Occlusion Boundary: A Formal Definition & Its Detection via Deep Exploration of Context

Chaohui Wang, Huan Fu, Dacheng Tao, Michael J Black

Occlusion Boundary: A Formal Definition & Its Detection via Deep Exploration of Context

Chaohui Wang*, Huan Fu*, Dacheng Tao *Fellow, IEEE,* and Michael J. Black

Abstract—Occlusion boundaries contain rich perceptual information about the underlying scene structure and provide important cues in many visual perception-related tasks such as object recognition, segmentation, motion estimation, scene understanding, and autonomous navigation. However, there is no formal definition of occlusion boundaries in the literature, and state-of-the-art occlusion boundary detection is still suboptimal. With this in mind, in this paper we propose a formal definition of occlusion boundaries for related studies. Further, based on a novel idea, we develop two concrete approaches with different characteristics to detect occlusion boundaries in video sequences via enhanced exploration of contextual information (*e.g.*, local structural boundary patterns, observations from surrounding regions, and temporal context) with deep models and conditional random fields. Experimental evaluations of our methods on two challenging occlusion boundary benchmarks (CMU and VSB100) demonstrate that our detectors significantly outperform the current state-of-the-art. Finally, we empirically assess the roles of several important components of the proposed detectors to validate the rationale behind these approaches.

Index Terms—Occlusion boundaries, convolutional neural networks, fully convolutional networks, conditional random fields

1 Introduction

Occlusions are ubiquitous in 2D images of natural scenes (Fig. 1). They are introduced in the 3D-to-2D projection process during image formation, due to overlap of the 2D extents (in the image plane) of 3D components/surfaces. In this paper, we are interested in these *occlusion boundaries*, each of which separates two 2D regions projected from two parts of scene surfaces that overlap locally in either of those two regions.

Occlusion boundary detection is of interest in computer vision, image analysis, and other related fields (*e.g.*, [1], [2], [3], [4]). On the one hand, occlusions constitute an obstacle to designing rigorous models and efficient algorithms in computer vision and image analysis. Besides the lack of information on invisible scene components, another main reason is that occlusions invalidate the assumption that two neighboring pixels in a 2D image correspond to two adjacent points lying on a common part of a 3D surface. Despite often being violated, this assumption is often made, either explicitly or implicitly, in existing methods (*e.g.*, the use of smoothness priors for aggregating spatial information in the 2D image). The localization of occlusion boundaries would, therefore, be very useful for overcoming this limitation and improving the solution in these tasks. On the other hand, since occlusion boundaries separate visible scene components from locally occluded components and usually correspond to an abrupt change in depth (along the line of sight), these boundaries contain rich perceptual information about the underlying 3D scene structure, the exploitation of which would be beneficial in various visual perception applications. For example, occlusion boundaries can serve as important cues for object discovery and segmentation (*e.g.*, [4], [5]), since an object is generally delimited from its environment by the isolation of its 3D surface. Indeed, psychologists have long studied their importance in human visual perception (*e.g.*, Biederman [6], Gibson [7]). Hence, a number of studies have been performed and various methods have been proposed to detect occlusion boundaries in images and videos (*e.g.*, [8], [9], [10], [11], [12], [13], [14], [15], [16]).

Fig. 1: Illustration of the ubiquity of occlusions and the variety of local occlusion patterns (source image from [17]).

In spite of their value, there is no unified definition of occlusion boundaries in the literature and it is usually regarded as a well-known concept. Previous works have provided their own definitions (*e.g.*, [10], [18], [19], [20], [21]), which nevertheless refer to different concepts and exhibit one or more of the following problems: (i) too abstract and lack clarity; (ii) not self-complete and require the definition of other term(s) which in themselves are

- *Asterisk indicates equal contributions.*
- *C. Wang (corresponding author) is with LIGM, Univ Gustave Eiffel, CNRS, ESIEE Paris, École des Ponts, Marne-la-Vallée, France*
  E-mail: chaohui.wang@univ-eiffel.fr
- *H. Fu and D. Tao are with UBTech Sydney AI Institute, School of IT, FEIT, The University of Sydney, Sydney, Australia.*
  E-mail: hufu6371@uni.sydney.edu.au, dacheng.tao@sydney.edu.au
- *M. J. Black is with the Perceiving Systems Department at the Max Planck Institute for Intelligent Systems, Tübingen, Germany.*
  E-mail: black@tuebingen.mpg.de

complicated; and (iii) inaccurate or even incorrect. For instance, the most formal definition that we can identify is the one given in [21]: "occlusion boundaries are object boundaries that occlude other parts of the scene - as opposed to within-object boundaries.". Nevertheless, such a definition is based on the definition of the object, which depends on the specific application (*e.g.*, tree detection regards the whole tree as an object, while leaf counter treats every leaf as an object). Also, from the viewpoint of linguistics, the term *occlusion boundary* should only involve the concepts of *occlusion* and *boundary*, so as to be distinguished from other terms commonly used in computer vision, such as: *object boundary*, *motion boundary*, *etc*. Hence, here we propose a formal definition of occlusion boundaries and illustrate its major connection with depth boundaries, object boundaries and motion boundaries, so as to serve for further exploration of this valuable concept in visual perception and understanding.

Regarding the occlusion boundary detection, despite the considerable number of studies, its state of the art is still unsatisfactory. We believe that one main reason for this is that contextual information has not been sufficiently explored in an efficient way. With this in mind, here we are particularly interested in exploring three main types of contextual information useful for occlusion boundary detection: (i) local contextual correlations in pixel labeling[1] (*e.g.*, [9], [10], [11], [12], [13], [14], [22]); (ii) contextual correlations between the labeling of pixels (*e.g.*, patches) and the observations from the surrounding area of the region (*e.g.*, [9], [22]); and (iii) temporal contextual information contained in video sequences (*e.g.*, [10], [14]). Moreover, we aim to jointly model these three types of information with advanced modeling tools (*e.g.*, deep models, graphical models) to better explore them and boost occlusion boundary detection performance.

To better explore type (i) and (ii) of contextual information, we propose the following main idea (referred to as *L2S*): let us learn to establish a structural labeling map that estimates the state of a relatively small patch "S" by performing the reasoning on the observation in a relatively large image patch "L" with the same center as "S" based on an advanced model (*e.g.*, a deep model). With respect to type (iii) contextual information, we consider the scenario in which a video sequence is the input data[2] (similar to many existing works such as [10], [11], [12], [13], [14]) and design an effective way to make use of such information in the whole model. Finally, we develop two detectors with different characteristics. The first consists of a convolutional neural network (CNN) [23] and a conditional random field (CRF) [24], [25], referred to as *L2S-C*^{2} (see Fig. 2). It benefits from a low requirement for training data, but suffers from additional intrinsic errors introduced in the motion feature computation and the CRF inference. Therefore, we attempt to further improve the performance and show that our L2S idea can be implemented via a specifically designed *fully convolutional network (FCN)* [26], [27]. Based on this, we develop a second approach, referred to as *L2S-FCN* (see Fig. 6), which performs better than L2S-C^2 but requires much more training data.

Experimental evaluations based on two challenging occlusion boundary benchmarks (*CMU* and *VSB100*) demonstrate that our detectors significantly outperform the current state-of-the-art. Further, we empirically demonstrate the importance of spatial and temporal contextual information in occlusion boundary detection, compare our methods with several alternatives to validate the underlying rationale of L2S, and show how the performance of L2S-FCN varies with respect to its two main parameters. These studies are helpful for addressing related visual perception tasks such as object localization and semantic segmentation.

The remainder of the paper is organized as follows. Following the presentation of related work in the remainder of this section, we introduce our definition of occlusion boundaries in Section 2. We then present the two proposed approaches for occlusion boundary detection in Section 3. Experimental evaluations are provided in Section 4, and finally we conclude in Section 5.

Related Work

Regarding the definition of occlusion boundaries, besides the definition in [21] discussed above, some other works have provided their own versions. For example, occlusion boundaries are defined as "extremal boundaries, where the viewing ray is tangent to the object's surface" in [10], "pixels where the flow forward from a frame is inconsistent with the flow back into the frame or where the flow gradient has large magnitude" in [18], "non-occluded boundary pixels that circumscribe a dis-occluded region" in [19], and "the border between two different depth planes" in [20]. It can clearly be seen that all these definitions exhibit one or more of the aforementioned issue(s) with definitions.

Occlusion boundary estimation has attracted extensive attention in computer vision over the past few years. Several methods have been proposed to detect occlusion boundaries in a single image. For example, Saxena *et al*. [28] learn an MRF to capture 3D scene structure and depth information from single images. Hoiem *et al*. [9] demonstrate the importance of 2D perceptual cues and 3D surface and depth cues for occlusion boundary detection, and compute these geometric contexts to reason about occlusions within their CRF model.

Due to the fact that occlusion boundary detection from a single 2D image is ambiguous, many applications consider videos or image sequences as inputs and extend occlusion boundary detection to the temporal dimension. Apostoloff and Fitzgibbon [29] observe that the T-junction is a particularly strong occlusion indicator, and thus learn a relevance vector machine (RVM) T-junction classifier on spatiotemporal patches and fuse Canny edges and T-junctions to detect occlusion edges in the spatial domain. Feldman and Weinshall [4] define the average of the second moment matrix around a pixel as a gradient structure tensor by regarding the video sequence as a spatiotemporal intensity function, and demonstrate that the smallest eigenvalue of this tensor is the occlusion indicator. Stein and Hebert [10] exploit subtle relative motion cues present at occlusion boundaries during a sequence of frames and develop a global boundary model that combines these motion cues and standard appearance cues based on an initial edge detector [30]. Black and Fleet [31] represent occlusion boundaries via a generative model that explicitly encodes the orientation of the boundary, the velocities on either side, the motion of the occluding edge over time, and the appearance/disappearance of pixels at the boundary. Based on this model, the motion of occlusion boundaries is predicted and thus information over time is integrated. Stein and Hebert [32] utilize a spatio-temporal edge detector to detect edglets and estimate edge strength, orientation, and normal motion. Based on these, patches from either side of

1. Like most existing methods, we formulate the problem by endowing each pixel with a binary variable denoting whether the pixel is on occlusion boundaries.

2. Note that one proposed approach, *L2S-C*^{2}, can also be applied directly to the scenario where an individual 2D image is the input data (see Table 3).

each detected edglet are extracted, and then their motions are estimated and compared to determine whether the edglet belongs to an occlusion boundary. Further, both motion boundaries and image boundaries are combined within an MRF framework in [33] to better reason about the occlusion structure in the scene over time.

