This CVPR 2020 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

# Super-BPD: Super Boundary-to-Pixel Direction for Fast Image Segmentation

Jianqiang Wan[1], Yang Liu[1], Donglai Wei[2], Xiang Bai[1], Yongchao Xu[1*]

[1]Huazhong University of Science and Technology [2]Harvard University

{jianqw,yangl_,xbai,yongchaoxu}@hust.edu.cn, donglai@seas.harvard.edu

## Abstract

*Image segmentation is a fundamental vision task and a crucial step for many applications. In this paper, we propose a fast image segmentation method based on a novel super boundary-to-pixel direction (super-BPD) and a customized segmentation algorithm with super-BPD. Precisely, we define BPD on each pixel as a two-dimensional unit vector pointing from its nearest boundary to the pixel. In the BPD, nearby pixels from different regions have opposite directions departing from each other, and adjacent pixels in the same region have directions pointing to the other or each other (*i.e.*, around medial points). We make use of such property to partition an image into super-BPDs, which are novel informative superpixels with robust direction similarity for fast grouping into segmentation regions. Extensive experimental results on BSDS500 and Pascal Context demonstrate the accuracy and efficiency of the proposed super-BPD in segmenting images. In practice, the proposed super-BPD achieves comparable or superior performance with MCG while running at ∼25fps vs. 0.07fps. Super-BPD also exhibits a noteworthy transferability to unseen scenes. The code is publicly available at https://github.com/JianqiangWan/Super-BPD.*

## 1. Introduction

Image segmentation aims to decompose an image into non-overlapping regions, where pixels within each region share similar perceptual appearance, *e.g.*, color, intensity, and texture. Image segmentation is a crucial step for many vision tasks such as object proposal generation [33, 46], object detection [21], semantic segmentation [18]. However, an efficient and accurate segmentation remains challenging.

There are many unsupervised image segmentation methods that can be roughly categorized into early merging and clustering methods [48, 14], active contours [22, 7, 8], variational approaches [29, 37], watersheds [42, 31], segmentation with graphical models [19, 38, 5]. Though these classical methods are mathematically rigorous and achieve desirable results in some applications, as depicted in Fig. 1, they usually do not perform well in segmenting natural images or are not very efficient. Superpixel segmentation [1, 41] is an efficient alternative that over-segments an image into small and compact regions. A grouping process [25] is usually involved in producing final segmentation.

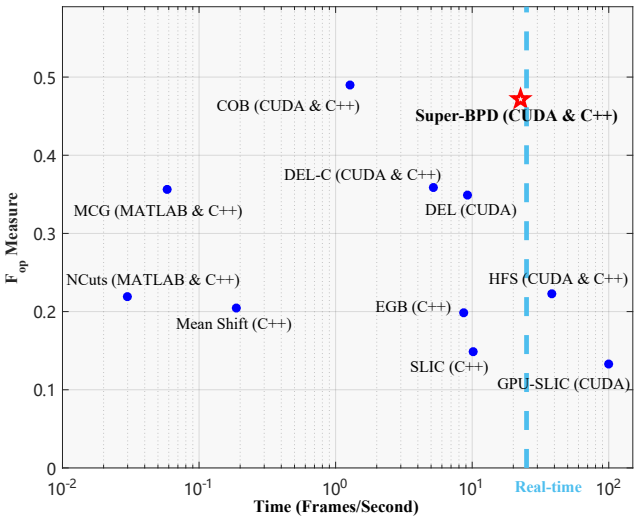
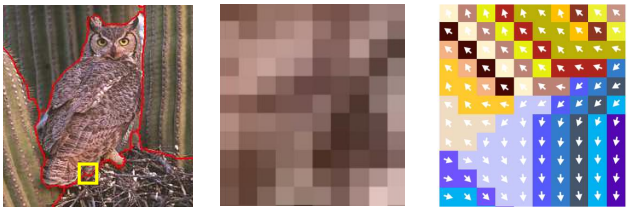



Figure 1. Super-BPD achieves competing while near real-time performance on the PASCAL Context dataset [28]. We plot the trade-off between efficiency and region F-measure accuracy ($F_{\text{op}}$) [34].

Thanks to convolutional neural networks (CNNs), semantic segmentation [26, 10, 47] that classifies each pixel into a predefined class category has witnessed significant progress in both accuracy and efficiency. Nevertheless, it does not generalize well to unseen object categories. Alternatively, for image segmentation, some methods [3, 33, 27, 17] resort to learn contours, followed by a transformation to bridge up the gap between contours and segmentation. As shown in Fig. 1, though these methods achieve impressive performances, the inevitable contour to segmentation transformation takes great effort to remedy leakage problems at weak boundaries and is usually time-consuming.

Different from previous methods that directly learn con-

*Corresponding author



(a) (b) (c)

Figure 2. Illustration of super-BPD results. (a) Given an image with super-BPD segmentation boundary (red), we zoom into a region with weak image boundaries (yellow). (b) Although pixels have similar values, (c) super-BPD can link pixels by the robustly predicted boundary-to-pixel direction (BPD), generating stripe-like segments on either side of the boundary for later grouping.

tours and transform contours to segmentation, we propose a novel super boundary-to-pixel direction (super-BPD) and an efficient segmentation algorithm with super-BPD. Specifically, we introduce a boundary-to-pixel direction (BPD) on each pixel in terms of a two-dimensional unit vector, pointing from its nearest boundary to the underlying pixel. The BPD not only provides contour positions but also encodes the relative position of each pixel to the corresponding region boundary and thus the relationship of neighboring pixels. This allows us to efficiently partition an image into super-BPDs such that each pixel and the pixel it points to and having similar directions are in the same super-BPD. The super-BPD can be regarded as a novel alternative of the classical superpixel, which provides robust direction for further grouping into segmentation regions.

The set of super-BPDs form a region adjacency graph (RAG), where the edges are weighted by the direction similarity along the boundaries of adjacent super-BPDs. Nearby pixels within different regions have approximately opposite BPD, and hence small direction similarity. Such property also holds even at weak boundaries, where the learned BPD smoothly diverges to roughly opposite directions along the direction (see Fig. 2 for an example). This equips the super-BPDs with robust direction similarity that helps to group similar super-BPDs within the same perceptual region and separate super-BPDs from different perceptual regions. We leverage such direction similarity between adjacent super-BPDs to partition the RAG into different clusters, resulting in a segmentation. As shown in Fig. 1, the proposed super-BPD achieves a good trade-off between accuracy and efficiency on PASCAL Context dataset [28].

