UNIVERSITY OF CALIFORNIA SAN DIEGO

Unseen Object Perception for Robots

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Darren M. Chan

Committee in charge:

Professor Laurel D. Riek, Chair
Professor Manmohan Chandraker
Professor Virginia de Sa
Professor Ryan Kastner
Professor Jishen Zhao

2021

The dissertation of Darren M. Chan is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

DEDICATION

For fun.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Many people have supported me during my Ph.D., and I would like to thank them.

I thank my advisor, Dr. Laurel Riek, who has supported and guided every step of my research, and for continually challenging me to bring out the best of my abilities. I also thank my dissertation committee, Dr. Manmohan Chandraker, Dr. Virginia de Sa, Dr. Ryan Kastner, and Dr. Jishen Zhao for their invaluable feedback and suggestions that have strengthened my work.

I thank the current and former members in my lab, Angelique Taylor, Tariq Iqbal, Hee Rin Lee, Sachiko Matsumoto, Alyssa Kubota, Maryam Pourebadi, Auriel Washburn, Michael Gonzales, Cory Hayes, and Maryam Moosaei for their help in strengthening my writing, and for their friendship during my time in graduate school.

I thank my family, Don, Wanda, and Kevin Chan, for their unending support, and for ultimately providing me the opportunity to further my education.

I thank Masako Okimura for her unwavering love, patience, and for her uplifting humor during the course of all my failed experiments.

I also thank my kind housemates from Lighthouse Bible Church, for their encouragement, prayers, and spiritual guidance during my time in San Diego.

My research was made possible by funding support from the National Science Foundation. Chapter 3 of this dissertation contains material from "Faster Robot Perception Using Salient Depth Partitioning", by D. M. Chan, A. Taylor, L. D. Riek, which appears in the Proceedings of IEEE/RAS International Conference on Intelligent Robots and Systems, 2017 [28]. The dissertation author was the primary investigator and author of this paper.

Chapter 4 contains material from "Object Proposal Algorithms in the Wild: Are they Generalizable to Robot Perception?", by D. M. Chan and L. D. Riek, which appears in the Proceedings of IEEE/RAS International Conference on Intelligent Robots and

Systems, 2019 [25]. The dissertation author was the primary investigator and author of this paper.

Chapter 5 contains material from “Unseen Salient Object Discovery for Monocular Robot Vision?”, by D. M. Chan and L. D. Riek, which appears in Robotics and Automation Letters and in the Proceedings of IEEE/RAS International Conference on Intelligent Robots and Systems, 2020 [27]. The dissertation author was the primary investigator and author of this paper. This chapter also contains material from ”Unsupervised Salient Object Discovery for Robots”, by D. M. Chan and L. D. Riek, which appears in the Proceedings of Robotics: Science and Systems Pioneers [26]. The dissertation author was the primary investigator and author of this paper.

Chapter 6 contains material from “RaccooNet: Real-time Object Proposal Generation for Robot Vision”, by D. M. Chan and L. D. Riek, which is currently in review for publication. The dissertation author was the primary investigator and author of this paper.

VITA

2013	Bachelor of Science in Electrical Engineering, California Polytechnic State University, San Luis Obispo, California
2015	Master of Science in Electrical Engineering, California Polytechnic State University, San Luis Obispo, California
2021	Doctor of Philosophy in Computer Science, University of California San Diego, La Jolla, California

ABSTRACT OF THE DISSERTATION

Unseen Object Perception for Robots

by

Darren M. Chan

Doctor of Philosophy in Computer Science

University of California San Diego, 2021

Professor Laurel D. Riek, Chair,

Robots have tremendous potential to help us in our daily lives. However, one key obstacle to facilitating their autonomy is that they lack the ability to perceive novel, or unseen objects. The de-facto solution to this problem is to pre-program robots using a large corpus of prior-known objects in hope that they will understand every object that they encounter. However, if robots need to understand new objects, they must be manually re-programmed to do so, which has proven to be time consuming and expensive, and is fundamentally intractable. Alternatively, a more direct approach to this problem is to leverage a robot's context, e.g., its immediate surroundings, which can be a rich source of information from which to learn about unseen objects in a scalable manner.

The goal of my research is to design algorithms and systems that enable robots to automatically discover unseen objects from their surroundings in a manner that is fast and robust to real-world vision challenges. In this dissertation, I discuss four key contributions of my work. First, I designed Salient Depth Partitioning (SDP), a novel depth-based region cropping algorithm which substantially improves the computation time of existing object detectors by up to 30%, with no discernible change in accuracy. SDP achieves real-time performance and is designed to give robots a better sense of visual attention, guiding them to visual regions that are likely to contain semantically important elements, which are also known as salient. Consequently, SDP can be used as a preprocessing algorithm to improve the computational efficiency of depth-based object detectors on mobile robots.

Second, I demonstrated that object proposal algorithms, a ubiquitous algorithmic component in machine vision systems, do not translate well to real-world contexts, which can negatively impact the performance of robots. I conducted a study to explore how algorithms are influenced by real-world robot vision challenges such as noise, blur, contrast, and brightness. I also investigated their performance on hardware with limited memory, CPU, and GPU resources to mimic constraints faced by mobile robots. To my knowledge, I am the first to investigate object proposal algorithms for robotics applications. My results suggest that object proposal algorithms are not generalizable to real-world challenges, in direct contrast to what is claimed in the computer vision literature. This work contributes to the field by demonstrating the need for better evaluation protocols and datasets, which will lead to more robust unseen object discovery for robots.

Third, I developed Unsupervised Foraging of Objects (UFO), a novel, unsupervised method that can automatically discover unseen salient objects. UFO is substantially faster than existing methods, robust to real-world noise (e.g., noise and blur), and achieves state-of-the-art performance. Unlike existing approaches, UFO leverages object propos-

als and a parallel discover-prediction paradigm. This allows UFO to quickly discover arbitrary, salient objects on a frame-by-frame basis, which can help robots to engage in scalable object learning. I compared UFO to two of the fastest and most accurate methods (at the time of writing) for unsupervised salient object discovery (Fast Segmentation and Saliency-Aware Geodesic), and found it to be 6.5 times faster, achieving state-of-the-art precision, recall, and accuracy. Furthermore, I show that UFO is robust to real-world perception challenges encountered by robots, including moving cameras and moving objects, motion blur, and occlusion. This work lays the foundation for faster online object discovery for robots which contributes toward future methods that will enable robots to learn about new objects via observation.

Fourth, I designed RaccooNet, a new real-time object proposal algorithm for robot perception. To my knowledge, RaccooNet is the current fastest object proposal algorithm at a runtime of 47.9 fps while also achieving comparable recall performance to the state-of-the-art (e.g., RPN, Guided Anchoring). Additionally, I introduced a novel intersection over union overlap confidence prediction module, which allows RaccooNet to recall more objects using a lesser number of object proposals, thus improving its efficiency. I also designed a faster variant, RaccooNet Mobile, which is over ten times faster than the state-of-the-art (171 fps). Conducting experiments on an embedded device, I demonstrated that my algorithm is suitable for computationally resource-constrained mobile robots. I validated RaccooNet and RaccooNet Mobile on three real-world robot vision datasets (e.g., RGBD-scenes, ARID, and ETH Bahnhof) and showed that they are robust to vision challenges, for example, blur, motion, lighting, object scale. This work contributes to the field by introducing a real-time object proposal algorithm, which will serve as a foundation to new real-time object discovery methods for mobile robots.

Summarizing my doctoral research, my work contributes to building real-time object perception systems that can be deployed on real-world robotic systems that operate

in the wild. This work will ultimately lead to more scalable object perception frameworks that can learn directly from the environment, on-the-fly. Moreover, my research will allow roboticists to build smarter robots that will one day become more seamlessly integrated into our daily lives, and become the useful machines that we envisioned for our future.

Chapter 1

Introduction

Robots have the potential to transform society in ways that can improve the daily lives of people. For example, robots can provide social assistance to support people with cognitive impairments and their caregivers [143, 78, 109], and provide personalized training and treatments [110, 228]. In clinical settings, robots can assist and train healthcare workers [211, 144, 168, 213, 142]. More generally, they can assist people with day-to-day tasks such as cooking and cleaning [12], worker support [220, 230], or even provide companionship [17, 76, 159].

However, before robots are ready to transition to these domains, they will need a robust understanding of their surroundings so that they can make coherent decisions while maintaining a safe environment for people [163, 181, 28, 94, 151, 149].

It is particularly challenging to build robots that can operate robustly in human spaces, because they are prone to change frequently and little can be known about them in advance [180, 234, 212] (Figure 1.1). For instance, people are prone to constantly interact with their surroundings, which can cause the context, objects, and environment to change quickly over time [67, 180, 149, 204, 197]. Moreover, variations in appearance (and function) between objects, places, and people can be difficult to understand, and

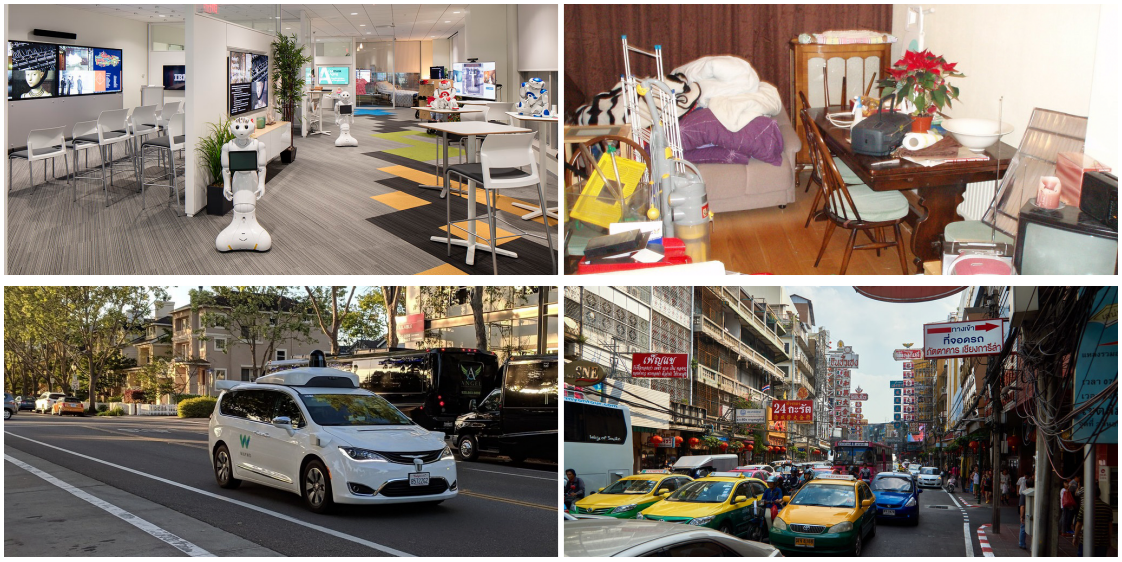

Figure 1.1: Recently, mobile robots and autonomous vehicles are successfully transitioning to human spaces, where it is easy to perceive objects and people around them (left column). However, in more cluttered and chaotic environments, enabling robots to perceive objects and people around them is not straight-forward (right column). Image references shown in clockwise order starting from upper left: [183, 171, 93, 150]

even more difficult to computationally model. Consequently, robots need to be adaptable to these factors.

One key challenge that prevents robots from adequately understanding their surroundings is that they cannot perceive novel objects. This innate ability is one that people often take for granted, since we can seamlessly discern when objects are unfamiliar to us, and moreover reason about them. When encountering novel objects, we can make sense of their shape and form, and also hypothesize their functions using our knowledge about prior known objects. For example, a person might be unfamiliar with what an eighteenth-century carbon microphone looks like, and might not be able to immediately identify one when presented (shown in Figure 1.2, left). However, they might be able to associate it with modern-day microphones (shown in Figure 1.2, right) by observing the parts and attributes that they have in common (e.g., diaphragm, ability to input sounds) [182, 198].

Figure 1.2: An antique carbon microphone [177] (left) and modern dynamic microphone [156] (right). Despite looking very different, we can infer that they have common attributes such as a diaphragm and ability to input sounds. However, most computer vision systems cannot easily perceive novel objects in this way.

If perhaps people stumble upon novel objects that have no resemblance to prior-known objects, their curiosity can inspire them to interact and learn about them [74]. This capability allows us to discern how objects can be useful to us and ultimately enables us to expand our knowledge about the physical world [184]. For example, if a person were to have never seen a microphone, they might interact with one to learn its utility as a tool (i.e., one that captures sound), which can then later be applied to solve more complex problems (e.g., its utility as a sensor for a robot to acquire speech). While this ability to learn about novel objects might be easy for people, robots are unable to do the same.

In addition to increasing their capacity to learn from their surroundings, enabling robots to perceive novel objects is critical to their performance and autonomy in real-world environments. For instance, robots will be able to perform more robustly and independently if they are able to reason about unfamiliar objects that might be helpful to their tasks (e.g., appraising items, identifying entities that can be manipulated). Consequently, until robots are equipped with the intelligence to perceive novel objects and learn about them in a scalable manner, their comprehension of the world and their ability

to adapt to it remains constrained.

1.1 Motivation and scope

Robots are typically equipped with one or more cameras that allow them to perceive their surroundings as a sequence of images. Computer vision algorithms then process these images to enable robots to gather information about their environment. In particular, object detection algorithms (i.e., object detectors) are often used to support their ability to visually perceive nearby objects, which can inform them about possible actions to take (e.g., which objects to inspect, which objects to interact with). However, object detectors are predominantly designed with the assumption that all objects must fit within a set of predetermined categories, or *classes*, which organize objects by common properties such as appearance or function. When objects are visually dissimilar from those seen in training, or do not fall into any one of these predefined classes, i.e., *unseen objects*, they are either misclassified or completely ignored. Consequently, existing object detectors constrain robots to perceive a limited set of items, which hinders their intelligence.

Some researchers address the problem of detecting unseen objects, i.e., *unseen object discovery*, where the goal is to recover the boundaries of some generic object in image sequences. The most common approaches require some degree of semi-supervision, for example, manually annotating an object in at least one image from the sequence. This annotation then provides a training example (e.g., one-shot object learning), so that the object can be discovered at later points of the image sequence or across multiple viewpoints [31, 225]. However, these methods are intractable for robots because they require a human to manually initialize them (i.e., provide the annotation) each time that a robot encounters an unseen object.

To date, there is little work that addresses unseen object discovery in an automatic manner that is suitable for robots. The most recent and closely related methods function by uniformly sampling key features (e.g., optical flow boundaries) across bounded image sequences [227, 158]. Subsequently, many methods use constrained optimization to compute object boundaries that correlate to unseen objects. While these methods are accurate, they often require a large number of image frames that need to be post-processed (i.e., computed offline) before object discovery can occur, making them prohibitively slow for real-time robot perception [158]. Consequently, these approaches can disrupt reactive decision-making behaviors of robots, which are essential for time-sensitive tasks (c.f., [94]).

Designing algorithms to discover unseen objects is challenging. This problem is oftentimes ambiguous, having multidisciplinary underpinnings in fields such as computer science, cognitive science, psychology, and philosophy. Thus, there are a number of diverse and sometimes divergent theories about how this problem should be best approached. While there are many facets to this problem, **this dissertation explores how to design unseen object discovery algorithms for robots using computational methods**. Specifically, this dissertation explores several themes:

- How to design algorithms that can discover generic objects by exploiting low-level sensory information.
- How to enable robots to selectively tune their vision to objects that are potentially meaningful or useful.
- How to design algorithms that are fast for real-time robot perception.
- How to design perception algorithms for ubiquitous vision sensors and systems.
- How to design robot vision algorithms that are robust to real-world vision chal-

lenges such as noise, blur, or camera motion.

- How to validate the real-world performance of robot and computer vision algorithms.

1.2 Contributions

The contributions of this work are as follows:

- **Developed a new computationally efficient algorithm, *Salient Depth Partitioning* (SDP), which extracts regions of interest from color and depth (RGB-D) images [28].** SDP serves as a basis for "where a robot should look" and is a preliminary step to object detection, to reduce computation time and minimally affect detection accuracy. This algorithm requires no *a priori* knowledge of the environment, or reliance on geometric constraints, and can work with existing object detectors given any calibrated RGB-D image. SDP was tested with four state-of-the-art detectors (HOG and SVM [39], Aggregate Channel Features [50], Checkerboards [243], and RCNN [73]), improving computation time by up to 30%, with no discernible change in accuracy. When deployed on computationally-constrained robots, SDP can serve as a preprocessing algorithm to improve the efficiency of RGB-D object detectors.

- **Conducted an investigation on object proposal algorithms to demonstrate that the standard evaluation dataset is flawed for generalizing algorithm performance, particularly within the context of robotics [25].** This study was the first to conduct an evaluation of object proposal algorithms on robot vision datasets, to gain insight into how they perform in more realistic settings with real-world vision challenges. It also includes a controlled study to see how object proposal

algorithms perform when exposed to image perturbations (i.e., brightness, gamma correction, Gaussian blur, and Gaussian noise), reflecting robot vision challenges. This is also the first study that considers algorithm execution time on portable hardware, showing that most algorithms are suitable for real-time robotics applications. This research ultimately shows that many prevailing object proposal algorithms are not as generalizable as the computer vision literature purports, which can profoundly impact how they perform in real-world robotic systems. The results of this work will be useful to the robotics community, which can be used to gauge performance and computation time trade-offs.

- **Developed a novel salient object discovery method *Unsupervised Foraging of Objects* (UFO), which is designed for monocular robot vision [26, 27].** UFO is designed with a parallel discovery-prediction paradigm, permitting it to discover arbitrary, salient objects on a frame-by-frame (i.e., online) basis, which can help robots to engage in scalable object learning. UFO is 6.5 times faster than the two current fastest and most accurate methods (at the time for writing) for unsupervised salient object discovery (Fast Segmentation [158] and Saliency-Aware Geodesic [227]), achieving state-of-the-art precision, recall, and accuracy. Furthermore, UFO is robust to real-world robot perception challenges, including moving cameras and moving objects, detractor objects, motion blur, and occlusion. This work ultimately lays the groundwork for enabling robots to learn about new objects on-the-fly.