Although some aforementioned methods attempt to develop discriminative occlusion features on a spatiotemporal volume, recent works have shown that directly using flow-based occlusion features as the temporal information is more convenient and efficient. To name a few, Sargin *et al*. [11] introduce a probabilistic cost function to generate a spatiotemporal lattice across multiple frames to produce a factor graph. Boundary feature channels are then learned using this factor graph by taking some independent flow-based occlusion feature channels into account. He and Yuille [12] argue that image depth discontinuities often occur at occlusion boundaries and estimate the pseudo-depth using the singular value decomposition (SVD) technique from motion flow as a cue for their occlusion detector. Sunberg *et al*. [13] recompute occlusion motion flows on each edge fragment at region boundaries from the initial optical flow [34] based on the observation that an occlusion boundary can be handled by comparing the difference in optical flow in regions on either side. Based on the Hedge algorithm [35], Jacobson *et al*. [36] propose an online learning approach to classifying each particle into one of the predefined occlusion types (including non-occlusion) by utilizing three consecutive images and the corresponding two flow fields as the input. Reporting that local patch features are unable to handle highly variable appearances or intra-object local motion, Raza *et al*. [14] estimate temporally consistent occlusion boundaries via an MRF model whose potentials are learned by random forests using global occlusion motion features and a high-level geometric layout on segmentation boundaries.

Contextual information already plays a proven important role in many computer vision tasks such as object detection, localization, and recognition [37], [38], [39], [40]. Recently, context modeling has also been introduced to boundary detection. Dollár and Zitnick [41] adopt random decision forests [42] to capture structural information of local patch edges. Weinzaepfel *et al*. [43] extend [41] to video datasets and exploit temporal information and static image cues to learn correlations between motion edges within local patches (edges between motion objects).

Previous studies have suggested that the brain encodes contextual information, and biologically inspired deep CNNs have been shown to be powerful for feature extraction and description [44], [45]. This has motivated us to learn the internal correlation of an occlusion boundary in local patches using the CNN framework. Patch-level CNNs have been widely used in a variety of computer vision tasks, with excellent progress made over recent years. For example, Fan *et al*. [46] combine local image patches and a holistic view in a CNN framework to learn contextual information for human pose estimation. Wang *et al*. [47] exploit physical constraints in local patches using a CNN-based model for surface normal estimation. Sun *et al*. [48] and Li *et al*. [49] learn convolutional features from multiple local regions for facial trait recognition. Further, Sun *et al*. [50] formulate an MRF model to remove non-uniform motion blur using the patch-level probabilistic motion blur distribution by CNNs. Motivated by nearest neighbor relationships within a local patch, Ganin and Lempitsky [22] detect edges by learning a 4×4 label feature vector for each patch and matching against a sample CNN output dictionary corresponding to training patches with known annotation. Shen *et al*. [51] make use of the structural information of object contours in contour detection by classifying image patches into different boundary types and accordingly defining a special loss function for training a CNN.

2 Definition of Occlusion Boundaries

Computer vision studies (except studies based on 3D volumetric image, transparency-related works, *etc*.) usually assume that the scene of interest consists of completely and totally opaque components, such that the image generation of such a scene can be modeled as a set of colored 3D surfaces[3] in $\mathbb{R}^3$ ($\mathbb{R}^n$ denotes n-dimensional Euclidean space) and a camera model [52]. Let us use $\mathcal{S} \subset \mathbb{R}^3$ to denote the union of all the 3D surfaces included in the underlying scene, and $\Pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ the 3D-to-2D projection mapping involved in the camera model. Below, we first define *occlusion boundaries* in the context of the continuous image and then explain how to apply it to digital images.

Given a continuous image I of interest with a 2D image domain $\Omega \subset \mathbb{R}^2$, *occlusion boundary point* is defined as follows:

Definition 1. *Occlusion boundary point*: $\forall \mathbf{x} \in \Omega \setminus \partial(\Omega)$, $\mathbf{x}$ is *occlusion boundary point* iff there is no neighborhood $\mathcal{N}_\mathbf{x}$ of $\mathbf{x}$ and $\mathcal{S}_\mathbf{x} \subset \mathcal{S}$, such that: $\mathcal{S}_\mathbf{x}$ is entirely visible and its 2D projection $\Pi(\mathcal{S}_\mathbf{x})$ is $\mathcal{N}_\mathbf{x}$.

In the above definition, we do not consider the points on the image border as occlusion boundary points. Broadly speaking, the camera's aperture boarder can also be regarded as a part of the scene, in which case we can simply replace "$\forall \mathbf{x} \in \Omega \setminus \partial(\Omega)$" by "$\forall \mathbf{x} \in \Omega$" in the definition.

Definition 2. *Occlusion boundaries*: the set of all occlusion boundary points of the image I.

To apply definition 1 to a digital image, we only need to recover the continuous neighborhood $\mathcal{N}_\mathbf{x}$ in the continuous image domain for the smallest discrete neighborhood $\mathcal{N}_\mathbf{x}^d$ of a pixel $\mathbf{x}$ (*e.g*., in case of 4-neighborhood, the smallest discrete neighborhood consists of five pixels: $\mathbf{x}$ itself and its four neighbors), the test of which is sufficient to determine whether $\mathbf{x}$ is occlusion boundary point. To this end, we can simply regard a pixel occupies a square in the continuous image domain and $\mathcal{N}_\mathbf{x}$ is the union of the squares of the pixels included in $\mathcal{N}_\mathbf{x}^d$. Note that the definition leads to two-pixel-width boundaries, composed of the (inner) boundary points of the two (local) regions on the two sides of boundaries (referred to as the *occluder-side* and the *occludee-side* hereafter). If we want to have a one-pixel-width boundaries, we can simply take into account only the points of the occluder-side or the occludee-side.

Note that a piece of occlusion boundary corresponds to the end of some piece of occluding surface and usually implies abrupt increase in depth (*i.e*., the distance to the camera) from the occluder-side to the occludee-side. And *depth boundaries*, which are the discontinuities of the ground-truth depth map for an image, provide a great approximation of occlusion boundaries in practice, in particular when the ground-truth 3D surfaces in the scene are unknown. Furthermore, there is often relatively big variation of other features (such as surface normal, brightness, *etc*.) between

3. For images of outdoor scenes, one can use a plane lying far enough away from the camera or a ball of sufficiently large radius centered at the camera to model the background such as sky.

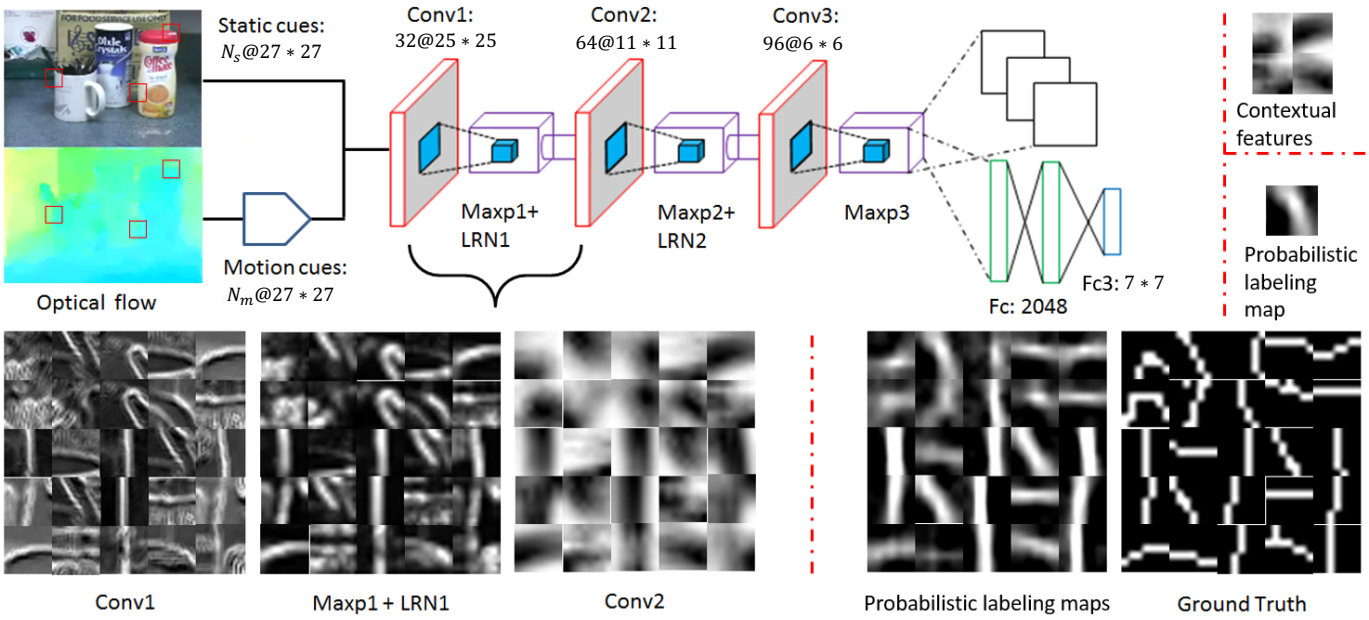

Fig. 2: **Illustration of the CNN architecture and the output of several layers.** For a 27×27 patch of the given input sequence, we first extract N_s static and N_m motion feature maps, which serve as the input of the CNN ($N_s = 2$ and $N_m = 3$ in the current implementation). The output of $Maxp3$ layer corresponds to deep features that aggregate the high-level contextual information (referred to as *deep contextual features*). $fc3$ layer outputs a probabilistic labeling map on a 7×7 patch.

the two sides of the occlusion boundary. From these, we can see that occlusion boundaries contain rich information about the underlying scene structure (in particular about the extent and shape of the underlying objects corresponding to those occluding surfaces), and provide very useful cues for scene construction and understanding from 2D images.

Based on definition 1, according to whether the surface projected onto the two sides of a piece of occlusion boundary are from the same object of interest in the scene, we can further get the definitions of the *self-occlusion boundaries* (in the positive case) and the *object boundaries* (in the negative case) of the image I. This also indicates that the occlusion boundaries are the union of the self-occlusion boundaries and the object boundaries. However, different applications may have different object boundaries and self-occlusion boundaries, since the definition of *object* depends on how we specify the object in the application. For example, (1) in tree detection, we regard the whole tree as an object, and the occlusion boundaries between leaves as self-occlusion boundaries; (2) in leaf counter, every leaf is an object and the occlusion boundaries caused between leaves are object boundaries. This demonstrates that, unlike object boundaries, the concept of occlusion boundaries is application-independent.

