Oriented Object Proposals

Shengfeng He and Rynson W.H. Lau
City University of Hong Kong
shengfeng_he@yahoo.com, rynson.lau@cityu.edu.hk

Abstract

In this paper, we propose a new approach to generate oriented object proposals (OOPs) to reduce the detection error caused by various orientations of the object. To this end, we propose to efficiently locate object regions according to pixelwise object probability, rather than measuring the objectness from a set of sampled windows. We formulate the proposal generation problem as a generative probabilistic model such that object proposals of different shapes (i.e., sizes and orientations) can be produced by locating the local maximum likelihoods. The new approach has three main advantages. First, it helps the object detector handle objects of different orientations. Second, as the shapes of the proposals may vary to fit the objects, the resulting proposals are tighter than the sampling windows with fixed sizes. Third, it avoids massive window sampling, and thereby reducing the number of proposals while maintaining a high recall. Experiments on the PASCAL VOC 2007 dataset show that the proposed OOP outperforms the state-of-the-art fast methods. Further experiments show that the rotation invariant property helps a class-specific object detector achieve better performance than the state-of-the-art proposal generation methods in either object rotation scenarios or general scenarios. Generating OOPs is very fast and takes only $0.5s$ *per image.*

1. Introduction

Object detection is one of the main research problems in computer vision in recent years. In order to detect and recognize the objects in an image, the sliding window paradigm is commonly used in the state-of-the-art detectors. However, exhaustively searching for objects across different sizes, positions and aspect ratios requires about 10^6 times predictions per image, which restricts the detection tasks from using more sophisticated classifiers.

The object proposal technique (or "objectness estimation") is proposed recently to cope with this barrier. Instead of directly recognizing objects in the images, a small set of object proposals (10^3 to 10^4 image windows) are first produced to cover most of the regions that are likely to contain an object. Taking advantages of recent efforts on this topic [1, 26, 10, 2, 25, 6, 32], several object detection algorithms use object proposals as input to significantly speed up the detection process, including the top detectors [28, 15] in the PASCAL VOC challenge [11].

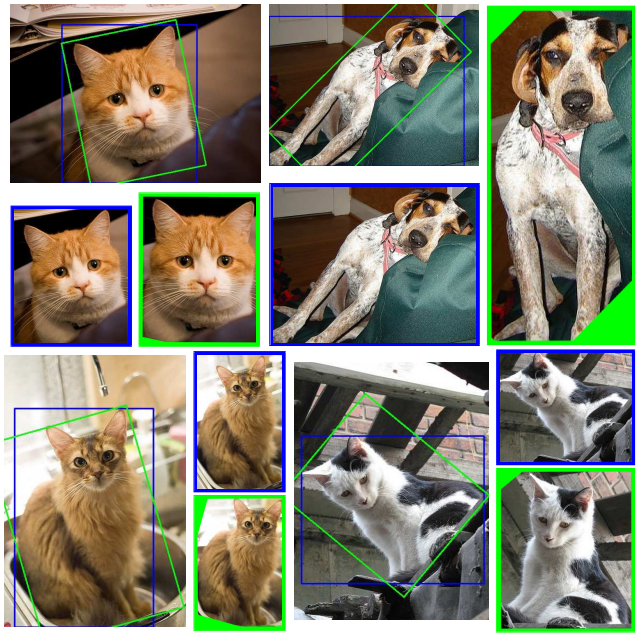


Figure 1: Examples of oriented object proposals (OOPs). Blue boxes are the ground truth, and green boxes are the OOPs. The proposed method is able to tightly locate objects of different orientations and achieve better detection performance, as OOPs reduce the ambiguity caused by orientation variation (see the realigned images with green borders).

Notwithstanding the demonstrated success, how proposals may help improve object detection performance is still being explored. An important problem, which has not been considered by existing proposal methods, is object rotation. Most of the object detectors are trained on a dataset with a small number of object orientations. Thus, objects with larger orientation differences may not be detected at runtime, as object appearances may vary largely due to orientation changes (see Figure 1). The traditional solution to this problem is to specifically train multiple classifiers at different orientations [18]. This, however, may significantly

increase the computation time and the number of false positives. In addition, larger bounding boxes may be produced to cover rotated objects (e.g., the ground truth boxes in Figure 1), which may increase background distraction.

In this paper, we propose a new approach to generate oriented object proposals (OOPs), which reduces object ambiguity and thus leads to better detection performance. Instead of measuring objectness of a set of windows, we take a different approach to generate proposals according to pixelwise objectness. Our idea is to actively search for objects across different positions, scales and aspect ratios to avoid exhaustive scoring every window. (The number of windows can be large especially when orientation is taken into consideration.) We formulate the object proposal generation problem as a generative probabilistic model, where the likelihood function corresponds to how likely a pixel belongs to an object. In order to describe the objectness of a pixel without bounding boxes, we introduce a background prior, together with a gradient map and structured edges [9], to construct the likelihood function. The objects are searched and located as the local maximum log-likelihood and the shape of the object is described by a full covariance matrix. The optimum position and shape are obtained by an efficient iterative searching algorithm.

