method does

(a) Input image (b) Our result

Figure 5.13: An example showing the limitation of our method. Without knowing the strong light source, our method is not able to remove the reflection area.

not perform well. One solution to address these issues is to estimate the sunlight direction and perform shadow detection [49]. Such information can be used as prior during the sky search step to ensure that images with similar sunlight directions are retrieved, which will be addressed in our future work.

5.7 Summary

In this work, we propose an automatic method that utilizes semantic information for rendering images with stylized sky backgrounds. We present an accurate sky segmentation algorithm that is effective in delineating boundaries between foreground and background

regions. To find proper images for replacement, we construct a new database that contains various scenes and skies, and search images with similar semantic content. During the sky replacement step, we show that our semantic-aware transfer method can generate realistic results compared to existing approaches. We exploit semantic information through each component of this work, and show that it facilitates the rendering process. For future work, it is of great interest to explore how to utilize semantic cues for image editing problems such as scene completion or photo re-coloring.

Chapter 6

Deep Semantic-guided Image Harmonization

6.1 Introduction

Compositing is one of the most common operations in image editing. To generate a composite image, a foreground region in one image is extracted and combined with the background of another image. However, the appearances of the extracted foreground region may not be consistent with the new background, making the composite image unrealistic. Therefore, it is essential to adjust the appearances of the foreground region to make it compatible with the new background (Figure 6.1). Previous techniques improve the realism of composite images by transferring statistics of hand-crafted features, including color [48, 116] and texture [87], between the foreground and background regions. However, these techniques do not take the contents of the composite images into account, leading to unreliable results when appearances of the foreground and background regions are vastly different.

In this work, we propose a learning-based method by training an end-to-end deep convolutional neural network (CNN) for image harmonization, which can capture both the

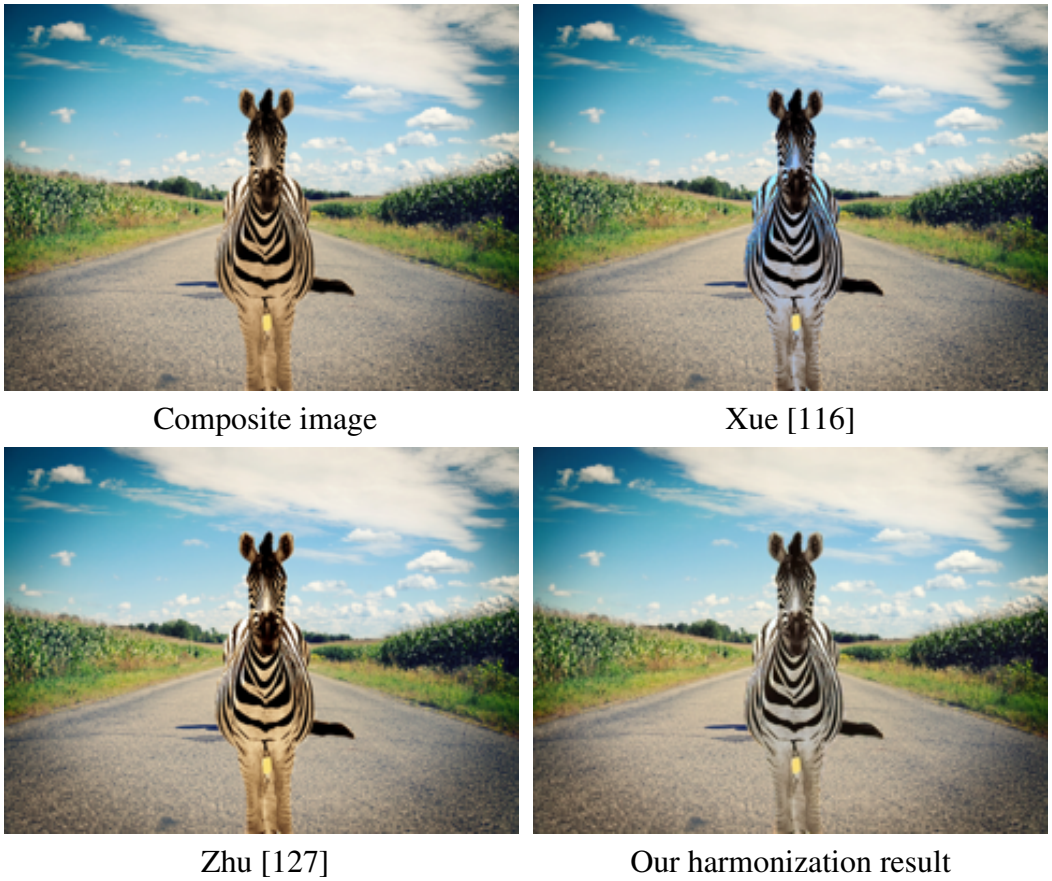


Figure 6.1: Our method can adjust the appearances of the composite foreground to make it compatible with the background region. Given a composite image, we show the harmonized images generated by [116], [127] and our deep harmonization network.

context and semantic information of the composite images during harmonization. Given a composite image and a foreground mask as the input, our model directly outputs a harmonized image, where the contents are the same as the input but with adjusted appearances on the foreground region. Context information has been utilized in several image editing tasks, such as image enhancement [34, 117], image editing [99] and image inpainting [67]. For image harmonization, it is critical to understand what it looks like in the surrounding background region near the foreground region. Hence foreground appearances can be ad-

justed accordingly to generate a realistic composite image. Toward this end, we train a deep CNN model that consists of an encoder to capture the context of the input image and a decoder to reconstruct the harmonized image using the learned representations from the encoder.

In addition, semantic information is of great importance to improve image harmonization. For instance, if we know the foreground region to be harmonized is a sky, it is natural to adjust the appearance and color to be blended with the surrounding contents, instead of making the sky green or yellow. However, the above-mentioned encoder-decoder does not explicitly model semantic information without the supervision of high-level semantic labels. Hence, we incorporate another decoder to provide scene parsing of the input image, while sharing the same encoder for learning feature representations. A joint training scheme is adopted to propagate the semantic information to the harmonization decoder. With such semantic guidance, the harmonization process not only captures the image context but also understands semantic cues to better adjust the foreground region.

Training an end-to-end deep CNN requires a large-scale training set including various and high-quality samples. However, unlike other image editing tasks such as image colorization [121] and inpainting [67] where unlimited amount of training data can be easily generated, it is relatively difficult to collect a large-scale training set for image harmonization, as generating composite images and ground truth harmonized output requires professional editing skills and a considerable amount of time. To solve this problem, we develop a training data generation method that can synthesize large-scale and high-quality training pairs, which facilitates the learning process.

To evaluate the proposed algorithm, we conduct extensive experiments on synthesized and real composite images. We first quantitatively compare our method with different settings to other existing approaches for image harmonization on our synthesized dataset, where the ground truth images are provided. We then perform a user study on real composite images and show that our model trained on the synthesized dataset performs favorably in real cases.