The main contribution of this paper is two-fold: 1) We present a novel super boundary-to-pixel direction (super-BPD), which is a powerful alternative of the classical superpixel. super-BPD provides robust direction similarity between adjacent super-BPDs, which allows for efficient image segmentation. 2) We propose an efficient segmentation algorithm with super-BPDs in a coarse-to-fine way based on the direction similarity, leading to a good trade-off between segmentation accuracy and efficiency.

## 2. Related Work

We shortly review some works on image segmentation and other vision tasks leveraging direction information.

### 2.1. Image Segmentation

**Unsupervised Methods.** Many image segmentation methods have been proposed in the past two decades and can be roughly classified into several categories. Early segmentation methods are driven by region merging and clustering methods. Typical examples are region competition [48] and mean shift [14]. Active contours [22, 7, 8] are another type of popular segmentation methods that evolve region contours by minimizing some energy functions. Variational approaches [29, 37] also attempt to minimize some energy functions based on some appropriate hypotheses about the underlying image (*e.g.*, piece-wise constant in [29]). A set of watersheds [42, 31] has been proposed from the community of mathematical morphology. They segment image domain into catchment basins (*i.e.*, regions) and watershed lines (*i.e.*, contours). Another popular family of segmentation methods is based on graphical models [38, 5, 19], which model image domain as a graph and attempt to cut graphs based on some energy minimization. Besides these segmentation methods, superpixel methods [1, 35] aim to over-segment an image into small and compact regions.

**Supervised Methods.** Many learning-based image segmentation methods have been proposed. Different from semantic segmentation that can be regarded as a pixel-wise category classification problem, the mainstream learning-based segmentation methods [3, 33, 27, 17] start with learning contours. They then resort to oriented watershed transformations and globalization via spectral clustering to alleviate the leakage problem at weak boundaries. However, such a contour-to-segmentation process is usually time-consuming. In [44], the authors propose mutex watershed (MWS) by learning local attractive/repulsive affinities, followed by a modified maximum spanning tree to segment images. Another direction is to learn a feature embedding [25] for SLIC superpixels [1], where superpixels within the same region (*resp.*, different regions) have similar (*resp.*, very different) embedded features. A simple merging algorithm based on the embedded features is then adopted to group superpixels into perceptual regions.

The proposed super-BPD falls into supervised methods. Different from the existing learning-based methods, super-BPD does not rely on contours and is free of the time-consuming process to handle weak boundaries in transforming contours to segmentation. Super-BPD is a powerful alternative to classical superpixels. It provides robust di-

rection similarity for efficiently grouping pixels within the same region, and separating nearby regions even with weak boundaries between them. This results in a good trade-off between accuracy and efficiency. Compared with [25], super-BPD does not require a separate superpixel generation and embedding step, more efficient to separate different nearby regions with weak boundaries.

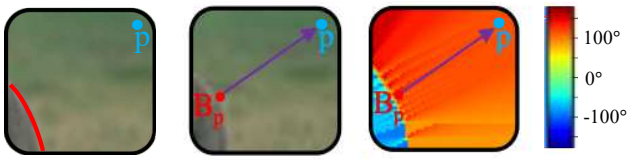
### 2.2. Direction Cues for Vision Applications

The direction information has been recently explored in different vision tasks. Some methods rely on similar direction fields defined on regions of interest. For instance, deep watershed transform [4] propose to learn the direction field on semantic segmentation and then regress the distance to boundaries based on the direction information, followed by a classical watershed to produce instance segmentation. Textfield [45] and DeepFlux [43] defines a similar direction field on text areas and skeleton context for scene text detection and skeleton extraction, respectively. The direction cue is also explored in MaskLab [9] and IRnet [2] for improving instance segmentation and weakly instance segmentation, respectively. PifPaf [24] and PVNet [32] leverage direction cue for 2D human pose estimation and 6 DoF pose estimation, respectively.

The proposed super-BPD builds upon boundary-to-pixel direction (BPD), which is similar to [4, 45, 43] but defined on the whole image domain instead of regions of interest. The BPD learning confirms that it is possible to learn the direction field encoding the relative position of each pixel with respect to region contours in natural images. In this sense, the BPD can be seen as an extension of the flux for binary object skeletonization [39] to natural images for image segmentation. super-BPD differs a lot with [4, 45, 43] in how to use BPD defined on the whole image. The major contribution is the extension of BPD to super-BPDs that enhances the robustness of direction information of BPD and consequently the robust direction similarity between neighboring super-BPDs. Based on it, we propose an efficient coarse-to-fine RAG partition algorithm, leading to an accurate and efficient generic image segmentation.

## 3. Super Boundary-to-Pixel Direction

Current segmentation methods achieve great performances with time-consuming post-processing, which limits their usages in practice. Whereas efficient segmentation methods provide degenerated results. We propose to remedy this issue by introducing a novel super boundary-to-pixel direction (super-BPD). The boundary-to-pixel direction (BPD) is defined on each pixel $p$ as the two-dimensional unit vector pointing from its nearest boundary pixel $B_p$ to $p$. Such BPD encodes the relative position between each pixel $p$ and the region (containing $p$) boundary. We adopt a CNN to learn such BPD, which is then used to partition the image into super-BPDs, a powerful alternative of classical superpixels. Super-BPDs provides robust direction similarity between adjacent super-BPDs, thus allowing fast image segmentation by partitioning the region (*i.e.*, super-BPD) adjacency graph (RAG).




(a) (b) (c)

Figure 3. Illustration of boundary-to-pixel direction (BPD). (a) For each pixel $p$ (ground truth boundary in red), (b) we find its nearest boundary pixel $B_p$. The BPD $\mathcal{D}_p$ is defined as the two-dimensional unit vector pointing from $B_p$ to $p$. (c) We predict BPD densely for each pixel and its direction is color-coded.

