Convolutional Oriented Boundaries: From Image Segmentation to High-Level Tasks

Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and Luc Van Gool

Abstract—We present Convolutional Oriented Boundaries (COB), which produces multiscale oriented contours and region hierarchies starting from generic image classification Convolutional Neural Networks (CNNs). COB is computationally efficient, because it requires a single CNN forward pass for multi-scale contour detection and it uses a novel sparse boundary representation for hierarchical segmentation; it gives a significant leap in performance over the state-of-the-art, and it generalizes very well to unseen categories and datasets. Particularly, we show that learning to estimate not only contour strength but also orientation provides more accurate results. We perform extensive experiments for low-level applications on BSDS, PASCAL Context, PASCAL Segmentation, and NYUD to evaluate boundary detection performance, showing that COB provides state-of-the-art contours and region hierarchies in all datasets. We also evaluate COB on high-level tasks when coupled with multiple pipelines for object proposals, semantic contours, semantic segmentation, and object detection on MS-COCO, SBD, and PASCAL; showing that COB also improves the results for all tasks.

Index Terms—Contour detection, contour orientation, hierarchical image segmentation, object proposals, semantic contours

arXiv:1701.04658v2 [cs.CV] 28 Apr 2017

1 Introduction

The adoption of Convolutional Neural Networks (CNNs) has caused a profound change and a large leap forward in performance throughout the majority of fields in computer vision. In the case of a traditionally category-agnostic field such as contour detection, it has recently fostered the appearance of systems [1], [2], [3], [4], [5], [6] that rely on large-scale category-specific information in the form of deep architectures pre-trained on Imagenet [7] for image classification [8], [9], [10], [11].

This paper proposes Convolutional Oriented Boundaries (COB), a generic CNN architecture that allows end-to-end learning of multiscale oriented contours, and we show how it translates top performing base CNN networks into high-quality contours; allowing to bring future improvements in base CNN architectures into semantic grouping. We then propose a sparse boundary representation for efficient construction of hierarchical regions from the contour signal. Our overall approach is both efficient (it runs in 0.8 seconds per image) and highly accurate (it produces state-of-the-art contours and regions on PASCAL and on the BSDS). Figure 1 shows an overview of our system.

For the last fifteen years, the Berkeley Segmentation Dataset and Benchmark (BSDS) [12] has been the experimental testbed of choice for the study of boundary detection and image segmentation. However, the current large-capacity and very accurate models have underlined the limitations of the BSDS as the primary benchmark for grouping. Its 300 train images are inadequate for training systems with tens of millions of parameters and, critically, current state-of-the-art techniques are reaching human performance for boundary detection on its 200 test images.

In terms of scale and difficulty, the next natural frontier for perceptual grouping is the PASCAL VOC dataset [13], an influential benchmark for image classification, object detection, and semantic segmentation which has a *trainval* set with more than 10 000 challenging and varied images. A first step in that direction was taken by Hariharan et al. [14], who annotated the VOC dataset for category-specific boundary detection on the foreground objects. More recently, the PASCAL Context dataset [15] extended this annotation effort to all the background categories, providing thus fully-parsed images which are a direct VOC counterpart to the human ground truth of the BSDS. In this direction, this paper investigates the transition from the BSDS to PASCAL Context in the evaluation of image segmentation.

We derive valuable insights from studying perceptual grouping in a larger and more challenging empirical framework. Among them, we observe that COB leverages increasingly deeper state-of-the-art architectures, such as the recent Residual Networks [11], to produce improved results. This indicates that our approach is generic and can directly benefit from future advances in CNNs. We also observe that, in PASCAL, the globalization strategy of contour strength by spectral graph partitioning proposed in [16] and used in state-of-the-art methods [1], [17] is unnecessary in the presence of the high-level knowledge conveyed by pre-trained CNNs and oriented contours, thus removing a significant computational bottleneck for high-quality contours.

We conduct two types of experiments, the first of which regards low-level vision applications, such as contour detection and generic segmentation on PASCAL Context and the BSDS500. We extend the evaluation to the NYUD RGB-D dataset, showing that the pipeline of COB can benefit from depth embeddings. We also include evaluation of object contour detection on the PASCAL VOC'12 database. In all cases, COB demonstrates state-of-the-art performance on contours and regions while being computationally efficient.

In a second set of experiments, we study the interplay of COB with various downstream recognition applications. We

• *K.-K. Maninis, J. Pont-Tuset, and L. Van Gool are with the Computer Vision Laboratory, ETHZ, Switzerland. P. Arbeláez is with the Department of Biomedical Engineering, Universidad de los Andes, Colombia. Contacts: see www.vision.ee.ethz.ch/~cvlsegmentation/*

ALIGN – EXHIBIT 1007

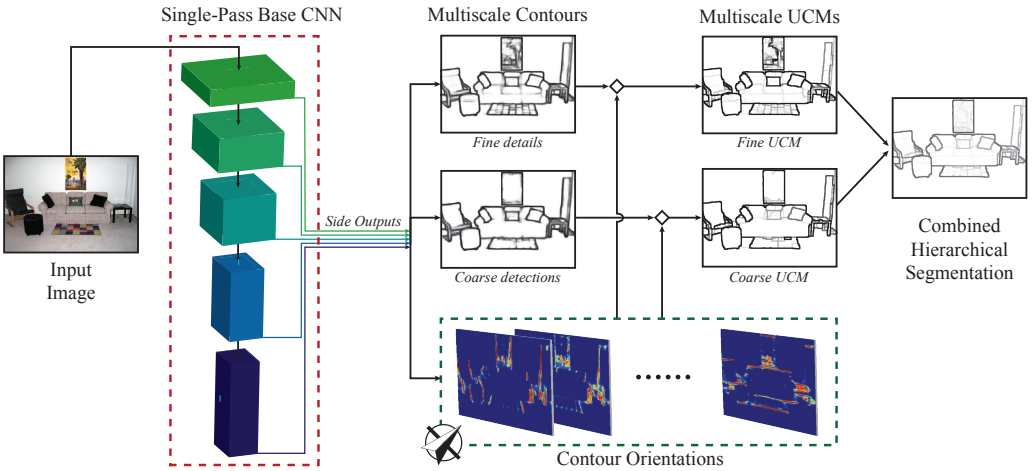


Fig. 1. **Overview of COB**: From a single pass of a base CNN, we obtain multiscale oriented contours. We combine them to build Ultrametric Contour Maps (UCMs) at different scales and fuse them into a single hierarchical segmentation structure.

use our hierarchical regions as input to the combinatorial grouping algorithm of [17] and obtain state-of-the-art segmented object proposals on PASCAL VOC'12 Segmentation by a significant margin. Furthermore, we provide empirical evidence for the generalization power of COB by evaluating our object proposals without any retraining in the even larger and more challenging MS-COCO [18] dataset, where we also report competitive results compared to the state of the art. We have also studied the effects of COB when coupled with well-known pipelines, showing that injecting COB detections to them lead to improvements on Semantic Segmentation and Object Detection. Finally, we report a new state of the art on Semantic Boundary detection.

Our approach to segmentation has also found application in retinal image segmentation [19], obtaining state-of-the-art and super-human performance in vessel and optic disc segmentation, which further highlights its generality.

The COB code, pre-computed results, pre-trained models, and benchmarks are publicly available at www.vision.ee.ethz.ch/~cvlsegmentation/.

2 Related Work

Contour Detection: Early approaches to contour detection relied on local gradient measurements in an image [20], [21], [22]. These simple edge detectors operate by applying local derivative filters on grayscale images. Gradient filtering was followed by detection of zero crossings [23], or by non-maximum suppression [24].

Such simple gradient techniques are unable to handle information captured by richer features such as color and texture [25], or Statistical Edges [26]. Martin et al. [25] define rich gradient operators out of color, brightness and texture, and use them as input to a logistic regression classifier. Their approach is extended by Arbeláez et al. [16], to combine contours at multiple scales.

Machine Learning techniques contributed to learnable features and classifiers that boosted contour detection performance, especially after the manual annotation of the BSDS database [16], [25]. The BEL algorithm [27] attempts to learn an edge classifier in the form of a probabilistic boosting tree. Kokkinos [28] trains an orientation-sensitive boundary detector using Multiple-Instance Learning. Ren and Bo [29] use patch representations automatically learned through sparse coding. Sketch Tokens [30] and Structured Edges [31] tackle both accuracy and speed, by using random forests to classify patches.

The latest wave of contour detectors takes advantage of deep learning to obtain state-of-the-art results [1], [2], [3], [4], [5], [6], [32]. Ganin and Lempitsky [6] use a deep architecture to extract features of image patches. They approach contour detection as a multi-class classification task, by matching the extracted features to predefined ground-truth features. The authors of [3], [4] make use of features generated by pre-trained CNNs to regress contours. They prove that object-level information provides powerful cues for the prediction of contours. Shen et al. [5] learn deep features using shape information. Xie and Tu [2] provide an end-to-end deep framework to boost the efficiency and accuracy of contour detection, using convolutional feature maps and a novel loss function. An extended version of their work, with many additional experiments can be found in [33]. Kokkinos [1] builds upon [2] and improves the results by tuning the loss function, running the detector at multiple scales, and adding globalization.

What many of the aforementioned methods have in common is that several simple components contribute to increased performance: (i) information at multiple scales [1], [16], [17], [34], (ii) contour orientation [16], [28], [30], [35], and (iii) end-to-end deep learning [1], [2]. COB is able to combine all of the above in a single pass of a CNN, producing an output that is richer than a linear combination of cues at different scales.

At the core of all these deep learning approaches lies a *base CNN*, starting from the seminal AlexNet [8] (8 layers), through the more complex VGGNet [10] (16 layers) and inception architecture of GoogLeNet [9] (22 layers), to the very recent and very deep ResNets [11] (up to 1001 layers). Image classification results, which originally motivated these architectures, have been continuously improved by exploring deeper and more complex networks. In this work, we present results both using VGGNet and ResNet, showing that COB is modular and can incorporate and benefit from future improvements in the base CNN.