Besides, *motion boundaries*, another type of boundaries largely studied in computer vision, are defined as the discontinuities of the ground-truth optical flow between two consecutive frames [53]. We can see that the definition of motion boundaries depends on how we define/measure the discontinuities. This is usually done via a thresholding process and different thresholds lead to different ground-truth motion boundaries. Actually, such discontinuities usually occur across the occlusion boundaries of which the two (local) regions on the two sides exhibit a significant motion difference, since the motion of the 2D projection of a piece of entirely visible surface usually is usually smooth. In this sense, if the discontinuity of optical flow is measured properly, the set of motion boundaries boils down into the set or a subset of occlusion boundaries, for the setting where the input data is video sequence.

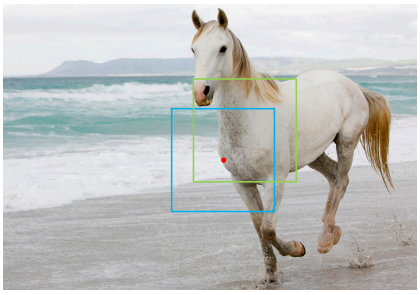

Fig. 3: **Position sensitive box.** The green box can provide more discriminative features than the blue box, if we use either of them to predict the category-level label of the red pixel.

3 Occlusion Boundary Detection via Deep Exploration of Context

3.1 Why L2S?

The L2S idea is mainly motivated by the belief that learning boundaries in a small patch "S" from a large patch "L" with the same center can better explore: (i) local contextual correlations in pixel labeling; and (ii) contextual correlations between the labeling of pixels and the observations from the surrounding area [54]. Here, we further explain one more rationale for L2S from the semantic segmentation perspective, since scene parsing and occlusion boundary detection are highly correlated tasks (occlusion boundaries often lie between neighboring regions of different categories in the image domain). Scene parsing operates at the region level, and a specific pixel may have different semantic

labels in the prediction based on different regions, especially pixels in boundary regions. For example, as shown in Fig. 3, to accurately predict the category label of the red pixel, the features from the green box are more discriminative than those from the blue box since the green box contains more foreground information.

The above observation is reminiscent of position-sensitive score maps (PSSM) proposed in [55], [56], which reduce the translation-invariance of deep convolutional architectures to better address object detection and image segmentation with deep models. The PSSM scheme of [55], [56] is in fact a variant of L2S, and L2S also implies the motivation of PSSM. In [55], [56], the PSSM scheme learns M^2 position-sensitive score maps by considering M^2 neighboring regions from the feature maps. In our L2S scheme, given an input patch, we learn and predict the weighted occlusion boundary map on a smaller patch (of size $M \times M$) with the same center. That is to say, a specific pixel has M^2 scores predicted from M^2 neighboring patches. From this perspective, L2S and PSSM share a similar concept (Fig. 4 illustrates their connection). Finally, based on the aforementioned observation regarding scene parsing, the proposed L2S makes the regions on the two sides of the boundary more distinguishable and thus is efficient for boundary detection, especially for object-level boundaries (such as occlusion boundaries).

3.2 L2S-C^2 Approach

First, to accomplish L2S, we: (i) consider each individual pixel patch as the unit of interest, and (ii) adopt a CNN to learn and predict a patch's occlusion boundary map based on the observation of a larger patch of pixels with the same center. Second, we efficiently explore and encode temporal contextual information within the whole framework by adopting effective motion features in the CNN. Finally, we use a CRF model to integrate patch-based occlusion boundary maps and soft contextual correlations between neighboring pixels to achieve occlusion boundary estimation for the entire image. Each part is described below.

3.2.1 Patch-based Labeling using CNNs

We are interested in modeling and predicting labeling of a patch of pixels based on the observation of a larger patch with the same center via a structured learning/prediction process. Mathematically, given the K-channel observed data on an $N \times N$ patch centered at pixel c, denoted $\mathbf{x}_c \in \mathbb{R}^{K*N^2}$, we aim to obtain the weighted occlusion boundary map $\mathbf{y}_c \in \mathbb{R}^{M^2}$ on an $M \times M$ ($M < N$) patch that is also centered at pixel c, which is achieved via our structured CNN illustrated in Fig. 2. Below we first briefly describe the architecture of our structured CNN and then discuss the initial input features/cues used for occlusion boundary detection.

CNN Architecture

We train a CNN using a cross entropy loss function to predict the probability distribution in a small 7×7 patch from a large 27×27 image patch (*i.e.*, $M = 7$ and $N = 27$ in our experiments). The overall CNN architecture is shown in Fig. 2. The input of our CNN consists of 3 static color channels and 2 temporal channels of size 27×27 (detailed in Section 3.2). The CNN structure can be described by the size of the feature map at each layer as follows: $conv1$ (32@25*25) $\rightarrow$ $maxp1$ $\rightarrow$ $LRN1$ $\rightarrow$ $conv2$ (64@11*11) $\rightarrow$ $maxp2$ $\rightarrow$ $LRN2$ $\rightarrow$ $conv3$ (96@6*6) $\rightarrow$ $maxp3$ $\rightarrow$ $fc1$ (2048) $\rightarrow$ $dropout1$ $\rightarrow$ $fc2$ (2048) $\rightarrow$ $dropout2$ $\rightarrow$ $fc3$ (49), which corresponds to a probabilistic labeling map of size 7×7. Here, $conv$, $maxp$, LRN, fc and $dropout$ denote the convolutional layer, max pooling layer, local response normalization layer, fully-connected layer, and dropout layer, respectively. The LRN scheme implements a form of lateral inhibition, encouraging competition for large activations in the neuronal output [57], [58]. The dropout layer is used to prevent units from co-adapting too much when training a large neural network [59].

In our CNN architecture, the rectified linear units ($ReLUs$) non-linear active function, $f(x) = max(0, x)$, is followed by all $conv$ and fc layers except $fc3$. A sigmoid function is applied to $fc3$ to obtain a probabilistic labeling map; and accordingly the cross entropy loss function is adopted for the training process. Furthermore, the output of $Maxp3$ provides learned deep features that aggregate the high-level contextual information (referred to as *deep contextual features*), which are then used in the CRF model (see Section 3.2.2) to globally reason about occlusion boundaries.

Initial Features for Occlusion Reasoning

Many previous occlusion boundary detection methods have attempted to extract various specific features that characterize occlusions in raw images such as T-junctions, relative depths, and other useful 3D scene properties [9], [12], [14]. However, accurate automatic extraction of such features is also challenging. To avoid these difficulties, we aim to perform occlusion reasoning by using simple but effective initial features/cues. To this end, we first convert an RGB image to Lab space and consider the gradient magnitudes of the Lab maps as three feature maps for the CNN model. In addition, we include optical flow-based motion features to efficiently encode temporal contextual information in video sequences to further improve detection performance. In particular, the following two motion features, together with that of [60], are exploited in the experiments:

- **Motion Feature 1**. The first occlusion motion feature OMF_1 aims to capture optical flow discontinuity, which suggests occlusion boundaries. We use $f_{t,t+t_0}$ ($t_0 \in \mathcal{N}^*$) to denote the optical flow map from frame t to frame $t+t_0$. To capture the discontinuity of $f_{t,t+t_0}$, we compute the unoriented gradient magnitude $GF_{t,t+t_0}$:

$$GF_{t,t+t_0} = |\nabla f_{t,t+t_0}| \tag{1}$$

  Since both forward flow $f_{t,t+t_0}$ and backward flow $f_{t,t-t_0}$ provide motion information from frame t, in order to achieve robustness, we compute $GF_{t,t-t_0}$ similarly together with $GF_{t,t+t_0}$ and consider their geometric mean as one occlusion motion feature OMF_1:

$$OMF_1 = \sqrt{GF_{t,t+t_0} * GF_{t,t-t_0}} \tag{2}$$

- **Motion Feature 2**. The second occlusion motion feature OMF_2 models the fact that the consistency of the flow $f_{t,t+t_0}$ and reverse flow $f_{t+t_0,t}$ is not satisfied when occlusion occurs [61]. We measure these inconsistencies using both location and angle, illustrated in Fig. 5. Let f_l and f'_l denote the flow values at location l in the forward and reverse flow maps $f_{t,t+t_0}$ and $f_{t+t_0,t}$, respectively. If a point located

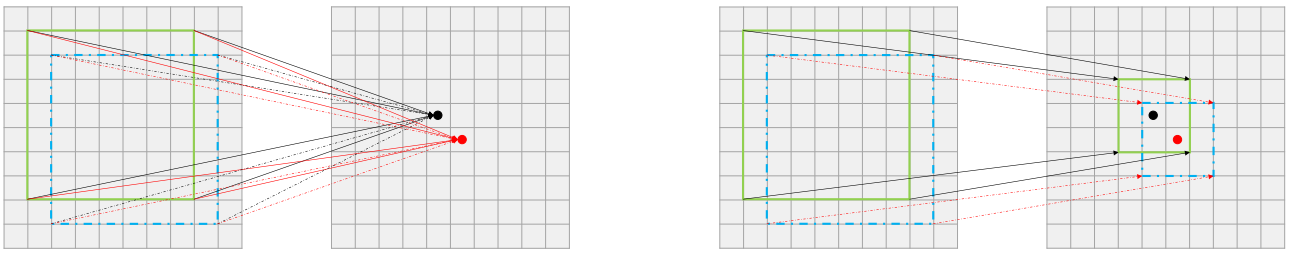

Fig. 4: **Illustration of the connection between PSSM (left) and L2S (right).** Gray grid: image. Larger box: receptive field (or input patch). Smaller box: label patch. Let us take the red pixel as an example. In PSSM, the red pixel receives M^2 scores from M^2 neighboring larger boxes. In L2S, these M^2 neighboring larger boxes are mapped to M^2 neighboring label patches, all of which contain the red pixel. In other words, they provide M^2 scores to the red pixel. PSSM and L2S are equivalent in this sense.