Our new formulation of the object proposal generation problem leads to three main advantages. First, as OOPs can capture objects of different orientations, they help the detectors efficiently recognize them. Second, OOPs have tighter bounding boxes, as their positions and sizes may vary to fit the objects. Third, the number of proposals needed to achieve a high recall is reduced, as the required number of initial positions/sizes is significantly smaller (e.g., 5K vs. 100K [1]) due to the proposed efficient searching mechanism. Figure 1 shows some examples of our approach. The detection of these objects can be challenging even if we feed the detector with ground truth bounding boxes, due to object orientation variations. In contrast, OOPs are invariant to rotation, making it easier for the detector to detect. Extensive experiments on the PASCAL VOC 2007 dataset show that the proposed sampling strategy performs better than the state-of-the-art window scoring methods. We further examine the rotation invariant property of our approach on the Freestyle Motocross dataset [27] with two experiments. The first experiment shows that the class-specific object detector [14] performs better in terms of accuracy and efficiency (fewer classifiers) when fed with our proposals than with those from the state-of-the-art methods in in-plane object rotation scenarios. The second experiment shows that when training a class-specific object detector [14] with arbitrarily oriented objects, our method is still able to achieve significant better detection rates among the state-of-the-art proposal methods. The proposed method is fast and takes only $0.5s$ to process one image.

2. Related Work

Object proposal methods aim at efficiently generating a set of windows on an image for object detection. Depending on how the proposals are generated, object proposal methods can be roughly divided into *window scoring methods* and *superpixel grouping methods*. They are summarized as follows. (See [17] for a comprehensive review.)

Window scoring methods. They generate object proposals by ranking the objectness scores of a bunch of candidate windows. Alexe *et al.* [1] first sample a set of candidate windows according to a saliency map, and then measure objectness scores of the candidate windows by combining different cues, including multi-scale saliency, color contrast and superpixels straddling. Zhang *et al.* [31] and Rahtu *et al.* [24] extend this idea to use cascaded ranking SVM and structured output ranking, respectively, to combine different cues. Recently, Cheng *et al.* [6] propose an efficient method to generate object proposals. They use a simple linear classifier with the learned normed gradient features to measure the objectness scores of the candidate windows obtained from sliding windows. Zitnick and Dollár [32] also sample candidate windows using sliding windows, but then compute the objectness score by measuring the number of contours completely enclosed within a bounding box. These scoring methods typically run very fast and are more suitable for real-time detections. However, as they quantize the initial bounding boxes into fixed sizes, they may not produce accurate bounding boxes with a small number of proposals.

Superpixel grouping methods. They aim at localizing the objects by combining multiple superpixels into object proposals. Gu *et al.* [16] first propose to recognize objects using bounding box proposals obtained from hierarchically grouped superpixels. Carreira *et al.* [3] and Endres *et al.* [10] apply binary foreground-background segmentation on each initialized seed region. While [3] solves a constrained parametric min-cut (CPMC) problem to produce a set of proposals, [10] combines a set of cues to train a regressor to predict if a group of segments belong to an object. Uijings *et al.* [26] propose a learning-free approach to produce proposals by greedily merging adjacent superpixels according to low-level features. Two works [25, 21] extend this idea, where Rantalankila *et al.* [25] further considers global consistency, and Manén Manén *et al.* [21] applies a randomized Prim's algorithm for superpixel merging. Arbelaez *et al.* [2] propose a fast normalized cut hierarchical segmenter to combine multi-scale regions into object proposals. Krhenbhl and Koltun [19] train classifiers to place seeds in the image by computing level sets in geodesic distance transform. While these methods may produce object proposals that well fit the objects due to the use of superpixels, they usually have a much higher processing time than the window scoring methods.

We formulate the proposal generation problem in a different way than both types of methods. Our OOP approach can be conceptually regarded as a combination of both, as we start with some seed bounding boxes and each of them would gradually adapt to the object region. Although the proposed method does not rely on superpixels, our efficient greedy localization of object regions shares a similar principle to the greedy superpixel merging algorithm. As a result, our approach is both efficient and accurate.

3. The Proposed Approach

In this section, we present how our approach adaptively locate objects. Given a set of initial seed bounding boxes on an image I, each of them would actively search for objects. The final proposals are produced by determining the positions and shapes of the local maximums of a likelihood function. For a bounding box centered at θ, its shape can be approximated by an ellipse described by a full covariance matrix C. The log-likelihood of a certain observed set of features for this bounding box can be written as:

$$\log \mathcal{L}(\theta, C) = \sum_{m=1}^{M} \log p(f_m(\theta, C)), \tag{1}$$

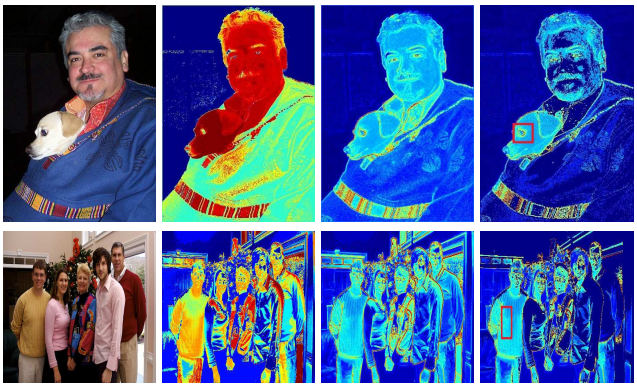
where M is the total number of features within a local area. $p(f_m)$ is the probability density function. f_m is the independent feature that describes an object and can be considered as a stochastic variable. The maximum of Eq. (1) can be achieved by maximizing the likelihood of a set of neighbor pixels [13, 23]:

$$\sum_{m=1}^{M} \log p(f_m(\theta, C)) \propto \sum_{n=1}^{N} l_n K(x_n; \theta, C), \tag{2}$$

where l_n is the likelihood of a pixel belonging to an object. x_n is the pixel position. K is the kernel describing the elliptical region. N is total number of pixels within a finite neighborhood of the kernel center. Our goal is to find a local position θ and shape C that maximize the log-likelihood function in Eq. (1) to estimate an object proposal. To do this, we first define object likelihood l_n.

