Object Proposal Algorithms in the Wild: Are they Generalizable to Robot Perception?

Darren M. Chan and Laurel D. Riek
Computer Science and Engineering, UC San Diego

Abstract—The recent emergence of object proposal algorithms in the computer vision community shows great promise to addressing difficult problems in robotic such as object discovery and salient object detection. However, it is difficult to determine how these algorithms actually perform for real-world robot vision applications, because the standard evaluation protocol uses datasets which do not adequately account for real-world noise (motion blur, occlusion, etc.). We evaluated several state-of-the-art object proposal algorithms using naturalistic datasets from the robotics community, and found a substantial performance drop across all algorithms. This suggests that many object proposal algorithms are not as generalizable as the computer vision literature purports, which can have a significant impact on how they are applied to robotics. We also conducted a study on how each algorithm is influenced by specific kinds of real-world robot vision challenges, including variable brightness, gamma correction, Gaussian blur, and Gaussian noise. Our results provide insight into certain weaknesses of object proposal algorithms, which can be used to gauge how they might be suitable for different robotics applications. It is our intent that this work will motivate future research about how to design more flexible and robust object proposal algorithms for the robotics community.

I. INTRODUCTION

Robots are becoming ubiquitous in manufacturing, healthcare, and entertainment. Rather than being separated from people, strides are being made to enable robots to work in close proximity to them. However, this has proven to be difficult, because human spaces are often unpredictable and prone to abrupt changes, where robots need to be adaptable [1], [2]. To enable this capability, robots must robustly understand their surroundings so that they can make coherent decisions about how to physically interact with them, while maintaining a safe environment for people [3]–[8].

One hurdle that prevents robots from being effective in the real world (i.e. outside of controlled laboratories), is the ability to visually perceive people and meaningful objects around them. To become truly useful, vision algorithms need to be generalizable to any situation, to allow robots to autonomously operate in the real world, i.e., in the wild.

To robustly perceive objects and pedestrians (herein: objects) in the world, robots must efficiently perform two tasks: localize objects and recognize them. In computer vision, object localization falls under the broad area of problems in object detection research, where the pixel coordinates of all objects in images must be located.

Object recognition, or classification, attempts to solve the problem of identifying an object in question— one that has already been localized. For example, if a robot already detected and localized all animals in a scene, its next task would be to classify each of them (e.g. dog, cat, bird).

In the past, vision algorithms incorporated the sliding window paradigm, which performed both object localization and recognition in one simultaneous step by searching across all possible image positions and scales in a brute-force manner (c.f. HOG [9]). However, by delineating localization and recognition, object detection can be achieved much more quickly, demonstrating superior accuracy over the sliding window approach [10].

Object proposal algorithms emerged to solve object localization more efficiently. In modern object detection algorithms, classification is typically performed over fewer, select regions (i.e. proposals) to determine the presence of class specific objects [10]–[12].

In fact, roboticists are beginning to incorporate object proposal algorithms into their systems. For example, Sunderhauf et al. [13] used object proposals to predict semantic mappings of objects and places. Object proposals also show great promise toward solving unseen object discovery, because they excel at hypothesizing class-agnostic objects [14].

Despite the achievements of object proposal algorithms, we argue that there is a problem regarding their generalizability: the standard evaluation data are too clean to make assumptions about how well object proposal algorithms actually perform in most applications. A close inspection of the PASCAL and Microsoft COCO datasets reveals that the images contain objects of interest that are center biased with minimal amounts of occlusion. Moreover, it is difficult to make a decision about which algorithms work well for robotics, because there is little study about how they perform with the influence of real-world image degradation such as noise, contrast, and blur.

This paper discusses our evaluation of the top-performing object proposal algorithms in the context of robotics, where we address three research questions: (1) How well do the state-of-the-art algorithms perform on datasets that include...
real-world noise, object occlusion, and motion blur? (2) Are deep learning approaches accurate and computationally inexpensive enough to be used in robotics? (3) How fast are the top-performing algorithms, when used on a small computing platform that might be mounted on a mobile robot?

Our contributions are fourfold. First, we showed that the standard evaluation dataset, PASCAL, is flawed for determining the generalizability of object proposal algorithms, particularly in the context of robotics. This raises some concern about how object proposals algorithms are currently being evaluated.

Second, we tested two recent deep learning approaches (DeepMask [15] and SharpMask [16]) and showed that they produced high recall on more naturalistic datasets using only a small number of proposals, suggesting that they have great potential for real-time robotics applications.