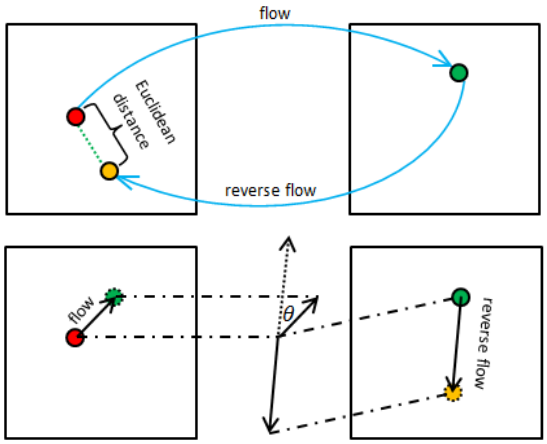

Fig. 5: **Illustration of flow inconsistencies.**

at l in frame t is visible at t and $t+t_0$, its correspondence in frame $t+t_0$ should be located at $l+f_l$ and should return to its start position l after being transported to frame t by the reverse flow f'_{l+f_l}. And with respect to angle, flow f_l and reverse flow f'_{l+f_l} should be π apart if consistent. Hence, we measure these inconsistencies as follows:

$$\Gamma_l = |f_l + f'_{l+f_l}| \tag{3}$$

$$\Lambda_l = \begin{cases} 0 & P|Q \\ arccos\{\frac{-f_l \cdot f'_{l+f_l}}{|f_l||f'_{l+f_l}|}\} & \text{others} \end{cases} \tag{4}$$

where P and Q represent $|f_l| < \delta$ and $|f'_{l+f_l}| < \delta$, respectively, and are used to filter out the likely static pixels and prevent the denominator of the formulation above from being 0 ($\delta = 0.01 * \max_l(|f_l|)$ for each frame in the experiments). Since both Γ and Λ describe inconsistent properties when occlusions occur, we combine them to obtain our inconsistency descriptor $IC_{t,t+t_0}$, via a Gaussian smoothness process:

$$IC_{t,t+t_0}(l) = \sum_{l^*} \sigma(d - |l^* - l|) e^{-\frac{|l^*-l|^2}{2}} \sqrt{\Gamma_{l^*} \Lambda_{l^*}} \tag{5}$$

where $\sigma(x) = 1$ when $x \geq 0$ and 0 otherwise, and $d = 2$ in the experiments. Similar to OMF_1, OMF_2 also takes the backward flow into consideration and is defined as:

$$OMF_2 = \sqrt{IC_{t,t+t_0} * IC_{t,t-t_0}} \tag{6}$$

3.2.2 Image-level Reasoning via CRFs

We then adopt CRFs to efficiently integrate patch-based occlusion boundary maps and soft contextual correlations between neighboring pixels, so as to achieve global occlusion boundary estimation for the entire image. Here, we consider the most common pairwise CRF with 4-neighborhood system used in computer vision and image analysis[4]. In the CRF model, the nodes correspond to the pixel lattice and the edges to pairs of neighboring nodes. Let $\mathcal{V}$ and $\mathcal{E}$ denote the node set and the edge set, respectively. The CRF energy is defined as:

$$E(\mathbf{x}) = \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{\{i,j\} \in \mathcal{E}} \theta_{ij}(x_{ij}) \tag{7}$$

Unary potentials $(\theta_i(\cdot))_{i \in \mathcal{V}}$ are used to encode the data likelihood on individual pixels based on the patch-based probabilistic labeling maps provided by the CNN presented in Section 3.2.1, by defining $\theta_i(\cdot)$ as the negative logarithm of the average probability $\bar{p}_i(\cdot)$ over all output patches that cover the pixel i:

$$\theta_i(x_i) = -\log \bar{p}_i(x_i) \tag{8}$$

Let R_i denote the deep contextual features of the local patch centered at pixel i provided by the $maxp3$ layer of our CNN, and the l^2 norm between R_i and R_j is measured to capture the dissimilarity between neighboring pixels i and j. To penalize different labels between neighboring pixels, the *pairwise potentials* $(\theta_{ij}(\cdot))_{\{i,j\} \in \mathcal{E}}$ between pairs of nodes are defined as:

$$\theta_{ij}(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j \\ w \cdot \exp\{- \parallel R_i - R_j \parallel\} & \text{otherwise} \end{cases} \tag{9}$$

where w is a weight coefficient balancing the importance of the unary and pairwise terms ($w = 2.1$ in the experiments).

3.2.3 Implementation Details

We adopt the region detector provided in [62], which produces a large number of small regions, so as to preserve nearly all types of boundaries, including occlusion boundaries. The occlusion boundary detection boils down to a binary classification problem which determines whether the boundary between regions is or is not an occlusion boundary, which is the same setting as many previous works (*e.g.*, [9], [10], [13], [14]). In order to address

4. The whole method is modular with respect to the choice of CRFs. A main reason to adopt the pairwise CRF with 4-neighborhood system in the experiments, instead of more sophisticated CRFs, is to demonstrate more clearly the effectiveness of the whole method.

the ground truth labeling bias (*e.g.*, the original set of boundaries created by [62] are often 1 or 2 pixels away from the corresponding ground truth boundaries drawn by hand [10]), we consider all pixels within 2 pixels of the boundaries obtained by [62] to produce image patches[5]. To balance the number of positive patches (patches containing an occlusion boundary curve) and negative patches during training, we randomly sample 100,000 training patches in a 1:1 ratio.

The optical flow computation is done via *FlowNet* [63][6]. The 3 image and 3 motion cue channels are the CNN input to learn internal correlations around occlusion boundaries and predict probabilistic labeling maps and extract deep contextual features (see Section 3.2.1 for the motion cues computation and structured CNN framework). Our structured CNN model is built based on *Caffe* [64], developed by the Berkeley Vision And Learning Center (BVLC) and community contributors. The CRF model is then constructed to globally estimate occlusion boundaries for each image using the probabilistic labeling maps and the deep contextual features provided by the learned CNN model. Regarding CRF inference, many powerful off-the-shelf algorithms can be directly applied to solve the CRF model [25]. We simply used sum-product loopy belief propagation [65] to estimate approximate-marginal probabilities of all nodes/pixels via message passing over the graph, so as to get a probabilistic boundary labeling map on the entire image and directly compare with previous methods using the same quality metric, *i.e.*, F-measure.

In the final step, we apply the method in Arbelaez *et al*. [62] to remove isolated pixels and connect disconnected short lines that might belong to a long boundary in our probabilistic occlusion boundary map η. This produces contour boundary map Ω. We then combine η and Ω by learning a weight factor α using SVM to get obtain our final occlusion boundary detector ξ:

$$\xi = \alpha * \eta + (1 - \alpha) * \Omega \tag{10}$$

The value of α is 0.65 in our experiments[7].

3.3 L2S-FCN Approach

The L2S-C^2 approach presented in Section 3.2 performs well in occlusion boundary detection. However, it introduces additional intrinsic errors caused by the following two facts:

- The occlusion motion features are modeled based on the spatial discontinuity of the optical flow and the consistency between the original and reverse flow maps. However, optical flow estimation itself is an open problem in computer vision and its state-of-the-art accuracy is generally still not satisfactory. Further computation based on the computed flow would lead to more errors and further debase the motion information involved in optical flow.
- The image-level inference with CRFs is generally an approximate process due to its NP-hardness, which can also introduce additional computational errors.

Hence, we attempt to further improve our L2S-C^2 and develop a second approach referred to as *L2S-FCN*. We first introduce the approach based on the structure (shown in Fig. 6) used in the experiments by presenting the technical details with respect to two main aspects, and then the implementation details.

5. This operation also prevents the CNN from paying too much attention to the center of the label patch and assigning a high probability to it.

6. Except the experiment estimating how the performance of L2S-C^2 depends on the accuracy of optical flow computation, as shown in Section 4.3.1.

7. The code will be released open-source on publication, same for L2S-FCN.

3.3.1 L2S via FCNs

Assuming that the feature map $\mathcal{C}$ (consisting of c channels of size $h \times w$) and the feature vector $\mathcal{F}$ (consisting of f variables) are fully connected, there should be f corresponding convolutional kernels of size $h \times w \times c$ (as illustrated in Fig. 7). Then, for $\mathcal{C}'$, we can obtain f channels of feature maps of spatial dimension $H \times W$ via those convolutional kernels and some zero-padding operations (to maintain the spatial dimension). In doing so, a feature vector at a specific spatial location in $\mathcal{F}'$ is fully connected to the corresponding feature maps of spatial dimension $h \times w$ within $\mathcal{C}'$. Thus, the feature vectors for all those pixels are computed in a single feed-forward step without any information loss. Moreover, the connection between latent variables to latent variables is a special case of the connection between $\mathcal{C}$ and $\mathcal{F}$ with $h = w = 1$. It should be noted that L2S-FCN predicts M^2 position-sensitive score maps, and for each position/pixel, the M^2 scores in the channel dimension correspond to those of the small label patch ($M \times M$) around it in the original L2S-C^2 scheme.

As shown in Fig. 6, L2S-FCN takes two images (frame t and $t + t_0$) and the optical flow map between them as inputs, and outputs the detected occlusion boundary map for the reference frame t. The training of those five convolutional layers (illustrated in Fig. 6) is supervised by the M^2 position-sensitive score maps to consider features from different receptive fields. Finally, all the score maps are combined together via a group convolutional layer to output the estimated occlusion boundary map. Last but not least, in the whole scheme, the dilation strategy is adopted to extend the receptive fields; the last convolutional layer is added to make the network a little deeper; and those deconvolutional layers are used to upsample the score maps.

3.3.2 Dense Supervision

To predict the label of a specific pixel, it is not true that the larger the receptive field the better. Receptive fields of different sizes may be suitable for recognizing different objects and thus the boundaries between objects. This observation has previously been exploited so as to enhance the performance of deep models. For instance, Xie and Tu [66] suggest giving supervision to each convolutional block of a deep network architecture in their *holistically-nested edge detection* (HED) approach. Liu *et al*. [67] give supervision to each convolutional block in their *richer convolutional features* (RCF) model for edge detection by jointly using all the convolutional features within a convolutional block to capture rich information. However, since the direct concatenation of all the features within a convolutional block would have a high memory cost, RCF resorts to further dimensionality reduction operations, resulting in information loss. Here, to make full use of these convolutional features, we choose to densely supervise (DS) all individual convolutional layers in our network by providing supervision to the outputs of those deconvolutional layers in Fig. 6 and then fuse all output scores.