The contributions of this work are as follows. First, to the best of our knowledge, this is the first attempt to have an end-to-end learning approach for image harmonization. Second, we demonstrate that our joint CNN model can effectively capture context and semantic information, and can be efficiently trained for both the harmonization and scene parsing tasks. Third, an efficient method to collect large-scale and high-quality training images is developed to facilitate the learning process for image harmonization.

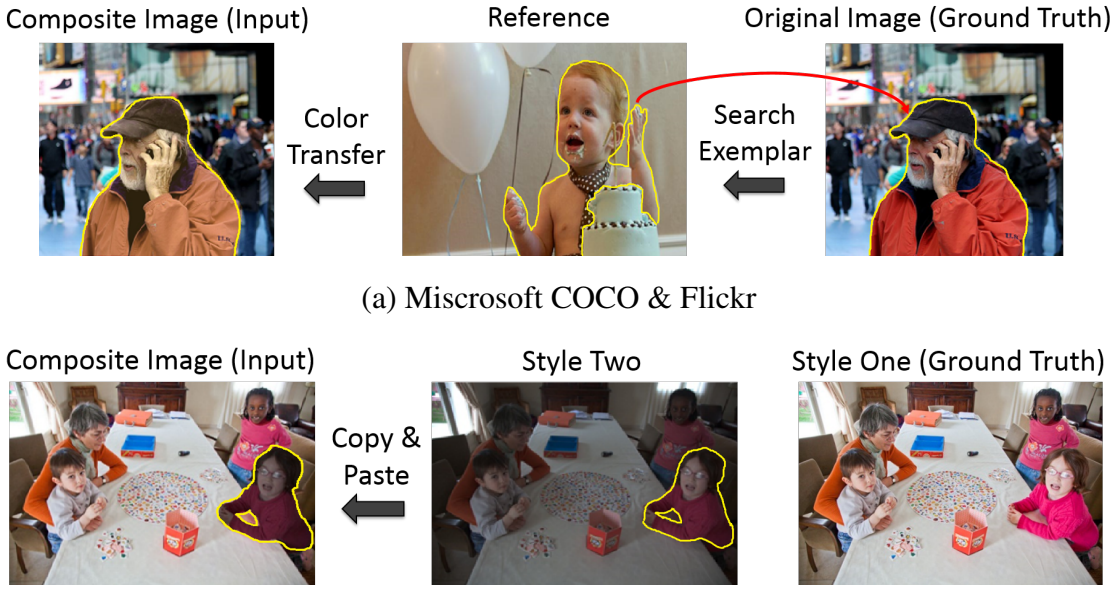
6.2 Deep Image Harmonization

In this section, we describe the details of our proposed end-to-end CNN model for image harmonization. Given a composite image and a foreground mask as the input, our model outputs a harmonized image by adjusting foreground appearances while retaining the background region. Furthermore, we design a joint training process with scene parsing to understand image semantics and thus improve harmonization results. Figure 6.3 shows an overview of the proposed CNN architecture. Before describing this network, we first introduce a data collection method that allows us to obtain large-scale and high-quality training pairs.

6.2.1 Data Acquisition

Data acquisition is an essential step to successfully train a CNN. As described above, an image pair containing the composite and harmonized images is required as the input and ground truth for the network. Unlike other unsupervised learning tasks such as [121, 67] that can easily obtain training pairs, image harmonization task requires expertise to generate a high-quality harmonized image from a composite image, which is not feasible to collect large-scale training data.

To address this issue, we start from a real image which we treat as the output ground truth of our network. We then select a region (e.g., an object or a scene) and edit its



(a) Miscrosoft COCO & Flickr

(b) MIT-Adobe FiveK

Figure 6.2: Data acquisition methods. We illustrate the approaches for collecting training pairs for the datasets (a) Miscrosoft COCO and Flickr via color transfer, and (b) MIT-Adobe FiveK with different styles.

appearances to generate an edited image which we use as the input composite image to the network. The overall process is described in Figure 6.2. This data acquisition method ensures that the ground truth images are always realistic so that the goal of the proposed CNN is to directly reconstruct a realistic output from a composite image. In the following, we introduce the details of how we generate our synthesized dataset.

Images with Segmentation Masks. We first use the Microsoft COCO dataset [57], where the object segmentation masks are provided for each image. To generate synthesized composite images, we randomly select an object and edit its appearances via a color transfer method. In order to ensure that the edited images are neither arbitrary nor unrealistic in color and tone, we construct the color transfer functions by searching for proper reference objects.

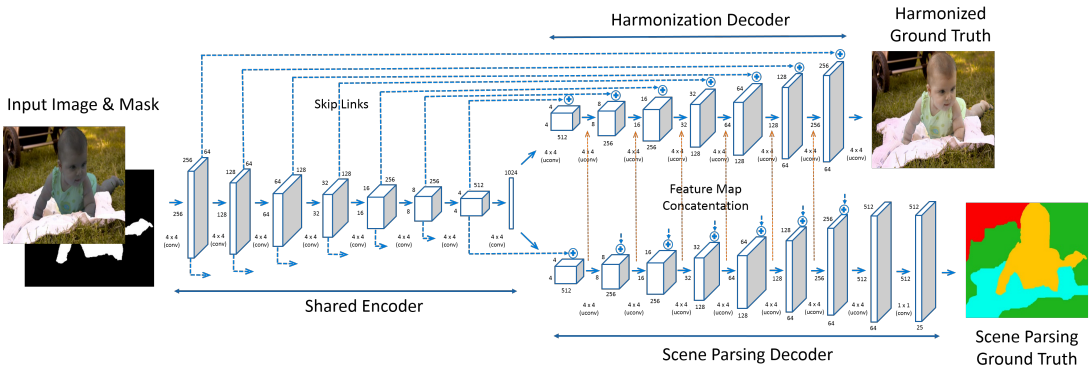


Figure 6.3: The overview of the proposed joint network architecture. Given a composite image and a provided foreground mask, we first pass the input through an encoder for learning feature representations. The encoder is then connected to two decoders, including a harmonization decoder for reconstructing the harmonized output and a scene parsing decoder to predict pixel-wise semantic labels. In order to use the learned semantics and improve harmonization results, we concatenate the feature maps from the scene parsing decoder to the harmonization decoder (denoted as dot-orange lines). In addition, we add skip links (denoted as blue-dot lines) between the encoder and decoders for retaining image details and textures. Note that, to keep the figure clean, we only depict the links for the harmonization decoder, while the scene parsing decoder has the same skip links connected to the encoder.

Specifically, given a target image and its corresponding object mask, we search a reference image which contains the object with the same semantics. We then transfer the appearance from the reference object to the target object. As such, we ensure that the edited object still looks plausible but does not match the background context. For color transfer, we compute statistics of the luminance and color temperature, and use the histogram matching method [54].

To generate a larger variety of transferred results, we apply different transfer parameters for both the luminance and color temperature on one image, so that our learned

Table 6.1: Number of training and test images on three synthesized datasets.

	MSCOCO	MIT-Adobe	Flickr
Training set	51187	4086	4720
Test set	3842	68	96

network can adapt to different scenarios in real cases. In addition, we apply an aesthetics prediction model [43] to filter out low-quality images. An example of generated synthesized input and output pairs are shown in Figure 6.2(a).