3.1. Object Features

In [1], the distinctive characteristics of an object are explored. Cheng *et al.* [6] and Zitnick and Dollár [32] further show that the most effective feature for locating an object is a well-defined closed boundary (i.e., edges). Edge information is also suitable for our case, as edge pixels likely belong to some objects. However, using only edge information is not enough to locate the entire object as edges do not describe object interiors, and the searching would be stacked at textured regions if their densities are not continuous. In addition, we would expect the features to be able to at least partially represent most of the object regions so that no objects are missed. Hence, techniques that focus on certain objects, e.g., saliency map, are not appropriate. For the sake of efficiency, we propose to use a background prior to address this problem.



(a) (b) (c) (d)

Figure 2: Object confidence maps: (a) two input example images, (b) the background prior images, (c) the combined images of the three confidence maps, and (d) the locality-aware confidence maps of the red bounding boxes.

3.1.1 Background Prior

The background prior has been shown to be effective in object-related tasks, such as salient object detection [29, 30]. It suggests that most pixels near to the image boundaries unlikely belong to salient objects. The same is true for object proposal generation that majority of the colors appeared at image boundaries unlikely belong to the colors of an object in the image. Based on this assumption, we propose a histogram-based method as follows.

We first extract a histogram H_{bg} of the image boundary region, computed as $H_I - H_{ir}$, where H_I is the histogram of image I and H_{ir} is the histogram of the inner region of I (size is computed as $0.9\times$ size of I) . The object weight of pixel n is then computed as:

$$l_n^{bg} = 1 - \frac{1}{k^{bg}} \sum_{b=1}^{B} G(I_b, I_n) \cdot H_{bg}(b), \tag{3}$$

where k^{bg} is the normalization factor. B is the total number of bins. I_n is the color of pixel n and I_b is the corresponding quantized color of bin b. $G(I_b, I_n) = \exp(-\frac{\|I_b - I_n\|_2}{2\sigma^2})$ is the Gaussian function to control the influence of color distance. To obtain a more informative and distinct color histogram, we apply the strategy in [5] to get a quantized color histogram constructed by the most frequent colors.

Figure 2b shows two results of the background prior. This approach is able to assign non-zero values to most of the foreground pixels. More importantly, it serves as a complement to the edges, to provide reliable interior object information. Furthermore, it is able to produce a continu-

ous likelihood map to prevent bounding boxes emphasizing only on textured regions.

3.1.2 Final Object Likelihood

We combine the gradient map l^g and the structured edges l^e [9] to obtain the final object likelihood. The structured edge detector is proposed to efficiently detect object boundaries, and is considered as a complement to the gradient map here to reduce edge errors. The final weight of pixel n is computed as:

$$l_n = a_1 l_n^{bg} + a_2 l_n^e + a_3 l_n^g, \tag{4}$$

where a_1, a_2, a_3 are the normalized weights learned from the training set of PASCAL VOC 2007 using linear regression to maximize the likelihood in ground truth regions[1].

Figure 2c shows the resulting confidence maps. We can see that the objects in the two images are well described by the object features. However, if a bounding box tries to cover the regions with highest likelihood, it may become too large to cover them as one object due to the multiple local maxima around the initial position (e.g., covering both the dog's and the man's heads in the first row of Figure 2). To prevent all bounding boxes being too large, we introduce locality-awareness to the likelihood function. Given the size of an initial seed bounding box S_{bb}, we only consider the densities with the same colors as those within a local region of $3 \times S_{bb}$. As a result, the bounding box is constrained to cover local maxima, while it can still be able to cover a large region with similar colors. Figure 2d shows the locality-aware confidence maps. As can be seen, even multiple objects (the second row of Figure 2d) can be separated if an initial bounding box is located within one of them. Due to the proposed background prior, this separated region is assigned with non-zero continuous values, and thus it can be easily located by the following algorithm.

3.2. Finding Local Maximum Likelihood

Once the likelihood function is determined, finding the local maximum likelihood is equivalent to seeking the mode in the kernel density function. Mean-shift [7] is arguably the best known algorithm for this purpose, but it is typically limited to estimating the optimum position. To obtain the position θ and shape C that maximize the log-likelihood function in Eq. (2), we use a two-step iterative algorithm similar to [8, 4]. For each iteration i, we use a gradient descent procedure to find better θ and C. Our algorithm iterates towards the gradient direction and converges always to a local maximum. According to the Jensen's inequality and Eq. (2), we have:

$$\log \sum_{n=1}^{N} l_n K(x_n; \theta, C) \geq \sum_{n=1}^{N} \log l_n K(x_n; \theta, C). \tag{5}$$

[1]The learned weights are: $a_1 = 0.46, a_2 = 0.33, a_3 = 0.21$.