Third, we conducted a study of algorithm execution time on portable hardware to find that most algorithms are suitable for real-time robotics applications.

Fourth, we gained insight into algorithm weaknesses by conducting a controlled study to see how they performed when introduced to four kinds of image perturbations (brightness, gamma correction, Gaussian blur, and Gaussian noise). Our results will be useful to the robotics community, which can be used to gauge performance and computation time trade-offs. It is our intent that this work will be used as a guideline for how roboticists should select object proposal algorithms to be incorporated in their system designs.

II. RELATED WORK

Object proposal algorithms are designed to extract image regions that are most likely to contain objects. In this way, background regions can be filtered out to enable vision pipelines to attain higher speed. Object proposal algorithms can follow many approaches, but generally fall under one of two types: Segmentation and Window-scored (see Figure 1).

Segmentation-based approaches are designed to produce object candidates at the pixel level, typically grouping adjacent pixels by some measure of similarity.

Window-scored algorithms use various imaging strategies to generate a set of hypothesized object regions, or windows. Each window is evaluated with a scoring function, which predicts the likelihood that the region contains an object. Windows containing low prediction scores can then be quickly rejected with low computational expense.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Ego-centric</th>
<th>Spatio-temporal</th>
<th>Blur</th>
<th>Occluded Objects</th>
<th># Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>B3DO [25]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>849</td>
</tr>
<tr>
<td>RGBDMV [26]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>18.4k</td>
</tr>
<tr>
<td>PASCAL [22]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>11.5k</td>
</tr>
<tr>
<td>PASCAL+</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>124k</td>
</tr>
</tbody>
</table>

TABLE II
A LIST OF DATASETS USED IN OUR EVALUATION, COMPARING THE PRESENCE OF SOME ROBOT VISION CHALLENGES.

In this paper, we motivate discussions about how well object proposal algorithms generalize to robotics, by testing them with more realistic datasets. To our knowledge, we are the first to conduct a comprehensive study of object proposal algorithms on multi-view datasets, which include spatio-temporal image sequences and naturally occurring perturbations (noise, occlusion, blur, etc.)—factors that are elemental to robot vision.

III. METHODOLOGY

Using standard evaluation metrics in computer vision [23], we measured the recall performance of seven top-performing object proposal algorithms [16]–[21], [27] (shown in Table I). Among them are three window-scored methods, four non-deep learning segmentation methods, and two deep learning segmentation methods.

To explore these algorithms in the context of mobile robotics, we tested them on three egocentric datasets containing images with real-world noise, occlusion, and motion blur. This differs from other approaches, where evaluations were performed on datasets containing clean images and capture bias [22], [24], [28].

We also wanted to study how the algorithms might be weak to specific types of perturbations. However, we found that naturally occurring image perturbations are difficult to quantify in such a way to create a controllable experiment that fairly assesses their effects in isolation. Thus, we modified the PASCAL dataset to create our own version, PASCAL+, which we augmented with controlled amounts of brightness, gamma correction (simulating contrast), Gaussian blur, and Gaussian noise.

To give researchers insight into how these algorithms might be designed on smaller, more mobile robotic platforms, we performed our experiments using less powerful hardware than those used in other evaluations.

A. MEASURING PROPOSAL PERFORMANCE

To quantitatively measure the detection performance of a proposal algorithm, two principles must be considered: Intersection over Union (IoU) and Recall.

IoU is indicative of an object proposal algorithm’s ability to accurately localize objects, which measures the similarity between an algorithm’s generated hypothesis and the dataset’s ground truth.

Recall measures an algorithm’s ability to recover image regions, or candidate windows, that match all of the actual object locations in a dataset. Several varieties of the recall metric have been introduced in the literature, so we briefly discuss the intuition behind each of them [23], [24].

To analyze the behavior of an algorithm’s ability to recall and localize objects, a graphical function of recall vs. IoU is derived by computing the recall of each algorithm at various IoU thresholds. All hypothesis and ground truth pairings having an IoU greater than or equal to a threshold value are marked as correct detections, and contribute to the recall for that IoU threshold.

The general consensus among researchers in the computer vision community is that recall computed at IoU values less than 0.5 do not provide substantial information about an algorithm’s performance. Thus, in practice, IoU values are only evaluated in the range of 0.5 to 1, where 0.5 indicates loose overlap, and 1 indicates perfect overlap.