### 3.1. Boundary-to-Pixel Direction (BPD)

**Definition.** As shown in Fig. 3, for each pixel in the image domain $p \in \Omega$, we find its nearest boundary pixel $B_p$. Then, the BPD at pixel $p$, $\mathcal{D}_p$, is defined as a two-dimensional unit vector pointing from $B_p$ to $p$ given by

$$\mathcal{D}_p = \overrightarrow{B_p p}/|\overrightarrow{B_p p}|, \tag{1}$$

where $|\overrightarrow{B_p p}|$ is the distance between $B_p$ and $p$. The BPD provides cues about contour positions and relative position of each pixel $p$ to its region boundary. Note that generating BPD from ground-truth annotation could be efficiently achieved by the distance transform algorithm.

**Architecture and Learning.** We adopt a Fully Convolutional Network (FCN) to predict BPD as a two-channel map with the same spatial size as the input image (Fig. 4a). For a fair comparison with other methods, we adopt VGG16 [40] as the backbone network, where the last max-pooling layer and all following layers are discarded. We also leverage ASPP layer [10] to enlarge the receptive field, better coping with large regions. We extract features from different stages of VGG16 to aggregate multi-scale information. Specifically, we apply $1 \times 1$ convolutions to $conv3$, $conv4$, $conv5$, and ASPP layers, followed by a concatenation of these side output features after resizing them to the size of $conv3$. Finally, we apply three consecutive $1 \times 1$ convolutions on the fused feature maps, followed by an upsampling with bilinear interpolation to predict the BPD.

We define the loss function in terms of both $L_2$-norm distance and angle distance for the BPD prediction $\hat{\mathcal{D}}$:

$$L = \sum_{p \in \Omega} w(p)(\|\mathcal{D}_p - \hat{\mathcal{D}}_p\|^2 + \alpha \|\cos^{-1}\langle \mathcal{D}_p, \hat{\mathcal{D}}_p \rangle\|^2), \tag{2}$$

where the adaptive weight at pixel $p$, $w(p) = 1/\sqrt{|GT_p|}$, is proportional to the inverse square root of the size of ground

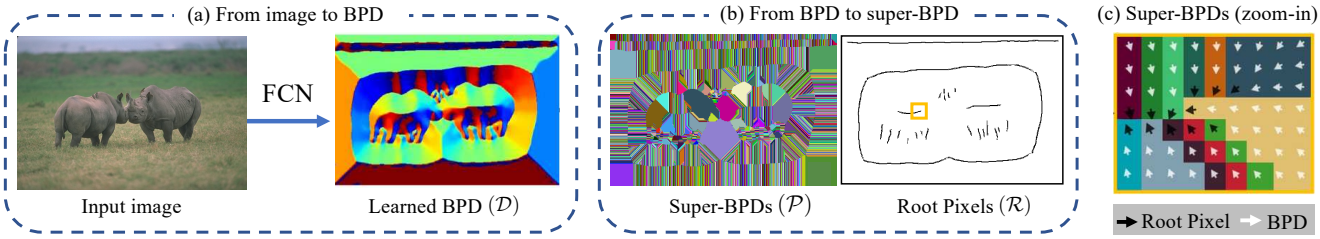



Figure 4. Overview of super-BPD computation. (a) We adopt a Fully Convolutional Network (FCN) to learn the BPD field from the input image. (b) We then group BPDs into super-BPDs (colored regions) by the direction similarity threshold (Algo. 1) and extract root pixels. (c) We zoom in a region near the symmetry axis of the segment, where there are root pixels (black arrow) and regular BPDs (white arrow).

**Algorithm 1:** Generate super-BPDs from the learned BPDs (Sec. 3.2).

**Input:** Learned BPD ($\hat{\mathcal{D}}$), threshold $\theta_a$
**Output:** Super-BPD ($\mathcal{P}$) and root pixel ($\mathcal{R}$)

1 **function** Get_Super-BPDs($\hat{\mathcal{D}}, \theta_a$)
2  // initialization
3  $\mathcal{P} \leftarrow p_0, \mathcal{R} \leftarrow \emptyset$
4  // From BPD to super-BPD
5  **foreach** $p \in \Omega$ **do**
6  **if** $\cos^{-1}\langle \hat{\mathcal{D}}_p, \hat{\mathcal{D}}_{n_p} \rangle < \theta_a$ **then**
7  $\mathcal{P}(p) \leftarrow n_p$
8  **else**
9  $\mathcal{P}(p) \leftarrow p, \mathcal{R} \leftarrow p$
10  **return** $\mathcal{P}$, $\mathcal{R}$

truth segment $GT_p$ containing $p$ and $\alpha$ is a hyper-parameter to balance the loss terms. In practice, we set $\alpha = 1$.

### 3.2. BPD Grouping into Super-BPDs

From the learned BPD, we extract super-BPDs, stripe-like segments encoded by a parent image $\mathcal{P}$, and their root pixels $\mathcal{R}$ close to regions' symmetry axes (Fig. 4b).

Precisely, inspired by the algorithms of computing component trees [36, 30, 6], we adopt a parent image $\mathcal{P}$ to encode the relationship between neighboring pixels. Initially, the parent of each pixel $p$ is set to itself, $\mathcal{P}(p) = p$ and the set of root pixels $\mathcal{R}$ is empty. For each pixel $p$, we define its next pixel $n_p$ as the neighboring pixel that is pointed to by $\hat{\mathcal{D}}_p$. As depicted in Algo. 1 (line 5-9), for each pixel in the raster order, if the angle between its BPD and that of its next pixel $n_p$, $\hat{\mathcal{D}}_p$ and $\hat{\mathcal{D}}_{n_p}$, is smaller than a given threshold $\theta_a$, we group them together by setting $\mathcal{P}(p)$, the parent of $p$, to $n_p$. Otherwise, we insert $p$ into the set of root pixels $\mathcal{R}$. The final parent image $\mathcal{P}$ partitions the image into a forest of trees, each of which is rooted at a root pixel in $\mathcal{R}$. We define each tree as a super-BPD.

Following the definition of BPD in Sec. 3.1, the direction around boundary pixels departs from each other, forming repulsive edges for separating neighboring pixels of different regions (even at weak boundaries, see Fig. 2). Pixels near the symmetry axis of each region also have approximately opposite directions, resulting in root pixels. Therefore, as shown in Fig. 4b, the root pixels lie close to the symmetry axis of each region, allowing a safe merging of nearby root pixels that are usually within the same region.