3.3.3 Implementation Details

As shown in Fig. 6[8], after two basic convolutional layers (including two local normalization layers and two max pooling layers), we obtain a set of convolutional features with spatial dimension $H/4 \times W/4$, where H and W are the height and the width of the inputs. On top of the basic convolutional layers, three

8. The second image is only used to compute brightness warping errors, following [63].

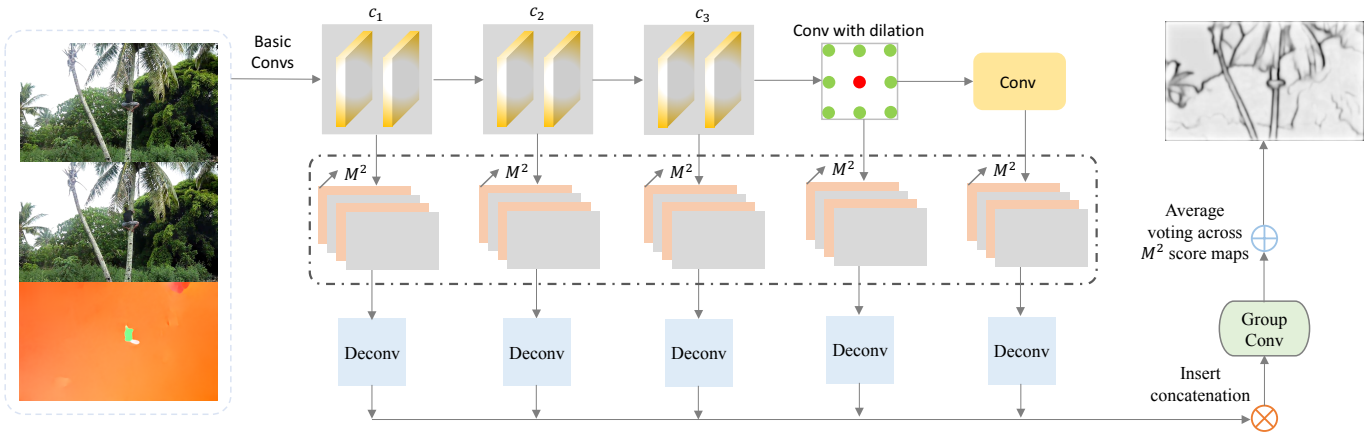

Fig. 6: **L2S-FCN.** The illustration of the L2S-FCN approach (see text).

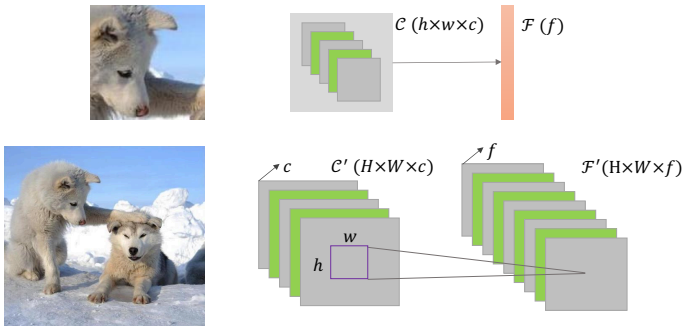

Fig. 7: **Fully-connected layer to convolutional layer.** Top: a patch-level fully-connected layer (in L2S-C^2). Bottom: the corresponding image-level convolutional layer (in L2S-FCN).

additional convolutional layers ($\mathcal{C} : c_1 \sim c_3$) with kernel size 3×3 and channel number 256 are used to make the network a little deeper[9]. To perform reasoning on occlusion boundaries at the image level without further downsampling operations, a typical pooling strategy would make the network lose too much information, while convolution with large kernels would seriously challenge limited GPU memory. Thus, we choose to adopt a dilated convolution technique, which introduces additional zeros to extend the covered areas of the convolutional kernels. Our experiments show that L2S-FCN performs well when the dilation is 6. Moreover, an additional convolutional layer of kernel size 1×1 and channel number 1024 is adopted to make the mapping function more complex. The insert concatenation is specifically designed for L2S-FCN, so that those $M \times M$ score maps can concatenate channel-by-channel according the label patch positions. Then, a group convolutional layer learns $5 \times M^2$ weights for fusing the five M^2-score maps in a weighted manner.

4 Experimental Results

In this section, we first introduce the experimental setting, and then demonstrate the effectiveness of our L2S-C^2 and L2S-FCN approaches in both qualitative and quantitative assessments.

9. In our experiments, we observe that more convolutional layers sharply decrease the performance since our training images are limited and more parameters lead to overfitting.

Last but not least, a set of ablation studies on L2S-C^2 and L2S-FCN are presented to show several of their important properties.

4.1 Experimental Setting

4.1.1 Benchmark

We evaluate the performance of L2S-C^2 and L2S-FCN on two challenging occlusion boundary benchmarks: *CMU* [10] and *VSB100* [13].

CMU[10] is a widely used occlusion boundary benchmark containing 30 video clips and where one single frame of resolution 480×640 is labeled for each video clip. Most previous works on occlusion boundary detection (*e.g.*, [10], [11], [12], [13], [68]) report their performance on this benchmark. It should be noted that this benchmark is not officially split into training and testing sets.

VSB100[11] was proposed by the Berkeley Vision Group for boundary/motion boundary detection and video segmentation and contains 100 videos (40 for training and 60 for testing). Manually labeled boundaries are provided for several frames of each video. For a labeled frame, several types of boundaries are provided according to the segmentation, and one of these is close to occlusion boundary (referred to as *pseudo occlusion boundary*). Aiming to provide an occlusion boundary benchmark, Sundberg *et al.* [13] chose 100 frames (one for each video) to further process their pseudo occlusion boundaries to improve the quality of the occlusion boundary labeling.

4.1.2 Data Augmentation

Since the available occlusion boundary benchmarks only provide a limited number of labeled images, various data augmentation techniques are employed to prevent overfitting and to make L2S-FCN converge to a better solution, including: (i) *size-scaling*: we pre-scale the data (*i.e.*, the inputs and the corresponding labeled boundary map) via 5 fixed factors: 0.5, 0.75, 1.0, 1.25, and 1.5; (ii) *random cropping*: we randomly crop rectangles with predefined sizes from the data; (iii) *flipping*: we horizontally flip the data during the training process, and (iv) *time-scaling*: for a reference image with a labeled boundary map, we compute optical flow maps across different neighboring frames to constitute

10. http://www.cs.cmu.edu/~stein/occlusion_data/
11. https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/

additional training inputs for the same labeled boundary map. Only *flipping* was used for L2S-C^2 in our experiments thanks to its low requirement for training data.

4.1.3 Performance Criteria

As many previous works on edge/boundary detection (including those competitors [43], [62], [66], [67], [69], [70]), we use the following three criteria as quality measures: (i) *optimal dataset scale F-score* (*ODS*): the best F-score (*i.e.*, *F-measure*) on the dataset obtained using a common threshold for all involved test images; (ii) *optimal image scale F-score* (*OIS*): the aggregate F-score on the dataset obtained with image-dependent thresholds; and (iii) *average precision* (*AP*): the average of the precision on the full recall range.

4.1.4 Other Details

The interval t_0 between the two input frames is fixed as 5. Both L2S-C^2 and L2S-FCN are randomly initialized, driven by some cross-entropy loss terms, and learned in an end-to-end fashion during the training process. The stochastic gradient descent optimization involved in deep learning follows a polynomial decay strategy with a base learning rate of 0.0001, power of 0.9, momentum of 0.9, and weight decay of 0.0005. Finally, L2S-FCN is trained with a batch size of 4 on 4 TITAN X (*Pascal*) GPUs with an input resolution of 320×512 for 50K iterations.

4.2 Qualitative and Quantitative Results

4.2.1 CMU

Algorithm	Metric		
	ODS	OIS	AP
OBFM [10]	0.48	–	0.47
POBD-SPL [11]	0.57	–	0.58
Pseudo-depth [12]	0.47	–	0.43
gPb+mg+δ [13]	0.62	–	0.69
Gb [68]	0.62	–	–
gPb-owt-ucm [62]	0.466	0.536	0.425
SE [69]	0.423	0.430	0.368
LDMB [43]	0.577	0.623	0.680
HED [66]	0.542	0.563	0.519
RDS [70]	0.533	0.539	0.514
RCF [67]	0.577	0.601	0.39
L2S-C^2, S1	0.637	0.689	0.765
L2S-C^2, S2	0.634	0.674	0.745
L2S-C^2, S3	0.651	0.697	0.795
L2S-FCNs-NMS	**0.656**	**0.700**	**0.796**

TABLE 1: **Quantitative comparison on the CMU benchmark.** The top and middle subtables show the quantitative results of several previous occlusion boundary detectors and representative edge detectors, respectively. The bottom subtable shows the quantitative results of L2S-C^2 and L2S-FCN. "Sk" ($k = 1, 2, 3$) denotes the index of the experimental setting for L2S-C^2. "NMS" represents standard non-maximum suppression [69] (used hereafter), which is widely used to thin detected edges.

Since the CMU benchmark only provides 30 labeled images, which is not enough for image-level training, we train L2S-FCN by adopting 2530 labeled images from the original VSB100 benchmark and then test on the 30 labeled images of CMU. For L2S-C^2, we evaluate the method in the following three settings, where the quantity of the training data was much smaller and $100,000$ patches randomly extracted from the training images are used for training: (1) use all 100 frames labeled by Sundberg *et al*. [13] for the BVSD benchmark as training images and test on all labeled images in the CMU benchmark; (2) similar to (1), but instead of all 100 labeled frames, use only 15 frames randomly taken from the 40 training frames as training images; and (3) perform 2-fold cross-validation (the 15 images with even/odd indices are regrouped into one fold).

A representative set of qualitative occlusion boundary detection results and a detailed quantitative comparison[12] are shown in Fig. 8 and Table 1. Our L2S-C^2 and L2S-FCN approaches outperform the state-of-the-art for all metrics. Moreover, the *Pre* vs. *Rec* curves of the different methods are shown in Fig. 10, which further demonstrate that the two L2S-based detectors outperform those competitors by a significant margin, especially within the recall interval of [0.4, 0.7]. With more training data, it is expected that L2S-FCN would be able to further improve performance by making the whole network deeper. The performance of L2S-C^2 in all three settings demonstrates the weak dependence of L2S-C^2 on the quantity of training data and that L2S-C^2 can achieve good results when very limited labeled images are available for training.