We describe the three recall metrics that we used to evaluate the object proposal algorithms:

- Recall vs. IoU with fixed number of proposals
- Recall at IoU vs. number of proposals
- Area under recall vs. number of proposals

1) Recall as a Function of IoU with Fixed # of Proposals:
To observe the localization accuracy of object proposal algorithms, recall as a function of IoU is preferred because it represents an algorithm’s ability to localize objects as the IoU criteria becomes more strict. In this metric, recall is computed for each IoU threshold between 0.5 and 1. Here, the top K proposals are used in the evaluation, where K is a user-defined number of proposals.

In our evaluation, we selected a maximum value of $K = 1000$ for GOP, DeepMask, Objectness, Selective Search, and Rahtu. However, Randomized Prim does not allow us to have direct control over this parameter, so we report $K = 900$ proposals, which were automatically generated.

We note that the implementation of SharpMask, provided by Facebook, contained memory leak issues at time of writing but, decided to include it in our study because it remains to be one of the most prominent object proposal algorithms in the computer vision literature. However, we were not able to set the $K$ value of SharpMask to $K = 1000$, as we did with the other algorithms. Instead, we systematically decreased the $K$ parameter starting at $K = 1000$ until the algorithm was stable enough for evaluation at $K = 300$.

2) Recall at IoU as a Function of # Proposals: In some circumstances, it is desirable to place a lower bound on localization accuracy when studying object proposal algorithm behavior. For instance, an IoU value of 0.5 is commonly
regarded as the optimal balance between loose and tight fitting. When IoU is set to 0.5, all hypothesized regions must at least have an IoU score greater than or equal to 0.5 to be considered correct detections. We conduct two separate evaluations by fixing IoU to 0.5 and 0.7.

3) Area Under Recall (AUC) as a Function of # Proposals: In AUC, the area under the recall curve (for all IoU in range 0.5 to 1) is computed for a fixed number of proposals (in the range of 10 to 1000). AUC combines recall, IoU, and the number of proposals in a single function, and is indicative of general recall performance.

To measure algorithm performance on PASCAL+, we computed AUC with respect to perturbation type and amount. Moreover, we fixed the number of proposals to a practical value of 100. This mitigates the undesirable effect that a larger number of proposals increases the likelihood that objects are recalled at random, rather than by design.

B. Computation Time

To gain insight about the computational performance of these algorithms when used on a mobile robot, we conducted our evaluation on a high performance laptop equipped with a quad-core i7-6600HQ CPU, an Nvidia GTX970M 3GB GPU, and 8GB RAM.

DeepMask and SharpMask were evaluated using the laptop’s GPU. Randomized Prim, Geodesic, Objectness, Selective Search, and Rahtu were evaluated using the laptop’s CPU because they were not designed for GPU operation.

C. Datasets

We conducted our experiment using video frames and images that are representative of real-world robot vision challenges, which include naturalistic motion blur, occlusion, camera noise, non-centric biasing, and non-ideal lighting conditions. We were also interested in studying the performance of object proposal algorithms from an egocentric and spatiotemporal point of view that captures the morphology of objects in space and time.

We selected two datasets commonly used to evaluate object detection for robotics: RGB-D Multi-view [26], and Berkeley 3-D Object [25] (shown in Table 1).

The RGB-D Multi-view (RGBDMV) dataset contains over 300 different objects, and is frequently used to validate scene understanding ([29], [30]) and SLAM ([31], [32]) algorithms. Images of objects were captured using a PrimeSense RGB-D camera from over 24 unique viewpoints, where they may be occluded, contain motion blur, and are prone to disappear and reappear within the image sequence.

The Berkeley 3-D Object (B3DO) dataset contains Kinect V1 images depicting occluded objects in cluttered indoor environments, with natural lighting. Some images in this dataset are spatiotemporally sequenced, where motion blur and a large amount of camera noise is present.

To study the effects of image perturbations in isolation, we create our own version of the PASCAL dataset, PASCAL+. We split the PASCAL dataset in half, applying four kinds of perturbations: brightness, gamma correction, Gaussian blur, and Gaussian noise (shown in Figure 3). In total, PASCAL+ contains over 124k images.

To mimic how brightness affects imaging in robot vision, we applied additive illumination to each image. This was achieved by adding scalar intensity values (for each RGB channel) to each pixel, ranging from $-200$ to 200 in increments of 40. To preserve the 8-bit format of the images, pixels with intensity less than 0 or pixels with intensity exceeding 255 were scaled to 0 and 255, respectively.