Images with Different Styles. Although the Microsoft COCO dataset provides us with rich object categories, it is still limited to certain objects. To cover more object categories, we augment it with the MIT-Adobe FiveK dataset [10]. In this dataset, each original image has another 5 different styles that are re-touched by professional photographers using Adobe Lightroom, resulting in 6 editions of the same image. To edit the original image, we begin with one randomly selected style and manually segment a region. We then crop this segmented region and overlay on the image with another style to generate the synthesized composite image. An example set is presented in Figure 6.2(b).

Flickr Images with Diversity. Since images in the MIT-Adobe FiveK and Microsoft COCO datasets only contain certain scenes and styles, we collect a dataset from Flickr with larger diversity such as images containing different scenes or stylized images. To generate input and ground truth pairs, we apply the same color transfer technique described for the Microsoft COCO dataset. However, since there is no semantic information provided in this dataset to search proper reference objects for transfer, we use a pre-trained scene parsing model [124] to predict semantic pixel-wise labels. We then compute a spatial-pyramid label histogram [52] of the target image and retrieve reference images from the ADE20K dataset [124] with similar histograms computed from the ground truth annotations.

Next, we manually segment a region (e.g., an object or a scene) in the target image.

Based on the predicted scene parsing labels within the segmented target region, we find a region in the reference image that shares the same labels as the target region. The composite image is then generated by the color transfer method mentioned above (Figure 6.2(a)).

Discussions. With the above-mentioned data acquisition methods on three datasets, we are able to collect large-scale and high-quality training and test pairs (see Table 6.1 for a summarization). This enables us to train an end-to-end CNN for image harmonization with several benefits. First, our data collection method ensures that the ground truth images are realistic, so the network can really capture the image realism and adjust the input image according to the learned representations.

Another merit of our method is to enable quantitative evaluations. This is, we can use the synthesized composite image to measure errors by comparing to the ground truth images. Although there should be no single best solution for the image harmonization task, this quantitative measurement can give us a sense of how closer the images generated by different methods are, to a truly realistic image (discussed in Section 6.3), which is not addressed by previous approaches.

6.2.2 Context-aware Encoder-decoder

Motivated by the potential of the Context Encoders [67], our CNN learns feature representations of input images via an encoder and reconstruct the harmonized output results through a decoder. While the proposed deep network bears some resemblance, we add novel components for image harmonization. In the following, we present the objective function and proposed network architecture with discussion of novel components.

Objective Function. Given a RGB image $I \in \mathbb{R}^{H \times W \times 3}$ and a provided binary mask $M \in \mathbb{R}^{H \times W \times 1}$ of the composite foreground region, we form the input $X \in \mathbb{R}^{H \times W \times 4}$ by concatenating I and M, where H and W are image dimensions. Our objective is to predict an output image $\hat{Y} = \mathcal{F}(X)$ that optimizes the reconstruction ($L2$) loss with respect to the

ground truth image Y:

$$\mathcal{L}_{rec}(X) = \frac{1}{2} \sum_{h,w} \parallel Y_{h,w} - \hat{Y}_{h,w} \parallel_2^2 . \tag{6.1}$$

Since the $L2$ loss is optimized with the mean of the data distribution, the results are often blurry and thus miss important details and textures from the input image. To overcome these problems, we show that adding skip links from the encoder to the decoder can recover those image details in the proposed network.

Network Architecture. Figure 6.3 shows basic components of our network architecture with an encoder and a harmonization decoder. The encoder is a series of convolutional layers and a fully connected layer to learn feature representations from low-level image details to high-level context information. Note that as we do not have any pooling layers, fine details are preserved in the encoder [67]. The decoder is a series of deconvolutional layers which aim to reconstruct the image via up-sampling from the representations learned in the encoder and simultaneously adjust the appearances of the foreground region.

However, image details and textures may be lost during the compression process in the encoder, and thus there is less information to reconstruct the contents of the input image. To retain those details, it is crucial that we add a skip link from each convolutional layer in the encoder to each corresponding deconvolutional layer in the decoder. We show this method is effectively useful without adding additional burdens for training the network. Furthermore, it can alleviate the problem of the $L2$ loss that prefers a blurry image solution.

Implementation Details. We implement the proposed network in Caffe [39] and use the stochastic gradient descent solver for optimization with a fixed learning rate 10^{-8}. In addition, we compute the loss on the entire image rather than the foreground mask to account for the reconstruction differences in the background region. We also try a weighted loss that considers the foreground region more important, but the results are similar and thus we use a simple loss function. Since the entire network is trained from

scratch, we use the batch normalization [36] followed by a scaling layer and an ELU layer [19] after each convolutional and deconvolutional layers to facilitate the training process.

Discussions. We conduct experiments using the proposed network architecture with different input sizes. Interestingly, we find that the one with larger input size performs better in practice, and thus we use input resolution of 512×512. This observation also matches our intuition when designing the encoder-decoder architecture with skip links, where the network can learn more context information and details from a larger input image. To generate higher resolution results, we can up-sample the output of the network with joint bilateral filtering [69], in which the input composite image is used as the guidance to keep clear details and sharp textures.

6.2.3 Joint Training with Semantics

In the previous section, we propose an encoder-decoder network architecture for image harmonization. In order to further improve harmonization results, it is natural to consider the semantics of the composite foreground region. The ensuing question is how to incorporate such semantics in our CNN, so that the entire network is still end-to-end trainable. In this section, we propose a modified network that can jointly train the image harmonization and scene parsing tasks simultaneously, while propagating semantics to improve harmonization results. The overall architecture is depicted in Figure 6.3, which adds the scene parsing decoder branch.

Joint Loss. In addition to the reconstruction loss described for image harmonization in (6.1), we introduce a pixel-wise cross-entropy loss with the standard softmax function $\mathbb{E}$ for scene parsing:

$$\mathcal{L}_{cro}(X) = -\sum_{h,w} \log(\mathbb{E}(X_{h,w};\theta)). \tag{6.2}$$

We then define a combined loss for both tasks and optimize it jointly:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{cro}, \tag{6.3}$$

where λ_i is the weight to control the balance between losses for image harmonization and scene parsing.

Network Architecture. We design the joint network by inheriting the encoder-decoder architecture described in the previous section. Specifically, we add a decoder to predict scene parsing results, while the encoder is to learn feature representations and is shared for both decoders. To extract semantic knowledge from the scene parsing model and help harmonization process, we concatenate feature maps from each deconvolutional layer of the scene parsing decoder to the harmonization decoder, except for the last layer which focuses on image reconstruction. In addition, skips links [61] are also connected to the scene parsing decoder to gain more information from the encoder.

Implementation Details. To enable the training process for the proposed joint network, both the ground truth images for harmonization and scene parsing are required. We then use a subset of the ADE20K dataset [124], which contains 12080 training images with the top 25 frequent labels. Similarly, training pairs for harmonization are obtained in a way described in the data acquisition section via color transfer.