- **Developed a new real-time deep learning-based object proposal algorithm, *RaccooNet*, which is designed for mobile robot perception applications.** RaccooNet is the only method capable of true real-time performance (at time of writing), which is approximately three times faster than the currently top-performing object proposal algorithms at 47.9 frames per second (fps). Additionally, this

research led to RaccooNet Mobile, which at 171 fps, is approximately ten times faster than top-performing methods. RaccooNet achieves state-of-the-art recall performance for a variety of real-world robot perception challenges, including camera motion, blur, and noise, across a range of dynamic scenes. This research also introduced a novel intersection over union (IoU) overlap confidence prediction module, which improves object proposal algorithm efficiency. The ideas introduced in this work is foundational toward future research in deep learning-based real-time object discovery algorithms.

1.3 Publications

Some of the work presented in this dissertation is based on the following publications; some of the work in this dissertation is based on research that has yet to be published.

1. **Chan, D. M.** and Riek, L. D. (2021). RaccooNet: Real-time Object Proposal Generation for Robot Vision. In review.

2. **Chan, D. M.** and Riek, L. D. (2020). Unseen Salient Object Discovery for Monocular Robot Vision. IEEE Robotics and Automation Letters (RA-L). Also appears in IEEE International Conference on Robotics and Automation (ICRA).

3. **Chan, D. M.** and Riek, L. D. (2019). Object Proposal Algorithms in the Wild: Are they Generalizable to Robot Perception? In Proceedings of the IEEE/RAS International Conference on Intelligent Robots and Systems (IROS).

4. **Chan, D. M.** and Riek, L. D. (2019). Unsupervised Salient Object Discovery for Robots. In Proceedings of Robotics: Science and Systems (RSS) Pioneers.

5. Taylor, A., **Chan, D.M.**, and Riek, L. D. (2019). Robot-Centric Perception of Human Groups. ACM Transactions on Human-Robot Interaction (THRI).

6. **Chan, D. M.**, Taylor, A. and Riek, L. D. (2017). Faster Robot Perception Using Salient Depth Partitioning. In Proceedings of the IEEE/RAS International Conference on Intelligent Robots and Systems (IROS).

7. **Chan, D. M.** and Riek, L. D. (2017). A Study on Object Proposal Algorithms for Robots. Kyoto Prize Symposium at Qualcomm San Diego.

8. **Chan, D. M.** and Riek, L. D. (2016). A Biologically-Inspired Salient Region Filter for Faster High-Fidelity Robot Perception. Midwest Robotics Workshop (MWRW) at the Toyota Technical Institute of Chicago (TTIC).

1.4 Ethical procedures

The acquisition of the dataset involving images of human pedestrians described in Chapter 3 of this dissertation was formally reviewed by the Institutional Review Board at the University of Notre Dame. The board found this process straightforward to review as it did not cause any harm to participants or result in non-identifiable data being disclosed. The dataset was stored securely and has not been published or publicly distributed.

All other research reported in this dissertation used publically available datasets.

1.5 Dissertation overview

This dissertation is organized as follows:

- **Chapter 2** provides a brief overview of related work in the areas of object detection and object discovery, including a discussion of metrics and datasets.

- **Chapter 3** describes SDP, a new method for faster object detection with salient depth partitioning.
- **Chapter 4** discusses an investigation on the generalizability of object proposal algorithms for robots.
- **Chapter 5** discusses a new unsupervised method for object discovery for robots, UFO.
- **Chapter 6** discusses RaccooNet, a new real-time object proposal algorithm designed for robot perception.
- **Chapter 7** summarizes the main contributions of this dissertation, discusses plans for future work and open research questions in the field, and delivers concluding remarks.

Chapter 2

Background

Object perception consists of algorithmic processes that transform visual inputs (e.g., images, video) to meaningful information about objects. By virtue of its practicality, a broad range of applications leverage object perception such as video surveillance [38, 194], autonomous vehicles [32, 235], human-computer interaction [160], and robotics [239, 214, 164].

Often considered a superset of complementary research problems, object perception can have many theoretical and algorithmic components. This chapter presents a brief overview of object perception, discussing foundational work from the computer vision and robotics literature. This includes a discussion about how objects are defined within the context of computational models, and a discussion about prominent algorithmic components and theoretical insights that are foundational to my research. Moreover, this chapter discusses common algorithm evaluation protocols, datasets, and metrics that I employ in my research.

2.1 Definition of an object

According to Merriam-Webster, a physical *object* generally means something material that can be perceived by the senses [45]. However this definition can be onerous for computational object perception contexts.

In computer and robot vision research, one contention with the definition of *objects* concerns *scale*, or how cameras capture the appearance of objects with respect to their distances [170, 64]. Imaging systems project objects onto two dimensional planes (i.e., images), where closer objects appear larger, while those that are further away appear smaller. These differences in camera perspectives introduce ambiguity to how objects appear, which are difficult for computational models to resolve [118]. For example, it is easy for many modern-day computer vision systems to determine the image of a house from a further distance (e.g., a full view of the house), since they can also detect several relevant cues (e.g., windows, doors, roofing). In contrast, computer vision systems cannot make the same determination given an image of a close-up view of the same house (e.g., view of the front porch); rather, they will infer parts of the house such as windows and doors.

Another problem with the standard definition of *object* relates to *context*, or how systems should perceive objects with respect to their surroundings [6, 151]. One important consideration regarding context for intelligent robots, is that it is not only important that they detect objects, but that they detect those that are meaningful [218, 95]. Since object perception algorithms can utilize a great deal of computational resources, it can be impractical to design them to perceive objects that have little relevance to robots [222]. For example, trees within the context of a forest can be disregarded as objects since there are many that naturally blend in with one another; rather, trees can be considered *background* entities. In contrast, a lone tree in a barren desert can be much more easily

distinguished as a *foreground* entity. For similar reasons, solid entities such as walls and flooring within the context of indoor environments are often classified as background elements rather than as objects.

Another complication concerning the definition of *objects* is that it conflicts with how the performance of object perception algorithms are measured. To validate new algorithms and to compare their performance to that of existing ones, they must be evaluated using repeatable, quantitative metrics. However, this necessitates that objects be precisely defined and disambiguated in a way that algorithms can be fairly and consistently evaluated.

Despite the need for clarity about how objects should be defined, there is no universal agreement about what an object actually is in the robotics and computer vision literature; instead, its definition is often nuanced depending on the research problem. In pick-and-place tasks, objects are entities having known affordances, or in this case, those that can be possibly manipulated [105, 240]. In the object detection literature, objects are regarded as entities consisting of a set of arbitrarily-chosen classes within a large database of images [60, 108, 122]. Within the context of mobile robots, objects are loosely defined as non-static entities that are more likely to change in their environment over shorter periods of time [223, 113, 164].

In my own research and throughout this dissertation, I adopt the general definition of *object* from Alexe et al. [2], which synthesizes decades of computer vision and robotics research, and is characterized by three axioms:

Axiom 1 *An object has closed boundaries [248, 114]. This means that an unobstructed view of the object is fully visible within an image or video frame. Furthermore, there are finite edges that define the boundaries of the object.*

Axiom 2 *An object must differ in appearance from its surroundings [124, 131]. That is, an object is clearly disambiguated from its background.*

Axiom 3 *An object sometimes has a unique appearance that is salient [19, 65]. In other words, the object can possess visual characteristics that have an effect on the attention of the observer.*

2.2 Saliency estimation

In computer vision, *saliency estimation* is a process that seeks to extract pixels from images that are *salient*, or "stand out" as a means to mimic human visual attention [95]. In robotics, saliency estimation is often used to filter images so that computational resources can be efficiently allocated to regions that are more visually interesting (e.g., foreground objects). For example, these regions can correspond to vibrant, high-contrast, or conspicuous pixels in an image. Some prominent applications of saliency estimation in robotics include semantic segmentation [10, 138] and waypoint detection for navigation [40].

2.3 Object detection

Object detection is a computational process that determines the presence and locations of specific kinds (i.e., those belonging to a class) of objects in images or video.

Fundamentally, an object detection algorithm, i.e., *object detector*, performs two primary subtasks: localization and classification. *Localization* addresses how to generate object regions in the form of segmentation masks (i.e., individual pixels that form the volume or area of objects) or bounding boxes (i.e., rectangular coordinates that encapsulate objects). *Classification* is responsible for assigning each object region to a set of predetermined *classes*, or categories of objects that share one or more similarities in appearance or function.

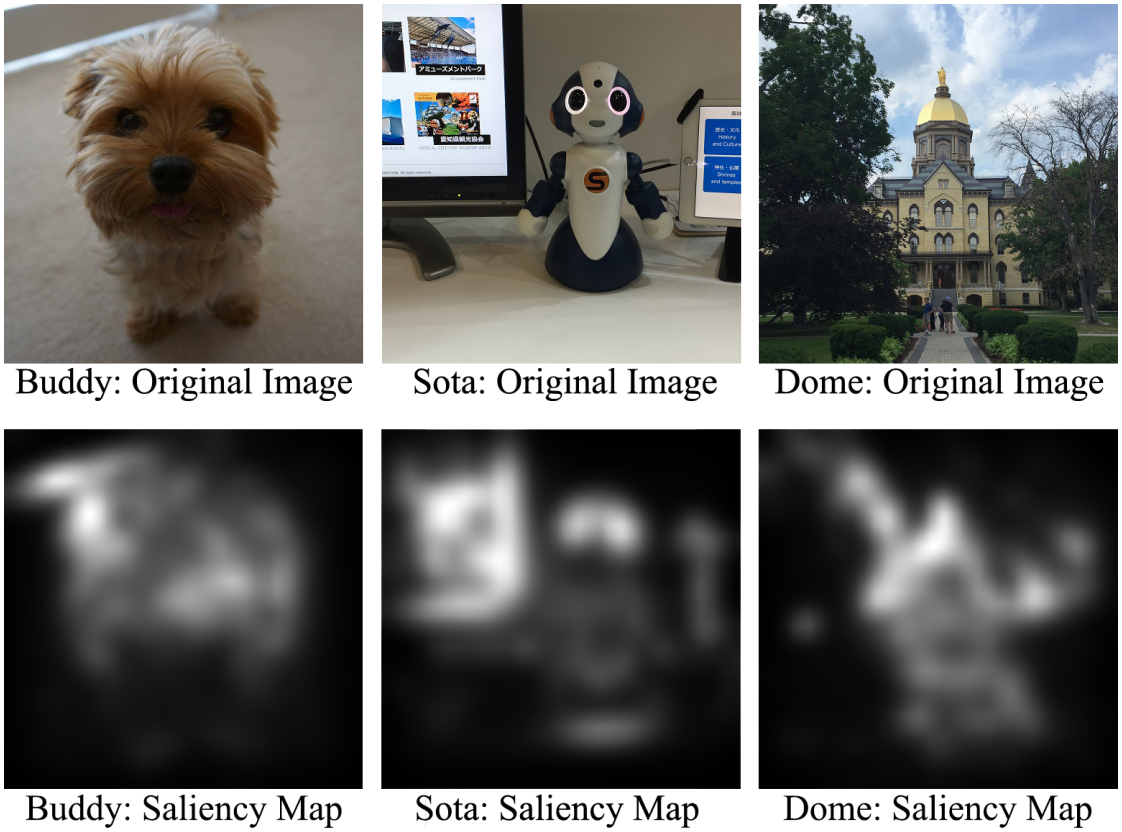

Figure 2.1: Saliency predicts pixels of an image that attract visual attention. Original images are shown on top and their saliency maps are shown on the bottom.

While a number of object detection systems were introduced in the past several decades [249, 123], they typically follow one of two general approaches: two- or one-stage. The earliest object detectors were developed as two-stage detectors, which achieve localization and classification separately. This involves using distinct algorithms to perform each of these processes one after the other [72, 71, 178].

One-stage object detectors are a more recent innovation developed using modern data-driven techniques. In contrast to two-stage detectors, one-stage detectors leverage convolutional neural networks that are designed to simultaneously perform regression and classification to predict object locations and classes [175, 125].

One-stage networks are generally faster than two-stage networks, since they are built using a single, unified network to allow them to more efficiently perform detection.

Consequently, they are more often used in smaller-scale applications such as mobile robotics [34, 56, 210, 231]. In contrast, modern two stage networks can attain higher accuracy since their localization and classification processes can be individually tuned. As such, two stage networks are most often used on systems that have less computational resource restrictions [242, 77].

2.4 Object discovery

Object discovery is a computer vision problem that investigates how to design systems that can locate arbitrary objects in images. Unlike the similar problem of object detection, object discovery methods are not explicitly trained to classify objects that are known beforehand [141]. Rather, they are designed to locate objects that are *unseen*, or objects that they have not been trained to explicitly recognize. Due to the open-ended nature of the problem, object discovery is challenging, and is largely unsolved. Consequently, ideas about how to best approach the problem can be both diverse and disparate in the literature.

Weber et al. [229] are often credited as the first to build a model toward object discovery, modeling objects as "constellations" of rigid object parts. Akin to graph networks, constellations are used to match keypoints across unlabeled image sets, to construct a probabilistic mixture model based on k-means clustering. This work later became the basis for weakly supervised object discovery [31, 35, 92, 225, 196, 195].

Other researchers approach object discovery using one-shot learning [20, 96, 101, 161, 173, 75, 226]. This requires a human to manually annotate objects from at least one frame of an image sequence. Subsequently, an algorithm uses the annotation to automatically locate the same generic objects from images in the rest of the sequence. However, because one-shot approaches require manual initialization to discover each

new object, they can cause robots to become over-reliant on humans, making them both impractical and intractable for mobile robot perception.

To that end, some researchers are working toward approaches that can automatically discover objects in video. Some researchers apply motion boundaries to separate foreground objects from the background [132, 16].

Figure 2.2: Example outputs of a typical object discovery method shown for eight image sequences. Constrained optimization is used to determine the most consistent segmentation boundaries which have high correlation to objects (shown in red) [9]. However, this approach makes assumptions that the image sequence is bounded, which is incompatible with robot perception applications.

Extracting distinguishable markers (e.g., edges, key points) across equally spaced frame intervals, researchers often apply constrained optimization to segment unseen object candidates (shown in Figure 2.2) [227, 158]. However, one disadvantage to these approaches is that they operate by processing bounded video sequences offline before

object discovery can occur. Thus, they are unable to discover objects in an online manner, making them unsuitable for real-time mobile robot perception.

2.5 Object proposals

Object proposal algorithms serve to extract image regions that are most likely to contain objects while pruning less meaningful background regions to reduce computational complexity. They often involve predicting object locations using one of two representations: segmentation masks or bounding boxes (see Figure 2.3).

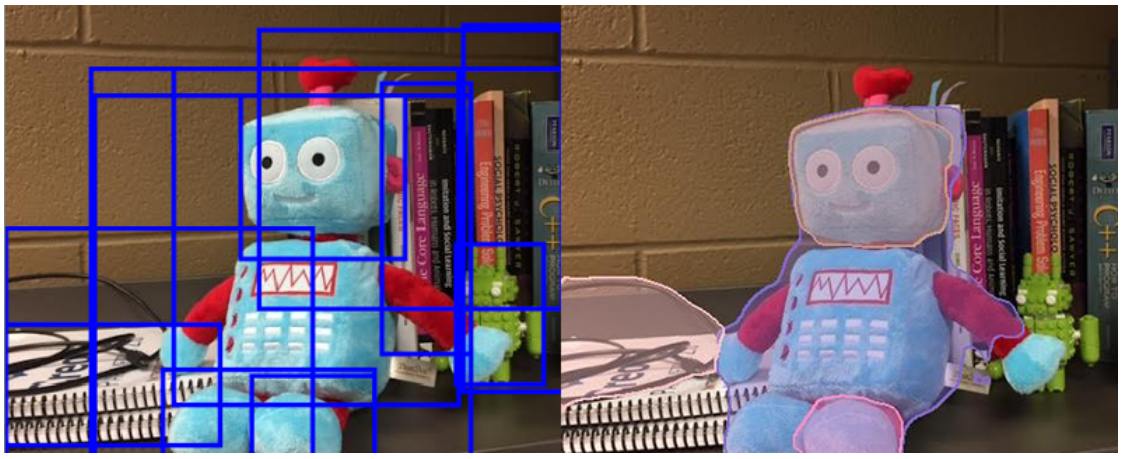

Figure 2.3: Bounding box (left) and segmentation mask (right).

Segmentation masks consist of pixel-wise labels that articulate the shape of objects and can delineate spatial boundaries with fine-grained accuracy. Segmentation-based approaches are designed to produce object candidates at the pixel level, typically grouping adjacent pixels by some measure of similarity. Earlier methods were designed using image processing techniques, for example, clustering [37], region growing [81], and connected component analysis [8]. More modern approaches use convolutional neural networks to perform classification on individual pixels to derive segmentation masks [165, 166]. In general, segmentation can be computationally expensive and slow,

which has prevented its widespread adoption in real-world applications [85]. In contrast, bounding boxes provide a coarser approximation, but they are significantly faster to compute and are thus ubiquitous in many robot vision systems.

Bounding boxes form rectangular perimeters around objects and are typically represented using a Cartesian-based coordinate system (e.g., x, y, width, and height). Each bounding box is evaluated with a scoring function, which predicts the likelihood that the region contains an object, i.e., *objectness confidence*. Bounding boxes containing low prediction scores can then be quickly rejected with low computational expense. Due to their versatility, object proposals that output bounding boxes are often used as part of a larger computational pipeline to address a large and diverse number of robot vision tasks such as object discovery [152, 153, 27], object detection [206, 193], and SLAM [164, 238].

Object proposal algorithms often depend on an input parameter that controls the number of object proposals that they generate, which can be anywhere from a few, to hundreds of object proposals (shown in Figure 2.4). A larger number of proposals increases the likelihood that all objects are recovered from the image, leading to higher recall. However, a larger number of proposals also leads to higher computation cost. Thus, it is desirable that object proposal algorithms achieve high recall with the least number of proposals possible.

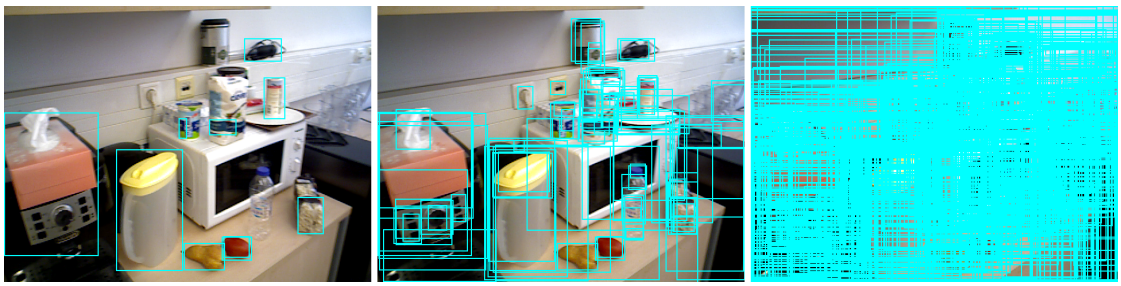

Figure 2.4: Object proposal algorithms can generate anywhere from a few, to hundreds of object proposals: 10 (left), 100 (center), 1000 (right).