To mimic various levels of dynamic contrast, we transformed each image using gamma correction, defined by $I_{out} = I_{in}^{\gamma}$. We applied gamma correction to each pixel of each image, varying $\gamma$ from 0.5 to 2.5 in increments of 0.5.

To simulate camera and motion blur, Gaussian blur was applied to each image with an adaptive square filter of size $2 \times \text{ceil}(2\sigma) + 1$, varying $\sigma$ from 2 to 10 in increments of 2.

Noise negatively impacts how illumination is perceived, which affects how object proposal algorithms perform. To this end, we applied zero-mean additive Gaussian white noise to each image, varying $\sigma^2$ from 0.02 to 0.1 in 0.02 increments.

D. Procedure

The robot vision datasets were repurposed for this experiment by removing category labels. Furthermore, we only evaluated on RGB images.

For each algorithm in Table 1, we generated $K$-proposals for each dataset listed in Table 1. Each of the recall scoring functions (previously described in Section III) were computed, and graphed (shown in Figures 4 and 5).

DeepMask and SharpMask are built on top of the ResNet-50 model (pre-trained on ImageNet [33]), and were additionally trained using segmentation annotations from the Microsoft COCO dataset [28]. The models were trained on data that are independent from PASCAL to evaluate their generalizability. We note that these training and evaluation procedures of DeepMask and SharpMask are identical to those from their respective papers.

IV. RESULTS

In this section, we present a summary of our results (see Table 1 and Figures 4 and 5).
A. Recall as a Function of IoU with Fixed # of Proposals

Across all datasets (see Figure 4), DeepMask and SharpMask produced the highest recall, which suggests that they are able to locate the most objects.

At IoU ≥ 0.7, all non-deep learning approaches substantially dropped in performance, while DeepMask and SharpMask had a shallower performance cutoff.

B. Recall at Fixed IoU as a Function of # Proposals

At IoU = 0.5 (see Figure 5), all algorithms achieve high recall because objects are detected with more forgiving overlap criteria between the hypothesized object regions and the ground truths. At IoU = 0.7, the recall of all algorithms decreased, because the overlap criteria are more strict. We found that the recall of Objectness suffered to a greater degree than the other algorithms from IoU = 0.5 across all datasets.

Comparing algorithm performance on B3DO to the performance on PASCAL, our results show that a substantial performance difference at # proposals ≤ 100, where the recall gap between deep learning methods and non-deep learning methods is considerably narrower.

In contrast, algorithm performance on the PASCAL and RGBDMV datasets are similar. This is somewhat expected, because RGBDMV contains a smaller number of objects that are less cluttered in the camera view.

C. Area Under Recall (AUC) as a Function of # Proposals

In general, most algorithms performed better on the PASCAL dataset at a lower number of proposals (see Figure 6), which highly suggests that the standard evaluation protocol does not account for realistic image noise.

However, it is interesting to see that the performance of the algorithms is generally consistent across all datasets at higher number of proposals. This implies that the algorithms are more generalizable when set to produce a large number of candidate object regions. However, this case is undesirable because this leads to higher computation cost, where these algorithms should be ideally designed to produce the highest AUC and recall using the least number of proposals.

D. Evaluation on PASCAL+

The results of our evaluation on PASCAL+ are presented in Figure 7. Most algorithms were less affected by gamma correction, blur, and noise than expected, though performance gradually declined as the perturbation magnitude increased. However, we found that all the algorithms performed worse when varying the brightness.

E. Computation Time

We present the execution time of each algorithm, corresponding to the configurations used in our evaluation (K values) in Table III. We also report the average time it takes for each algorithm to compute one proposal.

We found that Randomized Prim performed faster than the others by a substantial margin at 0.69ms per proposal, while Rahtu performed the slowest at 4.70ms per proposal.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>K</th>
<th>Cumulative Runtime per Image at K</th>
<th>Computation Time per Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Randomized Prim</td>
<td>900</td>
<td>0.63s</td>
<td>0.69ms</td>
</tr>
<tr>
<td>GOP</td>
<td>1000</td>
<td>1.25s</td>
<td>1.79ms</td>
</tr>
<tr>
<td>DeepMask</td>
<td>1000</td>
<td>3.13s</td>
<td>3.06ms</td>
</tr>
<tr>
<td>Objectness</td>
<td>1000</td>
<td>3.92s</td>
<td>3.29ms</td>
</tr>
<tr>
<td>SharpMask</td>
<td>300</td>
<td>1.28s</td>
<td>4.27ms</td>
</tr>
<tr>
<td>Selective Search</td>
<td>1000</td>
<td>4.45s</td>
<td>4.45ms</td>
</tr>
<tr>
<td>Rahtu</td>
<td>1000</td>
<td>4.70s</td>
<td>4.70ms</td>
</tr>
</tbody>
</table>

TABLE III

COMPUTATION TIME (PER 640 × 480 IMAGE) OF THE ALGORITHMS USED IN OUR EVALUATION.