We let:

$$\sum_{n=1}^{N} \log l_n K(x_n; \theta, C) = \sum_{n=1}^{N} \log \left(\frac{l_n K(x_n; \theta, C)}{\tau_n} \right)^{\tau_n}, \tag{6}$$

where τ_n are arbitrary constants that satisfy $\sum_{n=1}^{N} \tau_n = 1$ and $\tau_n \geq 0$. The maximum of Eq. (5) can be achieved if this equality sign is satisfied.

Given the current states θ^i and C^i, the first step is to keep θ^i and C^i fixed and to find τ_n to maximize Eq. (6). The equality sign in Eq. (6) can be met if:

$$\tau_n = \frac{l_n K(x_n; \theta^i, C^i)}{\sum_{n=1}^{N} l_n K(x_n; \theta^i, C^i)}. \tag{7}$$

In the second step, we keep τ_n fixed to maximize Eq. (6), which only needs to minimize the part that depends on the configuration of θ and C:

$$r(\theta, C) = \sum_{n=1}^{N} \tau_n \log K(x_n; \theta, C). \tag{8}$$

Local Optimum Position: For $K(x_n; \theta, C)$ is a Gaussian kernel, the θ that achieves local maximum in the current iteration can be estimated by $\frac{\partial}{\partial \theta} r(\theta, C) = 0$:

$$\begin{aligned} \theta^{i+1} &= \sum_{n=1}^{N} \tau_n x_n \\ &= \frac{\sum_{n=1}^{N} x_n l_n K(x_n; \theta^i, C^i)}{\sum_{n=1}^{N} l_n K(x_n; \theta^i, C^i)}. \end{aligned} \tag{9}$$

The local optimum position θ^{i+1} is recursively estimated from the current position θ^i. As K is a Gaussian kernel, this estimation is equivalent to mean-shift [4].

Local Optimum Shape: Similarly, the covariance matrix C that describes the local maximum shape with a Gaussian kernel assumption can be obtained by:

$$C^{i+1} = \sum_{n=1}^{N} \tau_n (x_n - \theta^i)(x_n - \theta^i)^T. \tag{10}$$

However, the local optimum shape cannot be appropriately estimated as the maximum likelihood cannot be compared across different Gaussian kernel scales. The local maximum value decreases with a larger C. This bias can be eliminated by 'γ-normalization', which is typically used to select the best filtering scale [20]. The unbiased likelihood function can be obtained by re-normalizing kernel $K(x_n; \theta, C)$, so that the total mass of the kernel within the area of C is equal to one:

$$K_\gamma(x; \theta, C) = |C|^{\gamma/2} K(x; \theta, C). \tag{11}$$

This normalization only changes the second step (Eq. (8)), which now becomes:

$$r(\theta, C) = \sum_{n=1}^{N} \tau_n \log |C|^{\gamma/2} K(x_n; \theta, C). \tag{12}$$

According to $\frac{\partial}{\partial C} r(\theta, C) = 0$, we have the covariance update equation:

$$C^{i+1} = \beta \sum_{n=1}^{N} \tau_n (x_n - \theta^i)(x_n - \theta^i)^T, \quad (13)$$

where $\beta = 1/(1-\gamma)$. For different kernels, β should be set differently to obtain an unbiased solution [22]. According to the cut-off range of our Gaussian kernel, where pixels beyond 3-sigma from the kernel center are neglected in our implementation, the unbiased solution can be obtained by setting $\beta \approx 1.5$.

3.3. Bounding Box Approximation

With the obtained full covariance matrix C, we now approximate it by an elliptical region and then a bounding box. We first start from the case of an axis-aligned ellipse (i.e., the x- and y-axes are uncorrelated and the input data have a zero covariance) with the lengths of its two axes being $2\sigma_x$ and $2\sigma_y$. It can be defined as:

$$\left(\frac{x}{\sigma_x}\right)^2 + \left(\frac{y}{\sigma_y}\right)^2 = s, \quad (14)$$

where s is a coefficient determining the size of the ellipse. In our implementation, since the input data are distributed in the form of a rectangle, s is set to 3 to obtain an approximated ellipse with a size similar to a rectangle. In this case, σ_x and σ_y can be defined as the standard deviations of the data, and the lengths of the axes can be computed as $2\sigma_x\sqrt{s}$ and $2\sigma_y\sqrt{s}$, which may also be approximated as the width and height of the bounding box.

In case if the data are correlated, the above definition holds if there is a temporary coordinate system in which the ellipse becomes axis-aligned. The variances of the data are parallel to the new axes and can be described by the eigenvectors/eigenvalues of the covariance matrix. As a result, the lengths of the major and minor axes are computed as:

$$A_{major} = 2\sqrt{s\lambda_1}, \quad A_{minor} = 2\sqrt{s\lambda_2}, \quad (15)$$

where λ_1 and λ_2 are the eigenvalues of the covariance matrix. The rotation angle can be obtained by:

$$\alpha = \arctan \frac{v_1(y)}{v_1(x)}, \quad (16)$$

where v_1 is the eigenvector corresponding to the largest eigenvalue of the covariance matrix.

3.4. Implementation

Having discussed how an initial bounding box locates the object in a local area, we now discuss some implementation issues. Given an image, we first seed some initial bounding boxes at different positions with different scales. Seeds are placed on a regular grid at each scale. Other seeding approaches like learning-based seeding [19] may also be used. The total number of scales for both width and length is 5. As the bounding boxes that are very long or tall are less likely to contain an object, we neglect the scales that $|Scale_w - Scale_l| > 2$. The initial sizes are set to relatively small, ranging from 25 to 150 (divided equally to 5 scales of widths or lengths), to allow flexible searching for either large or small objects. (It is more likely to enlarge than to shrink a bounding box, as a large initial bounding box may contain multiple local maxima.)