Recent work has also explored weakly supervised or

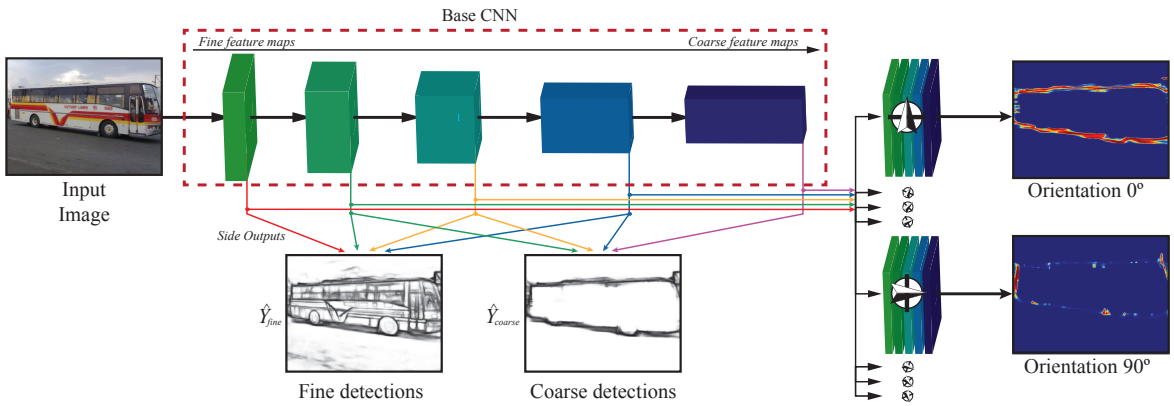


Fig. 2. **Our deep learning architecture** (best viewed in color). The connections show the different stages that are used to generate the multiscale contours. Orientations further require additional convolutional layers in multiple stages of the network.

unsupervised deep learning of contours: Khoreva et al. [36] learn from the results of generic contour detectors coupled with object detectors; and Li et al. [37] train contour detectors from motion boundaries acquired from video sequences. Yang et al. [38] use Conditional Random Fields (CRFs) to refine the inaccurately localized boundary annotations of PASCAL. Some works shift the domain of contours detection from abstract perceptual grouping to better defined tasks such as semantic or object contour detection [14], [36], [38]. Some methods also combine RGB-D cues for contour detection [31], [39], [40]. Extensive experiments on such benchmarks show that COB has an excellent performance even when shifting domains, showing state-of-the-art performance also in these new situations.

Hierarchical Image Segmentation and Grouping: One of the most studied category of methods for image segmentation are spectral methods, that rely on the generalized eigenvalue problem to solve a low-level pixel grouping problem. Notable approaches that fall into this category are Normalized Cuts [41], PMI [42], gPb [16], MCG [17]. Arbeláez et al. [16] showed the usefulness for jointly optimizing contours and regions (The duality between contours and regions was first studied by Najman and Schmitt [43]). Pont-Tuset et al. [17] leveraged multi-resolution contour detection and proved its interest for generating object proposals. COB also exploits the duality between contour detection and segmentation hierarchies. We differentiate from previous approaches mainly in two aspects. First, our sparse boundary representation translates into a clean and highly efficient implementation of hierarchical segmentation. Second, by leveraging high-level knowledge from the CNNs in the estimation of contour strength and orientation, our method benefits naturally from global information, which allows bypassing the globalization step (output of normalized cuts), a bottleneck in terms of computational cost, but a cornerstone of previous approaches.

3 Deep Multiscale Oriented Contours

CNNs are by construction multi-scale feature extractors. If one examines the standard architecture of a CNN consisting of convolutional and spatial pooling layers, it becomes clear that as we move deeper, feature maps capture more global information due to the decrease in resolution. For contour detection, this architecture implies local and fine-scale contours at shallow levels, coarser spatial resolution and larger receptive fields for the units when going deeper and, consequently, more global information for predicting boundary strength and orientation. CNNs have therefore a built-in globalization strategy for contour detection, analogous to the hand-engineered globalization of contour strength through spectral graph partitioning in [16], [17].

Figure 2 depicts how we make use of information provided by the intermediate layers of a CNN to detect contours and their orientations at multiple scales. Different groups of feature maps contain different, scale-specific information, which we combine to build a multiscale oriented contour detector. The remainder of this section is devoted to introducing the recent approaches to contour detection using deep learning, to presenting our CNN architecture to produce contour detection at different scales, and to explaining how we estimate the orientation of the edges; all in a single CNN forward pass at the image level.

Training deep contour detectors: The recent success of [2] is based on a CNN to accurately regress the contours of an image. Within this framework, the idea of employing a CNN in an image-to-image fashion without any post-processing has proven successful, and lead to a big leap in performance for the task of contour detection. Their network, HED, produces scale-specific contour images (side outputs) for different scales of a network, and combines their activations linearly to produce a contour probability map. Using the notation of the authors, we denote the training dataset by $S = \{(X_n, Y_n), n = 1, \ldots, N\}$, with X_n being the input image and $Y_n = \{y_j^{(n)}, j = 1, \ldots, |X_n|\}, y_j^{(n)} \in \{0, 1\}$ the predicted pixelwise labels. For simplicity, we drop the subscript n. Each of the M side outputs minimizes the objective function:

$$\ell_{side}^{(m)}\left(\mathbf{W},\mathbf{w}^{(m)}\right) = -\beta \sum_{j \in Y_+} \log P\left(y_j = 1 | X; \mathbf{W}, \mathbf{w}^{(m)}\right) - (1-\beta) \sum_{j \in Y_-} \log P\left(y_j = 0 | X; \mathbf{W}, \mathbf{w}^{(m)}\right) \quad (1)$$

where $\ell_{side}^{(m)}$ is the loss function for scale $m \in \{1, \ldots, M\}$, $\mathbf{W}$ denotes the standard set of parameters of the CNN, and $\{\mathbf{w}^{(m)}, m = 1, \ldots, M\}$ the corresponding weights of the the m-th side output. The multiplier β is used to handle the imbalance of the substantially greater number of background compared to contour pixels. Y_+ and Y_- denote the contour and background sets of the ground-truth Y, respectively. The probability $P(\cdot)$ is obtained by applying a sigmoid $\sigma(\cdot)$ to the activations of the side outputs

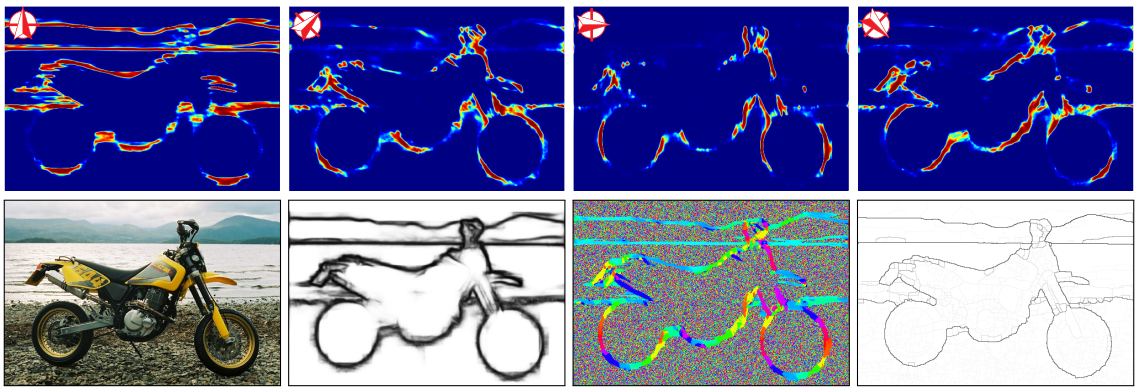


Fig. 3. Illustration of contour orientation learning. Row 1 shows the responses B_k for 4 out of the 8 orientation bins. Row 2, from left to right: original image, contour strength, learned orientation map into 8 orientations, and hierarchical boundaries.

$\hat{A}_{side}^{(m)} = \{a_j^{(m)}, j = 1, \ldots, |Y|\}$. In HED, the activations are finally fused linearly, as: $\hat{Y}_{fuse} = \sigma\left(\Sigma_{m=1}^{M} h_m \hat{A}_{side}^{(m)}\right)$ where $\mathbf{h} = \{h_m, m = 1, \ldots, M\}$ are the fusion weights. The fusion output is also trained to resemble the ground-truth applying the same loss function of Equation 1, by optimizing the complete set of parameters, including the fusion weights $\mathbf{h_m}$. We instead take advantage of the common CNN architectures to regress both the strength of the coarse and detailed (fine) contours, as well as the contour orientations. COB combines these output channels non-linearly to a single hierarchical segmentation. Inside this segmentation, the placement of each region in the hierarchy is determined by the strength of the boundaries to the neighbouring regions. All in all, COB efficiently combines contour strengths and orientations into a segmentation hierarchy which can further facilitate high-level vision tasks related to segmented object proposals. In the rest of the paper we use the class-balancing cross-entropy loss function of Equation 1.

Multiscale contours: We start from a deep network pre-trained on ImageNet [7], such as VGG [10] or ResNet [11]. The fully connected layers used for classification are removed, and so are the batch normalization layers, since we operate on one image per iteration. Therefore, the network consists mainly of convolutional layers coupled with ReLU activations, divided into 5 stages. We will refer to this architecture as the *base CNN* of our implementation. Each stage is handled as a different scale, since it contains feature maps of a similar size. At the end of a stage, there is a max pooling layer, which reduces the spatial dimensions of the produced feature maps to a half. As discussed before, the CNN naturally contains multiscale information, which we exploit to build a multiscale contour regressor.

We separately supervise the output of the last layer of each stage (side activation), comparing it to the ground truth using the loss function of Equation 1. This way, we enforce each side activation to produce an intermediate contour map at different resolution. The idea of supervising intermediate parts of a CNN has successfully been used in previous approaches, for a variety of tasks [2], [9], [44]. In the 5-scale base CNN illustrated in Figure 2, we linearly combine the side activations of the 4 finest and 4 coarsest scales to a fine-scale and a coarse-scale output ($\hat{Y}_{fine}$ and $\hat{Y}_{coarse}$, respectively) with trainable weights. The finer scale contains better localized contours, whereas the coarse scale leads to less noisy detections. To train the two sets of weights of the linear combinations, we freeze the pre-trained weights of the base CNN.

Estimation of Contour Orientations: In order to predict accurate contour orientations, we propose an extension of the CNN that we use to predict contour strength. We define the task as pixel-wise image-to-image multiscale classification into K bins. We connect K different branches (sub-networks) to the base network, each of which is associated with one orientation bin, and has access to feature maps that are generated from the intermediate convolutional layers at M different scales. We assign the parts of the CNN associated with each orientation a different task from the base network: classify the pixels of the contours that match a specific orientation. In order to design these orientation-specific subtasks, we classify each pixel of the human contour annotations into K different orientations. The orientation of each contour pixel is obtained by approximating the ground-truth boundaries with polygons, and assigning each pixel the orientation of the closest polygonal segment, as shown in Figure 5. As in the case of multiscale contours, the weights of the base network remain frozen when training these sub-networks.