V. DISCUSSION

Addressing the first research question (how the state-of-the-art object proposal algorithms perform on images containing real-world noise, occlusion, and motion blur), we found that all algorithms dropped in recall performance on the robot vision datasets, particularly for # proposals < 100.

Despite the claim that object proposal algorithms are designed to be generalizable [18], our results show that several of the algorithms did not perform consistently in relation to each other. For instance, GOP actually performed better on the robot datasets than on PASCAL. Additionally, we found that Rahtu performed better than Objectness on RGBDMV, while the inverse was true for B3DO and PASCAL.

In contrast to the findings in the computer vision literature, our results suggest that it is difficult to derive algorithm performance solely using the PASCAL dataset to conduct evaluations on object proposal algorithms. This is especially notable for proposal algorithms at a lower number of proposals, where the difference in recall performance is more dramatic. Arguably, this is most important for robotics and real-time systems, where high recall at low number proposals is desirable.

We found that the algorithms were more robust to perturbations than expected, as shown in our evaluation on PASCAL+. Nevertheless, we still show that perturbations can affect some algorithms more than others. For instance, although Objectness and Rahtu scored lower in overall recall, they were less affected by perturbations. These results open new and important discussions about how object proposal algorithms should be evaluated and designed for systems and robots operating in the wild.

Regarding the second research question (are deep learning approaches accurate and computationally inexpensive enough for robotics), we found that DeepMask and SharpMask performed the best across all metrics by a substantial margin on the robotic vision datasets. We also found this to be true on the PASCAL+ dataset, where DeepMask and SharpMask consistently attained the highest AUC scores, despite not being trained for the added perturbations. Moreover, even though SharpMask was limited by the number of proposals it could produce, it consistently performed better than the non-deep learning methods.

We also found that DeepMask and SharpMask achieved higher recall using only a small number of proposals (AUC ≈ 0.7 for # proposals < 100), which suggests that they might
be best-suited for robotic systems with GPU platforms that are powerful enough to support them. However, deep learning methods can be computationally expensive, particularly regarding GPU memory. Consequently, systems with lower computational power, such as those found on mobile robots, might be incompatible.

Regarding the third research question (are object proposal algorithms fast enough for computationally-limited mobile robots), we found that none of the algorithms in our evaluation took a substantial amount of time to run on a CPU (non-deep learning methods) or GPU (deep learning methods). We found that most of the algorithms were generally able to achieve high recall using only a few hundred proposals which is still practical within most robotics contexts [34].

In terms of practical insights, we suggest using DeepMask at 100 proposals for general vision applications on robotic systems that have sufficient GPU computing resources, which appears to be optimal for high recall and low computation time. For non-GPU systems, we suggest using GOp, which scored the highest AUC across the robot datasets. For applications that require high localization accuracy (e.g., pose estimation [35], [36]), we suggest using DeepMask, which has the highest recall at IoU thresholds $\geq 0.8$.

In summary, we found that evaluating object proposal algorithms on the PASCAL dataset is not as generalizable as the computer vision literature might claim, and is not necessarily suitable for designing and evaluating new algorithms for robotics contexts.

In our future work, we plan to deploy object proposal algorithms onboard mobile robots to explore their real-time performance on more constrained test hardware and in more realistic test environments [37], [38]. This will enable roboticists to be better informed regarding algorithm trade-offs between computation time and recall. It is our intent that this work will encourage roboticists and researchers from the computer vision community to work together to design robust and flexible algorithms that are truly generalizable.

REFERENCES

Fig. 4. Recall vs. IoU and Recall vs. # Proposals

Fig. 5. AUC vs. # Proposals on the PASCAL (left), B3DO (center), and RGBD-MV (right).

Fig. 6. AUC at 100 proposals vs. image perturbations from PASCAL+.