2.6 Feature extraction

The goal of feature extraction is to derive patterns or other meaningful information from images so that they can be used to solve machine learning problems (e.g., handwriting recognition [167], group detection [210]). Within the context of object perception, feature extraction is responsible for generating unique representations of objects, i.e., *features*, so that objects can be understood by machines.

Designing feature extractors for object perception tasks in robotics applications is challenging. Because robots need to perceive the many objects in their surroundings, feature extractors must have high representational power so that a broad range of objects can be more easily distinguishable. Moreover, features need to be robust so that they can account for changes in object appearances and viewpoints, which are dependent on the relative position between objects and a robot's vision systems (e.g., cameras). However, increasing feature complexity can also require a greater amount of computational resources, which can negatively impact how robots perform. For example, higher feature complexity can compete with the computational resources that robots need to perform other tasks (e.g., decision making, manipulation, navigation).

In this section, I provide a discussion about common feature design techniques that fall under one of two categories: hand-crafted and learned.

2.6.1 Hand-crafted features

The earliest feature extractors involve convolution filters, or *kernels*, which are designed using strict mathematical formulation and image processing techniques. When kernels are convolved with an image, they extract features such as lines, edges, corners, shapes, and other patterns or cues that can inform the presence of objects [23, 49, 82, 5]. Because designing feature extractors in this way requires a great deal of intuition to judge

their usefulness for various object perception tasks, this process is colloquially known in the literature as “hand-crafting”.

The following subsections provide a brief overview of foundational work in hand-crafted feature design, where many of the algorithmic processes are still used in modern object perception research.

Keypoints

Keypoints are features extracted from small image regions, i.e., *image patches*, that are robust to illumination, scale, and rotation (e.g., corner points). In general, keypoint feature extractors apply one or more convolutional kernels to determine pixels that are unique to their neighbors[1] [7, 21, 188, 174].

Within the context of object perception, keypoints are particularly useful for performing data association across spatially-connected sequences of images to infer objects or object parts [128, 129]. Consequently, keypoint extraction is essential to many computer and robot vision applications (e.g., random sampling consensus (RANSAC) [62] and simultaneous localization and mapping (SLAM) [145]).

Histogram of oriented gradients

To address technical challenges related to how object features can be made more robust to shift and rotation, Dalil and Triggs introduced a histogram of oriented gradients (HOG) [39]. HOG consists of generating a feature template derived from an ensemble of object features captured from multiple viewpoints. When used in conjunction with a classifier (i.e., support vector machine), a HOG feature template is moved at each position of an image to make a decision about whether or not an object exists; this process is

[1] Keypoint extraction differs from saliency estimation because keypoints are unique to their neighbors, which may not necessarily be unique from the rest of the image. In contrast, saliency correlates to pixels and regions that are unique with respect to the entire image.

popularly known as the *sliding window method*.

To account for non-rigid objects (e.g., pedestrians and animals), Felzenzwalb et al. [61] introduced the Deformable Part Model (DPM), which decomposes HOG templates into smaller components, or deformations. Deformations are individually detected using part-specific templates via the sliding window method, where the interaction between the parts are modeled using "spring-like" constraints.

Image pyramids

Designing feature extractors that are robust to scale is a challenging problem, and continues to be a major research hurdle in computer and robot vision. With respect to hand-crafted features, Felzenszwalb et al. [61] prominently showed that multi-scale object features can be derived from an *image pyramid* to characterize multiple sizes in one convenient structure. This is achieved using convolution and Gaussian kernels to repeatedly upsample and downsample images at various resolutions to form a "stacked" pyramid-like structure.

2.6.2 Learned features

The past decade introduced new data-driven methods as a way to enable computational models to extrapolate . In particular, *convolutional neural networks* (CNNs) leverage machine learning to directly derive object features that are robust to geometric transformations such as scale, shift, and rotation. Consequently, CNNs revolutionized the field of object perception, currently replacing or complementing traditional hand-crafted feature extraction techniques.

Convolutional neural networks

CNNs are biologically-inspired algorithms that loosely mimic the human visual cortex, designed to extract visual features from images. They are constructed from atomic building blocks of artificial neurons, or *perceptrons*. Intuitively, perceptrons behave as binary classification algorithms that map inputs to desired outputs using *activation functions*.

To increase their functionality, perceptrons are connected together to form more sophisticated topologies. Akin to graph theory, connections between perceptrons are assigned weights; intuitively, weights control the magnitude of perceptron outputs, which ultimately contribute to how CNNs function. When arranged into a formation of many interconnected perceptrons, or *fully-connected layers* (shown in Figure 2.5), they can output object features with a high degree of representational power [199].

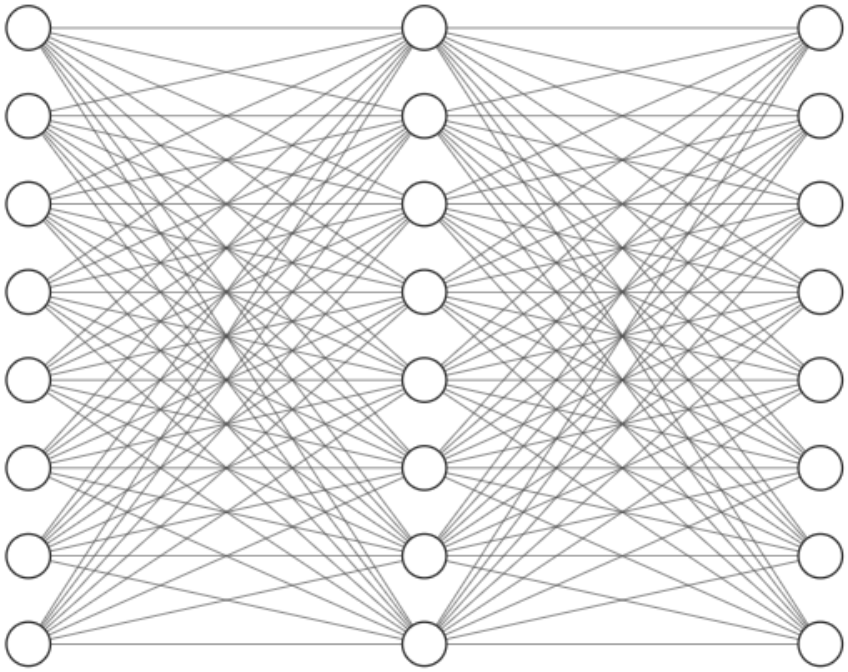

Figure 2.5: A simplified diagram showing three fully-connected layers. Each node represents a single perceptron.

An important component of CNNs are *convolutional layers*. Intuitively, convolutional layers are composed of an arrangement of perceptrons that form convolution kernels. Similar to hand-crafted feature extractors, convolutional layers extract features via convolution, which depend on their relative size and position in images (i.e., *receptive field*). However, one important distinction from hand-crafted kernels is that CNNs learn to generate their own kernel parameters, rather than needing them to be manually designed.

Within the context of object perception, CNNs consist of a chained sequence of interconnected convolutional layers, each with varying dimensions (i.e., width, height, and depth) that are designed to optimize feature representational power and memory footprint. In general, as this sequence of layers gets longer, the CNN's representational power increases, and is characterized as being *deeper* [199].

Many CNNs are constructed with one or more fully-connected layers, which are connected to the end of the convolutional layer chain (shown in Figure 2.6) [108]. This approach allows CNNs to condense the high-dimensional features from their convolutional layers, and convert them to compact (i.e, one-dimensional) feature vectors; these feature vectors are sometimes colloquially called *flattened features*.

More recently, some researchers replace fully-connected layers with convolutional filters to generate depth-wise features; these types of networks are called *fully convolutional networks* (FCN) [127].

Using the output feature vectors, CNNs make object predictions, i.e., *inferences*, using one of two fundamental operations: regression and classification. More generally, *regression* is a machining learning technique where CNNs use feature vectors to predict continuous variables; when CNNs predict individual object coordinates as continuous variables, this process is called *bounding box regression*.

Classification is a machine learning technique where CNNs use feature vectors to predict discrete variables, which correspond to a set of classes. Within the context of

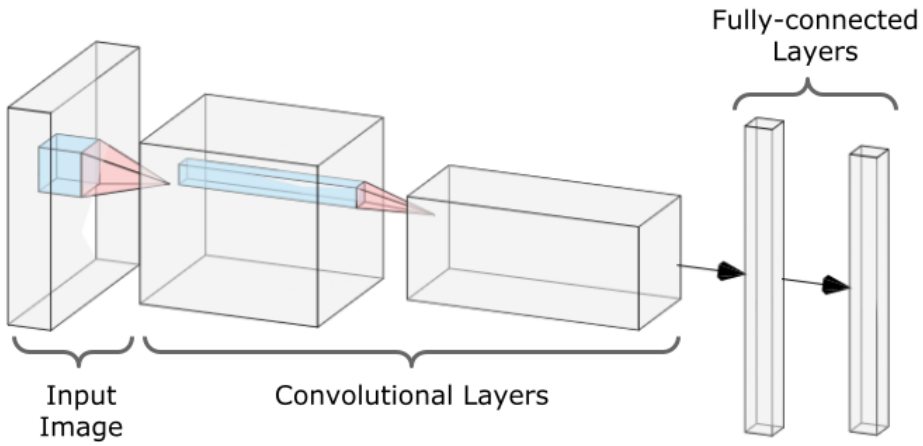

Figure 2.6: A simplified "AlexNet-style" [108] diagram showing a typical CNN architecture with main network components. An image is passed through two convolutional layers. The neurons (and their features) at the end of the convolutional layer chain are then flattened to allow them to connect to two fully-connected layers.

object perception, classification often determines how the feature vectors correlate to specific types of objects.

Feature pyramid networks

Recently, researchers are seeking ways to improve the representational power of CNNs by leveraging various parts of their structure. Intuitively, the shallower layers of CNNs possess greater spatial resolution (larger width and height channels) with shallower outputs (smaller depth channels). Features from these layers excel at characterizing smaller objects and lower-level structural information (e.g., edges, shapes) [55]. In contrast, the deeper layers have larger depth channels and smaller spatial resolutions. Features from the deeper layers of the network contain higher-level semantic content that have greater specificity and complexity (e.g., textures, specific object parts), making them more adapted for discriminating larger, more specific objects.

Feature pyramid networks (FPN) were introduced to augment existing CNNs so that they can extract and concatenate features from their various depths [120, 125]. In

this way, objects can be represented in multiple scales, making them more resilient in robot perception applications [33].

Training convolutional neural networks

As previously discussed in Section 2.6.2, CNNs are composed of perceptrons where their connections consist of weights. By modifying the values of these weights via *training*, the network can learn its intended behavior.

In general, training a CNN requires formulating a *loss function*, which computes how closely its predictions equal its desired output; CNN predictions that deviate further from their desired output correspond to a higher error, or *loss*. Training is achieved using *back propegation*, where the network's weights are iteratively adjusted via *gradient descent*, to learn a configuration that minimizes loss [190, 15]. This process also involves using an optimization algorithm (i.e., an *optimizer*), which applies a learning strategy for how the weights should be adjusted [189, 102]. Depending on many of its design factors (e.g., CNN architecture, training data, and optimizer), a CNN often requires manual fine-tuning; this involves modifying its hyperparameters, which are independent variables that directly influence how the network learns and performs.

Within the context of object perception, training a CNN involves curating a *dataset* that consists of a large collection of manually-labeled images. Each *label* characterizes various attributes of objects (e.g., locations, type), which provide instructions about what the CNN should learn. In general, training on many variations of image-to-label pairs, i.e., *training examples*, allows the CNN to learn more generalized behavior (c.f., over-fitting), and thus perform more robustly.

After training a CNN, its weights are frozen in such a way that they can no longer be updated. The CNN can then be deployed in applications, where they can infer new images.

2.7 Evaluation protocols

In this section, I discuss common metrics and datasets from the computer vision literature that are used to evaluate the performance of object perception methods.

2.7.1 Intersection over union

Intersection over union (IoU) quantitatively measures how well two bounding boxes overlap with each other (shown in Figure 2.7). Specifically, IoU is given by:

$$\text{IoU} = \frac{A \cap B}{A \cup B}, \quad 0 \leq \text{IoU} \leq 1 \tag{2.1}$$

where A and B are two arbitrary bounding boxes.

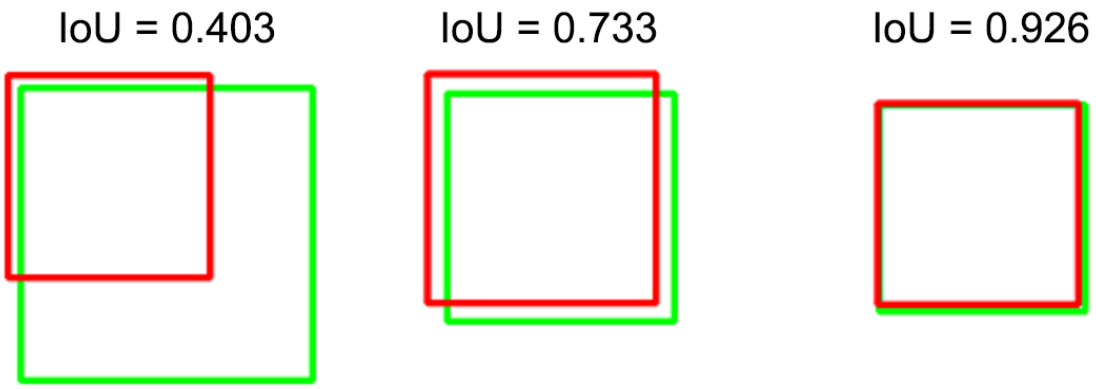

Figure 2.7: Visualization of *IoU* between two bounding boxes [186].

When used as an evaluation metric, IoU is indicative of an algorithm's ability to accurately localize objects, which measures the similarity between an algorithm's predicted bounding box and the dataset's ground truth. Predicted and ground truth bounding boxes are typically paired using best-fit linear assignment (e.g., Jonker-Volgenant algorithm [99]) to maximize the IoU overlap among all possible pairs.

Most commonly, an IoU threshold (IoU_t) is used as a decision boundary to decide how an algorithm's predictions fall into one of four outcome classifications: true positives, false positives, true negatives, or false negatives. For example, consider an algorithm that predicts bounding box A, and a ground truth bounding box B; when the IoU between bounding boxes A and B exceeds IoU_t, the outcome is considered a true positive. When the IoU between A and B is less than IoU_t, the outcome is designated a false positive In the special case of object detection, false positives are counted when the prediction assigns the incorrect object class regardless of whether or not the IoU of A and B exceeds IoU_t. A true negative indicates that there is no prediction A, and that there is no ground truth B; this criterion is generally not used, since it has little meaning in object perception contexts. A false negative indicates that A has no possible pairing with B; this condition can also happen when there is no prediction A, but there exists ground truth B.

In the computer vision and robotics literature, the de facto decision boundary for localization tasks is $IoU_t = 0.5$, because it accommodates the broadest range of objects, including those that are traditionally difficult to localize such as non-rigid objects (e.g., people, animals). However, it is also customary to compute metrics for $IoU_t \in \{0.5, 0.55 \ldots, 1\}$, which can give an indicator about an algorithm's ability to adapt to stricter localization specifications. In this way, algorithm performance can be presented in a way where researchers can consider their various performance tradeoffs (e.g., localization accuracy, recall, precision).

2.7.2 Measuring accuracy

Accuracy measures how successfully an algorithm can correctly predict object locations (and/or object classes), but is penalized for incorrect or missed predictions.

Accuracy (ACC) is computed as a function of IoU threshold IoU_t. At each IoU threshold, accuracy is computed by taking the sum of true positives and true negatives,

divided by the sum of true positives, true negatives, false positives, and false negatives. By definition, accuracy is given by Equation 2.2:

$$ACC(IoU_t) = \frac{TP(IoU_t) + TN(IoU_t)}{TP(IoU_t) + TN(IoU_t) + FP(IoU_t) + FN(IoU_t)} \tag{2.2}$$

where $TP(IoU_t)$ is the total number of true positives, $TN(IoU_t)$ is the total number of true negatives, $FP(IoU_t)$ is the total number of false positives, and $FN(IoU_t)$ is the total number of false negatives; in the literature, it is customary to let $TN(IoU_t) = 0$.

2.7.3 Measuring recall

Recall measures how successfully an algorithm can correctly predict object locations (and/or object classes), but is only penalized for missed predictions.

Recall is computed as a function of IoU threshold ($R(IoU_t)$). At each IoU threshold, recall is computed by dividing true positives by the sum of true positives and false negatives. By definition, recall is given by Equation 2.3:

$$R(IoU_t) = \frac{TP(IoU_t)}{TP(IoU_t) + FN(IoU_t)} \tag{2.3}$$

where $TP(IoU_t)$ is the total number of true positives and $FN(IoU_t)$ is the total number of false negatives.

To reduce the dimensionality of the recall vs. IoU metric, it is also common to compute average recall (AR), which computes the mean of $R(IoU_t)$ where $IoU_t \in \{0.5, 0.55, \ldots, 0.95\}$ [122]. Average recall is given by Equation 2.4:

$$AR = \frac{1}{10} \sum_{IoU_t \in \{0.5, 0.55, \ldots, 0.95\}} R(IoU_t) \tag{2.4}$$

2.7.4 Measuring precision

Precision (P) measures how successfully an algorithm can correctly predict object locations (and/or object classes), but is penalized for the number of false predictions.

Precision is computed as a function of IoU threshold. At each IoU threshold, precision is computed by dividing true positives by the sum of true positives and false positives, and is given by Equation 2.5:

$$P(IoU_t) = \frac{TP(IoU_t)}{TP(IoU_t) + FP(IoU_t)} \tag{2.5}$$

where $TP(IoU_t)$ is the total number of true positives and $FP(IoU_t)$ is the total number of false positives.