Once the bounding boxes are initialized, each of them would recursively search for the local maximum. The maximum number of iterations is set to 30, but the iteration may stop if the change of a bounding box is small. The typical number is around 15 to 20 iterations. The final position and shape of a bounding box form a final object proposal. As the proposed approach is not a scoring approach, we adopt the scoring function of [32] to rank the bounding boxes. Other scoring approaches [1, 6] may be used. Finally, we apply Non-Maximum Suppression (NMS) [14] to remove the redundant bounding boxes (IoU larger than 0.8). Algorithm 1 summarizes the proposed method.

Algorithm 1 Our Proposal Localization Algorithm.

Input: the object likelihood l_n and the initial bounding boxes ($i = 0$) with positions θ^i and shape C^i.
for *each bounding box* **do**
1: Compute the locality-aware likelihood – Eq. (4).
2: **for** $i < i_{max}$ **do**
3: Compute τ_n – Eq. (7).
4: Estimate new position θ^{i+1} – Eq. (9).
5: Estimate new shape C^{i+1} – Eq. (13).
6: Compute ellipse & bounding box – Eq. (15), (16).
7: **if** (the change of the bounding box is small) **then**
8: Store the new bounding box.
9: Break.
10: **end if**
11: **end for**
end for
Compute objectness score for each bounding box.
Run NMS to remove the redundant bounding boxes.

4. Experimental Results

In this section, we quantitatively and qualitatively evaluate the proposed OOP approach. We have implemented it in C++ and tested it on a PC with an i7 3.4GHz CPU and 8GB RAM. It takes on average $0.5s$ to process a 500×400 image, with the iteration process taking about $0.4s$. We first evaluate the detection rate of the proposed localization algorithm, and then test the detector performance when fed with OOPs on scenarios with in-plane object rotations. Based on the results of a recent comprehensive evaluation [17], we select the top three window scoring methods (Objectness [1], Bing [6], EdgeBoxes [32]) and the top three superpixel grouping methods (Selective Search [26], MCG [2], Geodesic [19]) for comparison. For fairness, we use the

implementations provided by the authors with default parameter settings.

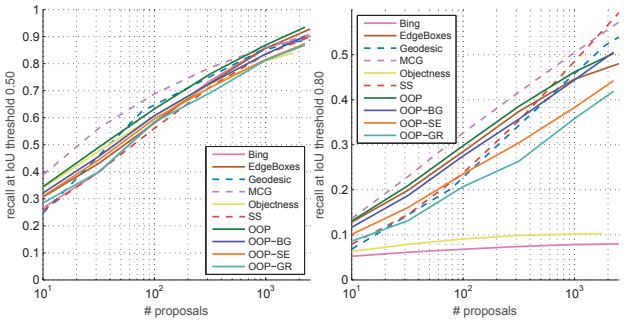
4.1. Detection Rate Comparison

We compare the detection rate of the proposed method with the state-of-the-art methods according to the evaluation results of [17] on the PASCAL VOC 2007 dataset [12], which consists of 4,952 test images. Note that the detection rate of the proposed OOP method is computed by ignoring orientation in the algorithm, as the ground truth does not consider object rotation. The effectiveness of the rotation invariant property is examined in Section 4.2.

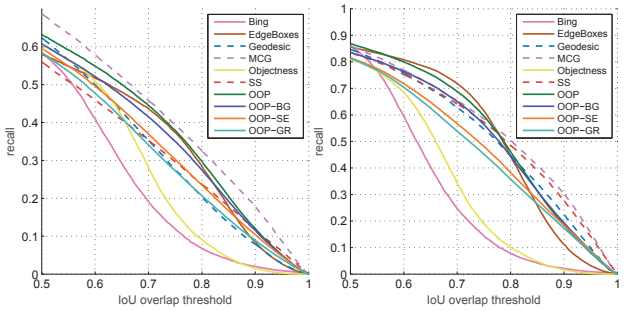
As a high detection rate is the most important property of a proposal method, we first evaluate the detection rate with respect to the number of proposals at intersection over union (IoU) above 0.5, as shown in Figure 3a. OOP consistently outperforms all the window scoring methods (solid lines) across different numbers of proposals, especially with a small number of proposals due to the active searching strategy. We further show the detection rate at IoU above 0.8 in Figure 3b. The superpixel grouping methods (dashed lines) generally perform well at this threshold, as they typically produce tight bounding boxes due to the use of superpixels. However, most of them are slow (Table 1) and not able to cover most of the objects even with a large number of proposals (e.g., above 1000), as shown in Figure 3a. As window scoring methods (solid lines) are restricted by the quantized window sizes, they may not produce tight bounding boxes. Edgeboxes [32] produces relative tighter ones, as it utilizes an exhaustive refinement after window sampling. However, this process is slow and thus it can only be applied to a small number of bounding boxes to maintain efficiency. Our OOP approach gets rid of massive window sampling, and our efficient searching algorithm enables windows to fit objects with a relative large range of scales and positions. As a result, OOP performs slightly better than Edgeboxes at IoU above 0.8.