To train the joint network, we start with the training data from the ADE20K dataset to obtain an initial solution for both the harmonization and scene parsing by optimizing (6.3). We set $\lambda_1 = 1$ and $\lambda_2 = 100$ with a fixed learning rate 10^{-8}. Next, we fix the scene parsing decoder with $\lambda_2 = 0$ and finetune the rest of the network using all the training data introduced in Section 6.2.1 to achieve the optimal solution for image harmonization. Note that, during this finetuning step, the scene parsing decoder is able to propagate learned semantic information through the links between two decoders.

Discussions. With the incorporated scene parsing model, our network can learn the color distribution of certain semantic categories, e.g., the skin color on human or the sky-like colors. In addition, the learned background semantics can help identify which region to match for better foreground adjustment. During harmonization, it essentially uses these

Table 6.2: Comparisons of methods with mean-squared errors (MSE) on three synthesized datasets.

	MSCOCO	MIT-Adobe	Flickr
cut-and-paste	400.5	552.5	701.6
Lalonde [48]	667.0	1207.8	2371.0
Xue [116]	351.6	568.3	785.1
Zhu [127]	322.2	360.3	475.9
Ours (w/o semantics)	80.5	168.8	491.7
Ours	**76.1**	**142.8**	**406.8**

learned semantic priors to improve the realism of output results. Moreover, the incorporation of semantic information through joint training not only helps our image harmonization task, but also can be adopted to benefit other image editing tasks [121, 67].

To validate our scene parsing model, we compare the proposed joint network to a deeplab model [13], MSc-COCO-LargeFOV, that has a similar model capacity and size to our model but is initialized from a pre-trained model for semantic segmentation. We evaluate the scene parsing results on the validation set of the ADE20K dataset with the top 25 frequent labels. The mean intersection-over-union (IoU) accuracy of our joint network is 32.2, while the MSc-COCO-LargeFOV model achieves IoU as 36.0. Although our model is not specifically designed for scene parsing and is learned from scratch, it shows that our method performs competitively against a state-of-the-art model for semantic segmentation.

6.3 Experimental Results

We present the main results on image harmonization with comparisons to the state-of-the-art methods in this section. More results and analysis can be found in the supplementary material.

Table 6.3: Comparisons of methods with PSNR scores on three synthesized datasets.

	MSCOCO	MIT-Adobe	Flickr
cut-and-paste	26.3	23.9	25.9
Lalonde [48]	22.7	21.1	18.9
Xue [116]	26.9	24.6	25.0
Zhu [127]	26.9	25.8	25.4
Ours (w/o semantics)	32.2	27.5	27.2
Ours	**32.9**	**28.7**	**27.4**

Synthesized Data. We first evaluate the proposed method on our synthesized dataset for quantitative comparisons. Table 6.2 and 6.3 show the results of mean-squared errors (MSE) and PSNR scores between the ground truth and harmonized image. Note that it is the first quantitative evaluation on image harmonization, which reflects how close different results are to realistic images. We show that our joint network consistently achieves better performance compared to the single network without combining scene parsing decoder and other state-of-the-art algorithms [48, 116, 127] on all three synthesized datasets in terms of MSE and PSNR. In addition, it is also worth noticing that our baseline network without semantics already outperforms other existing methods.

In Figure 6.4, we show visual comparisons with respect to PSNR of the harmonization results generated from different methods. Overall, the harmonized images by the proposed methods are more realistic and closer to the ground truth images, with higher PSNR values. In addition, Figure 6.5 presents one comparison of our networks with and without incorporating the scene parsing decoder. With semantic understandings, our joint network is able to harmonize foreground regions according to their semantics and produce realistic appearance adjustments, while the one without semantics may generate unsatisfactory results in some cases.

Real Composite Images. To evaluate the effectiveness of the proposed joint network in real scenarios, we create a test set of 52 real composite images and combine 48 examples

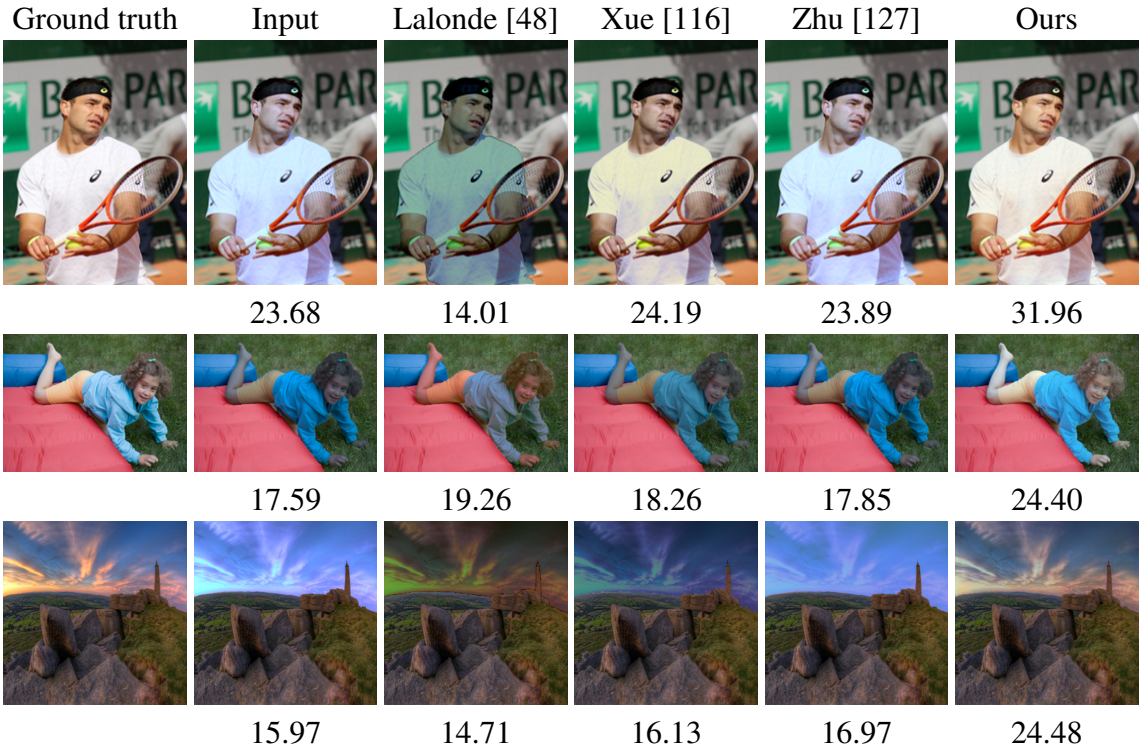


Figure 6.4: Example results on synthesized datasets for the input, ground truth, three state-of-the-art methods and our proposed network. From the first row to the third one, we show one example for the MSCOCO, MIT-Adobe and Flickr datasets. Each result is associated with a PSNR score. Among all the methods, our harmonization results obtain the highest score.

from Xue et al. [116], resulting in a total of 100 high-quality composite images. To cover a variety of real examples, we create composite images including various scenes and stylized images, where the composite foreground region can be an object or a scene.