To reduce the dimensionality of $P(IoU_t)$, it is also common to compute average precision (AP), which computes the mean of $P(IoU_t)$ where $IoU_t = 0.5, 0.55, \ldots, 0.95$ [122]. Average recall is given by Equation 2.6:

$$AP = \frac{1}{10} \sum_{IoU_t \in \{0.5, 0.55, \ldots, 0.95\}} P(IoU_t) \tag{2.6}$$

In the object detection literature, algorithms often need to detect objects belonging to one of several object classes, where it is common to compute AP for each object class ($k \in K$). Extending Equation 2.6 to include additional classes is given by Equation 2.7:

$$AP(k) = \frac{1}{10} \sum_{IoU_t \in \{0.5, 0.55, \ldots, 0.95\}} P(IoU_t, k), \quad k = 1 \ldots K \tag{2.7}$$

It is also common to reduce the dimensionality of $AP(k)$ by computing the mean average precision (mAP), which is the average of AP over all possible classes. Mean average precision is given by Equation 2.8:

$$mAP = \frac{1}{K}\sum_{k=1}^{K} AP(k) \tag{2.8}$$

Measuring F-score

F-score (F_1) is commonly used in the computer vision literature to evaluate binary classification algorithms, which combines recall and precision metrics. F_1 is computed as a function of IoU threshold ($F_1(IoU_t)$) and is given by Equation 2.9:

$$F_1(IoU_t) = 2\frac{R(IoU_t)P(IoU_t)}{R(IoU_t) + P(IoU_t)} \tag{2.9}$$

where $R(IoUt)$ is recall and $P(IoU_t)$ is precision.

2.7.5 Datasets

When analyzing the performance of object perception methods, it is critical to objectively and consistently evaluate them. Ideally, datasets serve as impartial testbeds which allow algorithm performance to be quantitatively measured with repeatable experiments and metrics.

In computer and robot vision applications, a good dataset should contain images that represent a number of diverse and challenging conditions, i.e., are *generalizable*, so that methods can be evaluated for robustness and reliability should they be deployed on robots or other real-world applications. For example, variable illumination, noise, contrast, and blur are desirable attributes.

Datasets are commonly partitioned to create three subsets (i.e., *splits*): training, validation, and test. The training split consists of images-label pairs which are used to train an algorithm; the goal of training is to have the algorithm fit to this data. The validation split consists of image-label pairs which are used to measure the success and

progress of an algorithm as it is being trained; this data can also allow researchers to tune hyperparameters without introducing bias. Finally, the test split consists of image-label pairs which are used to measure how an algorithm is likely to perform when deployed on unseen data. Some object perception benchmarks combine validation and test splits (e.g., object proposal algorithms).

In this section, I list several popular datasets commonly used to design and evaluate object perception methods, which I also used to conduct my dissertation research.

Microsoft COCO

The Microsoft COCO dataset [122] represents candid objects in everyday scenes. Each image consists of a unique background that contains an arbitrary number of objects belonging to 91 common object classes. In total, the dataset contains approximately 118k training images and 5k validation images. Due to its large size and for its high-quality annotations, COCO serves as the standard training and evaluation platform for a wide range of computer vision problems.

PASCAL visual objects challenge and PASCAL+

The PASCAL Visual Objects Challenge (VOC) [60] is an evaluation suite that includes a large object dataset. It is also the standard benchmarking tool for object detection algorithms. Included is a set of evaluative approaches, which have become standard in measuring an object perception algorithm's ability to detect objects, and classify them into one of 20 possible object classes.

In Chapter 4, I discuss my research where I modified the PASCAL dataset to study the effects of image perturbation on object proposal algorithm performance. I expanded the original dataset to formulate a new version, PASCAL+, by systematically introducing various levels of brightness, gamma correction, Gaussian blur, and Gaussian

noise [28].

DAVIS

DAVIS [162] is a standardized dataset for evaluating object tracking and object discovery methods. The dataset consists of 50 RGB videos, each decomposed into image frame sequences depicting a moving salient object (e.g., vehicle, pedestrian, or animal) captured at varying distances to the camera. Each image sequence consists of a unique outdoor scene with some containing non-salient detractor objects. Moreover, each sequence is captured from a moving camera under various lighting conditions, clutter, and occlusion, making it a suitable dataset to represent challenges in robot vision.

RGBD multi-view

The RGB-D Multi-view (RGBDMV or RGBD-Scenes) [113] dataset contains over 300 common household objects organized by 51 classes, and is frequently used to validate scene understanding ([80, 112, 22]) and SLAM ([164, 137, 117]) algorithms. This dataset is also primarily used to evaluate depth-based robot vision algorithms for multiple object detection and tracking applications. Images of objects are 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.

Autonomous robot indoor dataset

The Autonomous Robot Indoor 40K Scene (ARID) Dataset [126] contains 3298 RGB images collected from a mobile robot, which autonomously navigated around an indoor office environment to observe an assortment of objects. ARID consists of 10 unique scenes, which depict various household objects in cluttered environments.

ETH-Bahnhof

ETH Bahnhof Dataset [126] contains over 1000 RGB images, which was collected from a stereo camera mounted on top of a mobile platform, which was navigated around a crowded urban environment. The dataset is labeled with bounding boxes around all the pedestrians.

Berkeley 3D objects

The Berkeley 3D Objects (B3DO) [97] dataset contains 849 RGB-D image pairs captured from the Kinect V1, which consists of objects that belong to over 50 object categories. Moreover, the dataset contains 75 unique scenes that depict occluded objects in cluttered indoor environments, with natural lighting. Some scenes in the dataset are spatiotemporally sequenced, where various amounts of motion blur and camera noise are present.

2.8 Chapter summary

This chapter provided a brief discussion of various technical concepts, theories, and evaluation considerations that fall under the umbrella of object perception research. My research covers many of these aspects, which I used to design algorithms that enable robots to discover unseen objects. The following chapter presents a new RGB-D method that improves the computation time of object detection algorithms.

Chapter 3

A new real-time depth-based region of interest cropping algorithm

Pedestrian and object detection algorithms are computationally expensive for mobile robots. Many methods (at the time of writing) incorporate a multi-scaled sliding window approach, followed by a classifier to determine whether or not an object in question exists in an image. This equates to searching each pixel for detection candidates, which leads to computationally demanding loads that are counter-productive to robots which should ideally process images in real-time.

One solution to speed up detection algorithms is to use more powerful hardware (i.e. more GPUs). However, this is not always feasible because compact and power efficient computing systems are needed for mobile robots.

Rather than have a robot's attention divided among every possible location in an image, it can be given guidance about where to look. For example, autonomous vehicles require fast detection algorithms to ensure the safety of pedestrians and other inhabitants of the road. Vision-based algorithms (in contrast to LIDAR and RADAR techniques [185, 54, 68]) typically exploit road geometry for ground-plane and sky estimation to

increase detection speed [63, 18].

In more general applications where ground-plane estimation is not always tractable, proposal algorithms provide a better alternative. Object proposal algorithms operate to suggest regions of an image that are most likely to contain objects and people, to alleviate the high computational demands of object detectors [2, 24, 52, 221]. This idea stems from neurologically-inspired vision [95], where salient characteristics are detected and derived from low-level image features. However, object proposal algorithms are typically trained and designed for static images that do not represent real-world, naturalistic data (i.e. containing motion blur, rotation, and distortion).

In robotics, the prominence of RGB-D cameras has aided in the development of faster object segmentation and simultaneous localization and mapping (SLAM) algorithms [14, 69]. RGB-D is favored over solely RGB methods, because they allow geometric and structural information to be extracted from a scene without incurring high computational overhead [1]. Thus, the processing times of robot perception algorithms reduced by exploiting depth information.

We introduce a new algorithm, *Salient Depth Partitioning* (SDP), that extracts regions of interest from RGB-D images. Our algorithm serves as a basis for "where a robot should look" and is a preliminary step to object and people detection, to reduce computation time and minimally affect detection accuracy. In addition, our algorithm requires no *a priori* knowledge of the environment or reliance on geometric constraints. It can work with virtually any detection algorithm given any calibrated RGB-D image. To validate SDP's effectiveness, we integrated it with four state-of-the-art pedestrian detectors (HOG and SVM [39], Aggregate Channel Features [50], Checkerboards [243], and R-CNN [73]), and found that our algorithm decreases their runtimes by up to 30% with little to no degradation in detector accuracy.

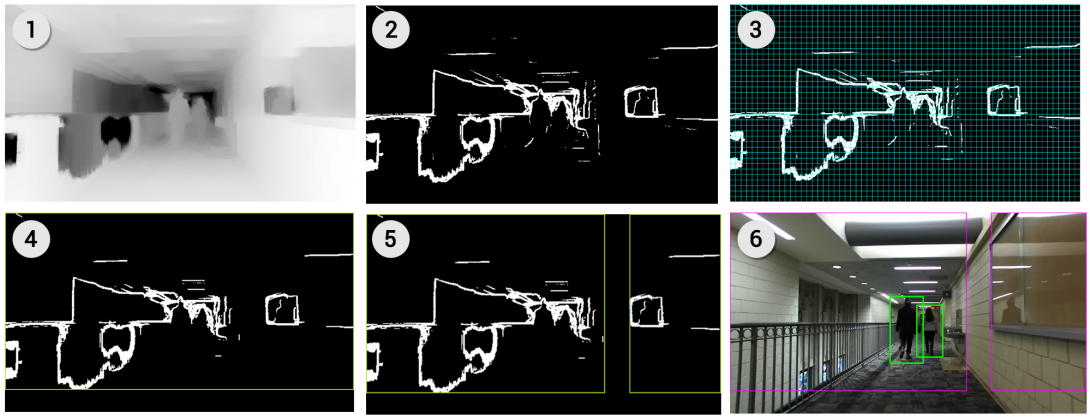

Figure 3.1: The steps of the SDP algorithm. (1) depicts the depth image in its unmodified form, and (2) shows the result of downsampling and Sobel edge-filtering. (3) removes noise via grid partitioning and edge density thresholding. (4) selects a region in the image using the remaining edges filtered by (3). The image is subpartitioned in (5), and transformed to accommodate the size of the original image. The transformed points are used in (6), where a detector is performed on subpartitions of the RGB image.

3.1 Methodology

SDP applies a multimodal approach on RGB-D images with the observation that adjacent pixels do not drastically vary in depth, when those pixels belong to the same object.

The main advantage to utilizing depth images for segmentation is that they do not contain textural information of people and objects, but instead are adept at relaying structural details about the scene [1]. This allows edges to establish clear delineations between objects of varying depth. Using edge detection, we are able to recover ROIs that contain strongly formed edges, or locations in the image that are most likely to contain people and objects. In contrast, large regions with uniform and gradual depths (e.g. walls, floors, buildings, sky, etc.) do not contain strong edges.

SDP does not require previous knowledge about the environment, nor does it require training. Instead, it adapts to what the robot sees. Furthermore, our method can be applied to virtually any detection algorithm, provided that the input is a calibrated

RGB-D image. In the remainder of this section, we describe our algorithm (depicted in Figure 3.1).

3.1.1 Faster edge detection

Edges describe abrupt variations in depth and can be used as predictors to object locations. Thus, we can also leverage detected edges to filter out large non-object regions such as roads, floors, walls, and the sky. However, edge detection and filtering adds considerable overhead to processing time, especially when they are applied via convolution. To reduce the number of computations, the depth map is first downsampled to a lower resolution.

Given an image I_{rgb} and its corresponding depth map, I_d, we downscale I_d by a factor, γ. The amount of downscaling largely depends on the resolution of the image. In the extreme case, downscaling the image by too much will yield a poor quality edge map because the image will be too small to be operated on by the convolution filter.

For images of 1280×720 resolution, we systematically tested the scale size to select $\gamma = 4$, which we found to provide an optimal balance of image preservation and mathematical simplicity (wholly divisible).

Finally, we apply an edge detector to I'_d. In our implementation, we adopted the 3×3 Sobel kernel for its low computation cost and adequate performance, in comparison to some state-of-the-art edge detectors [202]. We also tested our algorithm with better performing edge detectors (e.g. Canny [23] and Laplacian of Gaussian), but found that they negatively affected the runtime of SDP.

3.1.2 Edge filtering

After edge detection, the resultant edge map may still contain weakly formed edges due to noise. To remedy this, I'_d is divided into patches of size 5×5 pixels. The density of each patch is then found by computing the sum of its contents. If the density of any patch exceeds the edge sensitivity threshold, ρ, the contents of that patch are not modified. Conversely, if the density of the patch is less than ρ, the contents of the partition are erased to produce a filtered edge map, $P_{I'_d}$. Edge filtering is given by Equation 3.1.

$$\begin{aligned} &\mathcal{P}_{\mathbf{I'_d}}(n,m) = \begin{cases} \mathbf{I'_d}[\alpha,\beta] & if\ \sum \mathbf{I'_d}[\alpha,\beta] > \rho \\ \emptyset & otherwise \end{cases} \\ &\mathbf{I'_d}_{N\times M};\ \alpha = (5n-4,...,5n),\ \beta = (5m-4,...,5m) \\ &\forall n = 1,...,\frac{N}{5}, \forall m = 1,...,\frac{M}{5} \end{aligned} \tag{3.1}$$

where $\mathcal{P}_{I'_d}(n,m)$ is a 5×5 patch of $\mathbf{I'_d}$ and $\mathbf{I'_d}[\alpha,\beta]$ is a submatrix of $\mathbf{I'_d}$ bounded by rows α and columns β. ρ is the edge sensitivity threshold.

This method allows us to achieve localized image filtering with low computational cost. ρ describes the threshold ratio of edge pixels to background pixels of the depth image, and its value is selected to remove weakly formed edges. The impact of parameter ρ is discussed in Section 3.3.

3.1.3 Region of interest partitioning

Next, we find the minimum and maximum rows and columns $\mathcal{P}$, containing non-zero elements, to generate a bounded subregion, *ROI*. *ROI* is given by Equation 3.2.

$$ROI = \mathcal{P}[R,C] \tag{3.2}$$

where *ROI* is a subregion of $\mathcal{P}_{N\times M}$ bounded by minimum and maximum indices of row and column vectors, R and C, respectively. R is a row vector containing non-zero values of $\mathcal{P}$ and C is a column vector containing non-zero values of $\mathcal{P}$.

ROI may still contain large subregions, containing no edges, which may be removed by creating smaller partitions.

To accommodate a detector's ability to detect objects of minimum sizes, smaller partitions that cannot possibly contain detectable objects are removed. For example, if a detector is only able to detect objects of minimum size 128×64 pixels, we remove partitions measuring a height or width less than 128 and 64 pixels, respectively.

We achieve subpartitioning using a simple plane-sweep technique to detect the sparsity of empty row or column vectors in partition *ROI*, and use an efficient data structure to dynamically store region boundaries. The subpartitioning subroutine is presented in Algorithm 1.

Finally, the subpartitions are upscaled by γ to generate proposals that are consistent with the original image size.

3.2 Evaluation

SDP is designed to decrease the runtime of any object detection algorithm and preserve its accuracy. However, it would be impractical to evaluate the effects of SDP on all of them. Instead, we test SDP within the context of pedestrian detection, which we argue is a suitable and noisy testbed to allow us to evaluate it for robustness.

In our experiment, we selected four state-of-the-art (at the time of writing) pedestrian detection algorithms: Histogram of Oriented Gradients (HOG) and SVM, Aggregate Channel Features (ACF), Checkerboards, and R-CNN.

Despite its age, HOG [39] is still widely used in modern detection algorithms,

Algorithm 1: *Subpartitioning*(ROI, T)
Create partitions from a binary image by detecting sparsity of empty regions.

Input : ROI is a binary image mask of size $N \times M$.
T is the detector's minimum detection dimension.
Output : S list containing row or column indices of subpartitions.

$sweep\_list = \{\}$ // list containing the indices of non-zero valued row or column vectors.
$flag\_start\_count$ = false // if true, initialize the counting of consecutive zero-valued columns or rows.
$empty\_count = 0$ // counter corresponding to the number of consecutive zero-valued columns or rows.

for $m = 1$ *to* M **do**
    if $empty\_count > T$ **then**
        if $flag\_start\_count = true$ **then**
            $S.append(min(sweep\_list))$
            $flag\_start\_count = false$
        if $ROI[N,m].sum > 0$ **then**
            $S.append(max(sweep\_list))$
            $flag\_start\_count = true$
            $sweep\_list$ = $\{\}$
    if $ROI[N,m].sum = 0$ **then**
        $sweep\_list.append(m)$
        increment $empty\_count$ by 1
    else
        $empty\_count$ = 0
        $sweep\_list$ = $\{\}$
Return S

and is regarded by many as the best hand-crafted feature descriptor [244]. The HOG and SVM detector exploits gradient features, which are discretized to enable greater detector immunity, when objects are arbitrary oriented.

Stemming from gradient-based approaches, Dollár et al. [50] proposed ACF, which runs significantly faster and as accurately as the state-of-the-art object detectors. Their method derives feature pyramids to be automatically extrapolated from nearby scales. This allows for quick feature approximations, rather than have explicit time-

consuming computations performed at every scale.

Zhang et al. [244, 243] introduced the *Checkerboards* algorithm, using Filtered Channel Features as an extension of ACF. Using features such as an alternate color space (CIELUV) and HOG, they were able to show that their algorithm produced improved precision and recall scores over the current top-performing pedestrian detectors.

Lately, Convolutional Neural Networks (CNNs) have gained prominence in the computer vision and robotics communities due their adeptness at deriving highly discriminant features. As a result, they have been shown to achieve high recall and precision scores, when used for purposes like object and pedestrian detection. To increase the breadth of our study, we tested SDP with the *Region-based Convolutional Network* (R-CNN) proposed by Ross et al. [73], along with its pedestrian-trained variant, made public by Tome et al. [216].

3.2.1 Data acquisition

We required depth images of formidable quality, that are also calibrated to the RGB images to test our algorithm. However, existing datasets (such as PASCAL VOC [60], INRIA [39], ETH [58], TUD-Brussells [233], and Caltech [48]) do not contain spatiotemporal RGB-D images of pedestrians.

Therefore, we created our own RGB-D pedestrian dataset. Video was recorded with the ZED stereo camera system [200] at 1280×720 resolution at approximately 8 frames per second. We note here that we used the ZED camera as an alternative to the Microsoft Kinect, because it is the most economical camera with the farthest depth ranging ability (up to 30 meters) at the time of writing.