Each sub-network consists of M convolutional layers, each of them appended on different scales of the base network. Thus we need $M * K$ additional layers. In our setup, we use $K = 8$ and $M = 5$. All K orientations are regressed in parallel, and since they are associated with a certain angle, we post-process them to obtain the orientation map. Specifically, the orientation map is obtained as:

$$O(x, y) = \mathcal{T}\left(\arg\max_k B_k\left(x, y\right)\right), k = 1, \ldots, K \quad (2)$$

where $B_k(x, y)$ denotes the response of the k-th orientation bin of the CNN at the pixels with coordinates (x, y) and $\mathcal{T}(\cdot)$ is the transformation function which associates each bin with its central angle. For the cases where two neighboring bins lead to strong responses, we compute the angle as their weighted average. At pixels where there is no response for any of the orientations, we assign random values between 0 and π, not to bias the orientations. The different orientations as well as the resulting orientation map (color-coded) are illustrated in Figure 3.

In [16], [17], [31] the orientations are computed by means of local gradient filters. In Section 5 we show that our

learned orientations are significantly more accurate and lead to better region segmentations.

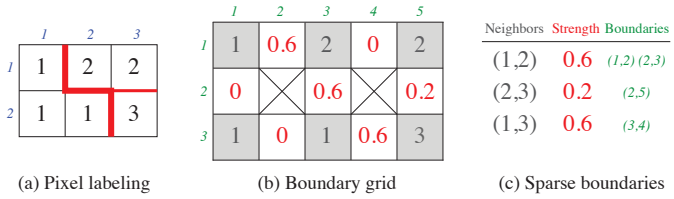
4 Fast Hierarchical Regions

This section is devoted to building an efficient hierarchical image segmentation algorithm from the multiscale contours and the orientations extracted in the previous section. We build on the concept of Ultrametric Contour Map (UCM) [16], which transforms a contour detection probability map into a hierarchical boundary map, which gets partitions at different granularities when thresholding at various contour strength values. Despite the success of UCMs, their low speed limits their applicability. We address this issue by using an alternative representation of an image partition which reduces the computation time of UCMs by an order of magnitude.

Sparse Boundary Representation of Hierarchies of Regions: An image partition is a clustering of the set of pixels into different sets, which we call regions. The most straightforward way of representing it in a computer is by a matrix of labels, as in the example in Figure 4(a), with three regions on an image of size 2×3. The boundaries of this partition are the edge elements, or *edgels*, between the pixels with different labels (highlighted in red). We can assign different *strengths* to these boundaries (thicknesses of the red lines), which indicate the *confidence* of that piece of being a boundary. By iteratively *erasing* these boundaries in order of increasing strength we obtain different partitions, which we call *hierarchy of regions*, or Ultrametric Contour Maps.

These boundaries are usually stored in the *boundary grid* (Figure 4(b)), a matrix of double the size of the image (minus one), in which the odd coordinates represent pixels (gray areas), and the positions in between represent boundaries (red numbers) and junctions (crossed positions). UCMs use this representation to store their boundary *strength* values, that is, each boundary position stores the threshold value beyond which that edgel *disappears* and the two neighboring regions merge. This way, simply *binarizing* a UCM we have a partition represented as a boundary grid. Continuing with the example in Figure 4, binarizing the UCM at 0.5 the edge between region 2 and 3 would disappear, that is, 2 and 3 would merge and create a new region.

This representation becomes very inefficient at run time, where the percentage of *activated* boundaries is very sparse. Not only are we wasting memory by storing those *empty* boundaries, but it also makes operating on them very inefficient by having to *sweep* over the entire matrix to perform a modification on a single boundary piece.



(a) Pixel labeling (b) Boundary grid (c) Sparse boundaries

Fig. 4. **Image Partition Representation:**
(a) Pixel labeling, each pixel gets assigned a region label. (b) Boundary grid, markers of the boundary positions. (c) Sparse boundaries, lists of boundary coordinates between neighboring regions.

Inspired by how sparse matrices are handled, we designed the *sparse boundaries* representation (Figure 4(c)). It stores a look-up table for pairs of neigboring regions, their boundary strength, and the list of coordinates the boundary occupies. Apart from being more compact in terms of memory, this representation enables efficient operations on specific pieces of a boundary, since one only needs to perform a search in the look-up table and scan the activated coordinates; instead of sweeping the whole boundary grid.

Fast Hierarchies from Multiscale Oriented Contours: We are inspired by the framework proposed in [17], in which a UCM is obtained from contours computed at different image scales and then combined into a single hierarchy. The motivation behind this work is that the UCMs obtained from downscaled images will focus on the coarse structures and ignore textures, so their localization accuracy will decrease. On the other hand, upscaled images will bring very good localization in the boundaries, but it will be harder to distinguish between the high- and low-level contents. To bring the best of the two worlds, [17] progressively *projects* the coarse hierarchies into the finer ones by adapting the high-level contours into the better localized ones. The final hierarchy keeps the high-level information while being snapped to the correctly localized low-level boundaries.

The deep CNN presented in Section 3 provides different levels of detail for the image contours, so instead of processing the image at multiple resolutions we use the different outputs that are computed in a single pass of the CNN to obtain different hierarchies that focus on high- and low-level features.

A drawback of the original framework [17], however, is that the manipulation of the hierarchies and their projection to different scales is very slow (in the order of seconds), so the operations on the UCMs had to be performed at a small subset of the contour strengths (from thousands to a few dozens). By using the fast sparse boundary representation, we can operate on all thousands of contour strengths, yielding better results at a fraction of the original cost. Moreover, we use the learned contour orientations for the computation of the Oriented Watershed Transform (OWT) [16], further boosting performance.

5 Experiments on Low-Level Applications

This section presents the empirical evidence that supports our approach for low-level applications (image segmentation and contour detection). First, Section 5.1 explores ablated and baseline techniques in order to isolate and quantify the improvements due to different components of our system. Section 5.2 further analyzes and evaluates the proposed contour orientations. In Section 5.3, Section 5.4, and Section 5.5 we compare our results against the state of the art in generic RGB image segmentation, RGB object boundary detection, and RGB-D image segmentation, respectively. In all three cases, we obtain the best results to date by a significant margin. Finally, Section 5.6 analyzes the effect of the various components in terms of speed on COB.

In terms of datasets, we extend the main BSDS benchmark [25] to the PASCAL Context dataset [15], which contains carefully localized pixel-wise semantic annotations for

the entire image on the PASCAL VOC 2010 detection trainval set. This results in 459 semantic categories across 10 103 images, which is an order of magnitude (20×) larger than the BSDS. In order to allow training and optimization of large capacity models, we split the data into train, validation, and test sets as follows: *VOC train* corresponds to the official PASCAL Context train with 4 998 images, *VOC val* corresponds to half the official PASCAL Context validation set with 2 607 images and *VOC test* corresponds to the second half with 2 498 images. In the remainder of the paper, we refer to this dataset division. Note that, in contrast to the BSDS, in this setting boundaries are defined between different semantic categories and not between their parts.

In all our experiments for boundary detection and image segmentation, we used the standard evaluation benchmark evaluating boundaries (F_b [25]) and regions (F_{op} [45]). Through the literature, the tolerance in the boundary localization metric F_b is altered (the `maxDist` parameter), depending on the database and the quality of the annotations. To avoid confusion, we list the value of this parameter for all our experiments in Table 1. Please also note that methods that produce open contours instead of regions can not be evaluated using the region measure F_{op}. In all the produced curves, markers indicate the optimal operating point that maximizes F_b and F_{op}. We used the publicly available *Caffe* [46] framework for training and testing CNNs, and all the state-of-the-art results are computed using the publicly-available code provided by the respective authors.

Training details: In our two-step training approach, we first train the base networks for the task of contour detection (coarse and fine). We use stochastic gradient descent with a momentum of 0.9 and weight decay of 0.0002 for 40k iterations. The base learning rate is set to 10^{-6}, and is divided by 10 after 30k iterations. After the first step is finished, the weights of the base network are frozen, and the layers of the orientation sub-network are connected and trained for an additional 10k iterations. Depending to the size of dataset we use different data augmentation strategies: flipping and rotation into 4 angles for PASCAL and NYUD-v2; flipping, rotation into 16 angles, and scaling into 3 scales [2] for BSDS500. In all cases, we initialize the network from ImageNet pre-trained weights. The same ground-truth boundaries are used for training both the fine and the coarse contours.

Database	Task	train	test	`maxDist`
BSDS500	Generic Segmentation	300	200	0.0075
VOC Context	Generic Segmentation	7 605	2 498	0.0075
VOC'12 Segm.	Object Contours	1 464	1 449	0.01
NYUD-v2	RGB-D Segmentation	795	654	0.011
SBD	Semantic Contours	8 498	2 857	0.02
VOC'12 Segm.	Semantic Segmentation	1 464	1 449	-
COCO	Object Proposals	-	40 504	-
VOC'07	Object Detection	5 011	4 952	-

TABLE 1
Datasets and Parameters: The list of databases used to evaluate our approach on various low-level and high-level tasks. We report the number of images used for training and testing our algorithm, along with the tolerance for contour localization used in the literature, when applicable. In all our experiments, we keep those numbers unchanged.

5.1 Control Experiments/Ablation Analysis

This section presents the control experiments and ablation analysis to assess the performance of all subsystems of our method. We train on *VOC train*, and evaluate on *VOC val* set. We report the standard F measure at Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS), as well as the Average Precision (AP), both evaluating boundaries (F_b [25]) and regions (F_{op} [45]).

Table 2 shows the evaluation results of the different variants, highlighting whether we include globalization and/or trained orientations. As a first baseline, we test the performance of MCG [17], which uses Structured Edges [31] as input contour signal. We then substitute SE by the newer HED [2], trained on *VOC train* as input contours and denote it MCG-HED. Note that the aforementioned baselines require multiple passes of the contour detector (3 scales).

In the direction of using the side outputs of the base CNN architecture as multiscale contour detections in one pass, we tested the baseline of naively taking the 5 side outputs directly as the contour detections. We trained both VGGNet [10] and ResNet50 [11] on *VOC train* and combined the 5 side outputs with our fast hierarchical regions of Section 4 (VGGNet-Side and ResNet50-Side).

We finally evaluate different variants of our system, as presented in Section 3. We first compare our system with two different base architectures: Ours(VGGNet) and Ours(ResNet50). We observe that the deeper architecture of ResNet translates into better boundaries and regions. Using the even deeper counterparts of ResNet lead to negligible gain in accuracy while significantly sacrificing speed.