4.2.2 VSB100

Algorithm	Metric		
	ODS	OIS	AP
gPb-owt-ucm [62]	0.540	0.575	0.527
gPb+mg+δ [13]	0.56	–	–
HED [66]	0.582	0.635	0.600
LDMB [43]	0.523	0.559	0.547
SE [69]	0.535	0.569	0.510
RDS [70]	0.597	0.660	0.637
RCF [67]	0.615	0.662	0.635
Boundary Flow [71]	0.597	0.632	0.566
L2S-C^2, S1	0.607	0.662	0.675
L2S-C^2, S2	0.601	0.660	0.665
L2S-FCN-NMS	**0.641**	**0.716**	**0.720**

TABLE 2: **Quantitative comparison on the VSB100 benchmark.** The quantitative results of several previous methods and ours. "Sk" ($k = 1, 2$) denotes the index of the training setting for L2S-C^2.

Since Sundberg *et al*. [13] is the only occlusion boundary detection study that reported quantitative performance on the VSB100 benchmark, we also used the 60 test images that they labeled as the testing set. Regarding the training set, L2S-FCN adopts 1225 labeled images from the officially-specified VSB100 training set and the CMU benchmark, while L2S-C^2 is evaluated in two settings where the model is trained using $100,000$ patches randomly extracted from: (1) those 40 training frames labeled by Sundberg *et al*. [13] and all 30 labeled samples in the CMU benchmark; and (2) 15 frames randomly taken from the aforementioned 40 training frames. We also test several related methods whose codes are available online. As shown in Table 2, L2S-C^2 and L2S-FCN outperform these competitors by a large margin. Fig. 9 shows a representative set of qualitative

12. The results of previous occlusion boundary detectors (see top rows in Table 1) are taken from the original papers (AP values are from [13]), while those of representative edge detectors (see middle rows in Table 1) are obtained via experimental evaluation using authors' codes, which are available online.

img ground truth gPb-owt-ucm RCF L2S-C^2 L2S-FCN

Fig. 8: **Representative qualitative results on the CMU benchmark [10].** Each row corresponds to one test sequence and consists of (from left to right): a reference frame, the occlusion boundary ground truth, the occlusion boundary maps obtained by gPb-owt-ucm [62], RCF [67], our L2S-C^2 and L2S-FCN approaches, successively.

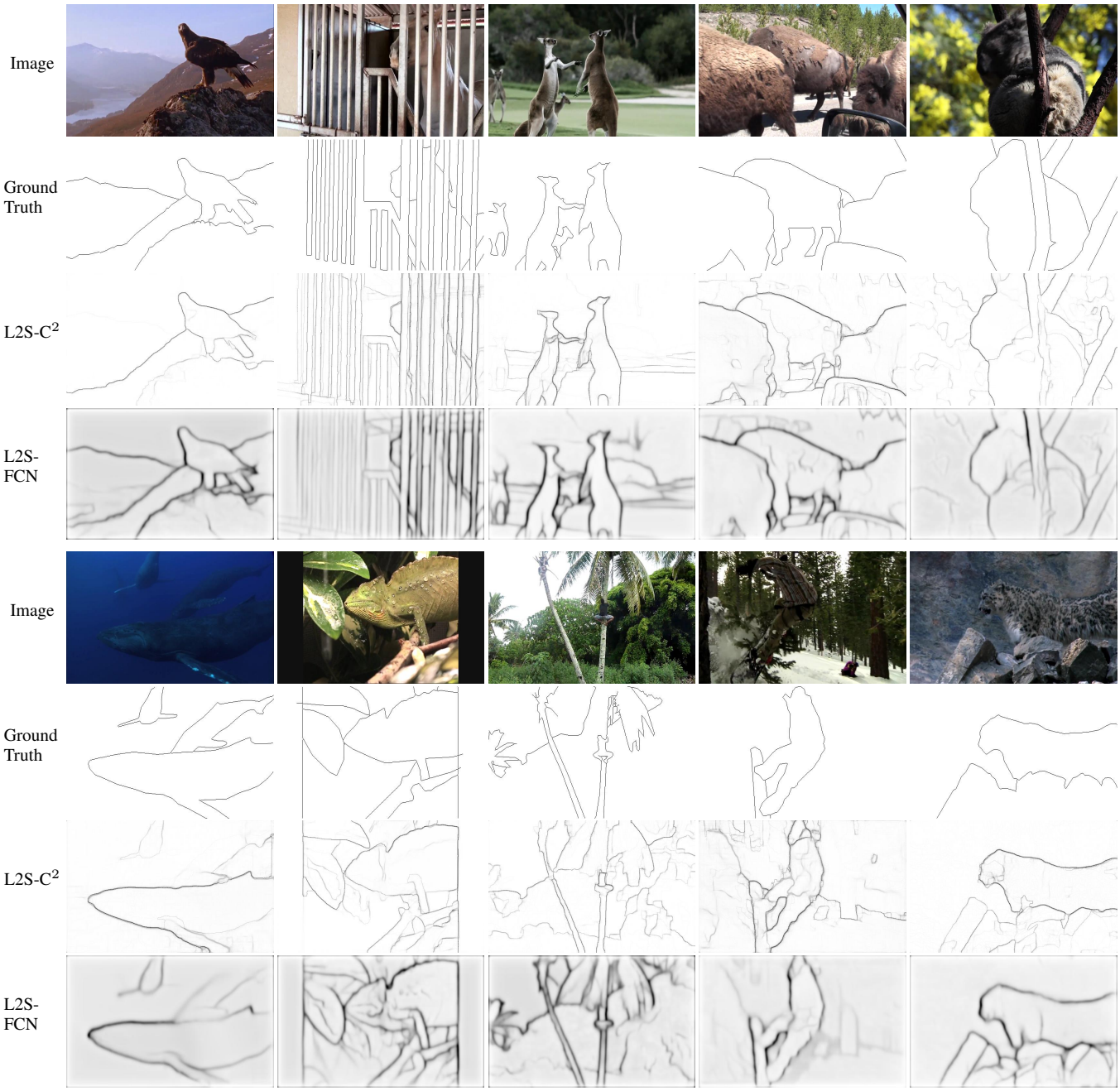

Fig. 9: **Representative qualitative results on the VSB100 benchmark [13]**. The whole figure consists of the upper and lower sub-figures. Each column of each sub-figure corresponds to one test sequence and consists of (from top to bottom): a reference frame, the occlusion boundary ground truth, and the occlusion boundary maps obtained by our L2S-C^{2}and L2S-FCN approaches.

results. It should be noted that, when learning L2S-FCN for the VSB100 benchmark, each convolutional layer only contains half the number of channels (fewer parameters) used for the CMU benchmark due to the decrease of the number of training images by about a half[13].

From the results on both the CMU and VSB100 benchmarks, we can conclude that L2S-FCN delivers better performance than L2S-C^2 in those cases in which enough data with ground truth are available to train the model, while L2S-C^2 provides a good approach to the scenario in which it is problematic to obtain enough training data for image-level training.

4.3 Ablation Studies of L2S-C^2

4.3.1 Contribution of Temporal Cues

We next present experimental results[14] that estimate the contribution of each type of motion feature to our algorithm. From the middle subtable of Table 3, it can be seen that: (i) the motion

13. Because of limited training samples and fewer parameters, we feed the first image to pre-trained VGG-16 [72] to extract 3 channels of convolutional features from *conv3-3*.

14. All ablation experiments are performed using the same data setting as the evaluation on the CMU benchmark presented in Section 4.2.1.

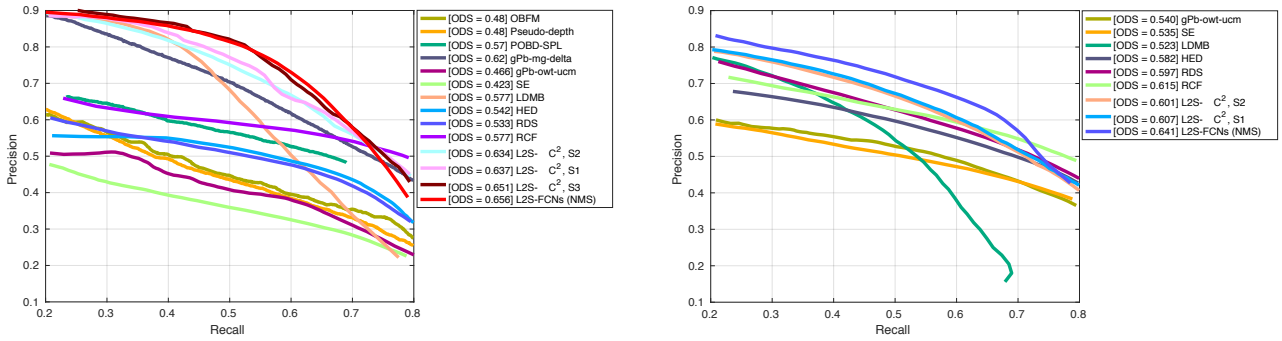

Fig. 10: **Precision-recall curves.** The left and right subfigures show the precision-recall curves on the CMU and VSB100 benchmarks, respectively, which are drawn following [62], [69].

context provides important cues that have a large impact on occlusion boundary detection performance, and (ii) each of the three motion features significantly contributes to the method's performance and jointly using them achieves the highest accuracy. Furthermore, in order to evaluate how the method's performance depends on the accuracy of optical flow computation, in the bottom subtable of Table 3, we report the quantitative performance achieved by using three typical optical flow algorithms proposed by: Brox *et al*. [34], Sun *et al*. [73] and Weinzaepfel *et al*. [74]. Those similar scores demonstrate that our method is quite robust with respect to the choice of optical flow algorithm.

Variants	Metric		
	ODS	OIS	AP
L2S-C^2(FlowNet)	0.637	0.689	0.765
Static	0.523	0.596	0.577
Omf1	0.618	0.656	0.678
Omf2	0.612	0.647	0.669
Omf3	0.611	0.647	0.665
L2S-C^2(LDOF)	0.619	0.678	0.714
L2S-C^2(ClassicNL)	0.634	0.710	0.755
L2S-C^2(Deepflow)	0.622	0.695	0.738

TABLE 3: **Comparison between different temporal cues and optical flow computation algorithms.** The top row shows the baseline setting "L2S-C^2(FlowNet)", where all three occlusion motion features are used (*i.e*., the two presented in Section 3.2 and that of [60]) and where the optical flow is obtained by FlowNet [63]. The middle subtable shows the performance when replacing the used features in the baseline setting. Static: L2S-C^2 without motion features. Omf$i$ ($i = 1, 2, 3$): L2S-C^2 with the i^{th} one of those three occlusion motion features. The bottom subtable shows the performance of L2S-C^2 with optical flow obtained by the LDOF [34], ClassicNL [73], and DeepFlow [74] algorithms, successively.