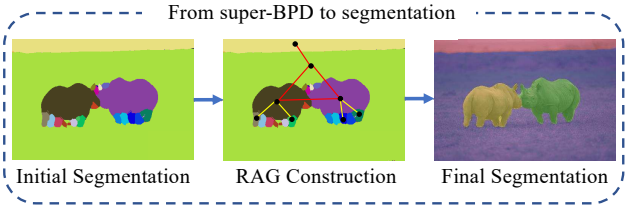



Figure 5. Image segmentation from initial segmentation. We construct a RAG on the set of initial segments, and compute the size of each initial segment and the graph edge weight. We then partition the edge-weighted RAG into perceptual regions based on repulsive and attractive edges following the direction similarity.

## 4. Image Segmentation with Super-BPD

We first obtain an initial segmentation via merging super-BPDs with nearby root pixels. Then, we construct an adjacency graph of regions (RAG) and apply the graph partitioning to merge initial segments as the final result (Fig. 5).

**Initial Segmentation.** As described above, root pixels of super-BPDs within the same ground truth region tend to be close to each other near the region's symmetry axis (Fig. 4b). Thus, we apply a simple dilation to group nearby root pixels and their corresponding super-BPDs to generate the initial segmentation. As depicted in Algo. 2 (line 3-6), for each root pixel $r$, we update its parent $\mathcal{P}(r)$ to the last root pixel within the bottom half of $3 \times 3$ window $\mathcal{N}_3^b$ centered at $r$. We also update $\mathcal{R}$ by removing the merged root pixels. With this simple method, we can group super-BPDs into a reasonable initial segmentation (Fig. 5, left).

**Region Adjacency Graph Construction.** We construct a region adjacency graph $(\mathcal{R}, E)$ from the initial segmentation, where $E$ stands for the set of edges linking the root pixels of adjacent segments (Algo. 2, line 9).

**Algorithm 2:** Generate image segmentation from super-BPDs (Sec. 4).

**Input:** $\mathcal{D}$, $\mathcal{P}$, $\mathcal{R}$, threshold $\theta_l$, $\theta_s$, $a_t$, $a_s$
**Output:** Set of linking edges $E_l$

**function** Super_BPD2SEG($\mathcal{D}, \mathcal{P}, \mathcal{R}, \theta_l, \theta_s, a_t, a_s$)
  // Initial segmentation
  **foreach** $r \in \mathcal{R}$ **do**
    // merge nearby root pixels of super-BPDs
    **foreach** $q \in \mathcal{N}_3^b(r)$ **do**
      **if** $q \in \mathcal{R}$ **then** $\mathcal{P}(r) \leftarrow q$, $\mathcal{R}$.pop(r)

  // Region adjacency graph construction
  $(E_A^{\downarrow}, \mathcal{S}, \mathcal{A}) \leftarrow$ Get_RAG($\mathcal{D}, \mathcal{P}, \mathcal{R}$)

  // Graph Partitioning
  $E_l \leftarrow \emptyset$
  **foreach** $e = (r_1, r_2) \in E_A^{\downarrow}$ **do**
    // merge similar large and small initial seg.
    **if** $\min(\mathcal{A}_{r_1}, \mathcal{A}_{r_2}) > a_t$ **and not** *Rep*$(r_1, r_2)$ **and** $\mathcal{S}(e) > h_{\theta_l,\theta_s,a_s}(\mathcal{A}_{r_1}, \mathcal{A}_{r_2})$ **then**
      Merge$(r_1, r_2)$, $E_l$.push(e) //updating
  **foreach** $e = (r_1, r_2) \in E_A^{\downarrow}$ **do**
    // merge tiny initial seg.
    **if** $\min(\mathcal{A}_{r_1}, \mathcal{A}_{r_2}) < a_t$ **and not** *Rep*$(r_1, r_2)$ **then**
      Merge$(r_1, r_2)$, $E_l$.push(e) //updating
  **return** $E_l$

For each edge $e = (r_1, r_2) \in E$, linking two regions $R_1$ and $R_2$, we compute its direction similarity $\mathcal{S}$. Let $B(e) = \{(p_i^1, p_i^2)\}$ be the set of neighboring pixels along the boundaries such that $p_i^1 \in R_1$ and $p_i^2 \in R_2$, we define $\mathcal{S}(e)$, the direction similarity on $e$, as following:

$$\mathcal{S}(e) = \pi - \frac{\sum_{i=1}^{|B(e)|} \cos^{-1}\langle \hat{\mathcal{D}}_{\mathcal{P}_s(p_i^1)}, \hat{\mathcal{D}}_{\mathcal{P}_s(p_i^2)} \rangle}{|B(e)|}, \quad (3)$$

where $|B(e)|$ denotes the number of boundary points between $R_1$ and $R_2$, and $\mathcal{P}_s(p)$ stands for the $s$-th step starting from the pixel $p$. For example, $s = 0$ refers to the pixel itself. With a direction similarity threshold $\mathcal{S}_0 = \frac{\pi}{18}$, we divide all edges into attractive edges $E_A^{\downarrow}$, sorted in decreasing order of direction similarity, and repulsive edges $E_R$. We define $Rep$ on a pair of adjacent segments to measure if they are repulsive. The $Rep$ is updated during the following merging process.

For each root pixel $r \in \mathcal{R}$, we compute the area $\mathcal{A}_r$ of its underlying initial segment. Similar to [19], we divide initial segments into large, small and tiny by the hyper-parameter area thresholds $a_s$ and $a_t$.

**Graph Partitioning.** We first greedily merge adjacent initial segments with either large or small sizes based on the direction similarity $\mathcal{S}$ (Algo. 2, line 13-16). Following [19], we make direction similarity thresholds adaptive to the size of the initial segments. For each edge $e = (r_1, r_2)$, linking adjacent initial segments $R_1$ and $R_2$, we use a piece-wise constant threshold function $h_{\theta_l,\theta_s,a_s}(\mathcal{A}_{r_1}, \mathcal{A}_{r_2})$ with values as $\theta_l$ if both $\mathcal{A}_{r_1}$ and $\mathcal{A}_{r_2}$ are large segments ($\mathcal{A} > a_s$), and as $\theta_s (< \theta_l)$ otherwise.

$$\mathrm{h}_{\theta_l,\theta_s,a_s}(\mathcal{A}_{r_1}, \mathcal{A}_{r_2}) = \begin{cases} \theta_l & \text{if } \min(\mathcal{A}_{r_1}, \mathcal{A}_{r_2}) \geq a_s \\ \theta_s & \text{otherwise} \end{cases} \quad (4)$$

Then, we iterate through edges $e = (r_1, r_2) \in E_A^{\downarrow}$ in the decreasing order of the direction similarity. We merge $R_1$ and $R_2$, if the direction similarity $\mathcal{S}(e)$ is larger than $h_{\theta_l,\theta_s,a_s}(\mathcal{A}_{r_1}, \mathcal{A}_{r_2})$ and the merge of $R_1$ and $R_2$ does not violate the repulsive rule, $Rep$. Note that each merge operation triggers the update of repulsive information $Rep$ between super-BPDs and super-BPD size $\mathcal{A}$.