We then evaluate the influence of our trained orientations and globalization, by testing the four possible combinations (the orientations are further evaluated in the next section). Our method using ResNet50 together with trained orientations leads to the best results both for boundaries and for regions. The experiments also show that, when coupled with trained orientations, globalization even decreases performance, so we can safely remove it and get a significant speed up. This behaviour arises from the fact that the image-to-image architecture of the base CNN already captures global information, addressing issues that could not be handled by local approaches, e.g., deleting internal contours of objects. Our technique with trained orientations and without globalization is therefore selected as our final system and will be referred to in the sequel as Convolutional Oriented Boundaries (COB).

5.2 Contour Orientation

We evaluate contour orientation results by the classification accuracy into 8 different orientations, to isolate their performance from the global system. We compute the ground-truth orientations as depicted in Figure 5 by means of the sparse boundaries representation. We then sweep all ground-truth boundary pixels and compare the estimated orientation with the ground-truth one. Since the orientations are not well-balanced classes (much more horizontal and vertical contours), we compute the classification accuracy per each of the 8 classes and then compute the mean.

Figure 6 shows the classification accuracy with respect to the confidence of the estimation. We compare our proposed

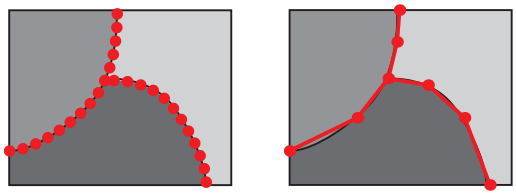


Fig. 5. **Polygon simplification:** From all boundary points (left) to simplified polygons (right), which are used to compute the ground-truth orientation robustly.

technique against the local gradient estimation used in previous literature [16], [17], [31]. As a baseline, we plot the result a random guess of the orientations would get. We observe that our estimation is significantly better than the previous approach. As a summary measure, we compute the area under the curve of the accuracy (ours 58.6%, local gradients 41.2%, random 12.5%), which corroborates the superior results from our technique.

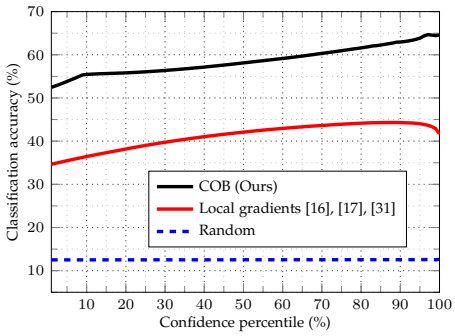


Fig. 6. **Contour orientation**: Classification accuracy into 8 bins.

5.3 Generic Image Segmentation

We present our results for contour detection and generic image segmentation on PASCAL Context [15] as well as on the BSDS500 [12], which is the most established benchmark for perceptual grouping.

PASCAL Context: We train COB in the *VOC train*, and perform hyper-parameter selection on *VOC val*. We report the final results on the unseen *VOC test* when trained on *VOC trainval*, using the previously tuned hyper-parameters. We compare our approach to several methods trained on the BSDS [2], [17], [31], [47] and we also retrain the current state-of-the-art contour detection methods HED [2] and the recent CEDN [38] on *VOC trainval* using the code provided by the respective authors.

Figure 7 presents the evaluation results of COB compared to the state of the art, showing that it outperforms all others by a considerable margin both in terms of boundaries and in terms of regions. The lower performance of the methods trained on the BSDS quantifies the difficulty of the task when moving to a larger and more challenging dataset.

BSDS500: We retrain COB using only the 300 *trainval* images of the BSDS, after data augmentation as suggested in [2], keeping the architecture decided in Section 5.1. For comparison to HED [2], we used the model that the authors provide online. We also compare with CEDN [38], by evaluating the results provided by the authors.

Figure 8 presents the evaluation results, which show that we also obtain state-of-the-art results in this dataset. The smaller margins are in all likelihood due to the fact that we almost reach human performance for the task of contour detection on the BSDS, which motivates the shift to PASCAL Context to achieve further progress in the field.

Qualitative Results: Figure 9 shows some qualitative results of our hierarchical contours. Please note that COB is capable of correctly distinguishing between internal contours and external, semantically meaningful boundaries.

Method	Global.	Orient.	Boundaries - F_b ODS	OIS	AP	Regions - F_{op} ODS	OIS	AP
MCG [17]	✓	✗	0.548	0.594	0.519	0.355	0.419	0.263
MCG-HED	✓	✗	0.691	0.727	0.693	0.459	0.520	0.374
VGGNet-Side	✓	✗	0.644	0.683	0.664	0.439	0.505	0.351
ResNet50-Side	✓	✗	0.676	0.711	0.681	0.456	0.521	0.374
Ours (VGGNet)	✗	✓	0.705	0.735	0.741	0.466	0.533	0.384
Ours (ResNet50)	✗	✗	0.734	0.767	0.757	0.475	0.545	0.405
Ours (ResNet50)	✓	✗	0.726	0.759	0.725	0.461	0.531	0.395
Ours (ResNet50)	✓	✓	0.732	0.763	0.731	0.481	**0.554**	**0.418**
Ours (ResNet50)	✗	✓	**0.737**	**0.768**	**0.758**	**0.483**	0.553	0.417

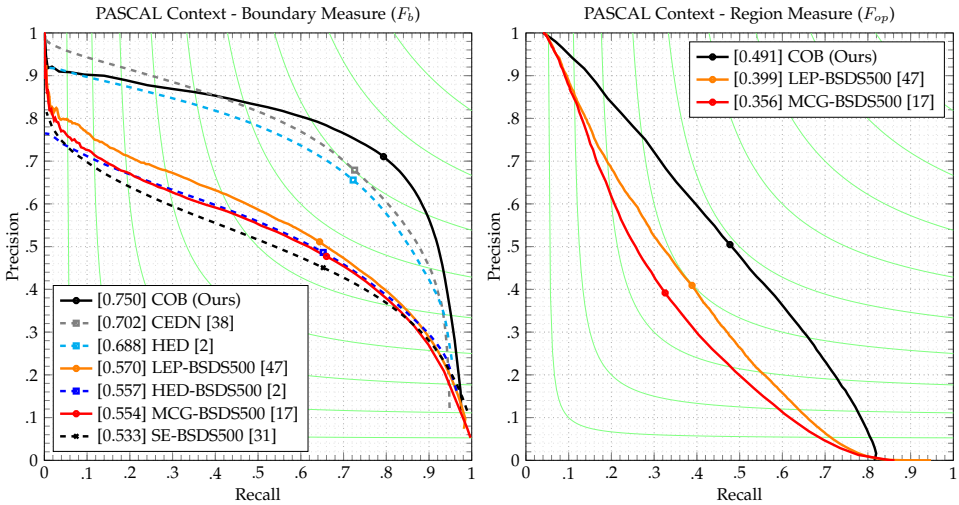
TABLE 2
Ablation analysis on *VOC val*: Comparison of different ablated and baseline versions of our system.

5.4 Object boundary detection

Concurrent works with the conference version of our paper [51] presented results on object boundary detection [36], [38] on the PASCAL VOC'12 Segmentation database. The database consists of 1464 training and 1449 validation images, including pixel-wise annotations of the instances and the semantic classes of the objects. The goal is to detect the boundaries of the objects that belong to the 20 classes of PASCAL, without distinguishing the semantics. Different from generic image segmentation, boundaries that do not belong to an object are treated as background.

We retrain COB on VOC'12 train set and report the results on the validation set. We use the instance level annotation of the database, and extract contours from the semantic segmentation annotations of the database. The uncertain areas (annotated with value of 255) are treated as background. We compare to several baselines, together with recent state-of-the-art results. Specifically, Khoreva et al. [36] retrained HED [2] on object contours, and Yang et al. [38] proposed a novel encoder-decoder architecture to tackle the same task. We evaluate the best pre-computed results provided by the authors in both cases. The results are quantified in Figure 11. We observe that COB obtains state-of-the-art results in all metrics. CEDN [38] performs better in the high precision regime. However, the authors used extra images from the SBD dataset [14] for training their detector. Also, CEDN is trained on an improved version of the ground truth, aligning the uncertain areas of VOC'12 with the the true image boundaries by using a CRF. We report results of COB trained only on VOC'12 train set, to be consistent with the results of Khoreva et al. [36]. In this experiment, we use `maxDist` of 0.01, as is adopted by the literature [36], [52].

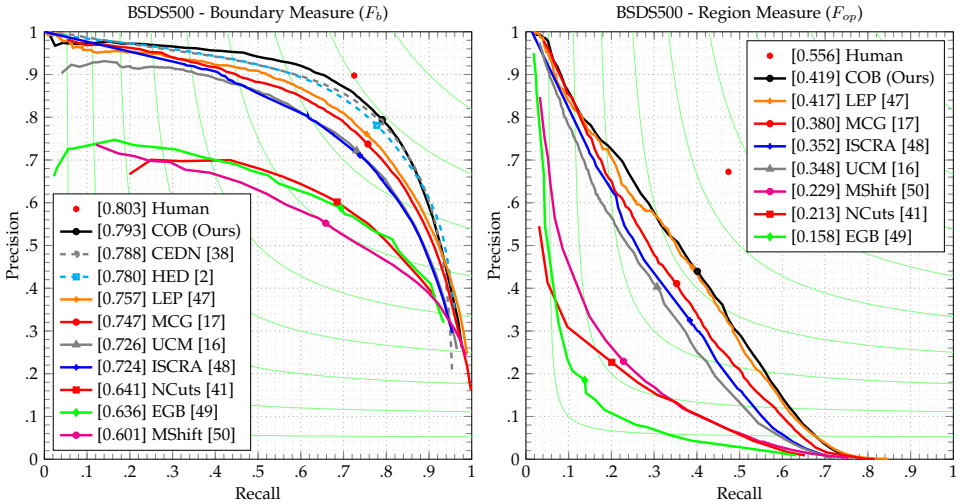
Figure 10 illustrates some qualitative results, as well as the differences of generic segmentation and object boundary



Method	Boundaries - F_b ODS	OIS	AP
COB	**0.750**	**0.781**	**0.773**
CEDN [38]	0.702	0.718	0.744
HED [2]	0.688	0.707	0.704
LEP-B [47]	0.570	0.636	0.547
HED-B [2]	0.557	0.588	0.518
MCG-B [17]	0.554	0.609	0.528
SE-B [31]	0.533	0.568	0.496

Method	Regions - F_{op} ODS	OIS	AP
COB	**0.491**	**0.565**	**0.439**
LEP-B [47]	0.399	0.461	0.304
MCG-B [17]	0.356	0.419	0.275

Fig. 7. **PASCAL Context *VOC test* Evaluation**: Precision-recall curves for evaluation of boundaries (F_b [25]), and regions (F_{op} [45]). Open contour methods in dashed lines and closed boundaries (from segmentation) in solid lines. ODS, OIS, and AP summary measures. Markers indicate the optimal operating point, where F_b and F_{op} are maximized.