4.3.2 *Validation of the choice of the L2S mapping*

The way in which contextual information is explored (*i.e*., in *L2S* we learn the mapping from a large patch "L" to a small patch "S" with the same center as "L") is a key component of the method, since both the edge and node potentials in the final CRF framework are related to the contextual information aggregated by CNNs. To validate our choice, we compare it with the following variants: (i) *L2P*: we learn the mapping from "L" to the pixel at the center of "L"; (ii) *L2SP*: we learn the mapping from "L" to "S" by independently learning the mapping to each individual pixel located within "S"; and (iii) *L2L*: we learn the mapping from "L" to "L", where the sizes of input and output patches are equal. The quantitative results reported in Table 4 show that the method works best when the mapping is learned from "L" to "S". This observation can be explained as follows: (i) In L2P and L2SP, the CNN concentrates more on learning the differences between input samples to binary classify the whole input patch without considering the structural correlation between the labeling of the pixels within a local image patch; (ii) In L2L, contextual correlations between the labeling of pixels and the observations from the surrounding area are not considered. Besides, the fixed training set would become over sparse for the model training if "L" is too large; and (iii) L2S properly handles the aforementioned issues exhibited in the variants so as to achieve better structural features and labeling results.

Variants	Metric		
	ODS	OIS	AP
L2P	0.601	0.660	0.680
L2SP	0.593	0.652	0.674
L2L	0.606	0.662	0.688
L2S	0.637	0.689	0.765

TABLE 4: **Comparison of different mapping methods.**

4.4 Ablation Studies of L2S-FCN

4.4.1 *Influence of the Label Patch Size*

To explore the influence of the label patch size for boundary detection tasks and to find an optimal label patch size, we perform experiments on label patches of side length 1, 3, 5, 7, and 9, respectively, and the obtained quantitative metrics[15] are shown in Table 5. L2S-FCN performs best when the side length is 5 and 7. When the side length is 1, which is simply an image to single boundary map structure (without L2S structure), the performance is obviously lower than those of 3, 5, and 7, demonstrating the effectiveness of our L2S idea. Moreover, L2S-FCN-9x9 performs the worst, mainly because a pixel lying far away from the center of a label patch cannot respond well to a fixed input region, which significantly affects the training process.

4.4.2 *Different Depths and Receptive Fields*

Different receptive fields in different depths all contribute to the final score, so here we report in Table 6 these scores for

15. Note that NMS [69] was not used in the experiments shown in Sec. 4.4, unless explicitly mentioned.

Variants	Metric		
	ODS	OIS	AP
L2S-FCN-1x1	0.610	0.668	0.718
L2S-FCN-3x3	0.619	0.675	0.739
L2S-FCN-5x5	0.646	0.689	0.764
L2S-FCN-7x7	0.640	0.689	0.761
L2S-FCN-9x9	0.605	0.673	0.711

TABLE 5: **Comparison of L2S-FCN with different label patch sizes.** L2S-FCN-MxM means that the label patch size is $M \times M$.

further analysis of L2S-FCN. The baseline model, which does not consider different receptive fields (without DS), leads to an ODS of 0.570. When adding dense supervision (different receptive fields at different depths), it yields an ODS of 0.646, which outperforms the baseline model by 13.3%. Also, we can conclude from Table 6 that the increase in the depth number within a certain range can include information from more scales of receptive fields and improve final performance. However, due to data limitations, a too deep network can result in overfitting.

Variants	Metric		
	ODS	OIS	AP
Baseline	0.570	0.646	0.682
L2S-FCN-$\mathcal{C}1$	0.577	0.643	0.700
L2S-FCN-$\mathcal{C}2$	0.598	0.649	0.705
L2S-FCN-$\mathcal{C}3$	0.646	0.689	0.764
L2S-FCN-$\mathcal{C}4$	0.640	0.682	0.756
L2S-FCN-$\mathcal{C}5$	0.622	0.670	0.743

TABLE 6: **L2S-FCN of different depths.** L2S-FCN-$\mathcal{C}n$ denotes L2S-FCN of depth n with DS. *Baseline* denotes L2S-FCN-$\mathcal{C}3$ without DS.

4.4.3 Study based on MPI Sintel Dataset

Last, but not least, we also conducted an ablation study based on *MPI Sintel* [75], which is a synthetic dataset that provides a set of animated films with ground-truth annotations for optical flow, depth, and motion occlusion regions. We use the discontinuity of the provided ground-truth depth maps to compute the pseudo-occlusion-boundaries, due to the lack of the ground-truth 3D surface. We split the original MPI Sintel videos into the training and testing sets, where the latter consists of four animated films, *i.e.*, "alley_2", "ambush_5", "bandage_2" and "temple_2". As a result, the training and testing set contain 748 and 195 images. We resize the images to the resolution of 512×1024, and train our L2S-FCNs model on a random crop of 512×512.

The quantitative results are reported in Table 7. On the one hand, we compare L2S-FCN with the RCF [67], OMD [76], and DOOBNet [16] [77] methods, and L2S-FCN achieves the best performance. On the other hand, we test L2S-FCN with optical flow obtained by the DeepFlow [74], FlowNet [78], FlowNet2 [63], RAFT [79] algorithms, and also the provided ground-truth optical flow, successively. The obtained results demonstrate that: (1) the performance of L2S-FCN depends on that of the optical flow algorithm, which is logical; and (2) L2S-FCN is robust with respect to the choice of the optical flow algorithm.

16. We reproduce DOOBNet on MPI Sintel dataset by removing the orientation smooth L1 loss and adding the optical flow feature.

Variants	Metric		
	ODS	OIS	AP
RCF [67]	0.489	0.548	0.359
OMD [76]	0.525	0.569	0.430
DOOBNet [77] (FlowNet2)	0.536	0.581	0.446
L2S-FCNs-NMS (FlowNet2)	**0.570**	**0.617**	**0.498**
L2S-FCNs-NMS (DeepFlow)	0.561	0.606	0.483
L2S-FCNs-NMS (FlowNet)	0.564	0.609	0.485
L2S-FCNs-NMS (FlowNet2)	0.570	0.617	0.498
L2S-FCNs-NMS (RAFT)	0.572	0.618	0.498
L2S-FCNs-NMS (GT)	0.586	0.631	0.506

TABLE 7: **Quantitate Evaluation on the MPI Sintel dataset.** The top subtable shows the comparison between L2S-FCN and three alternative algorithms, while the bottom subtable displays the performance of L2S-FCN with optical flow obtained by four representative algorithms, and also the ground-truth provided by the dataset, successively.

5 Conclusion

In this paper, besides a formal definition of occlusion boundaries, we have developed two occlusion boundary detectors following the L2S idea so as to exploit contextual information including local structural boundary patterns, observations from surrounding regions, temporal context, and soft contextual correlations between neighboring pixels. Experimental results demonstrate that both the detectors significantly outperform the current state-of-the-art. Last but not least, we empirically assess the roles of several important components of the proposed detectors in exploring contextual information to validate the rationale behind these approaches. In the future, we would like to seek appropriate approach to reducing the amount of human labor in labeling data for training L2S-FCN, and search for sophisticated approaches to applying the L2S idea to other challenging scene understanding problems.

Acknowledgments

This work is partly supported by Australian Research Council Projects (DP-140102164, FT-130101457, LE140100061) and CNRS INS2I-JCJC-INVISANA.

References

[1] F. Galasso, N. S. Nagaraja, T. Jimenez Cardenas, T. Brox, and B. Schiele, "A unified video segmentation benchmark: Annotation, metrics and analysis," in *Int. Conf. Comput. Vis.*, 2013.
[2] B. Taylor, V. Karasev, and S. Soatto, "Causal video object segmentation from persistence of occlusions," in *Comput. Vis. Pattern Recog.*, 2015.
[3] A. Owens, C. Barnes, A. Flint, H. Singh, and W. Freeman, "Camouflaging an object from many viewpoints," in *Comput. Vis. Pattern Recog.*, 2014.
[4] D. Feldman and D. Weinshall, "Motion segmentation and depth ordering using an occlusion detector," *IEEE Trans Pattern Anal. Mach. Intell.*, vol. 30, no. 7, pp. 1171–1185, 2008.
[5] A. Ayvaci and S. Soatto, "Detachable object detection: Segmentation and depth ordering from short-baseline video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 10, pp. 1942–1951, 2012.
[6] I. Biederman, *On the semantics of a glance at a scene*, 1981.
[7] J. Gibson, *The perception of surface layout: A classification*. Unpublished "Purple Perils" essay, Nov 1968.
[8] M. J. Black and P. Anandan, "Constraints for the early detection of discontinuity from motion." in *National Conf. on Artif. Intell.*, 1990.
[9] D. Hoiem, A. N. Stein, A. Efros, M. Hebert *et al.*, "Recovering occlusion boundaries from a single image," in *Int. Conf. Comput. Vis.*, 2007.
[10] A. N. Stein and M. Hebert, "Occlusion boundaries from motion: Low-level detection and mid-level reasoning," *Int. J. Comput. Vis.*, vol. 82, no. 3, pp. 325–357, 2009.