Finally, we merge initial segments with tiny sizes ($\mathcal{A} < a_t$) to their non-repulsive neighbors (Algo. 2, line 17-20). With this, we can clean up small crumb regions in the initial segmentation for better qualitative and quantitative results.

**Runtime Analysis.** The whole segmentation from BPD is composed of three stages: 1) Super-BPD partition (Algo. 1, line 5-9). The complexity is $O(N)$, where $N$ denotes the number of pixels in image. 2) Nearby root pixels grouping (Algo. 2, line 3-6), which has a linear complexity with the number of root pixels. 3) Graph partition (Algo. 2, line 12-20), which has a quasi-linear time complexity with respect to the number of edges (*i.e.*, in hundreds order) in RAG. Therefore, the whole post-processing has a near linear complexity, and is thus efficient.

## 5. Experiments

We conduct generic image segmentation experiments on PASCAL Context [28] and BSDS500 [3] dataset.

### 5.1. Implementation Details

For training on BSDS500 dataset, we adopt the same data augmentation strategy used in [25]. Specifically, the training images are rotated to 16 angles and flipped at each angle, then we crop the largest rectangle from the transformed images, yielding 9600 training images. Since PASCAL Context dataset has enough images, We only randomly flip images during training. The proposed network is initialized with the VGG16 model pretrained on ImageNet [16] and optimized using ADAM [23]. Both models are trained for first 80k iterations with initial learning rate $10^{-5}$ for backbone layers and $10^{-4}$ for extra layers. We then decay the learning rates to $10^{-6}$ and $10^{-5}$, respec-

| Dataset | Methods | $F_b$ | | $F_{op}$ | | Covering | | PRI | | VI | | Time (s) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ODS | OIS | ODS | OIS | ODS | OIS | ODS | OIS | ODS | OIS | |
| PASCAL Context | SLIC [1] | 0.359 | 0.409 | 0.149 | 0.160 | - | - | - | - | - | - | 0.099 |
| | Mean Shift [14] | 0.397 | 0.406 | 0.204 | 0.214 | - | - | - | - | - | - | 5.320 |
| | MS-NCut [15] | 0.380 | 0.429 | 0.219 | 0.285 | - | - | - | - | - | - | 33.40 |
| | EGB [19] | 0.432 | 0.454 | 0.198 | 0.203 | - | - | - | - | - | - | 0.116 |
| | DEL [25] | 0.563 | 0.623 | 0.349 | 0.420 | 0.600 | 0.640 | 0.790 | 0.810 | 1.600 | 1.390 | 0.108 |
| | DEL-C [25] | 0.570 | 0.631 | 0.359 | 0.429 | 0.610 | 0.660 | 0.800 | 0.820 | 1.580 | 1.330 | 0.193 |
| | MCG [33] | 0.577 | 0.634 | 0.356 | 0.419 | 0.577 | 0.668 | 0.798 | 0.854 | 1.680 | 1.332 | 17.05 |
| | COB [27] | **0.755** | **0.789** | **0.490** | **0.566** | **0.739** | **0.803** | **0.878** | **0.919** | **1.150** | **0.916** | 0.790 |
| | GPU-SLIC [35] | 0.322 | 0.340 | 0.133 | 0.157 | - | - | - | - | - | - | 0.010 |
| | HFS [12] | 0.472 | 0.495 | 0.223 | 0.231 | - | - | - | - | - | - | 0.026 |
| | Super-BPD **(Ours)** | **0.704** | **0.721** | **0.472** | **0.524** | **0.730** | **0.770** | **0.880** | **0.900** | **1.150** | **1.010** | 0.044 |
| BSDS 500 | SLIC [1] | 0.529 | 0.565 | 0.146 | 0.182 | 0.370 | 0.380 | 0.740 | 0.750 | 2.560 | 2.500 | 0.085 |
| | EGB [19] | 0.636 | 0.674 | 0.158 | 0.240 | 0.520 | 0.570 | 0.800 | 0.820 | 2.210 | 1.870 | 0.108 |
| | MS-NCut [15] | 0.640 | 0.680 | 0.213 | 0.270 | 0.450 | 0.530 | 0.780 | 0.800 | 2.230 | 1.890 | 23.20 |
| | Mean Shift [14] | 0.640 | 0.680 | 0.229 | 0.292 | 0.540 | 0.580 | 0.790 | 0.810 | 1.850 | 1.640 | 4.950 |
| | DEL [25] | 0.704 | 0.738 | 0.326 | 0.397 | 0.590 | 0.640 | 0.810 | 0.850 | 1.660 | 1.470 | 0.088 |
| | DEL-C [25] | 0.715 | 0.745 | 0.333 | 0.402 | 0.600 | 0.660 | 0.830 | 0.860 | 1.640 | 1.440 | 0.165 |
| | MWS [44] | - | - | - | - | - | - | 0.826 | - | 1.722 | - | 0.580 |
| | gPb-UCM [3] | 0.729 | 0.755 | 0.348 | 0.385 | 0.587 | 0.646 | 0.828 | 0.855 | 1.690 | 1.476 | 86.40 |
| | MCG [33] | **0.744** | **0.777** | **0.379** | **0.433** | 0.613 | **0.663** | 0.832 | **0.861** | 1.568 | **1.390** | 14.50 |
| | COB [27] | **0.782** | **0.808** | **0.414** | **0.464** | **0.664** | **0.712** | **0.854** | **0.886** | **1.380** | **1.222** | - |
| | GT* | 0.732 | 0.732 | 0.463 | 0.463 | 0.690 | 0.690 | 0.860 | 0.860 | 1.180 | 1.180 | - |
| | GPU-SLIC [35] | 0.522 | 0.547 | 0.085 | 0.132 | 0.340 | 0.370 | 0.730 | 0.750 | 2.950 | 2.810 | 0.007 |
| | HFS [12] | 0.652 | 0.686 | 0.249 | 0.272 | 0.560 | 0.610 | 0.810 | 0.840 | 1.870 | 1.680 | 0.024 |
| | Super-BPD **(Ours)** | 0.695 | 0.700 | 0.360 | 0.380 | **0.640** | 0.650 | **0.840** | 0.850 | **1.480** | 1.430 | 0.036 |

Table 1. In-dataset evaluation results. On both PASCAL Context and BSDS500 dataset, Super-BPD achieves the state-of-the-art performance among real-time methods (2nd row) and remains competitive against non-real-time methods. GT* is the rough annotation [25].

tively, and continue to train the model for another 320k iterations on both BSDS500 PASCAL Context.