Method	Boundaries - F_b ODS	OIS	AP
COB (Ours)	**0.793**	**0.820**	**0.859**
CEDN [38]	0.788	0.804	0.834
HED [2]	0.780	0.796	0.834
LEP [47]	0.757	0.793	0.828
MCG [17]	0.747	0.779	0.759
UCM [16]	0.726	0.760	0.727
ISCRA [48]	0.724	0.752	0.783
NCuts [41]	0.641	0.674	0.447
EGB [49]	0.636	0.674	0.581
MShift [50]	0.601	0.644	0.493

Method	Regions - F_{op} ODS	OIS	AP
COB (Ours)	**0.419**	**0.478**	**0.343**
LEP [47]	0.417	0.468	0.334
MCG [17]	0.380	0.433	0.271
ISCRA [48]	0.352	0.418	0.275
UCM [16]	0.348	0.385	0.235
MShift [50]	0.229	0.292	0.122
NCuts [41]	0.213	0.270	0.096
EGB [49]	0.158	0.240	0.080

Fig. 8. **BSDS500 Test Evaluation**: Precision-recall curves for evaluation of boundaries (F_b [25]), and regions (F_{op} [45]).

detection. We show our results on images of the VOC'12 val set using the model trained on PASCAL Context for generic image segmentation, and we compare qualitatively to the model retrained on the 20 classes of VOC'12 for object boundary detection. In the latter case, the detections are focused on the 20 object classes, disregarding strong contour cues of the background that are detected by the generic segmentation model.

5.5 RGB-D boundary detection on NYUD-v2 dataset

The NYUD (v2) dataset [53] consists of 1449 RGB-D indoor images, divided into splits of 795 training and 654 testing images, with the corresponding semantic and instance level segmentations. Gupta et al. [39] adopted this dataset for contour detection. In their experiments, they obtained the respective boundary annotations from the instance-level segmentations of the dataset. We evaluated the performance by using the standard benchmarks of BSDS. Following [2], [31], [40], we increased the tolerance for incorrect localizations from 0.0075 of the image diagonal to 0.11, to compensate for inaccurate annotations of boundaries.

We use the extra information of depth to train different variants of COB on the NYUD dataset. Gupta et al. [40] used the camera parameters of the images to encode the depth information in three channels: horizontal disparity, height above ground, and the angle of the local surface normal with the inferred gravity direction at each pixel (HHA). We retrain three different variants of the CNN: (a) Only using RGB data (ResNet50-RGB), (b) Incorporating depth information into a fourth channel (ResNet50-RGBD), and (c) Concatenating RGB and HHA channels and operate on 6 channels directly (ResNet50-RGB-HHA). Figure 13 illustrates an overview of the data, along with depth and HHA features that we used, as well as the results obtained by COB. In Figure 12 we show the ablation analysis by directly evaluating the CNN output, without any post-processing. We observe that the CNNs retrained on RGB and RGB-HHA channels obtain significantly better results than the one trained on RGB-D data, showing that HHA features provide an appropriate encoding for depth information. We retrain the full pipeline of COB (including orientations) on NYUD and we report the precision-recall curves. We

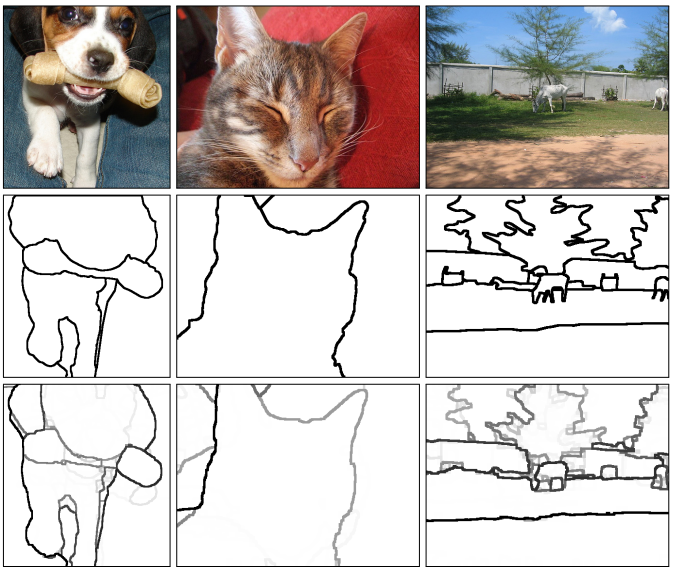


Fig. 9. **Qualitative results on PASCAL - Hierarchical Regions**. Row 1: original images, Row 2: ground-truth boundaries, Row 3: hierarchical regions with COB.

compare with various state-of-the-art methods, showing significant improvements. Specifically, we compare with the SE [31] detector retrained on the RGB-D data of NYUD, the detector proposed by [40] trained on RGB and depth normal gradients, and the best result reported on NYUD by HED [2], where the authors trained two different variants of the detector on RGB and HHA modalities respectively and averaged the obtained results. For completeness, we report results obtained by the original MCG [17] without any retraining on NYUD. The best result is obtained by the variant of COB trained on both RGB and HHA modalities. Compared to its RGB-only counterpart, the particular model achieves higher accuracy, suggesting that the depth embeddings are useful cues to discern contours when RGB modality alone is unable to do so. It is noteworthy that the post-processing step (orientations and UCMs) further boosts the performance of COB. For example, performance increases from 0.745 (ResNet50-RGB-HHA) to 0.784 (COB-RGB-HHA) by plugging in the orientations and the UCM pipeline to the trained ResNet50 architecture. We also report the results of the model trained on PASCAL Context (COB-PC) and operating only on RGB data, showing that it performs fairly well without any retraining on the NYUD dataset.

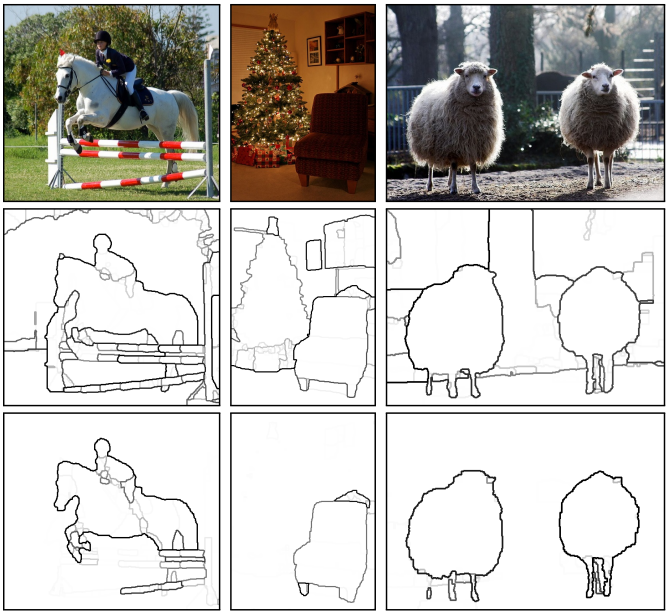


Fig. 10. **Qualitative results for Object Boundaries**. Row 1: original images, Row 2: Generic Image Segmentation results, Row 3: Object Boundary results.

5.6 Efficiency Analysis

Contour detection and image segmentation, as a preprocessing step towards high-level applications, need to be computationally efficient. The previous state-of-the-art in hierarchical image segmentation [16], [17] was of limited use in practice due to its computational load.

As a core in our system, the forward pass of our network to compute the contour strength and 8 orientations takes 0.28 seconds on a NVidia Titan X GPU. Table 3 shows the timing comparison between the full system COB (Ours) and some related baselines on PASCAL Context. We divide the timing into different relevant parts, namely, the contour detection step, the Oriented Watershed Transform (OWT) and Ultrametric Contour Map (UCM) computation, and the globalization (normalized cuts) step.

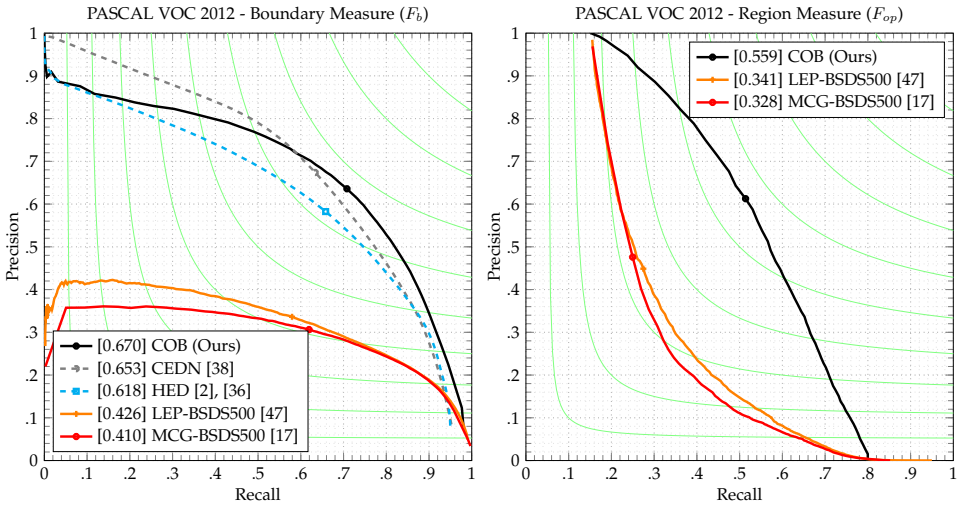
Steps	(1) MCG [17]	(2) MCG-HED	(3) Fast UCMs	(4) COB (Ours)
Contours	3.08	0.39*	0.39*	0.28*
OWT, UCM	11.33	11.58	1.63	0.51
Globalize	9.96	9.97	9.92	0.00
Total Time	24.37	21.94	11.94	**0.79**

TABLE 3
Timing experiments: Comparing our approach to different baselines. Times computed using a GPU are marked with an asterisk.