Figures 3c and 3d show the detection rate with respect to the IoU threshold for 100 and 1000 proposals. Like in Figure 3a, OOP performs well when using a small number of proposals. When using 1000 proposals, its performance is still comparable to the state-of-the-arts. Most of the window scoring methods show good results at low IoU thresholds but drop dramatically as the threshold increases. The superpixel grouping methods have relatively lower detection rates at low IoU thresholds. However, they only drop gradually. Although OOP does not perform as well as the superpixel grouping methods at high IoU thresholds, it achieves better or comparable performance for a wide range of thresholds, which is desirable for most practical applications.

Figure 3 also shows the performances of individual cues that we use. We observe that none of the three cues alone can produce a result matching the combined one. The background prior (OOP-BG) achieves very good performance,



(a) Detection rate at IoU above 0.5. (b) Detection rate at IoU above 0.8.



(c) Detection rate of 100 proposals. (d) Detection rate of 1000 proposals.

Figure 3: Quantitative comparison of the state-of-the-art methods. Solid lines refer to window scoring methods, and dashed lines refer to superpixel grouping methods. The first row shows the detection rate with respect to the number of proposals at IoU above 0.5 and 0.8. The second row shows detection rates with respect to the IoU threshold, using 100 and 1000 proposals.

Method	**OOP**	Obj[1]	Bing[6]	EB[32]	SS[26]	MCG[2]	GOP[19]
Time (s)	**0.5**	3	0.2	0.3	10	30	1.2

Table 1: Comparison on the computation time.

as it produces continuous likelihood. Structured edges (OOP-SE) and gradient (OOP-GR) cannot produce object response within object interior. Hence, only large bounding boxes (obtained by re-scaling the confidence map) are able to locate objects as the re-scaled maps are more compact and relatively continuous (similar to Bing [6]).

Table 1 compares the computation time of the methods. As can be seen, OOP has a comparable running time to the state-of-the-art methods due to our efficient maximum likelihood locating strategy. The qualitative results are shown in Figure 4.

Overall, we have shown in this subsection that the proposed approach performs better or comparably to the existing methods, without considering the rotation invariant property. This property is demonstrated to be important to the object detection process in the next subsection.