For the hyper-parameter settings, unless explicitly stated otherwise, we set $\theta_a$ to 45 for super-BPD partition. The threshold $a_s$ for small and $a_t$ for tiny regions are fixed to 1500 and 200. The step $s$ involved in computing the direction similarity in Eq. (3) is set to 3. The other two hyper-parameters $\theta_l$ and $\theta_s$ for merging large and small regions are tuned for the optimal dataset setting (ODS) on each dataset.

The proposed super-BPD is implemented with PyTorch platform. All experiments are carried out on a workstation with an Intel Xeon 16-core CPU (3.5GHz), 64GB RAM, and a single Titan Xp GPU. Training on PASCAL Context using a batch size of 1 takes about 6 hours.

## 5.2. Comparing with State-of-the-art Methods

**Datasets.** PASCAL Context [28] contains precisely localized pixel-wise semantic annotations for the whole image. We ignore the semantics of each region for benchmarking the generic image segmentation methods. This dataset is composed of 7605 *trainval* images and 2498 test images.

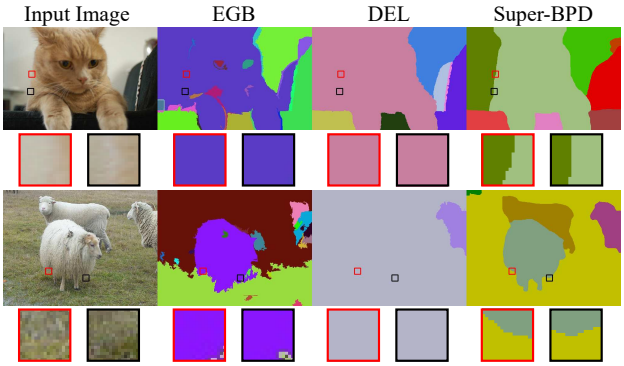
BSDS500 [3] is a benchmark dataset for image segmentation and boundary detection. It is divided into 200 training images, 100 val images, and 200 test images. Each image has 5-10 different segmentation ground-truth annotations.

**Metrics.** We use four standard benchmark measures adopted in [3]: $F$-measure for boundaries $F_b$, segmentation covering (Covering), Probabilistic Rand Index (PRI), and Variation of Information (VI). We also evaluate the proposed Sup-BPD using another widely adopted metric $F$-measure for objects and parts $F_{op}$ introduced in [34].

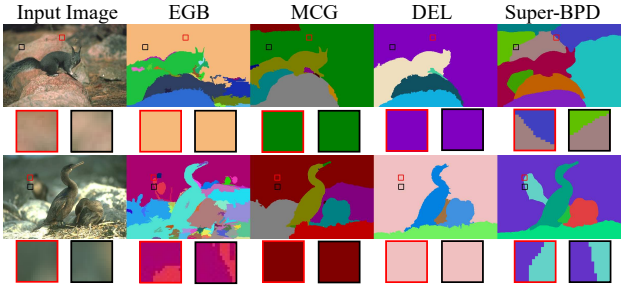
**In-Dataset Evaluation.** We compare the proposed super-BPD with other state-of-the-art image segmentation methods on both PASCAL Context and BSDS500. Some qualitative comparisons are shown in Fig. 6. Super-BPD correctly segments images into perceptual regions on both datasets.

| | PASCAL Context → BSDS500 | | | | BSDS500 → PASCAL Context | | | |
|---|---|---|---|---|---|---|---|---|
| Methods | $F_{op}$ | Covering | PRI | VI | $F_{op}$ | Covering | PRI | VI |
| DEL-C | 0.328 | 0.58 | 0.82 | 1.73 | 0.319 | 0.57 | 0.76 | 1.73 |
| Super-BPD | **0.347** | **0.61** | **0.83** | **1.53** | **0.356** | **0.62** | **0.81** | **1.59** |

Table 2. Cross-dataset evaluation results. We compare the performance of super-BPD and DEL [25] with the ODS metrics.




(a) Segmentation results on some images in PASCAL Context.




(b) Segmentation results on some images in BSDS500.

Figure 6. Qualitative comparisons with other methods on some images from BSDS and PASCAL Context dataset. Super-BPD correctly segments natural images into perceptual regions despite existing weak boundaries.

The quantitative comparison on PASCAL Context is depicted in Tab. 1 (top). Super-BPD achieves a pleasant trade-off between accuracy and efficiency on segmenting images in PASCAL Context. Specifically, super-BPD performs competitively with COB [27] while being much faster based on all evaluation metrics. Compared with MCG [33], super-BPD records an improvement of 11.6% in ODS $F_{op}$ while being much more efficient. Super-BPD improves DEL and DEL-C [25] by 12.3% and 11.3% in ODS $F_{op}$, respectively. Besides, super-BPD runs faster than DEL [25].

The quantitative comparison on BSDS500 dataset is shown in Tab. 1 (bottom). Super-BPD also achieves a good trade-off between segmentation accuracy and efficiency. Specifically, super-BPD achieves competitive or superior performance with COB [27], MCG [33], gPb-UCM [3], and MWS [44] while being more efficient. For a fair comparison with DEL [25], we also use the roughest annotation of each image for training. Super-BPD outperforms DEL and DEL-C [25] by 3.4% and 2.7% in ODS $F_{op}$, respectively. We also report the result of using the roughest annotation as segmentation denoted as GT*. Super-BPD approaches the "upper bound" of ground-truth segmentation. It is noteworthy to mention that the lower accuracy on BSDS500 than on PASCAL Context is due to inconsistent annotations between different subjects.