Column (1) shows the timing for the original MCG [17], which uses Structured Edges (SE) [31]. As a first baseline, Column (2) displays the timing of MCG if we naively substitute SE by HED [2] at the three scales (running on a GPU). By applying the sparse boundaries representation we reduce the UCM and OWT time from 11.58 to 1.63 seconds (Column (3)). Our final technique COB, in which we remove the globalization step, computes the three scales in one pass and add contour orientations, takes 0.79 seconds in mean. Overall, comparing to previous state-of-the-art, we get a significant improvement at a fraction of the computation time (24.37 to 0.79 seconds).

6 Experiments on High-Level Applications

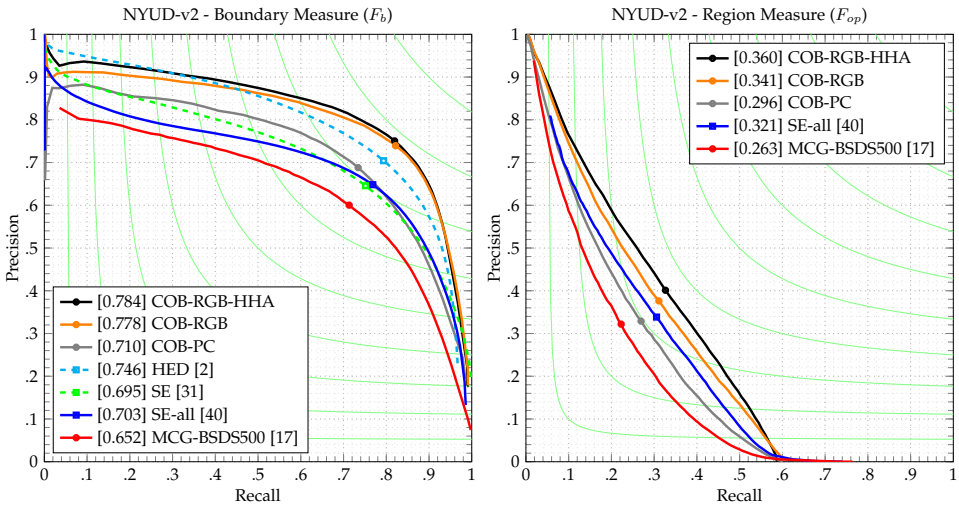
This section is dedicated to present the interaction of COB boundaries and segments with higher vision tasks. In Section 6.1 we evaluate COB as object proposals by plugging in the detected UCMs into the combinatorial goruping pipeline of MCG [17]. In Section 6.2 we study the interplay of our boundary detector with semantic contours and semantic segmentation by combining COB with Dilated Network [68] and PSPNet [69], and in Section 6.3 we couple the COB



Method	Boundaries - F_b ODS	OIS	AP
COB (Ours)	**0.670**	**0.728**	**0.687**
CEDN [38]	0.653	0.683	0.686
HED [2], [36]	0.618	0.650	0.628
LEP-B [47]	0.426	0.517	0.325
MCG-B [17]	0.410	0.486	0.294

Method	Regions - F_{op} ODS	OIS	AP
COB (Ours)	**0.559**	**0.644**	**0.391**
LEP-BSDS500 [47]	0.341	0.456	0.156
MCG-BSDS500 [17]	0.328	0.431	0.140

Fig. 11. **PASCAL VOC 2012**: Precision-recall curves for boundaries (F_b [25]), and regions (F_{op} [45]). ODS, OIS, and AP summary measures.



Method	Boundaries - F_b ODS	OIS	AP
COB-RGB-HHA	**0.784**	**0.805**	**0.825**
COB-RGB	0.778	0.799	0.814
COB-PC	0.710	0.735	0.734
HED [2]	0.746	0.761	0.786
SE [31]	0.695	0.708	0.719
SE-all [40]	0.703	0.716	0.707
MCG-B [17]	0.652	0.681	0.613

Method	Regions - F_{op} ODS	OIS	AP
COB-RGB-HHA	**0.360**	**0.398**	**0.270**
COB-RGB	0.341	0.376	0.252
COB-PC	0.296	0.332	0.203
SE-all [40]	0.321	0.354	0.179
MCG-BSDS500 [17]	0.263	0.296	0.159

Ablation: Method	Boundaries - F_b ODS	OIS	AP
ResNet50-RGB-HHA	0.745	**0.762**	**0.792**
ResNet50-RGB	**0.746**	0.761	0.789
ResNet50-RGBD	0.683	0.699	0.681

Fig. 12. **NYUD-v2 test**: Precision-recall curves for evaluation of boundaries (F_b [25]), and regions (F_{op} [45]). ODS, OIS, and AP summary measures.

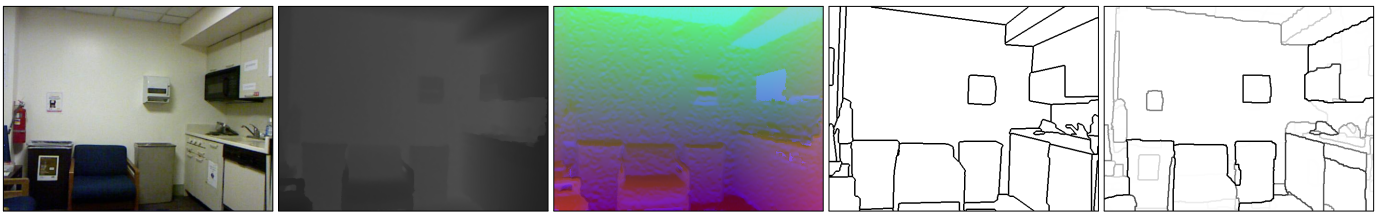


Fig. 13. **Data and results on NYUD-v2**. From left to right: RGB image, depth, HHA features [40], ground truth, and COB detections

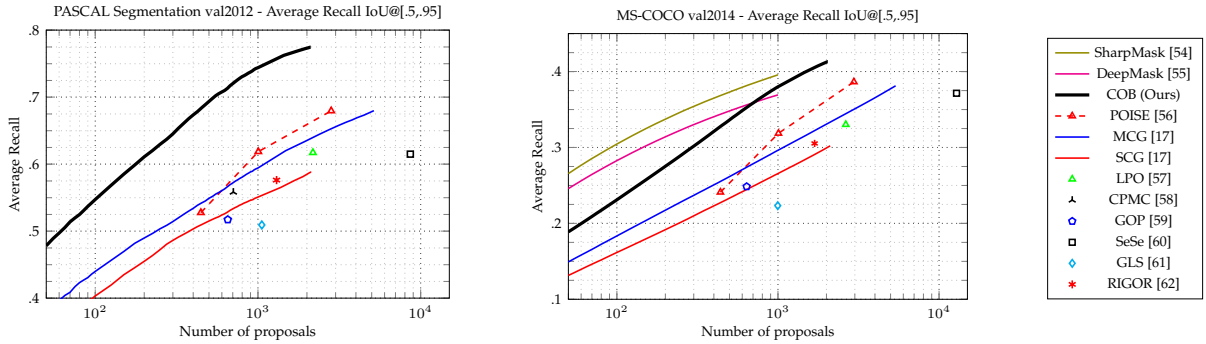


Fig. 14. **Segmented object proposals evaluation on PASCAL Segmentation val and MS-COCO val**: Dashed lines refer to methods that do not provide a ranked set of proposals, but they need to be re-parameterized.

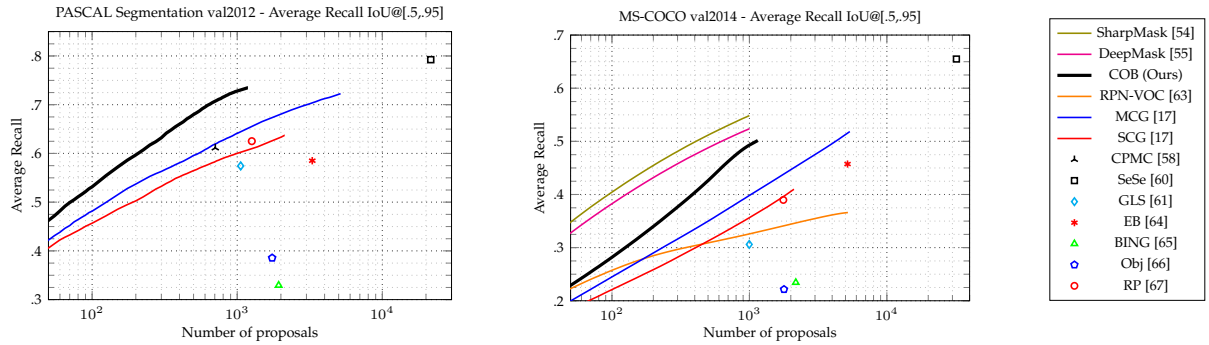


Fig. 15. **Bounding-box object proposals evaluation on PASCAL Segmentation val and MS-COCO val**: Note that COB is designed to detect segmented object proposals and not bounding-box proposals.

proposals with the Fast-RCNN [70] pipeline for object detection. In all cases, we show that COB co-operates well with existing approaches by improving their performance.

6.1 Object Proposals

Object proposals are an integral part of current object detection and semantic segmentation pipelines [63], [70], [71], as they provide a reduced search space on locations, scales, and shapes over the image. This section evaluates COB as a segmented and bounding box proposal technique, when using our high-quality region hierarchies in conjunction with the combinatorial grouping framework of MCG [17]. In terms of segmented object proposals, we compare against the most recent techniques SharpMask [54], DeepMask [55], POISE [56], MCG and SCG [17], LPO [57], GOP [59], SeSe [60], GLS [61], and RIGOR [62]. In terms of bounding box proposals, we compare also against SharpMask [54], DeepMask [55], EB [64], RPN [63] MCG and SCG [17], LPO [57], BING [65], SeSe [60], GLS [61], RIGOR [62], Obj [66], and RP [67]. Recent thorough comparisons of object proposal generation methods can be found in [72], [73].

We perform experiments on the PASCAL 2012 Segmentation dataset [13] and on the bigger and more challenging MS-COCO [18] (val2014 set). The hierarchies and combinatorial grouping are trained on PASCAL Context. To assess the generalization capability, we evaluate on MS-COCO, which contains a large number of previously unseen categories, without further retraining.

Figure 14 shows the average recall [73] with respect to the number of object proposals. In PASCAL VOC'12 Segmentation, the absolute gap of improvement of COB is at least of +13% with the second-best technique, and consistent in all the range of number of proposals. In MS-COCO, even though we did not train on any MS-COCO image, COB reaches competitive results for the task, with only very recent techniques [54], [55] reaching higher Average Recall when evaluating a low number of proposals. This shows that our contours, regions, and proposals are properly learning a generic concept of object rather than some specific categories.