We follow the same procedure as [116, 127] to set up a user study on Amazon Mechanical Turk, in which each user sees two randomly selected results at a time and is asked to choose the one that looks more realistic. For sanity checks, we use ground truth images from the synthesized dataset and heavily edited images to create easily distinguishable

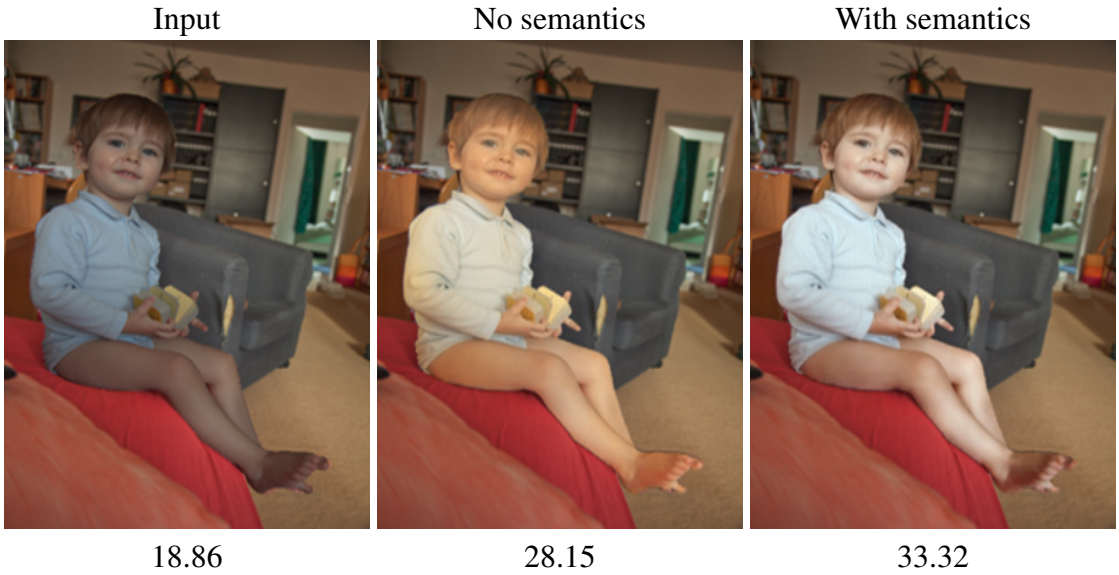
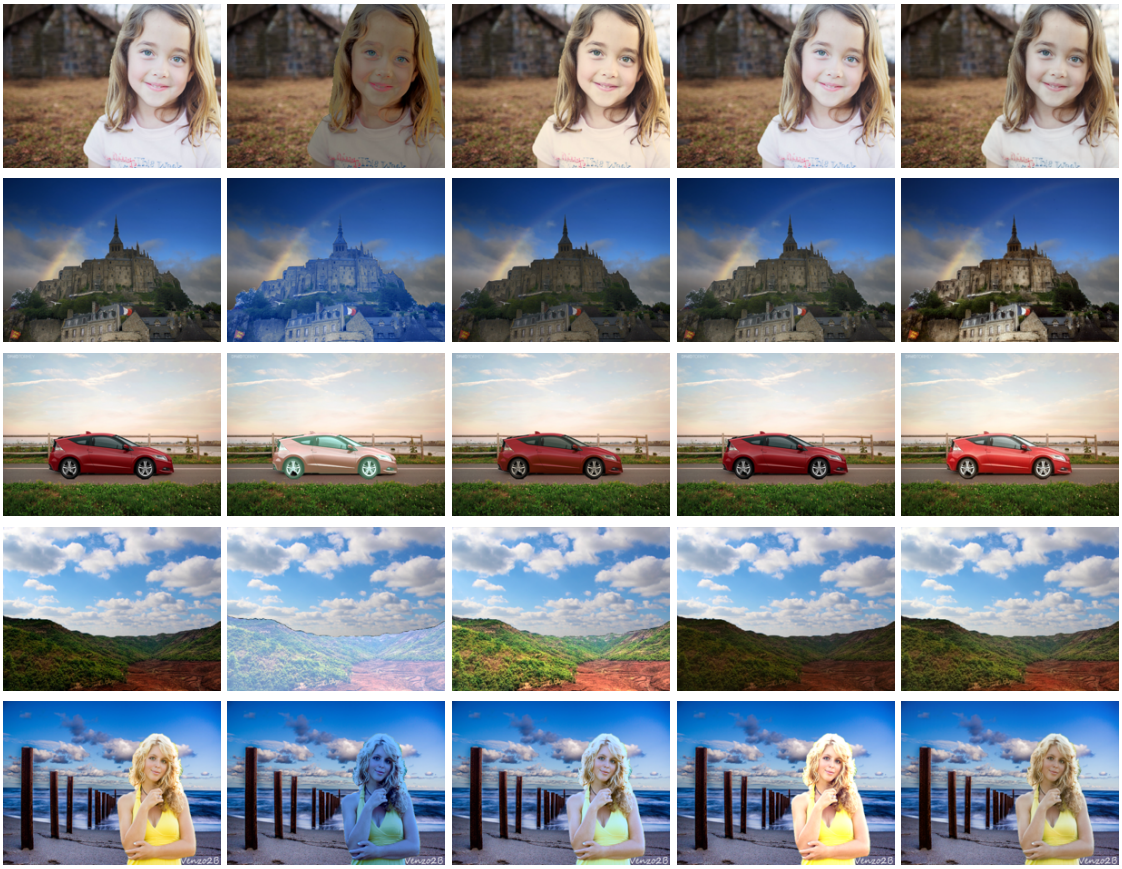


Figure 6.5: Example results to show the comparison of our network with or without incorporating semantic information. With semantics, our result can recover the skin color and obtain higher PSNR score.

Table 6.4: Comparisons of methods with B-T scores on real composite datasets.

Dataset	[116]	Our test set	Overall
cut-and-paste	1.080	1.168	1.139
Lalonde [48]	0.557	0.067	0.297
Xue [116]	1.130	0.885	1.002
Zhu [127]	0.875	0.867	0.876
Ours	**1.237**	**1.568**	**1.424**

pairs that are used to filter out bad users. As a result, a total of 225 subjects participate in this study with a total of 10773 pairwise results (10.8 results for each pair of different methods on average). After obtaining all the pairwise results, we use the Bradley-Terry model (B-T model) [7, 47] to calculate the global ranking score for each method.



Input | Lalonde [48] | Xue [116] | Zhu [127] | Ours

Figure 6.6: Example results on real composite images for the input, three state-of-the-art methods and our proposed network. We show that our method produces realistic harmonized images by adjusting composite foreground regions containing various scenes or objects.

Table 6.4 shows that our method achieves the highest B-T score in terms of realism compared to state-of-the-art approaches on both our created test set and examples from [116]. Interestingly, our method is the only one that can improve the harmonization result with a significant margin from the input image (by cut-and-paste).