**Cross-Dataset Evaluation.** To demonstrate the generalization ability of the proposed super-BPD, we evaluate the model trained on one dataset and test the trained model on another dataset. We mainly compare super-BPD with DEL-C [25], which is also dedicated for a good trade-off between accuracy and efficiency. As depicted in Tab. 2, super-BPD is robust in generalizing to unseen datasets. Specifically, super-BPD outperforms DEL-C [25] for both PASCAL Context to BSDS500 and BSDS500 to PASCAL Context setting. In fact, super-BPD even outperforms DEL-C [25] properly trained on the corresponding training set.

**Runtime Analysis.** Super-BPD has three stages: BPD inference, super-BPD partition, and segmentation with super-BPD. BPD inference on the GPU using VGG16 takes on average 22 ms for a $390 \times 470$ PASCAL Context image, and super-BPD partition and segmentation with super-BPD require on average 22 ms for a PASCAL Context image on the CPU. As depicted in Tab. 1, super-BPD is much more efficient than the other competing methods while achieving comparable or superior performance.

### 5.3. Ablation Studies

**ASPP Module.** We first study the effect of ASPP module that increases receptive field for coping with large regions. As shown in Tab. 3, when the ASPP module is not used, the performance slightly decreases on both PASCAL Context and BSDS500 dataset.

**Loss Functions.** We study the effect of different loss functions to train the network on PASCAL Context dataset. As depicted in Tab. 4, both $L_2$ loss function and the angular domain loss function achieve reasonable results, the combination of them by Eq. (2) further improves the performance, and can also boost the convergence in training.

**Number of Steps $s$.** We evaluate the influence of steps $s$ in computing direction similarity along boundaries between adjacent superpixels. As shown in Tab. 5, this step parame-

| Dataset | ASPP | $F_{op}$ |
|---|---|---|
| PASCAL Context | | 0.454 |
| PASCAL Context | ✓ | **0.472** |
| BSDS500 | | 0.348 |
| BSDS500 | ✓ | **0.360** |

Table 3. Effects of ASPP module on the performance in ODS $F_{op}$.

| Dataset | norm loss | angular loss | $F_{op}$ |
|---|---|---|---|
| PASCAL Context | ✓ | | 0.454 |
| | | ✓ | 0.465 |
| | ✓ | ✓ | **0.472** |

Table 4. Effects of loss functions on the performance in ODS $F_{op}$.

| Dataset | s=0 | s=1 | s=3 | s=5 | s=7 |
|---|---|---|---|---|---|
| PASCAL Context | 0.460 | 0.462 | **0.472** | 0.465 | 0.457 |
| BSDS500 | 0.353 | 0.354 | **0.360** | 0.350 | 0.339 |

Table 5. Effects of steps $s$ on the performance in ODS $F_{op}$.

ter $s$ has an impact on the segmentation performance. The best result is achieved with $s = 3$. In fact, adjacent pixels within different regions may have slightly different directions, but the overall direction trend is divergent. This explains the benefit of using $s = 3$.

## 6. Application: Object Proposal Generation

The object proposal generation task is a prerequisite step for a number of mid-level and high-level vision tasks such as object detection [20].

Following DEL [25], we use BING [13] as the baseline object proposal generation method. We further improve BING's results with the multi-thresholding straddling expansion method (MTSE) [11], referenced as M-BING. To compare generic segmentation methods, we replace the EGB segmentation results [19] used in MTSE [11] with DEL [25] and our proposed super-BPD respectively.

On PASCAL VOC2007 test set, we plot the detection recall with 0.8 IoU overlap versus the number of proposals in Fig. 7. M-BING with DEL has a slight improvement over M-BING until around 100 object proposals. On the other hand, M-BING with super-BPD significantly improves upon M-BING by nearly twice until 500 proposals.

## 7. Conclusion and Limitations

We propose a fast image segmentation method based on a novel super boundary-to-pixel direction (super-BPD) and a customized segmentation with super-BPD. Specifically, the BPD allows a fast image partition into super-BPDs, a powerful alternative of classical superpixels with

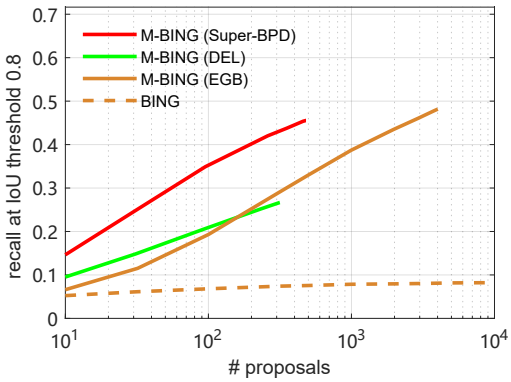



Figure 7. Object proposal results on PASCAL VOC2007. Super-BPD outperforms other methods using the BING framework [13].

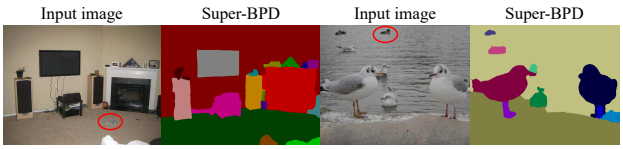



Figure 8. Some failure cases. Super-BPD does not perform well in segmenting very small perceptual regions.

robust direction similarity. We then construct a region adjacency graph and merge super-BPDs based on the direction similarity along their boundaries, resulting in a fast image segmentation. The proposed super-BPD achieves a good compromise between accuracy and efficiency, and can separate nearby regions with weak boundaries. In particular, super-BPD achieves comparable or superior performance with MCG but being near real-time. Besides, super-BPD also has an appealing transferability to unseen scenes. This allows potential use of super-BPD in many vision tasks. For example, we have verified the usefulness of super-BPD in object proposal generation. In the future, we would like to explore super-BPD in other applications.

Though the proposed super-BPD achieves a pleasant trade-off between generic image segmentation accuracy and efficiency, it is still difficult for super-BPD to accurately segment small regions. This is because that the prediction of BPD around small regions is not very accurate due to dramatic changes of direction. Some failure cases are illustrated in Fig. 8, where small regions are not accurately segmented. The segmentation of very small regions also remains a problem for other image segmentation methods.