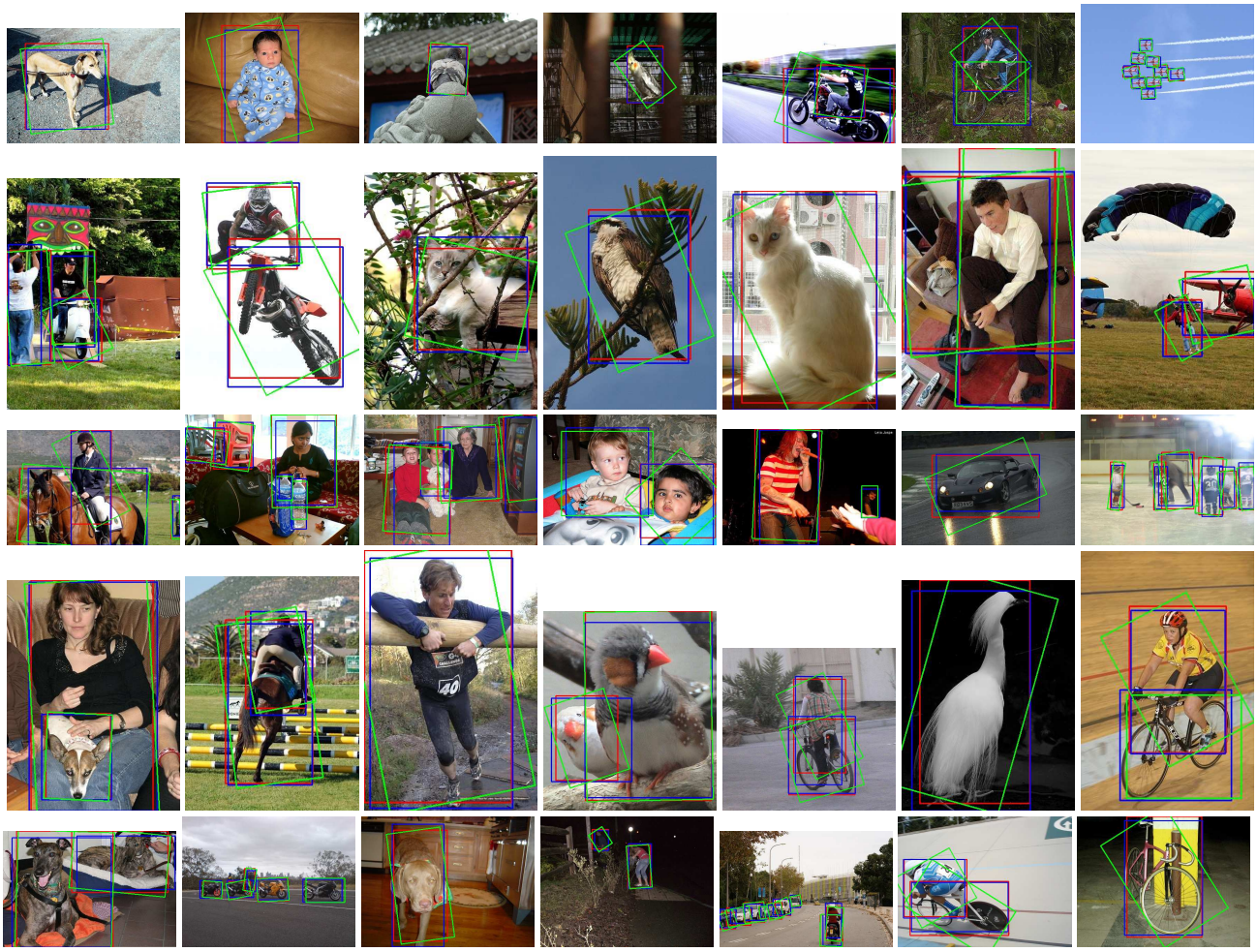


Figure 4: Qualitative evaluation of the proposed method. The red bounding boxes are the produced results closest to the ground truth, and the green boxes further consider orientation based on the red boxes. The blue boxes are the ground truth. Results are shown with 1000 object proposals.

4.2. Object Rotation Evaluation

We evaluate the rotation invariant property by conducting two experiments, training the detector with fixed number of orientations and arbitrary orientations to study how orientation affects detection performance.

Training with a fixed number of orientations. We first test the generated proposals using the DPM detector [14] on the Freestyle Motocross dataset [27], which consists of 100 images with in-plane rotated motorbikes for testing, 65 motorbike images (all without rotation) as positive samples and 900 random images as negative samples for training. We divide the evaluation into 3 tasks. Each task trains multiple classifiers with a different number of orientations by rotating the positive training data to 1, 4 and 16 orientations to detect rotated objects. All classifiers are trained individually with the corresponding oriented positive samples. Each proposal method is tested by feeding its proposals to the trained classifiers. For OOP, we crop the image against the bounding box and then rotate it to the closest axis before testing[2].

Method	**OOP**	Obj[1]	Bing[6]	EB[32]	SS[26]	MCG[2]	GOP[19]
mAP, 1 ori.	**55.5**	25.4	24.2	26.6	28.7	29.2	28.3
mAP, 4 ori.	**93.1**	54.9	54.1	57.7	59.5	60.4	59.7
mAP, 16 ori.	**93.3**	81.4	78.9	84.5	85.3	86.2	85.5

Table 2: Detection results on the Freestyle Motocross dataset with respect to the number of orientations used to train the classifiers. The proposed method consistently outperforms the state-of-the-art methods when encountering in-plane rotation, and is able to cover most of the cases when using only 4 orientations to train the classifiers.

Table 2 shows the detection results on the Freestyle Motocross dataset. All the compared methods perform poorly

[2]For 1 orientation, the bounding box will be rotated to align with the vertical or horizontal axis, but the detector is trained only using the horizontal orientation.

at 1 orientation, where they achieve lower than 30 mAP. This is because most of the motorbikes contain some degree of rotation and are treated as unseen objects by the detector. OOP achieves a much higher mAP, as the objects with a small amount of rotation (within $[-45°, 45°]$) can be placed in the correct orientation (i.e., horizontal). With 4 orientations, OOP is able to detect most of the objects. This is because most the object orientations can be easily approximated by these 4 orientations. As a result, the performance of 16 orientations for OOP is very similar to 4 orientations. Even with 16 orientations, the state-of-the-art methods are not able to achieve comparable detection rates as ours. One reason is that the rotated objects may still not be fully represented by 16 orientations. Another reason is that the bounding boxes produced by the other methods are not tight for rotated objects.

Training with arbitrary orientations. The previous experiment trains multiple classifiers with a fixed number of orientations. However, the orientation of an object can be arbitrary in practice. In this experiment, we rotate the positive training samples randomly, ranging between $[1°, 360°]$, to simulate a general scenario. Only one classifier is trained using 65 positive samples of arbitrary orientations and 900 negative samples, which have a similar statistic as a single category in a large dataset, such as PASCAL VOC 2007. Unlike the previous experiment, we rotate the ground truth bounding boxes to the closest axis here when training the detector for OOP. Obviously, this training process can only be used with oriented proposals, where the detector is trained with the 4 axis orientations.

Table 3 shows the detection performance of training with arbitrary orientations. OOP outperforms all the existing proposal methods in this more general evaluation. It effectively reduces object ambiguity caused by orientation changes in both training and testing. In addition, the object appearance can be learned and detected very easily with much less training data (i.e., training data needs to cover 4 orientations vs. arbitrary orientations). The training process of OOP in this experiment is similar to the last experiment using four orientations for training. The main differences are this experiment contains much less positive data (i.e., 65 vs. 4×65) and only one classifier is trained for each compared method (more efficient and practical). Figure 5 shows some results on the Freestyle Motocross dataset. For the sake of clarity, we only show the bounding boxes from the best window scoring method (i.e., EdgeBoxes [32]), the best superpixel grouping method (i.e., MCG [2]), and OOP. As can be seen, handling in-plane rotation is beyond the capability of the existing methods. The bounding boxes generated by the proposed method are able to fit the objects of different orientations well.

Method	**OOP**	Obj[1]	Bing[6]	EB[32]	SS[26]	MCG[2]	GOP[19]
mAP	**85.2**	64.8	63.3	71.5	69.6	72.1	70.5

Table 3: Detection results with training with arbitrary orientations on Freestyle Motocross. The proposed OOP reduces the object ambiguity in both training and testing, and thus significantly outperforms the existing methods.