Figure 6.6 shows sample harmonized images by the evaluated methods. Overall, our

(a) Input

(b) Mask

(c) Output

Figure 6.7: Given an input image (a), our network can adjust the foreground region according to the provided mask (b) and produce the output (c). In this example, we invert the mask from the one in the first row to the one in the second row, and generate harmonization results that account for different context and semantic information.

joint network produces realistic output images, which validates the effectiveness of using synthesized data to directly learn how to harmonize composite images from realistic ground truth images. The results from [116] may be easily affected by the large appearance difference between the background and foreground regions during matching. For the method [127], it may generate unsatisfactory results due to the errors introduced during realism prediction, which may affect the color optimization step. In contrast, our network adopts a single feed-forward scheme learned from a well-constructed training set, and utilizes semantic information to improve harmonization results. The complete results on the real composite test set are presented in the supplementary material.

Generalization to Background Masks. With the provided foreground mask, our network

can learn context and semantic information while transforming the composite image to a realistic output image. Therefore, our method can be applied to any foreground masks containing arbitrary objects, scenes or clutter backgrounds. Figure 6.7 illustrates one example, where originally the adjusted foreground region is the *child*. Instead, we can invert the mask and focus on harmonizing the region of *inverted child*. The result shows that our network can produce realistic outputs from different foreground masks.

Runtime Performance. Previous harmonization methods rely on matching statistics [48, 116] or optimizing an adjustment function [127], which usually require longer processing time (more than 10 seconds with a 3.4GHz Core Xeon CPU) on a 512×512 test image. In contrast, our proposed CNN is able to harmonize an image in 0.05 seconds with a Titan X GPU and 12GB memory, or 3 seconds with a CPU.

6.4 Summary

In this work, we present a novel network that can capture both the context and semantic information for image harmonization. We demonstrate that our joint network can be trained in an end-to-end manner, where the semantic decoder branch can effectively provide semantics to help harmonization. In addition, to facilitate the training process, we develop an efficient method to collect large-scale and high-quality training pairs. Experimental results show that our method performs favorably on both the synthesized datasets and real composite images against other state-of-the-art algorithms.

Chapter 7

Conclusion and Future Work

7.1 Summary

This thesis investigates problems related to visual analysis and synthesis, specifically in video object segmentation and image editing tasks. To address these challenging problems, we incorporate the temporal information and semantics to better understand visual contents in images and videos. Chapter 3 introduces a video object segmentation method that utilizes a joint model for both the segmentation and flow estimation. Flow connects segments temporally and provides motion cues of objects, while segmentation can benefit flow estimations with the guidance of accurate object boundaries. Toward this end, we formulate a principal and spatial-temporal model that iteratively updates segmentation and flow results. The optimized objective can thus predict temporally-smooth object segmentation and complete optical flow, especially around boundaries.

In Chapter 4, we take a further step by considering a collection of videos simultaneously, and segment the objects while understanding their visual semantics. The proposed co-segmentation framework first generates semantic tracklets with temporally consistent segments, and then discovers true objects within these tracklets. We construct a submodular function that takes semantics into account and model relations between all the tracklets.

By optimizing this function, we exploit semantic cues from the video collection, and hence true objects with higher mutual similarities can be discovered.

Starting from Chapter 5, we show two examples of image editing tasks guided by semantics. We first present a work of sky replacement method that can automatically generate a set of realistic and diverse results. The key component is to learn a scene parsing model that can predict pixel-wise semantic categories, and utilize such contents to guide the entire process including sky segmentation, search and replacement. We demonstrate that our algorithm can adjust the foreground appearance to fit the statistics of the new sky, and hence produce visually pleasing results.

In Chapter 6, we present an image harmonization algorithm for generic objects and scenes. Again, we utilize semantic cues by jointly learning an end-to-end deep network for scene parsing and harmonization. In order to train the model, we design a method to synthesize input and ground truth pairs efficiently. With the success of learning the model, we show that our network can handle real composite images with a significant improvement over the other state-of-the-art methods in terms of visual quality and accuracy. It proves that our models for visual analysis and synthesis tasks can be jointly learned and benefit each other.

7.2 Future work

Along the research direction of visual analysis and synthesis problems, we present three interesting topics for future development.

7.2.1 Learning-based Video Object Segmentation

To further improve the efficiency and accuracy for video object segmentation, one potential solution is to utilize learning-based deep networks that take advantage of a large scale data and annotations. For instance, following the idea in Chapter 3, either segmen-

tation or optical flow can be predicted using a single deep network. Since both networks contain similar object feature representations, where the ones from segmentation focus on objectness and the ones from optical flow exploit motions, it is reasonable to design a joint model that can communicate to each other, e.g., feature propagations between two networks.

Another interesting direction is to learn a segmentation network that can exploit temporal information in videos. Recently, most developed models only take a single image as the input at a time without learning temporal representations. However, such temporal cues are important to maintain smooth results in videos. Therefore, the problem of learning temporal representations would be critical for future applications in videos.

7.2.2 Semi-supervised Semantic Segmentation

Based on the discussion above for feature learning in the temporal domain, it would require a huge amount of annotations in videos. However, collecting such pixel-wise annotations is usually labor-intensive and may result in low-quality outcomes. Hence, one interesting research topic is to adopt semi-supervised learning for segmentation. Without annotations in every image, the designed system would need to self-explore useful information contained in unlabeled images. One can imagine that it would be challenging due to noise, but on the other hand, the model would not be limited to certain image spaces, and can exploit diverse data and evolve itself.

One idea to achieve this is to utilize an adversarial learning scheme [26], where a discriminator can be learned to distinguish ground truth and predicted segmentations using annotated images. Next, when feeding unlabeled images, this discriminator can recognize trust-worthy predicted regions and extract useful cues from these regions to improve the model.

7.2.3 Video-based Editing and Applications

In addition to image editing tasks shown in this thesis, video-based applications are more challenging. For instance, either replacing the background or harmonizing images needs to consider temporal consistency for generating realistic results. Another example is the generative task [72], where image generation has been studied widely, but video generation still remains a difficult problem. Therefore, understanding spatial-temporal cues is critical for these video-based tasks. Borrowing the ideas presented in this thesis, it would be interesting to utilize semantic and motion information in videos for creating/editing content and generating realistic results. In future work, developing such systems that incorporate visual analysis and synthesis would be interesting, where semantic cues can be utilized to jointly improve both tasks in videos.

Bibliography

[1] V. Badrinarayanan, F. Galasso, and R. Cipolla. Label propagation in video sequences. In *CVPR*, 2010.

[2] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *ICCV*, 2007.

[3] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *PAMI*, 35(8):1798 – 1828, 2013.

[4] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S. K. Nayar. Face Swapping: Automatically Replacing Faces in Photographs. *ACM Trans. Graph. (proc. SIGGRAPH)*, 2008.

[5] M. J. Black and A. Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *PAMI*, 18(10):972–986, Oct. 1996.

[6] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, pages 1124–1137, 2004.

[7] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39:324–345, 1952.