Figure 15 shows the evaluation in terms of bounding box object proposals. COB is less competitive in terms of box proposals, however the algorithm was not specifically designed for detecting bounding boxes. We also show the comparison to RPN [63], which is trained on VOC'07, and thus does not generalize well in the classes of COCO.

6.2 Semantic Boundaries and Semantic Segmentation

The task of Semantic Boundaries, introduced by [14], requires not only detecting the boundaries, but also associating a semantic class to them. It can be thought as a combination of Boundary Detection and Semantic Segmentation, where except for the binary information of boundaries, one needs to label each of the detected pixels with the corresponding semantic class. The common approach to this task is to separately approach semantic segmentation and contour detection, and fuse the results of the two tasks [4], [14], [32]. Hariharan et al [14] tackled the task with generic object detectors and bottom up contours. Bertasius et al. [4], [32] show that results can be significantly improved when using deep-learning based semantic segmenters and contour detectors. Kokkinos [74] approaches the task with fully-convolutional networks trained end to end, although the results do not reach the current state of the art.

We also follow the most common approach of mixing the two tasks. We couple the COB boundaries with Semantic Segmentation results by dilated convolutions [68]. Specifically, we mask the boundaries with Semantic Segmentation results, with a tolerance of 0.02 of the image diagonal.

We report results on the SBD [14] database, for semantic boundary detection, by using the standard benchmark. Tables 4 and 5 compare the results among various methods, in both metrics used in the benchmark (mean maximal F-measure and Average Precision) for all classes. The combination of COB with [68], denoted with COB-dil, achieves state-of-the-art results in both metrics. For fair comparison, we also include the results obtained by evaluating the semantic segmentation results obtained by [68] directly as contours. We show that COB fairly improves the result.

Having explored the performance of COB combined with the Dilated Convolution network on Semantic Boundaries, it is interesting to investigate the dual task: the effects of COB in semantic segmentation. We treat the COB UCMs as superpixels, by applying a low value threshold (0.1) to the hierarchy, which results in high recall. We then snap the semantic segmentation results to the superpixels by majority voting of the regions, i.e superpixels that overlap more than 50% with the semantic class, are assigned the corresponding label. Table 6 reports the effects of such snapping on Semantic Segmentation, on the validation split of PASCAL VOC Segmentation dataset. In addition to the Dilated network, we also explored the most recent PSPNet [69] as the base semantic segmenter. Results improve consistently

Technique	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	MBike	Person	Plant	Sheep	Sofa	Train	TV	Mean maxF
COB-dil	**84.2**	**72.3**	**81.0**	64.2	**68.8**	81.7	71.5	**79.4**	**55.2**	**79.1**	40.8	**79.9**	**80.4**	**75.6**	**77.3**	**54.4**	**82.8**	51.7	72.1	62.4	**70.7**
DilatedConv [68]	83.7	71.8	78.8	**65.5**	66.3	**82.6**	**73.0**	77.3	47.3	76.8	37.2	78.4	79.4	75.2	73.8	46.2	79.5	46.6	**76.4**	**63.8**	69.0
BNF [32]	76.7	60.5	75.9	60.7	63.1	68.4	62.0	74.3	54.1	76.0	**42.9**	71.9	76.1	68.3	70.5	53.7	79.6	**51.9**	60.7	60.9	65.4
HFL [4]	73.6	61.1	74.2	57.0	58.7	70.2	60.8	71.8	46.3	72.1	36.0	70.9	72.9	67.5	69.9	44.1	73.1	42.2	62.2	60.4	62.2
[36]	65.9	54.1	63.6	47.9	47.0	60.4	50.9	56.5	40.4	56.0	30.0	57.5	58.0	57.4	59.5	39.0	64.2	35.4	51.0	42.4	51.9
[14]	41.5	46.7	15.6	17.1	36.5	42.7	40.3	22.6	18.8	27.0	12.5	18.2	35.4	29.4	48.1	13.8	26.9	11.0	22.0	31.3	27.9

TABLE 4
SBD val evaluation: Semantic contours results: maximal F_b per class and mean maximal F_b is reported for all methods.

Technique	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	MBike	Person	Plant	Sheep	Sofa	Train	TV	Mean AP
COB-dil	**85.7**	**69.3**	**77.6**	**59.7**	**64.1**	**82.9**	**69.7**	**80.5**	**41.8**	**79.4**	**26.0**	**78.9**	**81.5**	**74.7**	**77.3**	**43.8**	**82.8**	**39.3**	**73.3**	**56.4**	**67.2**
BNF [32]	75.9	46.0	70.5	48.9	48.6	65.3	53.5	65.2	38.2	69.7	20.9	62.3	72.2	56.6	63.3	38.5	75.7	31.4	45.6	48.1	54.8
HFL [4]	71.3	54.9	68.8	45.6	48.3	70.9	56.5	65.6	29.0	65.8	17.6	64.3	68.3	64.0	65.6	28.8	66.5	25.8	59.5	49.8	54.3
[36]	67.1	50.5	62.2	42.1	38.9	57.8	47.7	53.7	32.1	52.3	17.5	53.1	56.0	53.2	57.7	29.4	62.2	24.0	46.2	32.8	46.8
[14]	38.4	38.9	8.6	9.3	23.0	37.1	33.6	18.4	11.5	16.0	5.1	12.2	29.0	21.3	46.9	7.2	15.8	5.6	14.4	21.4	20.7

TABLE 5
SBD val evaluation: Semantic contours results: Average Precision (AP) per class and mean AP (mAP) is reported for all methods.

Technique	BG	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	MBike	Person	Plant	Sheep	Sofa	Train	TV	Mean
COB-dil	**93.5**	**90.3**	**39.7**	**83.2**	**66.2**	**68.9**	**92.6**	**84.6**	**89.2**	**36.9**	**84.7**	**53.1**	**82.9**	**87.0**	**83.1**	**86.3**	**54.7**	**84.8**	**45.7**	**84.6**	**68.9**	**74.3**
DilatedConv [68]	92.8	87.1	39.2	79.6	65.9	66.3	90.0	82.5	85.3	36.2	81.7	51.7	78.1	83.8	80.2	83.4	50.5	82.6	43.1	83.8	65.3	71.9
COB-PSP	**95.4**	**90.9**	**44.8**	90.2	**76.1**	**84.1**	96.1	**92.1**	**95.3**	45.6	**95.4**	**59.9**	**92.0**	**93.2**	90.8	**90.1**	**68.0**	**93.4**	**50.2**	**93.3**	79.8	**81.7**
PSPNet [69]	95.3	90.7	44.4	90.2	74.8	83.4	**96.3**	92.0	95.0	**46.4**	94.6	59.1	91.9	92.5	**91.0**	89.9	66.0	91.6	50.2	93.0	**80.0**	81.3

TABLE 6
PASCAL VOC Segmentation val evaluation: Effect of COB on Semantic Segmentation. Per-class IoU and mean IoU are reported.

Technique	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	MBike	Person	Plant	Sheep	Sofa	Train	TV	Mean
COB	69.5	76.8	**69.7**	53.3	**44.6**	**80.5**	**81.3**	**83.1**	**45.3**	**74.2**	**69.4**	**80.1**	**84.2**	**76.7**	**72.8**	**35.9**	**67.1**	**68.4**	**75.1**	65.4	**68.7**
SeSe	**76.0**	76.8	65.3	**54.6**	38.0	76.5	78.2	81.6	40.1	74.1	66.5	78.9	81.8	74.5	66.2	32.9	65.6	67.7	73.4	**66.8**	66.8

TABLE 7
VOC 2007 test evaluation: Object Detection performance (mAP) of Fast-RCNN [70], using object proposals from [60] (original) or COB.

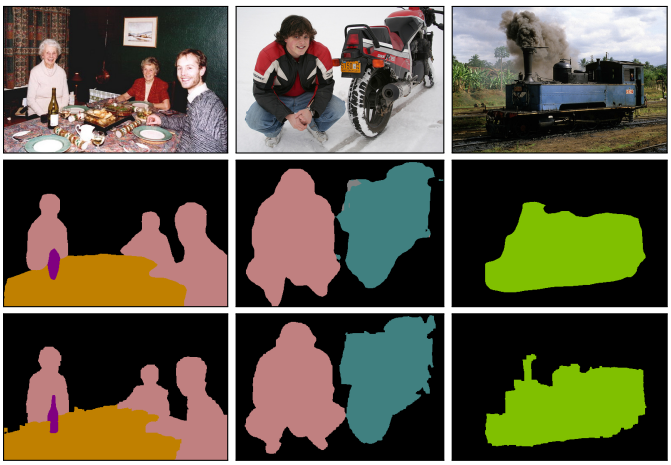


Fig. 16. **Qualitative results for Semantic Segmentation**. Row 1: original images, Row 2: Dilated Convolution Network, Row 3: Dilated Network with COB superpixels.

almost for all the classes in both cases, indicating that COB superpixels are further refining the semantic segmentation results on boundary locations. We observe a more moderate improvement in the PSPNet results, mainly because of the reduced false detections. We have excluded all images of VOC Segmentation val set for training the COB model.

In Figure 16 we present some qualitative results. Snapping to COB superpixels improves mainly on boundary locations, as well as on noisy semantic segmentation detections in places where COB superpixels are not present.

6.3 COB Object Proposals for Object Detection

Object Proposals have been extensively used to facilitate object detection [63], [70], [71]. Most common pipelines use object proposals in the form of a bounding box to regress a class score and a refined prediction of the bounding box locations. Even though our approach provides segmented object proposals from a hierarchy of regions, it is possible to study their effect on common object detection pipelines by simply extracting the bounding box around them.

We evaluate the bounding box proposals generated by COB by feeding them into the Fast-RCNN [70] pipeline for Object Detection. The original approach uses the VGG network [10] together with the box proposals generated by the Selective Search [60] algorithm to predict class probability and refine the localization for each of them. The final detection performance is evaluated by performing non-maximum suppression on the detections.

Experiments are performed on the VOC'07 detection database. The database consists of 5011 training, and we report the performance on its 4952 testing images. In our experiments, we change the box proposals of Selective Search, to the ones generated by COB. We keep all the hyper-parameters of the original approach unchanged, both at training and test times. Table 7 quantitatively evaluates the effects of COB proposals in performance. We observe improvements in object detection performance (mean Average Precision - mAP), which further proves the high quality of the proposals generated by COB. We would like to emphasize that the latest developments on Object Detection

use joint training of bounding box proposals and object class scores [63], [75], [76], [77], which together with training on external data achieves much higher results. Instead, we focus on proving the high quality of COB proposals compared to other object proposal techniques.