Additionally, we wanted to strictly control for consistent motion and speed across both indoor and outdoor data. To simulate mobile robot vision, we attached the ZED to a hand-wheeled cart which was pushed at a constant walking pace. The camera was also

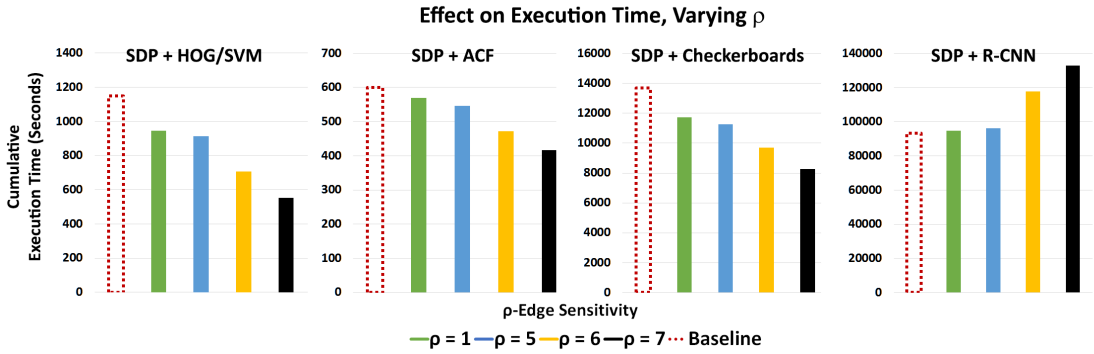

Figure 3.2: Cumulative execution times of several state-of-the-art detection algorithms integrated to SDP, for varying values of ρ - lower execution time is better. Note that we only select ρ values that depict interesting changes in detection behavior. The baseline (detection algorithms without SDP integration) algorithms are depicted in red, and do not have a ρ value.

panned to mimic active vision.

Our dataset consists of images recorded in two hallways and at two outdoor locations. Each location consists of unique geometric landscapes, corridors, occlusive objects, lighting, and variable foot traffic to simulate environments that an autonomous mobile robot may traverse. In total, our dataset contains approximately 3400 RGB-D images.

To obtain ground truth labels, every video frame was annotated with bounding boxes around the location of all visible upright humans in the scene.

For our experiments, we used a laptop equipped with an Intel i7-6700 CPU and 8GB RAM. We selected this platform for its compact size, which could reasonably be used on a mobile robotic platform. All experiments were conducted in MATLAB solely using CPU resources.

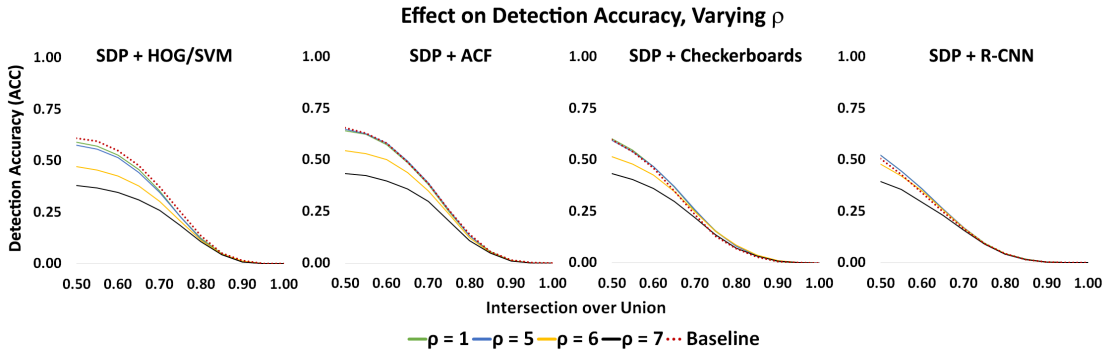

Figure 3.3: Detection accuracy vs. intersection over union threshold (IoU_t) of several state-of-the-art detection algorithms integrated with SDP, for varying values of ρ. These curves correspond to the execution times shown in Figure 3.2. The baseline (detection algorithms without SDP integration) algorithms are depicted in red, and do not have a ρ value. A ρ curve that more closely matches the baseline curve means lower degradation in detection performance.

3.3 Results

Here, we present our results for SDP with regards to execution time, detection accuracy, and computation cost, and discuss their implication in Section 3.4.

3.3.1 Execution time

We measured the cumulative execution time of SDP fused with each of the baseline algorithms (HOG and SVM, ACF, Checkerboards, R-CNN). The threshold parameter, ρ, was incrementally varied to show performance cutoff as edge sensitivity is decreased. Each of the baseline algorithms were also tested, independently of SDP (shown in Figure 3.2).

We found that SDP reduces the execution time of HOG and SVM, ACF, and Checkerboards by up to 30%. Our results show that execution time is inversely proportional to the value of ρ, which demonstrates that SDP was able to abstractly speed up these detectors without modification.

Interestingly, when compared with the other detectors, we found that SDP pro-

duced the opposite effect for R-CNN, where execution time is proportional to ρ.

3.3.2 Detection accuracy

We evaluated the detection accuracy of SDP to account for the detectors' ability to both generate correct predictions, while minimally generating false positives. Here, we measure the detection accuracy of SDP fused with each of the baseline algorithms performed on our dataset, using the IoU bounding box criteria found in [60, 248]. We also use the same evaluation on each of the baseline algorithms independently of SDP. To account for variability in human poses (e.g. stretched out arms, differences in gait), we select an IoU threshold of 0.5, which is the standard value in the pedestrian tracking literature [243].

SDP possesses differences from object proposal algorithms, and is instead characterized as a region cropping algorithm. Rather than generate a number of object proposals, our algorithm generates sparse regions containing multiple objects.

Consequently, we cannot apply the same metrics (i.e. recall/AUC/detection accuracy vs. number of proposals) as those of object proposal algorithms. Instead, we measure detection accuracy as a function of ρ, the edge threshold sensitivity (previously discussed in section 3.1.2), which directly correlates to the quality of proposed regions.

Furthermore, our algorithm extensively relies on depth information, so the performance of SDP also depends on the camera's ability to capture depth at longer ranges. Therefore, to test the performance of our algorithm, we remove predictions and ground truth labels of pedestrians captured beyond the range limitations of our camera. From our measurements, we estimate that the ZED can capture the depth of pedestrians within 13 meters distance at 1280×720 resolution, which measures to approximately 154×64 pixels. Figure 3.4 depicts the depth limitations of the ZED camera.

Figure 3.3 presents the detection accuracy of SDP in conjunction with each of

the selected algorithms; the baseline detection accuracy are also shown. Across all algorithms, there was minimal degradation in performance for $\rho = 5$, which suggests that SDP does not affect detection accuracy for $\rho < 6$. There is significant degradation for all algorithms at $\rho > 5$, which is to be expected, because the amount of image cropping (and number of cropped pedestrians) is directly proportional to ρ.

Figure 3.4: RGB-D images depicting the limitations of the ZED camera at various ranges. A pedestrian is observed from the ZED stereo camera at 5m (left), 10m (center), and 15m (right); the pedestrian is no longer visible in the depth map at 15m.

3.3.3 Computational throughput

To measure computational throughput of SDP, we executed it on our dataset without an object detector; in this test, regions are computed, but no classification is performed. Furthermore, we do not take into account the camera frame-rate, which we consider to be independent of the algorithm.

We report an average throughput of approximately 77 frames per second (approximately 13 ms per image), which is suitable for robots and real-time purposes.

3.4 Discussion

We introduced SDP, and showed that it substantially decreases the execution time of several state-of-the-art pedestrian detectors, without affecting detection accuracy. Thus, we expect that other detectors can also be optimized using region segmentation strategies.

When combined with SDP, the execution times of HOG and SVM, ACF, and Checkerboards are significantly reduced. We discovered that the proportional differences in execution time is dependent on the number of sliding window computations that each algorithm performs. For example, HOG and SVM uses the most sliding window operations of all the algorithms in our evaluation, so its execution time benefits most when more of the image is cropped out.

Contrary to what we expected, the execution time of R-CNN increases proportionally with ρ. We believe this to be caused by R-CNN's built-in object proposal algorithm, which caused it to produce extraneous proposals when the original image is partitioned. Inevitably, the computation time increased because classification is performed on each proposal.

However, it is interesting to see that R-CNN gained a slight increase in detection

performance. This highly suggests that a greater number of false positives are removed when more of the image (without sacrificing the integrity of the detector) is cropped, which supports the hypothesis found in [88].

Detection accuracy degradation was the highest for HOG and SVM. We postulate that this effect is caused when gradient features, neighbor to pedestrian locations, are removed after region cropping. Although actual pedestrians are not removed from the image, region cropping can have an effect on classification when localized histograms are altered.

Combined with SDP, ACF yielded lower degradation in detection accuracy at $\rho = 5$, when compared to $\rho = 1$. This is contrary to intuition, where a larger amount of pruning would expect to yield lower detection accuracy. However, in our experiments with ACF we found a lower number of false positives for $\rho = 5$, than $\rho = 1$. We attribute this to the same effect (false positives removed from region cropping) that was observed in the R-CNN and SDP experiments.

This suggests that the image features extracted by Checkerboards are more invariant to region cropping, and that SDP decreased the runtime of Checkerboards while preserving the optimal number of detectable pedestrians.

One limitation of our work is that our experiments were only studied within the context of pedestrian detection, which we believe to be a noisy testbed to evaluate our algorithm for robustness. However, the our results are encouraging, which motivates us to explore how SDP can be integrated to general object detection algorithms in our future work.

Furthermore, our experimental results showed that SDP does not perform as well with detectors that use an integrated object proposal algorithm (i.e. R-CNN). However, our results could vastly differ if we had direct control over the number of proposals that R-CNN generates, which we also plan to investigate in a future study.

One possible avenue of future work includes testing SDP on new RGB-D cameras as they become commercially available, to see how well our algorithm will scale to more precise depth data. Additionally, we can use SDP to pre-process existing low-power, low-computational algorithms for mobile robots, such as UV sky segmentation [201], RatSLAM [140], and SeqSLAM [139]. With the study of our algorithm in conjunction with these other algorithms, we can also gain a better sense of performance when robots exhibit different types of motion.

3.5 Chapter summary

In this chapter, we introduced SDP and demonstrated that it can substantially improve detection algorithms by decreasing their computation time, while sacrificing little to no detection performance. Moreover, compared to existing object detectors, the computational overhead of SDP is negligibly low (approximately 77 frames per second) and can be easily adapted to any RGB-D robot vision pipeline. This work motivated my later research, and inspired me to explore other computational mechanisms that can address low-level object perception for robots. The next chapter focuses on examining how object proposal algorithms can be useful for robotics domains, which I use to address the problem of object discovery in my later work.

3.6 Acknowledgements

I thank Angelique Taylor for sharing her insightful ideas and valuable feedback, which were influential to this project's success, and for helping me collect and annotate the data used to perform the experiments. I also thank Michael J. Gonzales for helping me create the image in Figure 3.1, which depicts an overview of SDP.

This chapter contains material from “Faster Robot Perception Using Salient Depth Partitioning”, by D. M. Chan, A. Taylor, L. D. Riek, which appears in the Proceedings of IEEE/RAS International Conference on Intelligent Robots and Systems, 2017 [28]. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Investigating object proposal algorithms for robots

Until recently, many object detection 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 [39]). However, by delineating localization and recognition, object detection can be achieved much more quickly, demonstrating superior accuracy over the sliding window approach [72].

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 [72, 175, 125].

Roboticists are beginning to incorporate object proposal algorithms into their systems for purposes other than object detection. For example, Sunderhauf et al. [205] 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 [26].

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 VOC 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 chapter 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 VOC, 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 [165] and SharpMask [166]) and showed that they produced high recall on more naturalistic datasets using only a small number of proposals, suggesting that they have high computational efficiency.

Third, we conducted a study of algorithm execution time on portable hardware to find that most algorithms are suitable for 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 are useful to the broader robotics community, because they 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.

4.1 Related work

When object proposal algorithms began to take prominence in the computer vision community, there were no publicly available datasets that were suitable to evaluate them. Consequently, the PASCAL evaluation protocol was modified to evaluate object proposal algorithms, to measure their ability to recall objects independently of categories. Ergo, object proposal algorithms became widely viewed as "category-independent object detectors", which also strongly implied their generalizability [219, 106].

Recently, several researchers questioned the effectiveness of object proposal algorithms, suggesting that the PASCAL VOC evaluation protocol is flawed. With a thought experiment, Chavali et al. [29] suggested that an object proposal algorithm could be built using a mixture-of-experts model to exploit the dataset's object categories. This would enable the model to excel in the PASCAL evaluation, but have no ability to perform well outside it.

Hosang et. al [87] performed a study of object proposal algorithms on the PASCAL VOC dataset and found that their repeatability is sensitive to some image perturbations such as rotation, shift, and image compression. However, their experiments did not directly address algorithm performance pertaining to real-world, adverse conditions.

Table 4.1: Recent top-performing object proposal algorithms used in our evaluation. W indicates that the algorithm is type window-scored, and S indicates segmentation-based.

Algorithm	Approach	Type
Rahtu [172]	Supervised	W
Objectness [2]	Supervised	W
Geodesic (GOP) [106]	Seed-based	S
Randomized Prim [135]	Supervised	S
Selective Search [219]	Region Similarity Clustering	S
DeepMask [165]	Supervised Deep Learning	S
SharpMask [166]	Supervised Deep Learning	S

Moreover, their evaluation did not consider image perturbations that might be more applicable to robot vision such as contrast, motion blur, and noise.

One major hurdle for developing robust object proposal algorithms for robots is that the datasets used in current evaluations are too clean. Robots do not typically view objects with capture bias, minimal noise, minimal occlusion, and choice lighting conditions – thus, the results of current evaluations are not reflective of real-world scenarios.

Thus, the purpose of this work was to 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 spatiotemporal image sequences and naturally occurring perturbations (noise, occlusion, blur, etc.)–factors that are elemental to robot vision.

4.2 Methodology

Using standard evaluation metrics in computer vision [29], we measured the recall performance of seven top-performing object proposal algorithms [172, 2, 107, 106, 135,

219, 166] (shown in Table 4.1). 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 [87, 60, 122].

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.

4.2.1 Measuring object proposal algorithm 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 [29, 87].

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[1] 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 threshold with fixed number of proposals
- Recall at IoU threshold vs. number of proposals
- Area under recall vs. number of proposals

Recall as a function of IoU threshold with fixed # of proposals

To observe the localization accuracy of object proposal algorithms, recall as a function of IoU threshold is preferred because it represents an algorithm's ability to localize objects as the IoU threshold 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

[1]For a more detailed summary of recall performance metrics, see [29].

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 $K = 300$.

Recall at IoU threshold 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 threshold of 0.5 is commonly regarded as the optimal balance between loose and tight fitting. When the IoU threshold is set to 0.5, all hypothesized regions must at least have an IoU overlap with ground truth objects greater than or equal to 0.5 to be considered correct detections. We conduct two separate evaluations by fixing the IoU threshold to 0.5 and 0.7.

Area under recall (AUC) as a function of # proposals

In AUC, the area under the recall curve (for all IoU thresholds 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 threshold, 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.

4.2.2 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.

4.2.3 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 [113], and Berkeley 3-D Object [97] (shown in Table 4.2).

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 4.2). In total, PASCAL+ contains over 124k images.

Figure 4.1: Sample images of bottles from the three datasets used in our evaluation: PASCAL [60], B3DO [97], and RGBDMV [113].

Table 4.2: A list of datasets used in our evaluation, comparing the presence of some robot vision challenges.

Dataset	Ego-centric	Spatio-temporal	Blur	Occluded Objects	# Images
B3DO [97]	Yes	Yes	Yes	Yes	849
RGBDMV [113]	Yes	Yes	Yes	Yes	18.4k
PASCAL [60]	No	No	No	No	11.5k
PASCAL+	No	No	Yes	No	124k

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 γ 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 $2 * \text{ceil}(2\sigma) + 1$, varying σ from 2 to 10 in increments of 2.

Noise negatively impacts how illumination is perceived, which affect how object

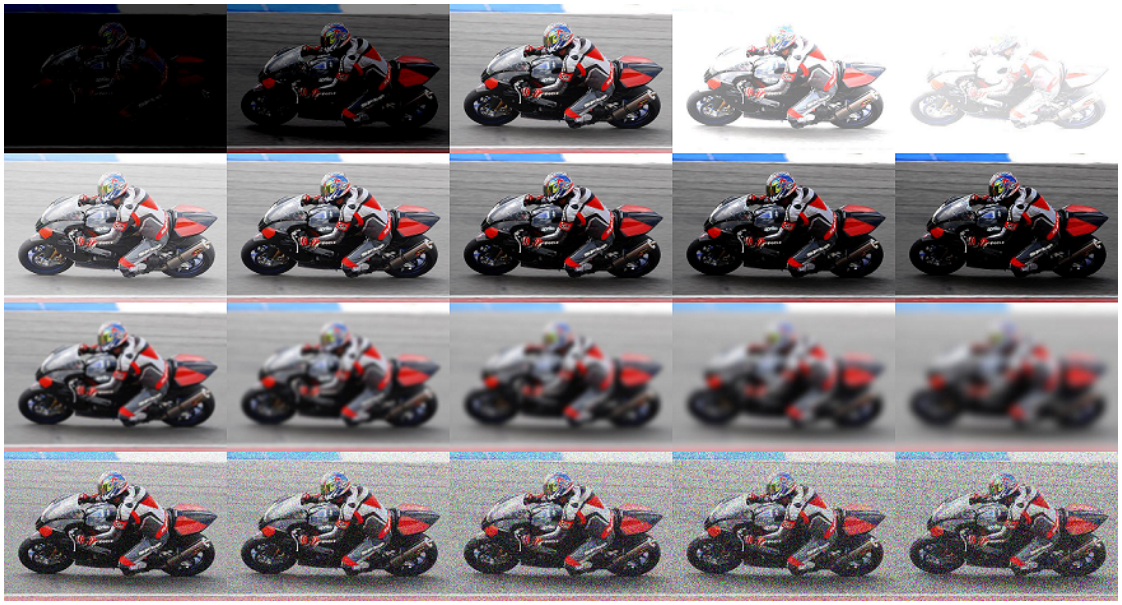

Figure 4.2: Sample images from PASCAL+, depicting perturbations shown in rows from top to bottom, with various magnitudes shown from left to right: brightness $(-200, -120, 0, 120, 200)$, gamma correction $(\gamma = 0.5, 1, 1.5, 2, 2.5)$, Gaussian blur $(\sigma = 2, 4, 6, 8, 10)$, and Gaussian white noise $(\sigma = 0.02, 0.04, 0.06, 0.08, 0.1)$.

proposal algorithms perform. To this end, we applied zero-mean additive Gaussian white noise to each image, varying σ^2 from 0.02 to 0.1 in 0.02 increments.