[8] W. Brendel and S. Todorovic. Video object segmentation by tracking regions. In *ICCV*, 2009.

[9] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *PAMI*, 33(3):500–13, 2011.

[10] V. Bychkovsky, S. Paris, E. Chan, and F. Durand. Learning photographic global tonal adjustment with a database of input/output image pairs. In *CVPR*, 2011.

[11] J. Chang and J. W. Fisher. Topology-constrained layered tracking with latent flow. In *ICCV*, 2013.

[12] J. Chang, D. Wei, and J. W. Fisher. A video representation using temporal superpixels. In *CVPR*, 2013.

[13] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

[14] X. Chen, A. Shrivastava, and A. Gupta. Enriching visual knowledge bases via object discovery and segmentation. In *CVPR*, 2014.

[15] Z. Chen, H. Jin, Z. Lin, S. Cohen, and Y. Wu. Large displacement optical flow from nearest neighbor fields. In *CVPR*, 2013.

[16] W. C. Chiu and M. Fritz. Multi-class video co-segmentation with a generative multi-video model. In *CVPR*, 2013.

[17] 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 *CVPR*, 2015.

[18] P. Chockalingam, N. Pradeep, and S. Birchfield. Adaptive fragments-based tracking of non-rigid objects using level sets. In *ICCV*, 2009.

[19] D. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *ICLR*, 2016.

[20] K. Dale, M. K. Johnson, K. Sunkavalli, W. Matusik, and H. Pfister. Image restoration using online photo collections. In *ICCV*, 2009.

[21] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen. Image melding: Combining inconsistent images using patch-based synthesis. *ACM Trans. Graph. (proc. SIGGRAPH)*, 31(4), 2012.

[22] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[23] H. Fu, D. Xu, B. Zhang, and S. Lin. Object-based multiple foreground video co-segmentation. In *CVPR*, 2014.

[24] F. Galasso, N. Nagaraja, T. Cardenas, T. Brox, and B. Schiele. A unified video segmentation benchmark: Annotation, metrics and analysis. In *ICCV*, 2013.

[25] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *ICCV*, 2011.

[26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[27] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010.

[28] J. Guo, L.-F. Cheong, R. T. Tan, and S. Z. Zhou. Consistent foreground co-segmentation. In *ACCV*, 2014.

[29] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *ACM Trans. Graph. (proc. SIGGRAPH)*, 30(4), 2011.

[30] G. Hartmann, M. Grundmann, J. Hoffman, D. Tsai, V. Kwatra, O. Madani, S. Vijayanarasimhan, I. Essa, J. Rehg, and R. Sukthankar. Weakly supervised learning of object segmentations from web-scale video. In *ECCV workshop*, 2012.

[31] J. Hays and A. A. Efros. Scene completion using millions of photographs. *ACM Trans. Graph. (proc. SIGGRAPH)*, 26(3), 2007.

[32] K. He, J. Sun, and X. Tang. Guided image filtering. *PAMI*, 35(6):1397–1409, 2013.

[33] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[34] S. J. Hwang, A. Kapoor, and S. B. Kang. Context-based automatic local image enhancement. In *ECCV*, 2012.

[35] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Trans. Graph. (proc. SIGGRAPH)*, 35(4), 2016.

[36] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[37] S. D. Jain and K. Grauman. Supervoxel-consistent foreground propagation in video. In *ECCV*, 2014.

[38] A. Jepson and M. J. Black. Mixture models for optical flow computation. In *CVPR*, 1993.

[39] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[40] M. K. Johnson, K. Dale, S. Avidan, H. Pfister, W. T. Freeman, and W. Matusik. Cg2real: Improving the realism of computer generated images using a large collection of photographs. *IEEE Trans. Vis. Comp. Graph.*, 17(9), 2011.

[41] N. Jojic and B. Frey. Learning flexible sprites in video layers. In *CVPR*, 2001.

[42] A. Joulin, F. Bach, and J. Ponce. Multi-class cosegmentation. In *CVPR*, 2012.

[43] S. Kong, X. Shen, Z. Lin, R. Mech, and C. Fowlkes. Photo aesthetics ranking network with attributes and content adaptation. In *ECCV*, 2016.

[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[45] S. Kwak, M. Cho, I. Laptev, J. Ponce, and C. Schmid. Unsupervised object discovery and tracking in video collections. In *ICCV*, 2015.

[46] P.-Y. Laffont, Z. Ren, X. Tao, C. Qian, and J. Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Trans. Graph. (proc. SIGGRAPH)*, 33(4), 2014.

[47] W.-S. Lai, J.-B. Huang, Z. Hu, N. Ahuja, and M.-H. Yang. A comparative study for single image blind deblurring. In *CVPR*, 2016.

[48] J.-F. Lalonde and A. A. Efros. Using color compatibility for assessing image realism. In *ICCV*, 2007.

[49] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan. Estimating the natural illumination conditions from a single outdoor image. *IJCV*, 98(2):123–145, 2011.

[50] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi. Photo clip art. *ACM Trans. Graph. (proc. SIGGRAPH)*, 26(3), 2007.

[51] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. In *ECCV*, 2016.

[52] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[53] N. Lazic, I. Givoni, B. Frey, and P. Aarabi. Floss: Facility location for subspace segmentation. In *ICCV*, 2009.

[54] J.-Y. Lee, K. Sunkavalli, Z. Lin, X. Shens, and I. S. Kweon. Automatic content-aware color and tone stylization. In *CVPR*, 2016.

[55] Y. J. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In *ICCV*, 2011.

[56] F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, 2013.

[57] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[58] S. Liu, J. Pan, and M.-H. Yang. Learning recursive filters for low-level vision via a hybrid neural network. In *ECCV*, 2016.

[59] X. Liu, D. Tao, M. Song, Y. Ruan, C. Chen, and J. Bu. Weakly supervised multiclass video segmentation. In *CVPR*, 2014.

[60] Y. Liu, M. Cohen, M. Uyttendaele, and S. Rusinkiewicz. Autostyle: Automatic style transfer from image collections to users image. *Comp. Graph. Forum*, 33(4), 2014.

[61] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[62] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015.

[63] T. Ma and L. J. Latecki. Maximum weight cliques with mutex constraints for video object segmentation. In *CVPR*, 2012.

[64] N. S. Nagaraja, F. Schmidt, and T. Brox. Video segmentation with just a few strokes. In *ICCV*, 2015.

[65] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 36(6):1187–1200, 2014.

[66] A. Papazoglou and V. Ferrari. Fast object segmentation in unconstrained video. In *ICCV*, 2013.

[67] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders : Feature learning by inpainting. In *CVPR*, 2016.

[68] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. Graph. (proc. SIGGRAPH)*, 22(3), 2003.

[69] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama. Digital photography with flash and no-flash image pairs. *ACM Trans. Graph. (proc. SIGGRAPH)*, 23(3), 2004.

[70] F. Pitié and A. Kokaram. The linear monge-kantorovitch linear colour mapping for example-based colour transfer. In *CVMP*, 2007.

[71] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012.