## Acknowledgement

This work was supported in part by the Major Project for New Generation of AI under Grant no. 2018AAA0100400, NSFC 61703171, and NSF of Hubei Province of China under Grant 2018CFB199. Dr. Yongchao Xu was supported by the Young Elite Scientists Sponsorship Program by CAST and Dr. Donglai Wei by NSF IIS-1835231.

## References


[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. 1, 2, 6

[2] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *Proc. of CVPR*, pages 2209–2218, 2019. 3

[3] Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 1, 2, 5, 6, 7

[4] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proc. of CVPR*, pages 5221–5229, 2017. 3

[5] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. 1, 2

[6] Edwin Carlinet and Thierry Géraud. A comparative review of component tree computation algorithms. *IEEE Transactions on Image Processing*, 23(9):3885–3895, 2014. 4

[7] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997. 1, 2

[8] Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001. 1, 2

[9] Liang-Chieh Chen, Alexander Hermans, George Papandreou, Florian Schroff, Peng Wang, and Hartwig Adam. Masklab: Instance segmentation by refining object detection with semantic and direction features. In *Proc. of CVPR*, pages 4013–4022, 2018. 3

[10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 1, 3

[11] Xiaozhi Chen, Huimin Ma, Xiang Wang, and Zhichen Zhao. Improving object proposals with multi-thresholding straddling expansion. In *Proc. of CVPR*, pages 2587–2595, 2015. 8

[12] Ming-Ming Cheng, Yun Liu, Qibin Hou, Jiawang Bian, Philip Torr, Shi-Min Hu, and Zhuowen Tu. HFS: Hierarchical feature selection for efficient image segmentation. In *Proc. of ECCV*, pages 867–882, 2016. 6

[13] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *Proc. of CVPR*, pages 3286–3293, 2014. 8

[14] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (5):603–619, 2002. 1, 2, 6

[15] Timothee Cour, Florence Benezit, and Jianbo Shi. Spectral segmentation with multiscale graph decomposition. In *Proc. of CVPR*, volume 2, pages 1124–1131, 2005. 6

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. of CVPR*, pages 248–255, 2009. 5

[17] Chaowei Fang, Zicheng Liao, and Yizhou Yu. Piecewise flat embedding for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(6):1470–1485, 2019. 1, 2

[18] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013. 1

[19] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004. 1, 2, 5, 6, 8

[20] Ross Girshick. Fast r-cnn. In *Proc. of ICCV*, pages 1440–1448, 2015. 8

[21] Mayank Juneja, Andrea Vedaldi, CV Jawahar, and Andrew Zisserman. Blocks that shout: Distinctive parts for scene classification. In *Proc. of CVPR*, pages 923–930, 2013. 1

[22] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988. 1, 2

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of ICLR*, volume 5, 2015. 5

[24] Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. PifPaf: Composite fields for human pose estimation. In *Proc. of CVPR*, pages 11977–11986, 2019. 3

[25] Yun Liu, Peng-Tao Jiang, Vahan Petrosyan, Shi-Jie Li, Jiawang Bian, Le Zhang, and Ming-Ming Cheng. DEL: Deep embedding learning for efficient image segmentation. In *Proc. of IJCAI*, pages 864–870, 2018. 1, 2, 3, 5, 6, 7, 8

[26] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. of CVPR*, pages 3431–3440, 2015. 1

[27] Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and Luc Van Gool. Convolutional oriented boundaries: From image segmentation to high-level tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):819–833, 2018. 1, 2, 6, 7

[28] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proc. of CVPR*, pages 891–898, 2014. 1, 2, 5, 6

[29] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989. 1, 2

[30] Laurent Najman and Michel Couprie. Building the component tree in quasi-linear time. *IEEE Transactions on Image Processing*, 15(11):3531–3539, 2006. 4

[31] Laurent Najman and Michel Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1163–1173, 1996. 1, 2

[32] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6dof pose estimation. In *Proc. of CVPR*, pages 4561–4570, 2019. 3

[33] Jordi Pont-Tuset, Pablo Arbeláez, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):128–140, 2017. 1, 2, 6, 7

[34] Jordi Pont-Tuset and Ferran Marques. Supervised evaluation of image segmentation and object proposal techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1465–1478, 2016. 1, 6

[35] Carl Yuheng Ren, Victor Adrian Prisacariu, and Ian D Reid. gSLICr: SLIC superpixels at over 250hz. *arXiv preprint arXiv:1509.04232*, 2015. 2, 6

[36] Philippe Salembier, Albert Oliveras, and Luis Garrido. Antiextensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, 7(4):555–570, 1998. 4

[37] James Albert Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999. 1, 2

[38] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 1, 2

[39] Kaleem Siddiqi, Sylvain Bouix, Allen Tannenbaum, and Steven W Zucker. Hamilton-jacobi skeletons. *International Journal of Computer Vision*, 48(3):215–231, 2002. 3

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. of ICLR*, 2015. 3

[41] Wei-Chih Tu, Ming-Yu Liu, Varun Jampani, Deqing Sun, Shao-Yi Chien, Ming-Hsuan Yang, and Jan Kautz. Learning superpixels with segmentation-aware affinity loss. In *Proc. of CVPR*, pages 568–576, 2018. 1

[42] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):583–598, 1991. 1, 2

[43] Yukang Wang, Yongchao Xu, Stavros Tsogkas, Xiang Bai, Sven Dickinson, and Kaleem Siddiqi. Deepflux for skeletons in the wild. In *Proc. of CVPR*, pages 5287–5296, 2019. 3

[44] Steffen Wolf, Constantin Pape, Alberto Bailoni, Nasim Rahaman, Anna Kreshuk, Ullrich Kothe, and FredA Hamprecht. The mutex watershed: efficient, parameter-free image partitioning. In *Proc. of ECCV*, pages 546–562, 2018. 2, 6, 7

[45] Yongchao Xu, Yukang Wang, Wei Zhou, Yongpan Wang, Zhibo Yang, and Xiang Bai. Textfield: Learning a deep direction field for irregular scene text detection. *IEEE Transactions on Image Processing*, 28(11):5566–5579, 2019. 3

[46] Ziming Zhang, Yun Liu, Xi Chen, Yanjun Zhu, Ming-Ming Cheng, Venkatesh Saligrama, and Philip HS Torr. Sequential optimization for efficient high-quality object proposal generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(5):1209–1223, 2017. 1

[47] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proc. of CVPR*, pages 2881–2890, 2017. 1

[48] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (9):884–900, 1996. 1, 2