4.2.4 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 4.1, we generated K-proposals for each dataset listed in Table 4.2. Each of the recall scoring functions (previously described in Section 4.2) were computed, and graphed (shown in Figures 4.3 and 4.4).

DeepMask and SharpMask are built on top of the ResNet-50 model (pre-trained on ImageNet [41]), and were additionally trained using segmentation annotations from the Microsoft COCO dataset [122]. 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.

4.3 Results

In this section, we present a summary of our results (see Table 4.3 and Figures 4.3, 4.4, and 4.5).

4.3.1 Recall as a function of IoU threshold with fixed # of Proposals

Across all datasets (see Figure 4.3), DeepMask and SharpMask produced the highest recall, which suggests that they are able to locate the most objects.

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

4.3.2 Recall at fixed IoU threshold as a function of # proposals

At IoU threshold = 0.5 (see Figure 4.3), 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 threshold = 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 threshold = 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 $\leq$ 100, where the recall gap between deep learning methods and non-deep learning methods is considerably narrower.

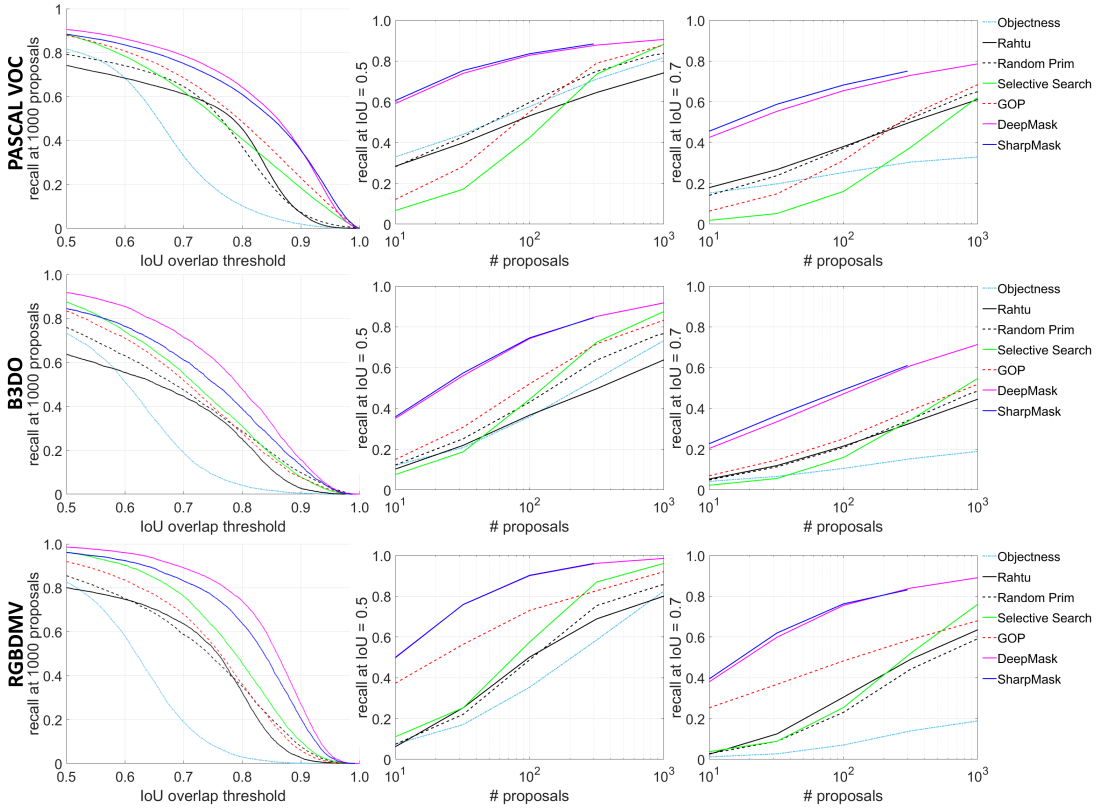

Figure 4.3: Recall vs. IoU Threshold and Recall vs. # Proposals

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.

4.3.3 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 4.4), 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.

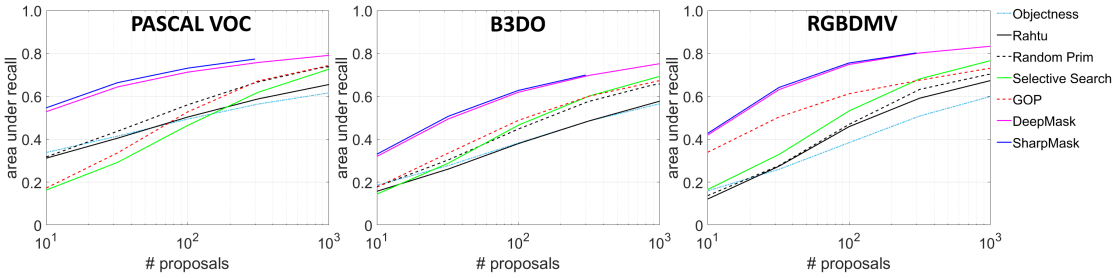

Figure 4.4: AUC vs. # Proposals on the PASCAL (left), B3DO (center), and RGBDMV (right).

4.3.4 Evaluation on PASCAL+

The results of our evaluation on PASCAL+ are presented in Figure 4.5. 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.

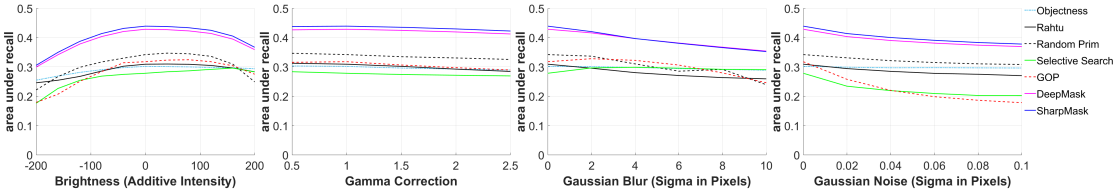

Figure 4.5: AUC at 100 proposals vs. image perturbations from PASCAL+.

4.3.5 Computation time

We present the execution time of each algorithm, corresponding to the configurations used in our evaluation (K values) in Table 4.3. We also report the average time it takes for each algorithm to compute one proposal.

Table 4.3: Computation time (per 640×480 image) of the algorithms used in our evaluation.

Algorithm	***K***	**Cumulative Runtime per Image at *K***	**Computation Time per Proposal**
Randomized Prim	900	0.63s	0.69ms
GOP	1000	1.25s	1.79ms
DeepMask	1000	3.13s	3.06ms
Objectness	1000	3.92s	3.29ms
SharpMask	300	1.28s	4.27ms
Selective Search	1000	4.43s	4.43ms
Rahtu	1000	4.70s	4.70ms

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.

4.4 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 [2], 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 [51].

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 [116, 215]), we suggest using DeepMask, which has the highest recall at IoU thresholds ≥ 0.8.

4.5 Chapter summary

In this chapter, I discussed our investigation into object proposal algorithms, where we found that the standard evaluation protocols were not suitable for designing and evaluating object proposal algorithms for robotics contexts. This work also provided us with insights into the robustness of different object proposal algorithms, which was inspirational to my current research. In the next chapter, I discuss a new object discovery method which leverages object proposals.

4.6 Acknowledgements

This chapter contains material from “Object Proposal Algorithms in the Wild: Are they Generalizable to Robot Perception?”, by D. M. Chan and L. D. Riek, which appears in the Proceedings of IEEE/RAS International Conference on Intelligent Robots and Systems, 2019 [25]. The dissertation author was the primary investigator and author of this paper.

Chapter 5

A new salient object discovery method for monocular robot vision

One challenge that robots must overcome "in the wild" is to discover unseen objects. This will play an important role for robots to learn about new objects to help them perform tasks (e.g., appraising anomalous parts or tools used for repair, retrieval of uncommon items, investigating new environments, identifying entities that can be manipulated, etc.). Furthermore, by exploring and interacting with unseen objects, robots can learn in a scalable manner.

Roboticists often leverage multi-modal data (e.g., via depth sensors) association to infer the presence of unseen objects and their properties [28]. For example, depth segmentation is prevalent in grasping [116], simultaneous localization and mapping (SLAM) [203], and multi-object tracking [152] topics. Recently, some researchers have also proposed using depth proposals to discover and track generic objects in street scenes [154, 104]. However, depth cameras can be particularly sensitive to placement, dynamic lighting conditions, and distance [53]. This can cause methods that rely on depth or 3D image to be more constrained to specific domains (e.g., close or far-range applications),

in contrast to standard RGB cameras which can be used for more general vision problems. As a consequence, some researchers show that depth is not necessary for robot perception, and that vision-related tasks can be achieved using monocular camera systems [136, 43].

Using solely RGB imaging, some researchers address the problem of detecting unseen objects that are visually salient, also known as *salient object discovery*. The most recent approaches require some degree of semi-supervision, for example, manually drawing a bounding box or segmentation mask that encapsulates the boundaries of an object. This annotation provides a training example (e.g., one-shot object learning), so that the object can be discovered from multiple viewpoints [31, 225]. However, these methods can be poorly suited for real-time robotics because they require a human to manually initialize them each time that a robot encounters a new object.

To date, little work addresses salient object discovery in an unsupervised manner, typically by aggregating multi-view images [227, 158]. These methods extract key features (e.g. optical flow boundaries) at spaced time intervals across entire video segments to determine the presence of salient objects. However, these methods often take many image frames to process, which can be prohibitively slow for real-time robots [158]. This can disrupt reactive decision-making behaviors of robots, which are essential for time-sensitive tasks (c.f., [94]).

To this end, we introduce an unseen salient object discovery method, Unsupervised Foraging of Objects (UFO). UFO is automatic and unsupervised in the sense that it does not require manual annotation or initialization to discover objects. Furthermore, UFO only requires a spatiotemporal stream of RGB image frames for input, making it a suitable method for robots with monocular RGB camera systems.

The contributions of this research are threefold. First, our method discovers unseen objects within a few image frames, in contrast to existing methods that require entire image sequences to be processed before object discovery can occur. By extension,

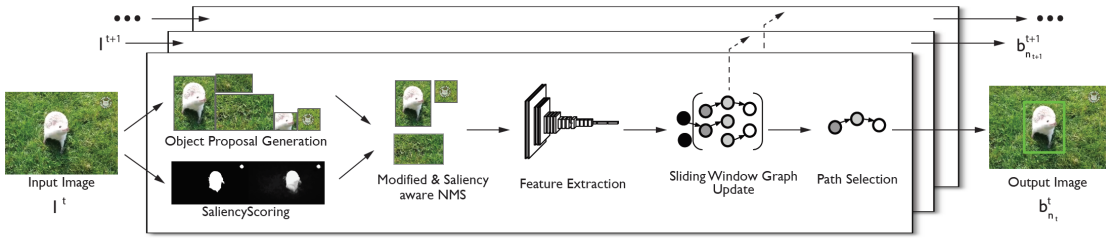

Figure 5.1: Our unsupervised object discovery framework, UFO, is composed of six processes: a) object proposal generation, b) saliency scoring , c) non-maximum suppression (NMS), d) feature extraction, e) sliding window graph update, f) path selection, and g) object proposal prediction.

UFO is able to discover salient objects in real-time image sequences, while also achieving state-of-the-art recall, precision, and accuracy.

Second, we designed a novel parallel discover-prediction paradigm to enforce the selection of strong object candidates, improving precision over state-of-the-art salient object discovery methods. Our method leverages the history of previously discovered objects to make new predictions about their locations while also re-discovering them using low-level image cues. In this way, previously discovered object instances can be used to make self-correcting predictions as objects change appearance over time.

Third, our method is less computationally expensive than predominant methods that employ motion cues. Instead, UFO leverages object proposals, exploiting their spatiotemporal consistency to obtain object boundaries. UFO can infer unseen objects in seconds, whereas optical flow-based methods can take on the order of *minutes*.

5.1 Methodology

Here, we describe UFO, which addresses unsupervised salient object discovery for RGB vision. UFO introduces the concept of an augmented GOP, a data structure that contains a bounding box (b), an objectness confidence score (o), a saliency score (s), and a feature embedding (f). A bounding box (b) corresponds to the location of

a potential object, and an objectness confidence score (o) measures the likelihood that the same bounding box tightly encloses an object. Saliency (s) measures how much a bounding box visually stands out in an image frame. A feature embedding (f) is a compact representation of an image region inside a bounding box, used to detect object correspondences for adjacent frames.

We developed UFO with the observation that GOPs corresponding to non-objects appear randomly, due to camera noise, lighting, or image artifacts. In contrast, GOPs containing objects appear more consistently, making it possible to detect salient object correspondences in image sequences.

Transforming GOPs to vertices and object correspondences to edges, we construct a sliding window graph. This graph is updated for each frame, tracking the histories of discovered objects, which are used to predict GOPs in the event that the OPA fails to make consistent predictions.

Figure 5.1 shows an overview of UFO and each of its aspects, which include: (a) object proposal generation, (b) saliency scoring, (c) saliency-aware non-maximum suppression, (d) feature extraction, (e) sliding window graph updating, (f) path selection, and (g) object proposal prediction. For the first frame of an image sequence, UFO performs Steps (a)-(f), generating an object prediction for the next frame in Step (g). Steps (a)-(f) repeat for the next frame, merging the object prediction from the previous frame after Step (c). This procedure repeats for incoming frames, where the sliding window graph is updated with the history of discovered objects in Step (e). These steps are described in detail in the following section.

5.1.1 Object proposal generation

Given an image sequence, we first apply an OPA to an image frame, I^t, at time t to generate a finite number (N) of GOPs. Each GOP consists of a bounding box which

we denote as $b^t_{n_t}$, and we denote the set of bounding boxes generated by the OPA as $B^t = \{b^t_{n_t} | n_t \in 1 \dots N_t\}$. For each GOP, the OPA assigns a confidence value that relates to the probability that the GOP correlates to an object, or *objectness score*. We denote the set of objectness scores as $O^t = \{o^t_{n_t} | n_t \in 1 \dots N_t\}$.

In our implementation, we selected DeepMask [165] for the OPA with $N = 100$, which we determined to provide an optimal balance of speed and performance.

5.1.2 Saliency scoring

To discover salient objects, we designed a method to measure the normalized saliency of each GOP. We first compute a saliency heat map, $\mathcal{U}^t$, for image frame I^t, using the Minimum Barrier Distance (MBD) Transform [241]. Next, we generate a binary mask, $\mathcal{U}^t_{msk}$, to compute the strongest salient pixels that highly correlate to object centers of mass. Since MBD generates a bimodal distribution of salient pixels centered around Gaussian distributed clusters, we can apply a globally-optimal threshold (e.g., Otsu's method [155]) to yield $\mathcal{U}^t_{msk}$, which represents the locations of the strongest salient pixels that correspond to "hot points" in $\mathcal{U}^t$. This approach allows us to compute a normalized measure of saliency for each GOP, which can adapt to changes in lighting and contrast that can affect the raw saliency values in $\mathcal{U}^t$.

There are two components in our saliency metric: saliency area (s_{area}) and saliency centeredness (s_{center}). Saliency area measures the number of salient pixels enclosed by each bounding box, $b^t_{n_t}$, with respect to (w.r.t.) the total number of salient pixels in the image frame (see Equation 5.1):

$$s^t_{area_{n_t}} = \frac{\sum_{x,y \in b^t_{n_t}} \mathcal{U}^t_{msk}(x,y)}{\sum_{x,y \in I^t} \mathcal{U}^t_{msk}(x,y)} \tag{5.1}$$

where x and y denote pixel coordinates w.r.t. I^t.

GOPs with bounding boxes that contain no salient pixels (i.e., $s^t_{area_{n_t}} = 0$) are immediately discarded. For sake of discussion and simplicity, we treat N as a constant, although N is time-dependent in practice.

S_{center} measures how closely located a GOP is to the center of a hot region in U^t (shown in Equation 5.2):

$$s^t_{center_{n_t}} = \max_{x,y \in b^t_{n_t}} \Big(\mathcal{U}^t(x,y) \circ g(x,y) \Big) \tag{5.2}$$

where $g(x,y)$ is a two dimensional Gaussian function centered-aligned with bounding box $b^t_{n_t}$. We require the standard deviations of $g(x,y)$ to be arbitrarily small to bias the center pixels, so we selected $\sigma_x = \frac{w}{10}$ and $\sigma_y = \frac{h}{10}$, respectively, where w is the width and h is the height of $b^t_{n_t}$ [1]. This allows maximally salient pixels at the center of $b^t_{n_t}$ to yield a saliency centeredness of 1, and non-salient pixels at the center of $b^t_{n_t}$ to yield a saliency centeredness of 0.

The saliency area and saliency centeredness metrics are then aggregated (shown in Equation 5.3) to construct a set of saliency scores, $S^t = s^t_{n_t} | n_t \in 1 \ldots N$ such that (s.t.) $0 \leq s^t_{n_t} \leq 1$.

$$s^t_{n_t} = s^t_{area_{n_t}} s^t_{center_{n_t}} \tag{5.3}$$

[1] We experimented with various standard deviation values and found that any value between $\frac{w}{20} \leq \sigma_x \leq \frac{w}{5}$ and $\frac{h}{20} \leq \sigma_y \leq \frac{h}{5}$ did not impact performance.

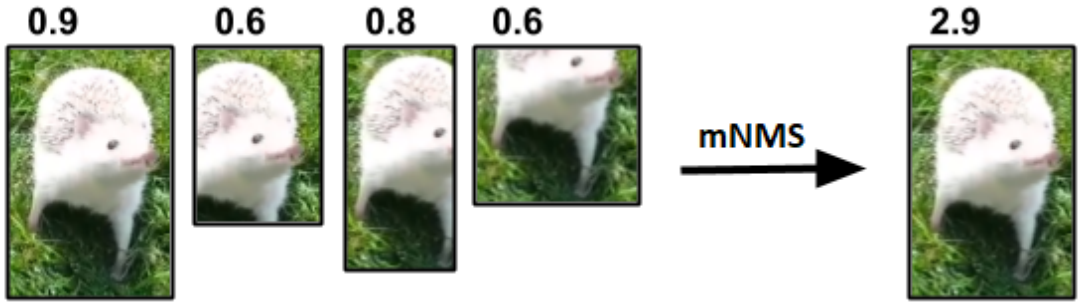

Figure 5.2: In modified non-maximum suppression (mNMS), the strongest bounding box is assigned with the cumulative sum of the scores of all overlapping neighbors.