[72] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[73] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley. Color transfer between images. *IEEE Comp. Graph. Appl.*, 21(5):34–41, 2001.

[74] X. Ren and J. Malik. Tracking as repeated figure/ground segmentation. In *CVPR*, 2007.

[75] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph. (proc. SIGGRAPH)*, 23(3), 2004.

[76] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, 2013.

[77] J. C. Rubio, J. Serrat, L. Antonio, and N. Paragios. Unsupervised co-segmentation through region matching. In *CVPR*, 2012.

[78] J. C. Rubio, J. Serrat, and A. López. Video co-segmentation. In *ACCV*, 2012.

[79] E. Shahrian, D. Rajan, B. Price, and S. Cohen. Improving image matting using comprehensive sampling sets. In *CVPR*, 2013.

[80] Y. Shih, S. Paris, F. Durand, and W. T. Freeman. Data-driven hallucination of different times of day from a single outdoorphoto. *ACM Trans. Graph. (proc. SIGGRAPH Asia)*, 32(6), 2013.

[81] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556:1187–1200, 2014.

[82] D. Sun, C. Liu, and H. Pfister. Local layering for joint motion estimation and occlusion detection. In *CVPR*, 2013.

[83] D. Sun, S. Roth, and M. J. Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *IJCV*, 106(2):115–137, 2014.

[84] D. Sun, E. B. Sudderth, and M. J. Black. Layered segmentation and optical flow estimation over time. In *CVPR*, 2012.

[85] D. Sun, J. Wulff, E. B. Sudderth, H. Pfister, and M. J. Black. A fully-connected layered model of foreground and background flow. In *CVPR*, 2013.

[86] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*, 2010.

[87] K. Sunkavalli, M. K. Johnson, W. Matusik, and H. Pfister. Multi-scale image harmonization. *ACM Trans. Graph. (proc. SIGGRAPH)*, 29(4), 2010.

[88] Y.-W. Tai, J. Jia, and C.-K. Tang. Local color transfer via probabilistic segmentation by expectation-maximization. In *CVPR*, 2005.

[89] K. Tang, A. Joulin, L.-J. Li, and L. Fei-Fei. Co-localization in real-world images. In *CVPR*, 2014.

[90] K. Tang, R. Sukthankar, J. Yagnik, and L. Fei-Fei. Discriminative segment annotation in weakly labeled video. In *CVPR*, 2013.

[91] L. Tao, L. Yuan, and J. Sun. Skyfinder: Attribute-based sky image search. *ACM Trans. Graph. (proc. SIGGRAPH)*, 28(3), 2009.

[92] M. W. Tao, M. K. Johnson, and S. Paris. Error-tolerant image compositing. *IJCV*, 103(2):178–189, 2013.

[93] J. Tighe and S. Lazebnik. Superparsing: Scalable nonparametric image parsing with superpixels. *IJCV*, 101(2):329–349, 2013.

[94] P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated Bayesian approach to layer extraction from image sequences. *PAMI*, 23(3):297–303, Mar. 2001.

[95] A. Torralba, A. Oliva, M. Castelhano, and J. M. Henderso. Contextual guidance of attention in natural scenes: The role of global features on object search. *Psychological Review*, 113(10):766–786, 2006.

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

[97] Y.-H. Tsai, O. Hamsici, and M.-H. Yang. Adaptive region pooling for object detection. In *CVPR*, 2015.

[98] Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, X. Lu, and M.-H. Yang. Deep image harmonization. In *CVPR*, 2017.

[99] Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, and M.-H. Yang. Sky is not the limit: Semantic-aware sky replacement. *ACM Trans. Graph. (proc. SIGGRAPH)*, 35(4), 2016.

[100] Y.-H. Tsai, J. Yang, and M.-H. Yang. Decomposed learning for joint object segmentation and categorization. In *BMVC*, 2013.

[101] Y.-H. Tsai, M.-H. Yang, and M. J. Black. Video segmentation via object flow. In *CVPR*, 2016.

[102] Y.-H. Tsai, G. Zhong, and M.-H. Yang. Semantic co-segmentation in videos. In *ECCV*, 2016.

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

[104] S. Vijayanarasimhan and K. Grauman. Active frame selection for label propagation in videos. In *ECCV*, 2012.

[105] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *TIP*, 3:625–638, 1994.

[106] L. Wang, G. Hua, R. Sukthankar, J. Xue, and N. Zheng. Video object discovery and co-segmentation with extremely weak supervision. In *ECCV*, 2014.

[107] S. Wang, H. Lu, F. Yang, and M.-H. Yang. Superpixel tracking. In *ICCV*, 2011.

[108] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013.

[109] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *ICCV*, 2013.

[110] L. Wen, D. Du, Z. Lei, S. Z. Li, and M.-H. Yang. Jots: Joint online tracking and segmentation. In *CVPR*, 2015.

[111] F. Wu, W. Dong, Y. Knog, X. Mei, J.-C. Paul, and X. Zhang. Content-based coulour transfer. *Comp. Graph. Forum*, 32(1), 2013.

[112] J. Wulff and M. J. Black. Efficient sparse-to-dense optical flow estimation using a learned basis and layers. In *CVPR*, 2015.

[113] C. Xu, C. Xiong, and J. J. Corso. Streaming hierarchical video segmentation. In *ECCV*, 2012.

[114] L. Xu, J. Chen, and J. Jia. A segmentation based variational model for accurate optical flow estimation. In *ECCV*, 2008.

[115] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. In *CVPR*, 2010.

[116] S. Xue, A. Agarwala, J. Dorsey, and H. Rushmeier. Understanding and improving the realism of image composites. *ACM Trans. Graph. (proc. SIGGRAPH)*, 31(4), 2012.

[117] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu. Automatic photo adjustment using deep neural networks. *ACM Trans. Graph.*, 35(2), 2016.

[118] J. Yang, Y.-H. Tsai, and M.-H. Yang. Exemplar cut. In *ICCV*, 2013.

[119] D. Zhang, O. Javed, and M. Shah. Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. In *CVPR*, 2013.

[120] D. Zhang, O. Javed, and M. Shah. Video object co-segmentation by regulated maximum weight cliques. In *ECCV*, 2014.

[121] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *ECCV*, 2016.

[122] Y. Zhang, X. Chen, J. Li, C. Wang, and C. Xia. Semantic object segmentation via detection in weakly labeled video. In *CVPR*, 2015.

[123] G. Zhong, Y.-H. Tsai, and M.-H. Yang. Weakly-supervised video scene co-parsing. In *ACCV*, 2016.

[124] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ADE20K dataset. *CoRR*, abs/1608.05442, 2016.

[125] Y. Zhou and H. Tao. A background layer model for object tracking through occlusion. In *CVPR*, 2003.

[126] F. Zhu, Z. Jiang, and L. Shao. Submodular object recognition. In *CVPR*, 2014.

[127] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Learning a discriminative model for the perception of realism in composite images. In *ICCV*, 2015.

[128] C. L. Zitnick, N. Jojic, and S. B. Kang. Consistent segmentation for optical flow estimation. In *ICCV*, 2005.