[11] M. E. Sargin, L. Bertelli, B. S. Manjunath, and K. Rose, "Probabilistic occlusion boundary detection on spatio-temporal lattices," in *Int. Conf. Comput. Vis.*, 2009.
[12] X. He and A. Yuille, "Occlusion boundary detection using pseudo-depth," in *Europ. Conf. Comput. Vis.*, 2010.
[13] P. Sundberg, T. Brox, M. Maire, P. Arbeláez, and J. Malik, "Occlusion boundary detection and figure/ground assignment from optical flow," in *Comput. Vis. Pattern. Recog.*, 2011.
[14] S. H. Raza, A. Humayun, I. Essa, M. Grundmann, and D. Anderson, "Finding temporally consistent occlusion boundaries in videos using geometric context," in *Winter Conf. Appl. Comput. Vis.*, 2015.
[15] P. Wang and A. Yuille, "Doc: Deep occlusion estimation from a single image," in *Europ. Conf. Comput. Vis.*, 2016.
[16] H. I. Cakir, C. Topal, and C. Akinlar, "An occlusion-resistant ellipse detection method by joining coelliptic arcs," in *Europ. Conf. Comput. Vis.*, 2016.
[17] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *Pattern Recog.*, 2014.
[18] N. Sundaram, T. Brox, and K. Keutzer, "Dense point trajectories by gpu-accelerated large displacement optical flow," in *Europ. Conf. Comput. Vis.*, 2010.
[19] H. Lim, Y. S. Kim, S. Lee, O. Choi, J. D. Kim, and C. Kim, "Bi-layer inpainting for novel view synthesis," in *Int. Conf. Image Process.*, 2011.
[20] G. Palou and P. Salembier, "Monocular depth ordering using t-junctions and convexity occlusion cues," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 1926–1939, 2013.
[21] M. Keuper and T. Brox, "Point-wise mutual information-based video segmentation with high temporal consistency," in *ECCV 2016 Workshops*. Springer, 2016.
[22] Y. Ganin and V. Lempitsky, "Nˆ 4-fields: Neural network nearest neighbor fields for image transforms," in *ACCV*, 2014.
[23] Y. Bengio, I. J. Goodfellow, and A. Courville, "Deep learning," 2015, book in preparation for MIT Press.
[24] A. Blake, P. Kohli, and C. Rother, *Markov random fields for vision and image processing*. Mit Press, 2011.
[25] C. Wang, N. Komodakis, and N. Paragios, "Markov random field modeling, inference & learning in computer vision & image understanding: A survey," *Comput. Vis. Image Underst.*, vol. 117, no. 11, pp. 1610–1627, 2013.
[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440.
[27] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, 2017.
[28] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Neural inf. Process. Syst.*, 2005.
[29] N. Apostoloff and A. Fitzgibbon, "Learning spatiotemporal t-junctions for occlusion detection," in *Comp. Vis. Pattern Recog.*, 2005.
[30] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 5, pp. 530–549, 2004.
[31] M. J. Black and D. J. Fleet, "Probabilistic detection and tracking of motion boundaries," *Int. J. Comput. Vis.*, vol. 38, no. 3, pp. 231–245, 2000.
[32] A. N. Stein and M. Hebert, "Local detection of occlusion boundaries in video," in *Brit. Mach. Vis. Conf.*, 2006, pp. 407–416.
[33] M. J. Black, "Combining intensity and motion for incremental segmentation and tracking over long image sequences," in *Europ. Conf. Comp. Vis.*, 1992.
[34] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 500–513, 2011.
[35] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth, "Using and combining predictors that specialize," in *STOC*. ACM, 1997, pp. 334–343.
[36] N. Jacobson, Y. Freund, and T. Q. Nguyen, "An online learning approach to occlusion boundary detection," *IEEE Trans. Image Process.*, vol. 21, no. 1, pp. 252–261, 2012.
[37] P. Carbonetto, N. de Freitas, and K. Barnard, "A statistical model for general contextual object recognition," in *Europ. Conf. Comput. Vis.*, 2004.
[38] Z. Tu and X. Bai, "Auto-context and its application to high-level vision tasks and 3d brain image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 10, pp. 1744–1757, 2010.
[39] H. Myeong, J. Y. Chang, and K. M. Lee, "Learning object relationships via graph-based context model," in *Comput. Vis. Pattern Recog.*, 2012.
[40] T. Malisiewicz and A. Efros, "Beyond categories: The visual memex model for reasoning about object relationships," in *Neural Inf. Process. Syst.*, 2009.
[41] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *Int. Conf. Comput. Vis.*, 2013.
[42] P. Kontschieder, S. Rota Buló, H. Bischof, and M. Pelillo, "Structured class-labels in random forests for semantic image labelling," in *Int. Conf. Mach. Learn.*, 2011.
[43] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Learning to detect motion boundaries," in *Comput. Vis. Pattern Recog.*, 2015.
[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Neural Inf. Process. Syst.*, 2012.
[45] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[46] X. Fan, K. Zheng, Y. Lin, and S. Wang, "Combining local appearance and holistic view: Dual-source deep neural networks for human pose estimation," in *Comput. Vis. Pattern Recog.*, 2015.
[47] X. Wang, D. F. Fouhey, and A. Gupta, "Designing deep networks for surface normal estimation," 2015.
[48] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Comput. Vis. Pattern Recog.*, 2014.
[49] S. Li, J. Xing, Z. Niu, S. Shan, and S. Yan, "Shape driven kernel adaptation in convolutional neural network for robust facial trait recognition," in *Comput. Vis. Pattern Recog.*, 2015.
[50] J. Sun, W. Cao, Z. Xu, and J. Ponce, "Learning a convolutional neural network for non-uniform motion blur removal," *Comput. Vis. Pattern Recog.*, 2015.
[51] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," in *Comput. Vis. Pattern Recog.*, 2015.
[52] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.
[53] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Learning to detect motion boundaries," in *Comput. Vis. Pattern Recog.*, 2015.
[54] H. Fu, C. Wang, D. Tao, and M. J. Black, "Occlusion boundary detection via deep exploration of context," in *Comput. Vis. Pattern Recog.*, 2016.
[55] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Neural Inf. Process. Syst.*, 2016.
[56] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," *Comput. Vis. Pattern Recog.*, 2016.
[57] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Comput. Vis. Pattern Recog.*, 2010.
[58] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Int. Conf. Mach. Learn.*, 2010.
[59] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.
[60] P. Sand and S. Teller, "Particle video: Long-range motion estimation using point trajectories," *Int. J. Comput. Vis.*, vol. 80, no. 1, p. 72, 2008.
[61] A. Humayun, O. Mac Aodha, and G. J. Brostow, "Learning to find occlusion regions," in *Comput. Vis. Pattern Recog.*, 2011.
[62] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.
[63] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," *arXiv preprint arXiv:1612.01925*, 2016.
[64] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *ACM Multimedia*, 2014.
[65] Y. Weiss, "Comparing the mean field method and belief propagation for approximate inference in mrfs," *Advanced Mean Field Methods - Theory and Practice*, pp. 229–240, 2001.
[66] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Int. Conf. Comput. Vis.*, 2015.
[67] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer convolutional features for edge detection," in *Comput. Vis. Pattern Recog.*, 2017.
[68] M. Leordeanu, R. Sukthankar, and C. Sminchisescu, "Generalized boundaries from multiple image interpretations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1312–1324, 2014.
[69] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 8, pp. 1558–1570, 2015.

[70] Y. Liu and M. S. Lew, "Learning relaxed deep supervision for better edge detection," in *Comput. Vis. Pattern Recog.*, 2016.
[71] P. Lei, F. Li, and S. Todorovic, "Boundary flow: A siamese network that predicts boundary motion without training on motion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
[72] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[73] D. Sun, S. Roth, and M. J. Black, "A quantitative analysis of current practices in optical flow estimation and the principles behind them," *Int. J. Comput. Vis.*, vol. 106, no. 2, pp. 115–137, 2014.
[74] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Deepflow: Large displacement optical flow with deep matching," in *Int. Conf. Comput. Vis.*, 2013.
[75] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.
[76] E. Ilg, T. Saikia, M. Keuper, and T. Brox, "Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
[77] G. Wang, X. Wang, F. W. B. Li, and X. Liang, "Doobnet: Deep object occlusion boundary detection from an image," in *Asian Conference on Computer Vision (ACCV)*, 2018.
[78] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
[79] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," *ArXiv*, vol. abs/2003.12039, 2020.

Dacheng Tao (F15) is Professor of Computer Science and ARC Future Fellow in the School of Information Technologies and the Faculty of Engineering and Information Technologies, and the Inaugural Director of the UBTech Sydney Artificial Intelligence Institute, at The University of Sydney. He mainly applies statistics and mathematics to Artificial Intelligence and Data Science. His research interests spread across computer vision, data science, image processing, machine learning, and video surveillance. His research results have expounded in one monograph and 500+ publications at prestigious journals and prominent conferences, such as IEEE T-PAMI, T-NNLS, T-IP, JMLR, IJCV, NIPS, CIKM, ICML, CVPR, ICCV, ECCV, AISTATS, ICDM; and ACM SIGKDD, with several best paper awards, such as the best theory/algorithm paper runner up award in IEEE ICDM07, the best student paper award in IEEE ICDM13, the 2014 ICDM 10-year highest-impact paper award, and the 2017 IEEE Signal Processing Society Best Paper Award. He received the 2015 Australian Scopus-Eureka Prize, the 2015 ACS Gold Disruptor Award and the 2015 UTS Vice-Chancellors Medal for Exceptional Research. He is a Fellow of the IEEE, OSA, IAPR and SPIE.

Chaohui Wang is Associate Professor at Université Gustave Eiffel, and researcher at LIGM Laboratory (UMR 8049), Université Gustave Eiffel, CNRS, ESIEE Paris, École des Ponts, France. He received his Ph.D. in applied mathematics and computer vision from Ecole Centrale Paris, France (in 2011), and was postdoctoral researcher at University of California, Los Angeles, USA (2012 $\sim$ 2013) and at Max Planck Institute for Intelligent Systems, Tübingen, Germany (2013 $\sim$ 2014), successively. His current research interests include computer vision, machine learning, and related fields.

Michael J. Black received the BSc from the University of British Columbia, in 1985, the MS degree from Stanford University, in 1989, and the PhD degree from Yale University, in 1992. After research at NASA Ames and the University of Toronto, he was at Xerox PARC as a member of research staff and area manager (1993-2000). From 2000 to 2010, he was on the faculty of Brown University (associate professor 2000-2004, professor 2004-2010). Since 2011, he has been one of the founding director at the Max Planck Institute for Intelligent Systems in Tübingen, Germany, where he leads the Perceiving Systems department. Significant awards include the 2010 Koenderink Prize (ECCV) and the 2013 Helmholtz Prize (ICCV). He is a foreign member of the Royal Swedish Academy of Sciences and is a co-founder and board member of Body Labs Inc.

Huan Fu is currently a PhD student with the School of Information Technologies and the Faculty of Engineering and Information Technologies, and a member in the UBTECH Sydney AI Centre, at The University of Sydney. He received the BEng degree in engineering computer science and technology from the University of Science and Technology of China. His research interests include deep learning and computer vision.