7 Conclusions

In this work, we have developed an approach to detect contours at multiple scales, together with their orientations, in a single forward pass of a convolutional neural network. We provide a fast framework for generating region hierarchies by efficiently combining multiscale oriented contour detections, thanks to a new sparse boundary representation. We shift from the BSDS to PASCAL to unwind all the potential of data-hungry methods such as CNNs and by observing that BSDS is close to saturation.

Our technique achieves state-of-the-art performance by a significant margin for contour detection, the estimation of their orientation, and generic (RGB and RGB-D) image segmentation. We show that our architecture is modular by using two different CNN base architectures, which suggests that it will be able to transfer further improvements in CNN base architectures to perceptual grouping. We also show that our method does not require globalization, which was a speed bottleneck in previous approaches. The generalization of COB was further demonstrated when applied to high-level vision tasks (object proposals, object detection, and semantic contours and segmentation) in combination with recent pipelines, where the results are improved in all cases.

All our code, CNN models, pre-computed results, dataset splits, and benchmarks are publicly available at www.vision.ee.ethz.ch/~cvlsegmentation/.

Acknowledgments

Research funded by the EU Framework Programme for Research and Innovation Horizon 2020 (Grant No. 645331, EurEyeCase), and by the Swiss Commission for Technology and Innovation (CTI, Grant No. 19015.1 PFES-ES, NeGeVA). The authors gratefully acknowledge support by armasuisse and thank NVidia for donating the GPUs used in this work.

References

[1] I. Kokkinos, "Pushing the boundaries of boundary detection using deep learning," in *ICLR*, 2016.
[2] S. Xie and Z. Tu, "Holistically-nested edge detection," in *ICCV*, 2015.
[3] G. Bertasius, J. Shi, and L. Torresani, "Deepedge: A multi-scale bifurcated deep network for top-down contour detection," in *CVPR*, 2015.
[4] ——, "High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision," in *ICCV*, 2015.
[5] 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 *CVPR*, 2015.
[6] Y. Ganin and V. Lempitsky, "N^4-fields: Neural network nearest neighbor fields for image transforms," in *ACCV*, 2014.
[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *IJCV*, 2015.
[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.
[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.
[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.
[12] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *ICCV*, 2001.
[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.
[14] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *ICCV*, 2011.
[15] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, "The role of context for object detection and semantic segmentation in the wild," in *CVPR*, 2014.
[16] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *TPAMI*, vol. 33, no. 5, pp. 898–916, 2011.
[17] J. Pont-Tuset, P. Arbeláez, J. Barron, F.Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *TPAMI*, vol. 39, no. 1, pp. 128 – 140, 2017.
[18] T. Lin, M. Maire, S. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *arXiv:1405.0312*, 2014.
[19] K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. V. Gool, "Deep retinal image understanding," in *MICCAI*, 2016.
[20] L. G. Roberts, "Machine perception of three-dimensional solids," Ph.D. dissertation, MIT, 1963.
[21] J. Kittler, "On the accuracy of the sobel edge detector," *Image and Vision Computing*, vol. 1, no. 1, pp. 37–42, 1983.
[22] J. M. Prewitt, "Object enhancement and extraction," *Picture processing and Psychopictorics*, vol. 10, no. 1, pp. 15–19, 1970.
[23] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Royal Soc. of London*, vol. 207, no. 1167, pp. 187–217, 1980.
[24] J. Canny, "A computational approach to edge detection," *TPAMI*, no. 6, pp. 679–698, 1986.
[25] D. Martin, C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *TPAMI*, vol. 26, no. 5, pp. 530–549, 2004.
[26] S. Konishi, A. L. Yuille, J. M. Coughlan, and S. C. Zhu, "Statistical edge detection: Learning and evaluating edge cues," *TPAMI*, vol. 25, no. 1, pp. 57–74, 2003.
[27] P. Dollar, Z. Tu, and S. Belongie, "Supervised learning of edges and object boundaries," in *CVPR*, 2006.
[28] I. Kokkinos, "Boundary detection using F-measure-, filter-and feature-(f3) boost," in *ECCV*, 2010.
[29] X. Ren and L. Bo, "Discriminatively trained sparse code gradients for contour detection," in *NIPS*, 2012.
[30] J. J. Lim, C. L. Zitnick, and P. Dollár, "Sketch tokens: A learned mid-level representation for contour and object detection," in *CVPR*, 2013.
[31] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *TPAMI*, vol. 37, no. 8, pp. 1558–1570, 2015.
[32] G. Bertasius, J. Shi, and L. Torresani, "Semantic segmentation with boundary neural fields," in *CVPR*, 2016.
[33] S. Xie and Z. Tu, "Holistically-nested edge detection," *International Journal of Computer Vision*, pp. 1–16, 2017.
[34] X. Ren, "Multi-scale improves boundary detection in natural images," *ECCV*, 2008.
[35] S. Hallman and C. C. Fowlkes, "Oriented edge forests for boundary detection," in *CVPR*, 2015.
[36] A. Khoreva, R. Benenson, M. Omran, M. Hein, and B. Schiele, "Weakly supervised object boundaries," in *CVPR*, 2016.
[37] Y. Li, M. Paluri, J. M. Rehg, and P. Dollár, "Unsupervised learning of edges," in *CVPR*, 2016.
[38] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang, "Object contour detection with a fully convolutional encoder-decoder network," in *CVPR*, 2016.
[39] S. Gupta, P. Arbeláez, and J. Malik, "Perceptual organization and recognition of indoor scenes from RGB-D images," in *CVPR*, 2013.

[40] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *ECCV*, 2014.
[41] J. Shi and J. Malik, "Normalized cuts and image segmentation," *TPAMI*, vol. 22, no. 8, 2000.
[42] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson, "Crisp boundary detection using pointwise mutual information," in *ECCV*, 2014.
[43] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation," *TPAMI*, vol. 18, no. 12, pp. 1163–1173, 1996.
[44] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," *arXiv preprint arXiv:1409.5185*, 2014.
[45] J. Pont-Tuset and F. Marques, "Supervised evaluation of image segmentation and object proposal techniques," *TPAMI*, vol. 38, no. 7, pp. 1465–1478, 2016.
[46] 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.
[47] Q. Zhao, "Segmenting natural images with the least effort as humans," in *BMVC*, 2015.
[48] Z. Ren and G. Shakhnarovich, "Image segmentation by cascaded region agglomeration," in *CVPR*, 2013.
[49] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, p. 2004, 2004.
[50] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *TPAMI*, vol. 24, no. 5, pp. 603 –619, 2002.
[51] K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. V. Gool, "Convolutional oriented boundaries," in *ECCV*, 2016.
[52] J. Uijlings and V. Ferrari, "Situational object boundary detection," in *CVPR*, 2015.
[53] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.
[54] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár, "Learning to refine object segments," in *ECCV*, 2016.
[55] P. O. Pinheiro, R. Collobert, and P. Dollar, "Learning to segment object candidates," in *NIPS*, 2015.
[56] A. Humayun, F. Li, and J. M. Rehg, "The middle child problem: Revisiting parametric min-cut and seeds for object proposals," in *ICCV*, 2015.
[57] P. Krähenbühl and V. Koltun, "Learning to propose objects," in *CVPR*, 2015.
[58] J. Carreira and C. Sminchisescu, "CPMC: Automatic object segmentation using constrained parametric min-cuts," *TPAMI*, vol. 34, no. 7, pp. 1312–1328, 2012.
[59] P. Krähenbühl and V. Koltun, "Geodesic object proposals," in *ECCV*, 2014.
[60] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.
[61] P. Rantalankila, J. Kannala, and E. Rahtu, "Generating object segmentation proposals using global and local search," in *CVPR*, 2014.
[62] A. Humayun, F. Li, and J. M. Rehg, "RIGOR: Recycling Inference in Graph Cuts for generating Object Regions," in *CVPR*, 2014.
[63] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.
[64] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014.
[65] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. H. S. Torr, "BING: Binarized normed gradients for objectness estimation at 300fps," in *CVPR*, 2014.
[66] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *TPAMI*, vol. 34, pp. 2189–2202, 2012.
[67] S. Manén, M. Guillaumin, and L. Van Gool, "Prime Object Proposals with Randomized Prim's Algorithm," in *ICCV*, 2013.
[68] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2016.
[69] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *CVPR*, 2017.
[70] R. Girshick, "Fast R-CNN," in *ICCV*, 2015.
[71] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.
[72] J. Pont-Tuset and L. Van Gool, "Boosting object proposals: From Pascal to COCO," in *ICCV*, 2015.
[73] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *TPAMI*, vol. 38, no. 4, pp. 814–830, 2016.
[74] I. Kokkinos, "Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *CVPR*, 2017.
[75] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, "Ssd: Single shot multibox detector," in *ECCV*, 2016.
[76] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.
[77] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *ECCV*, 2016.

Kevis-Kokitsi Maninis is a PhD candidate at ETHZ, Switzerland, in Prof. Luc Van Gool's Computer Vision Lab (2015). He received the Diploma degree in Electrical and Computer Engineering from National Technical University of Athens (NTUA) in 2014. He worked as undergraduate research assistant in the Signal Processing and Computer Vision group of NTUA (2013-2014).

Jordi Pont-Tuset is a post-doctoral researcher at ETHZ, Switzerland, in Prof. Luc Van Gool's Computer Vision Lab (2015). He received the degree in Mathematics in 2008, the degree in Electrical Engineering in 2008, the M.Sc. in Research on Information and Communication Technologies in 2010, and the Ph.D with honors in 2014; all from the Universitat Politècnica de Catalunya, BarcelonaTech (UPC). He worked at Disney Research, Zürich (2014).

Pablo Arbeláez received a PhD with honors in Applied Mathematics from the Université Paris-Dauphine in 2005. He was a Research Scientist with the Computer Vision Group at UC Berkeley from 2007 to 2014. He currently holds a faculty position at Universidad de los Andes in Colombia. His research interests are in computer vision, where he has worked on a number of problems, including perceptual grouping, object recognition and the analysis of biomedical images.

Luc Van Gool got a degree in electromechanical engineering at the Katholieke Universiteit Leuven in 1981. Currently, he is professor at the Katholieke Universiteit Leuven, Belgium, and the ETHZ, Switzerland, Switzerland. He leads computer vision research at both places, where he also teaches computer vision. He has authored over 200 papers in this field. He has been a program committee member of several major computer vision conferences. His main interests include 3D reconstruction and modeling, object recognition, tracking, and gesture analysis. He received several Best Paper awards. He is a co-founder of 5 spin-off companies.