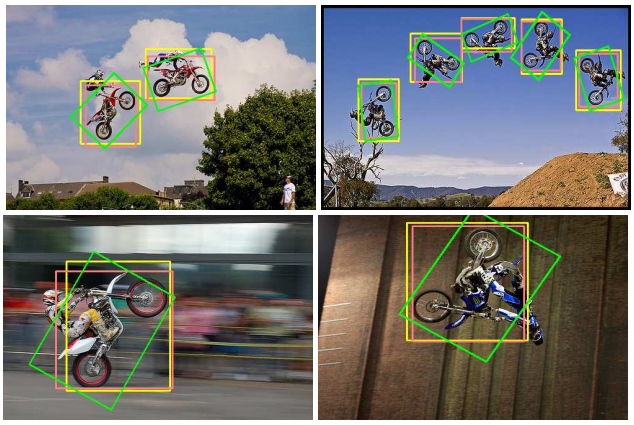


Figure 5: Examples of the Freestyle Motocross dataset. The green, pink and yellow bounding boxes are the best produced results of the proposed method, MCG [2] and EdgeBoxes [32], respectively. The proposed method is able to handle in-plane rotation of the motorbikes well.

5. Conclusion and Limitations

In this paper, we propose a new approach to generate oriented object proposals to reduce the detection errors caused by object orientation changes. We formulate the proposal generation problem as a generative probabilistic model, where the location, size and orientation of each object proposal can be determined by finding the local maximum likelihood. The likelihood function is constructed by a background prior as well as edge information. An iterative local maximum searching algorithm is used to efficiently locate objects of different shapes. Experiments show that the proposed method not only outperforms the state-of-the-art window scoring methods, but also resolves the object rotation problem in proposal estimation. One limitation of the proposed method is that it requires training with oriented ground truth to obtain superior detection performance, which is laborious than annotating regular bounding boxes. Although R-CNN [15] provides great flexibility to adapt with different object proposals, it is inaccurate to determine which box is used by comparing the IoU between OOPs and regular ground truth. Our future work includes annotating a large dataset and then exploring the effectiveness of OOP on such a large dataset.

Acknowledgements. This work was partially supported by a GRF grant from the RGC of Hong Kong (Project Number: CityU 115112).

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE TPAMI*, 34(11):2189–2202, 2012.

[2] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014.

[3] J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *IEEE TPAMI*, 34(7):1312–1328, 2012.

[4] M. Carreira-Perpindn. Gaussian mean-shift is an EM algorithm. *IEEE TPAMI*, 29(5):767–776, 2007.

[5] M. Cheng, N. Mitra, X. Huang, P. Torr, and S. Hu. Global contrast based salient region detection. *IEEE TPAMI*, 2014.

[6] M. Cheng, Z. Zhang, W. Lin, and P. Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, 2014.

[7] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE TPAMI*, 17(8):790–799, 1995.

[8] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of The Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[9] P. Dollár and C. Zitnick. Structured forests for fast edge detection. In *ICCV*, pages 1841–1848, 2013.

[10] I. Endres and D. Hoiem. Category-independent object proposals with diverse ranking. *IEEE TPAMI*, 36(2):222–234, 2014.

[11] M. Everingham, S. Eslami, L. van Gool, C. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes challenge: A retrospective. *IJCV*, pages 1–39, 2014.

[12] M. Everingham, L. Gool, C. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010.

[13] J. Fan, M. Farmen, and I. Gijbels. Local maximum likelihood estimation and inference. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(3):591–608, 1998.

[14] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE TPAMI*, 32(9):1627–1645, 2010.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[16] C. Gu, J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, pages 1030–1037, 2009.

[17] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *arXiv:1502.05082*, 2015.

[18] C. Huang, H. Ai, Y. Li, and S. Lao. Vector boosting for rotation invariant multi-view face detection. In *ICCV*, pages 446–453, 2005.

[19] P. Krhenbhl and V. Koltun. Geodesic object proposals. In *ECCV*, pages 725–739. 2014.

[20] T. Lindeberg. Feature detection with automatic scale selection. *IJCV*, 30(2):79–116, 1998.

[21] S. Manén, M. Guillaumin, and L. van Gool. Prime object proposals with randomized Prim's algorithm. In *ICCV*, 2013.

[22] R. Maronna. Robust m-estimators of multivariate location and scatter. *The Annals of Statistics*, 4(1):51–67, 1976.

[23] J. Polzehl and V. Spokoiny. Propagation-separation approach for local likelihood estimation. *Probability Theory and Related Fields*, 135(3):335–362, 2006.

[24] E. Rahtu, J. Kannala, and M. Blaschko. Learning a category independent object detection cascade. In *ICCV*, pages 1052–1059, 2011.

[25] P. Rantalankila, J. Kannala, and E. Rahtu. Generating object segmentation proposals using global and local search. In *CVPR*, 2014.

[26] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.

[27] M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, and A. Sanfeliu. Efficient rotation invariant object detection using boosted random ferns. In *CVPR*, pages 1038–1045, 2010.

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

[29] Y. Wei, F. Wen, W. Zhu, and J. Sun. Geodesic saliency using background priors. In *ECCV*, 2012.

[30] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, 2013.

[31] Z. Zhang, J. Warrell, and P. Torr. Proposal generation for object detection using cascaded ranking SVMs. In *CVPR*, pages 1497–1504, 2011.

[32] C. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.