5.1.3 Modified and saliency-aware non-maximum suppression

OPAs will generate redundantly overlapping GOPs that need to be removed. This is achieved by using non-maximum suppression (NMS), which selects the best GOP among overlapping ones. Traditional NMS is greedy [148], using the confidence scores directly generated by the OPA. While OPAs can produce high quality bounding boxes (i.e., those that tightly enclose objects), they can sometimes falsely assign parts of an object with higher confidence scores than the whole object. Additionally, OPAs can sometimes assign high objectness scores to background elements. These conditions can cause standard greedy NMS to incorrectly suppress GOPs that are essential to object discovery.

Thus, we designed a novel NMS procedure that accounts for both objectness and saliency; our approach is constructed in two stages: modified greedy NMS (mNMS) and saliency-aware greedy NMS (sNMS) shown in Algorithm 2. In mNMS, the maximally-selected GOPs are augmented with the sum of scores of their neighboring GOPs (illustrated in Figure 5.2). These sum-of-neighbor scores favor GOPs with more within-frame redundancy (i.e., GOPs with stronger correlations to objects). The outputs of the mNMS are then applied to sNMS.

For sNMS, a graph is constructed using GOPs as vertices, and the intersection

over minimum area (IoMA) of their bounding boxes as edges (shown in Equation 5.4). When used in tandem with the sum-of-neighbor scores from mNMS, sNMS suppresses non-redundant GOPs that more likely correlate with irrelevant entities (e.g., object parts or background regions), that also overlap with real objects.

$$\text{IoMA} = \frac{a \cap b}{min(a_{area}, b_{area})} \tag{5.4}$$

where a and b are bounding boxes and *area* denotes their area.

In sNMS, we sum-aggregate the scores from mNMS (i.e., sum-of-neighbors) and saliency scores, S^t, to select the best GOP among neighbors. This achieves selection of GOPs that have more redundant overlap that are also highly salient. To eliminate outlier bias, we apply feature scaling to normalize the objectness and saliency scores, shown in Equation 5.5:

$$\mathbf{Y} = \frac{\mathbf{X} - \overline{\mathbf{X}}}{max(\mathbf{X}) - min(\mathbf{X})} \tag{5.5}$$

where a bolded variable indicates a vector, $\overline{\mathbf{X}}$ denotes the mean of vector $\mathbf{X}$.

5.1.4 Feature extraction

For each GOP, we extract their image features to detect correspondences across adjacent image frames. We experimented with various CNN architectures (AlexNet [108], VGG19 [199], ResNet [86], and InceptionV3 [207]) to study how they perform as feature extractors for bipartite image feature matching (discussed in Section 5.1.5). In general, since image content does not drastically vary across adjacent image frames,

Algorithm 2: Saliency-Aware Greedy NMS (sNMS)

Inputs: A set of GOPs.

Initialization: Let $\mathbf{G}_s = (\mathbf{V}_s, \mathbf{E}_s)$ using GOPs as vertices and the **intersection over minimum area overlap** of their bounding boxes define edges. $\mathbf{V}'_s = \emptyset$.

while $|\mathbf{V}_s| > 0$ **do**

$$v_{max} \leftarrow v_i = \underset{i}{\operatorname{argmax}} \sum_{j \in \mathbf{V}_s} \Phi(i,j) = \begin{cases} 1, & \text{if } e(v_i, v_j) \\ 0, & \text{otherwise} \end{cases}$$

$$\mathbf{V}_{neighbors} \leftarrow \{v | e(v, v_{max}) \geq 0.5\}$$

$$v_{select} \leftarrow v_n = \underset{n | v_n \in \mathbf{V}_{neighbors}}{\operatorname{argmax}} (v_n^*.o + v_n^*.s)$$

$$v_{select}.o = \max(\mathbf{v}_{neighbors}.o)$$

$$\mathbf{V}'_s \leftarrow v_{select}$$

$$\mathbf{V}_s = \mathbf{V}_s - \mathbf{V}_{neighbors}$$

end

Return $\mathbf{V}'$

▷ *o is the objectness score corresponding to vertex v_n.*

▷ *s is the saliency score corresponding to vertex v_n.*

▷ *$*$ denotes scale-normalized (shown in Equation 5.5).*

we found that the performance differences of UFO were negligible (less than $0.01 mAP$) when substituting the CNN. We selected VGG-19 for its simplicity, speed, and object representational power. Features are extracted from the final fully connected layer ($fc7$), and stored in a set which we denote as $F^t = \{f^t_{n_t} | n_t \in 1 \ldots N_t\}$.

5.1.5 Updating the sliding window graph

Previously, we discussed bounding boxes (B^t), objectness scores (O^t), saliency scores (S^t), and feature vectors (F^t) at time t. We now group these components into a single structure, denoting a set of GOPs at time t as $V^t = \{v^t_{n_t} \supseteq b^t_{n_t}, o^t_{n_t}, s^t_{n_t}, f^t_{n_t} | n_t \in 1 \ldots N_t\}$. For example, the bounding box of the n-th GOP at time t, is expressed as $v^t_{n_t}.b$. Using this notation, we expand our discussion from a single image frame to a time-dependent sequence, where the current frame at time t is I^t, and a prior frame is $I^{t-\tau}$ for time τ.

To track the history of prior GOPs with a memory-scalable approach, we adapted a sliding window graph. This enables GOPs that fall outside of a temporal window to be removed from memory, allowing UFO to run indefinitely.

Given a window of size W, we construct a sliding window directed acyclic graph. In our implementation, we set $W = 3$ (we later discuss this parameter in Section 5.2.4). We denote this graph as $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, with $v^t_{n_t}$ as vertices, and edges defined by the spatiotemporal intersection over union (IoU) of their bounding boxes between adjacent frames. For example, the vertices in the window are denoted as $\mathbf{V} = \{V^{t-W} \dots V^t\}$. $\mathbf{V}$ is stored in a queue where V^t corresponds to the GOPs of the most recent frame, I^t.

Edges are generated in a directed matter from $t-1$ to t, where edges from previous time steps are moved further into the queue as new frames become available. Edges are only formed for GOPs if their bounding boxes are time-adjacent and spatially overlapping (i.e. $v^{t-1}_{n_i}.b \cap v^t_{n_j}.b > 0 | n_i \in 1 \dots |V^{t-1}|, n_j \in 1 \dots |V^t|$).

For each pair of GOPs in adjacent frames I^t and I^{t-1}, we compute their pairwise similarity score, Λ (shown in Equation 5.6), using their bounding box dimensions (i.e., width and height) and VGG19 features. $\Lambda = 1$ indicates little or no similarity and $\Lambda = 0$ indicates perfect similarity.

$$\begin{aligned} \Lambda &= \lambda(a,b), \qquad 0 \leq \lambda(a,b) \leq 1 \\ \lambda(a,b) &= 1 - e^{-zssd(a_f, b_f)} e^{-\left(\frac{|a_h - b_h|}{a_h + b_h} + \frac{|a_w - b_w|}{a_w + b_w}\right)} \end{aligned} \tag{5.6}$$

where a and b are bounding boxes corresponding to spatiotemporally adjacent GOPs, subscripts w and h refer to a bounding box's width and height, and subscript f denotes their feature embeddings. *zssd* computes the similarity of two feature vectors via zero-mean sum of square differences.

To find optimal edge assignments for vertices V^{t-1} and V^t we apply bipartite

minimum-cost matching using their similarity scores, $\lambda(v_{n_{t-1}}^{t-1}, v_{n_t}^{t})$, where $n_{t-1} \ldots N_{t-1}$ and $n_t \ldots N_t$. This procedure is repeated for incoming frames to form object paths. The time step is updated and the previous version of the sliding window graph is moved further into the FIFO queue (i.e., $V^{t-W} \leftarrow V^{t-W+1}, \ldots, V^{t-1} \leftarrow V^{t}$).

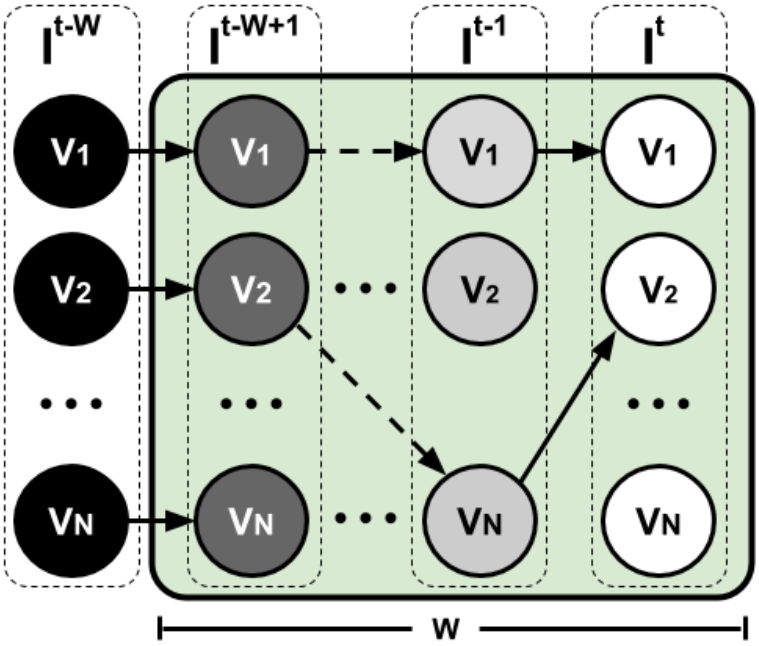

Figure 5.3: The sliding window graph of length W (shown in green). Vertices represent GOPs and edges represent similarity scores. Dashed lines show the resultant, non-adjacent connections of vertices between times $t-W+1$ and $t-1$. Solid lines show direct connections between frame-adjacent vertices.

5.1.6 Path selection

Finally, to discover objects, we compute the shortest paths in **G** which correspond to the greatest GOP correspondences in the image sequence. **G** contains a finite number (K) of shortest paths which we denote as $P = \{p_k | k \in 1 \ldots K\}$, where p_k contains a set of vertices: $p_k = \{v_{n_W}^{t-W+1}, \ldots, v_{n_t}^{t}\}$. From P, the goal is to find a path p_k, corresponding to the most salient object in the sequence.

We designed a greedy path selection strategy to find the path that contains vertices

with the highest objectness and saliency scores, which likely corresponds to the most salient object in the image sequence. To prevent outlier bias, we apply scaling (shown in Equation 5.5) to the set of objectness ($V^t.o$) and saliency scores ($V^t.s$) from each frame in interval $t-W \ldots t$. For each path p_k, the normalized objectness and saliency scores are used to derive sum-aggregated selection scores ($p_{k.score}$), shown in Equation 5.7.

The set of paths $P = \{p_k | k \in 1 \ldots K\}$ is sorted in descending order w.r.t. to $p_{k.score}$. Finally, the top-ranking path is selected, where the bounding box $v^t_{n_t}.b \in p_0$, is the output of UFO.

$$p_{k.score} = \sum_{\tau=t}^{t-WP+1} \sum_{v^\tau_{n_\tau} \in p_k} v^\tau_{n_\tau}.s + \sum_{\tau=t}^{t-WP+1} \sum_{v^\tau_{n_\tau} \in p_k} v^\tau_{n_\tau}.o \tag{5.7}$$

5.1.7 Object proposal prediction

While GOPs of objects tend to consistently appear throughout an image sequence, it is still unlikely that they will be present in every frame, since the appearance of objects can change dramatically over time. This can cause UFO to temporarily misdetect discovered objects until the corresponding path is regenerated in the sliding window.

To mitigate this problem, we generate a template using the bounding box from the previous frame. This template is cross-correlated with the current frame to predict the location of the object. Assuming small object displacement between adjacent frames, we form a search area two times the template, centered at the object's previously known location.

The resulting bounding box is then assigned with the mean objectness score of the vertices in its path to form a GOP prediction. We also apply a penalization factor to the mean objectness score, which enables the objectness score of a recurrent prediction to decay over time, preventing erroneous predictions from propagating due to drift.

The prediction is merged with the output of the OPA for the current frame (Section 5.1.1). Merging is achieved by computing the similarity score (shown in Equation 5.6) and solving bipartite matching for within-frame GOPs. Among matching pairs, the higher-scoring GOP is selected as the final merged candidate.

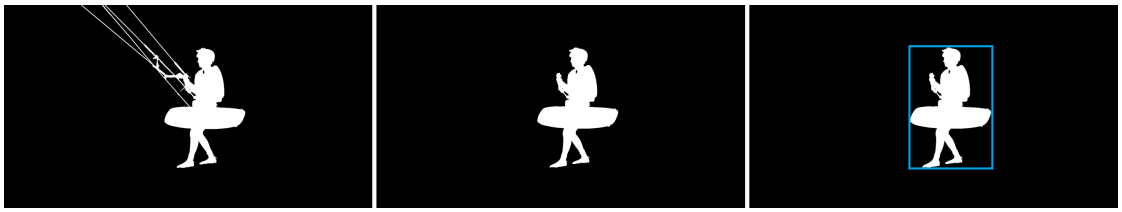

Figure 5.4: Image sequence depicting segmentation mask to bounding box conversion procedure. Left: original segmentation mask. Center: ropes are removed. Right: the final bounding box forms a perimeter around the mask.

5.2 Evaluation and results

5.2.1 Dataset

We use the DAVIS 2016 dataset [162], which is a standard testbed for evaluating salient object discovery methods.

The DAVIS dataset contains ground truth segmentation masks for each frame, which we converted to bounding box format[2]. To generate high quality bounding boxes (e.g., to support tighter fits around objects), we needed to adjust some segmentation masks by removing thin object parts (e.g., strings, ropes, chains) – for an example, see Figure 5.4. In total we adjusted 281 of 3455 images (i.e., from *paragliding-launch* (79), *kite-walk* (79), *kite-surf* (49), and *boat* (74) scenes).

[2]We note that while we made adjustments to DAVIS to make our experiments bounding box compatible, we compared our results to the recent survey by Caelles et al. [20], which also reported auxiliary bounding box evaluation results. We found no discernible differences in FST's performance. We note however, that we use the latest release of SAL which performs better than reported in their paper.

5.2.2 Comparison to the state-of-the-art

We selected two recent unsupervised salient object discovery methods to compare against UFO: Saliency-Aware Geodesic (SAL) [227] and Fast Segmentation (FST) [158], the fastest and most accurate methods reported in the literature [20]. We evaluated FST and SAL using their default parameters. Our results are shown in Figures 5.6, 5.1, 5.5, and 5.7.

To provide a fair comparison to UFO, we converted the segmentation masks from the output of SAL and FST to bounding boxes using the procedures in Section 5.2.1.

To measure performance, we employed widely-used metrics from the salient object discovery literature: precision, recall, F-measure, accuracy, mean average precision (*mAP*), and end-to-end computation per frame in seconds ($t(s)$) [209]. To measure the generalizability of each method, we computed the precision for each image sequence, then averaged them across all 50 sequences to compute the mean average precision (*mAP*).

We found that UFO was approximately 6.5 times faster than SAL (which took on average 35.7 seconds to infer object discovery predictions for each frame) and FST (which took on average 29.4 seconds). Comparing precision, recall, F-measure, and accuracy, we found that UFO scored similarly to FST, while SAL scored lower for all metrics.

Table 5.1: Comparison between UFO and two state-of-the-art methods on DAVIS. We measured precision, recall, F-score, accuracy, mean average precision (*mAP*) at IoU = 0.5. We report the average end-to-end computation time in seconds per frame ($t(s)$). Columns with upward arrows indicate that a higher score is better. Lower computation time is better. UFO scores best for computation time, precision, F-score, accuracy, and *mAP*.

Method	$t(s)\downarrow$	Precision ↑	Recall ↑	F-score ↑	Accuracy ↑	*mAP* ↑
UFO	**4.52**	**0.662**	0.645	**0.654**	**0.486**	0.568
FastSeg (FST) [158]	29.4	0.659	**0.647**	0.653	0.485	**0.586**
Salient Geodesic (SAL) [227]	35.7	0.517	0.597	0.597	0.425	0.517

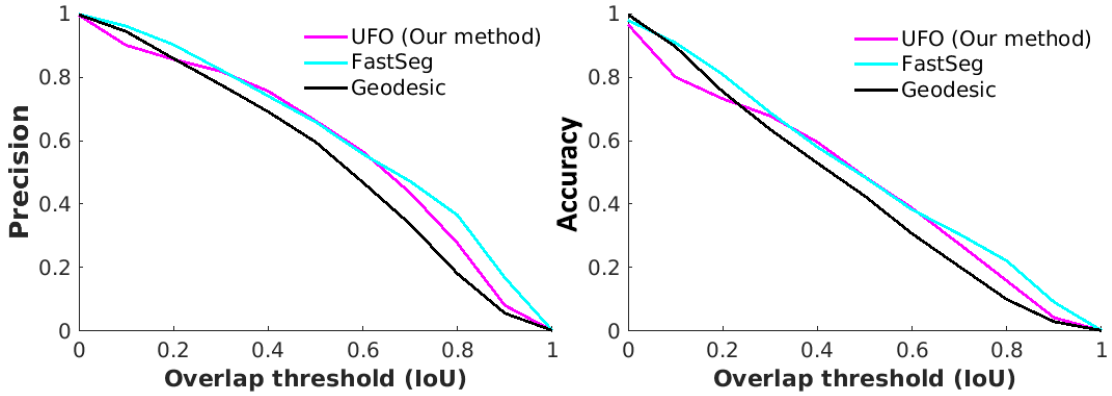

Figure 5.5: Precision (left) and Accuracy (right) measured over IoU threshold (IoU_t), which correlate to robustness to false-positives and overall accuracy, respectively (higher is better). For the standard overlap criterion ($IoU_t = 0.5$), UFO scores highest.

5.2.3 Ablation experiments

To analyze the importance of each system component, we evaluated ablated versions of UFO. Specifically, we investigated how UFO performs without the proposal prediction (UFO-P) and saliency-aware NMS (UFO-NMS) components. We also evaluated UFO without either of these components (UFO-P-NMS). We show our results in Figures 5.5 and 5.2.

When prediction is removed from UFO (UFO-P), performance declines across all metrics, with exception to computation time (4.41 seconds per frame). Our results suggest that the prediction component is important for correcting object discovery instances that can become corrupt over time.

When NMS is removed (UFO-NMS), performance again declines across all metrics. Moreover, UFO-NMS has longer computation time (6.41 seconds per frame). This suggests that saliency-aware NMS removes non-salient OPAs, reducing both the number of false positives and computation time.

Finally, we show that UFO-P-NMS has substantially longer computation time than UFO (6.53 seconds per frame). This further suggests that both components are

significant to UFO's design, such that the prediction component increases recall, while saliency-aware NMS reduces computation time.

Table 5.2: Ablation Study Findings: overall performance of UFO declines when prediction and/or NMS components are removed from the pipeline.

Method	$t(s)\downarrow$	Precision ↑	Recall ↑	F-score ↑	Accuracy ↑	$mAP\uparrow$
UFO	4.52	**0.662**	**0.645**	**0.654**	**0.486**	**0.568**
UFO-P	**4.41**	0.649	0.632	0.641	0.471	0.556
UFO-NMS	6.41	0.631	0.614	0.622	0.452	0.547
UFO-P-NMS	6.53	0.661	0.644	0.652	0.471	0.563

5.2.4 Performance due to window size

To explore how the size of the sliding window affects UFO, we incrementally varied parameter W (results shown in Figure 5.3). Our experiments show that as W increases, UFO can focus on false-positive or detractor objects instead of the main object, which reduces recall performance. Specifically, UFO will favor objects that remain in the window for a longer time, which possibly includes detractor objects. However, we also found that a larger W decreases computation time because it also reduces the number of object candidates.

When W is small, we found that UFO is more adaptable to new object candidates. This also enables it to recover previously discovered objects that were lost due to occlusion. We also found that a smaller W enables UFO to achieve higher recall when objects of interest are more easily discernible from the background (e.g., more salient).

Table 5.3: Effect of Window Size (W) Findings: overall performance of UFO declines as the window size increases.

Method	$t(s)\downarrow$	Precision ↑	Recall ↑	F-score ↑	Accuracy ↑	$mAP\uparrow$
UFO, W=3 (default)	4.52	**0.662**	**0.645**	**0.654**	**0.486**	**0.568**
UFO, W=5	4.36	0.646	0.629	0.638	0.468	0.555
UFO, W=10	4.17	0.629	0.612	0.620	0.450	0.541
UFO, W=20	**4.03**	0.618	0.601	0.609	0.438	0.534

5.2.5 Computation time of system components

To study which factors affect the speed of UFO, we measured the computation time of each of its system components. In general, we found that most components were computationally inexpensive, with exception to the OPA and NMS algorithm. However, we can expect the speed of the pipeline to improve by refining the OPA and NMS algorithm, since all other components are dependent on them. Our results are shown in Figure 5.4.

Table 5.4: Average per-image computation time of individual system components in UFO.

System Component	$t(s)$
Object Proposal Generation (OPA)	2.13
Saliency Scoring	0.23
Modified & Saliency-Aware NMS	1.08
Feature Extraction	0.63
Sliding Window Graph Update	0.27
Path Selection	0.01
Prediction	0.15

5.2.6 Discussion

UFO is a vision-based approach which can complement other perception methods that address object learning for robots. For example, UFO can be used with haptic-based approaches, to enable robots to autonomously explore novel objects by both means of touch and sight (c.f. [47]). UFO can also be suitable for detecting unfamiliar objects, to inspire robots to examine them via active perception.

Our method is designed for RGB vision, making it a viable perception framework for robots with monocular camera systems. Moreover, UFO is flexible in that it does not require depth data, which can be problematic for object discovery methods that rely on

range estimation.

UFO is approximately 6.5 times faster than recent unsupervised salient object discovery methods for RGB vision. Our method leverages an OPA to generate salient GOPs, exploiting their spatiotemporal consistency to discover objects in image sequences. We also designed UFO with a discover-prediction approach, which recovers previously discovered objects in the event that the OPA fails to generate suitable GOPs. With this approach, we show that object discovery can be achieved much more quickly than predominant approaches that rely on motion boundary detection. Since unsupervised salient object discovery methods require multiple frames and iterations to discover objects, optical flow-based methods take on the order of *minutes*, while UFO is able to reduce this time to seconds. To our knowledge, UFO is the fastest unsupervised salient object discovery method for RGB vision.

We evaluated UFO on the DAVIS dataset, which reflects real-world robot perception challenges including moving cameras and objects, motion blur, and occlusion. In terms of overall precision, F1-measure, and accuracy, UFO attained the highest performance among the methods studied. Moreover, UFO was able to perform consistently across nearly all of the scenes, suggesting that it can generalize to a broad range of robot vision contexts (see Figure 5.7).

We also found that UFO was robust to motion blur and dynamic lighting. In some image sequences (c.f., "*mallard-fly*" in Figure 5.6), the object of interest is visible at start of the sequence, but became heavily blurred when both the object and camera velocities suddenly changed. Because UFO does not rely on motion boundaries, it was still able to discover these objects, which suggests that it is robust to faster camera movement, suggesting its suitability for mobile robot vision.

One limitation was that we used DeepBox [111] to generate GOPs, where experimentation with other OPAs could have possibly improved our results. However, DeepBox

still enabled UFO to achieve state-of-the-art recall and precision, and we treat our current design as a lower bound for performance.

Figure 5.6: Sample object discovery sequence across a challenging scene (i.e., *mallard-fly*) from the DAVIS 2016 dataset. Our results suggests that UFO is robust to dynamic lighting, and fast camera and object motion, which is difficult for methods that rely on optical flow or motion boundaries.

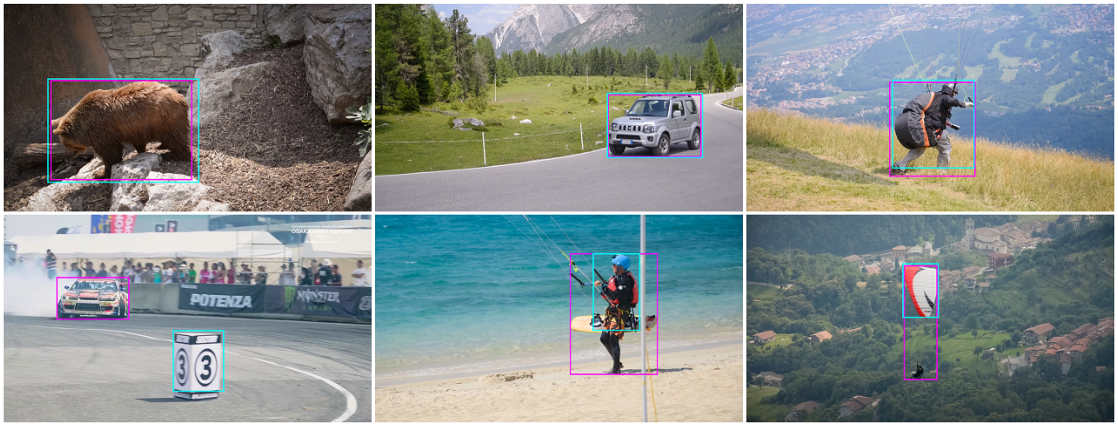

Figure 5.7: Examples of successful (top row) and less successful (bottom row) object discovery instances. Cyan boxes show the output of UFO, and magenta boxes correspond to ground truth objects.

5.3 Chapter summary

In this chapter, I discussed UFO, a new unsupervised salient object discovery method that can quickly and automatically discover unseen salient objects on-the-fly.

This research was influential to my current goal of achieving faster object discovery. In particular, this work motivated my exploration into faster methods for object proposal generation, which is an important factor for achieving this goal. The next chapter addresses this challenge by introducing a new real-time object proposal algorithm for robot vision.

5.4 Acknowledgements

I thank Hee Rin Lee for helping me design the image showing the overview of UFO in Figure 5.1.

This chapter contains material from “Unseen Salient Object Discovery for Monocular Robot Vision?”, by D. M. Chan and L. D. Riek, which appears in Robotics and Automation Letters and in the Proceedings of IEEE/RAS International Conference on Intelligent Robots and Systems, 2020 [27]. The dissertation author was the primary investigator and author of this paper.

This chapter also contains material from ”Unsupervised Salient Object Discovery for Robots”, by D. M. Chan and L. D. Riek, which appears in the Proceedings of Robotics: Science and Systems Pioneers [26]. The dissertation author was the primary investigator and author of this paper.

Chapter 6

A new real-time object proposal algorithm for robot vision

One promising approach toward faster object discovery concerns object proposal algorithms which can be used to query images for regions of interest that are likely to contain objects. Previously discussed in Chapter 5, object proposals can enable robots to perceive unseen objects with a high degree of accuracy and recall [152, 27]. However, current methods have high computational overhead, making them unsuitable for real-time, real-world robot perception [25].

To address this gaps, we introduce a new robot-optimized real-time object proposal algorithm, which can recall any conspicuous object using a one-stage network (RaccooNet[1]). We designed RaccooNet to address real-world robot vision challenges (i.e., camera motion, blur, noise) in contexts where robots need to quickly make decisions in uncertain environments [210, 213]. RaccooNet accepts standard RGB images as an input, allowing it to be adapted into existing vision pipelines and systems to support unseen object discovery for robots.

[1]Celebrating the unofficial mascot of our campus, and for its ability to Recall Any ConspiCuous Object using a One-stage Network, we named our method RaccooNet.

RaccooNet employs a one-stage, dense sampling approach to generate a diverse set of high-quality object proposals. Using a feature pyramid network to combine the advantages of low- and high-level features of deep convolutional neural networks, RaccooNet can robustly address object scale challenges. Moreover, RaccooNet incorporates a novel IoU prediction approach, making it more efficient in its ability to recall objects using a smaller number of object proposal predictions. In contrast to existing region proposal networks [178, 120], our method is designed for real-time operation, and for mobile robots.

The contributions of this work are four-fold. First, we introduce RaccooNet, a new real-time object proposal algorithm that perceives unseen objects, and is designed for mobile robot perception applications. To our knowledge, RaccooNet is the only method capable of true real-time performance, which is approximately three times faster than the currently top-performing object proposal algorithms at 47.9 fps, while also achieving state-of-the-art recall performance. We also show RaccooNet is robust when encountering a variety of real-world robot perception challenges, including camera motion, blur, and noise, across a range of dynamic scenes.

Second, to our knowledge, we are the first to design an object proposal algorithm with an intersection over union (IoU) overlap confidence module, which improves object recall using a smaller number of proposals (thus substantially improving its efficiency).

Third, we introduce RaccooNet Mobile, which at 171 fps, is approximately ten times faster than top-performing methods. We also show that RaccooNet Mobile is suitable for computationally-limited, low-powered systems by evaluating it on a popular embedded platform, the Nvidia Jetson TX2, without diminishing its recall performance.

Finally, we plan to release our implementations of both systems as open-source, which will benefit the broader robotics community to address real-time robot perception challenges.

6.1 Bounding box regression

Previously discussed in Chapter, 2.5 object proposal algorithms often involves predicting objects using one of two representations: segmentation masks or bounding boxes. In general, segmentation can be computationally expensive and slow, which has prevented its widespread adoption in real-world applications [85]. In contrast, bounding boxes provide a coarser approximation, but they are significantly faster to compute and are thus ubiquitous in many robot vision systems.

Bounding boxes form rectangular perimeters around objects, and are typically represented using a Cartesian-based coordinate system (e.g., x, y, width, and height). Recently, researchers used CNNs to predict bounding box coordinates as continuous variables (i.e., bounding box regression) [178, 125, 224]. In this manner, CNNs also learn parameters to invoke eccentric features that correspond to strong object boundaries.

Some researchers apply predefined boxes, i.e., anchor boxes, at various locations of the image to improve bounding box accuracy and training time. Using this approach, CNNs learn to make adjustments for each anchor box until they converge to their target object boundaries [208, 178]. Since anchor boxes can be adapted to possess unique properties (e.g., location, sizes, ratios), they can also specialize at detecting objects with similar characteristics. However, existing methods favor using multiple anchor boxes that simultaneously target the same objects [176, 247, 231]. While this increases the probability that those objects will be recalled, it also leads to redundant bounding box predictions that reduce the network's overall efficiency.

Addressing this limitation, we developed a new anchor-based bounding box regression technique which RaccooNet employs. RaccooNet uses an asymmetric assignment procedure to map anchor boxes to the feature pyramid according to their optimal receptive field dimensions. In effect, this forces overlapping anchors to selectively target

objects where their features are more prominent (e.g., smaller anchors projected onto higher resolution feature maps). With this approach, we mitigate the number of redundant predictions per target object, allowing the network to recall a diverse range of objects with a smaller number of object proposals.

6.2 Real-time object proposals with RaccooNet

RaccooNet is a one-stage, feed-forward CNN that predicts object proposals in real-time (shown in Figure 6.1). It uses an anchor-based approach to densely sample images for object-rich features. Composed of a three-level feature pyramid network (FPN), RaccooNet can generate object proposals over a large range of scales and aspect ratios, making it robust for real-world robot perception. Unique to RaccooNet, we use asymmetrically distributed anchors across feature pyramid levels, which we adapted to their respective field sizes. We also designed a novel approach to predict overlap confidence, allowing our method to be efficient in its ability to yield high quality object proposals using a small number of predictions.

In this section, we first discuss details of our target encoding procedure. We then discuss how we designed RaccooNet to predict object proposals, which we characterize by three attributes: object locations, objectness confidence, and overlap confidence. We then present the loss function which we use to optimize RaccooNet.

6.2.1 Encoding target objects

Taking a standard RGB image as an input, RaccooNet outputs tensors that correspond to object proposal predictions. Thus, a critical design challenge involves encoding its output tensors such that they map to target objects relative to their position in the image. We accomplish this goal in three steps: design the anchor boxes, map the anchor

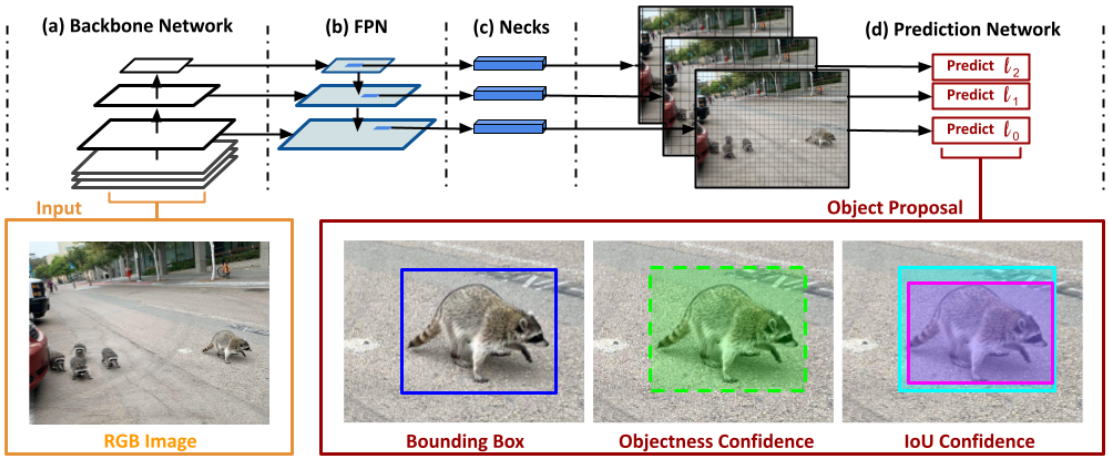

Figure 6.1: Overview of RaccooNet. An image is inputted into a pre-trained (a) backbone network, where its multi-scale features are extracted by a (b) feature pyramid network (FPN). From the FPN features from the receptive field are passed into the (c) necks to form spatially dependent feature encodings. Intuitively, this process subdivides the image into cells with respect to the FPN dimensions. Being fully convolutional, the (d) prediction network cells make dense object proposal predictions each consisting of: a bounding box (blue), objectness confidence interval, and IoU confidence score.

boxes to their FPN level according to size, then finally encode target objects to anchor boxes.

First, we compute a set of anchor boxes $\mathcal{A} = \{a_d \supseteq (a_d^w, a_d^h) | d = 0 \ldots D\}$ using k-means clustering on the Microsoft COCO [122] training set, where we use the width and height dimensions of all ground truth objects as sample points. Motivated by the widely accepted design choice of three anchors per FPN level [120], we parameterized nine clusters ($D = 9$).

We avoid the problem of using uniformly distributed anchor boxes (e.g., RPN) which can oversaturate the lower-levels of the FPN, leading to higher computational cost. We instead assign anchor boxes in an asymmetric fashion where we consider their positioning with regard to the width and height of the output layer of the backbone network, which is $\delta \times \delta$, or where $\delta = 7$ for most CNN architectures. Our intuition is that this will more efficiently assign anchors according to their appropriate size. For example,

smaller anchors are optimally paired with smaller objects; moreover, smaller objects are less likely to overlap with each other, which necessitates less anchor boxes. Starting from the bottommost FPN level, we assign anchors having a minimum dimension less than $\frac{2}{\delta}$ to l_0. Anchors having a maximum dimension greater than $\frac{4}{\delta}$ are assigned to the highest level, l_2; all other anchors are assigned to l_1; these assignment rules are shown in Algorithm 3.

Algorithm 3: Assign Anchors to FPN Level

Input: anchors, $\mathcal{A} = \{a_d \supseteq (a_d^w, a_d^h) | d = 0 \dots D\}$
Output: FPN levels, l_0, l_1, l_2
for $d \in D$ **do**
  if $min(a_d^w, a_d^h) < \frac{2}{\delta}$ **then**
    $l_0 \leftarrow a_d$
  else if $max(a_d^w, a_d^h) \geq \frac{4}{\delta}$ **then**
    $l_2 \leftarrow a_d$
  else
    $l_1 \leftarrow a_d$
end
Return l_0, l_1, l_2
▷ δ *is the native dimension of the backbone network's output layer.*

Once anchors are matched to an FPN level[2], they are assigned to their output tensors for every row $i \in I$ and column $j \in J$, where we now denote anchors $\mathcal{A} = \{a_{ijk} \supseteq (a_{ijk}^w, a_{ijk}^h)\}$, for $k \in K$ anchors.

Lastly, we encode target objects to anchors, where the goal is to generate an optimal pairing such that we achieve the maximum intersection-over-union (IoU) overlap for all pairs. We show the generalized form of IoU in Equation 6.1:

$$\text{IoU}(b_1, b_2) = \frac{b_1 \cap b_2}{b_1 \cup b_2} \tag{6.1}$$

[2]While our FPN structure possesses three levels, we only discuss one level for the remainder of this section for the sake of notational simplicity.

where b_1 and b_2 are two arbitrary bounding boxes.

We define the set of bounding boxes around target objects $Q^{gt} = \{q_m^{gt} \supseteq (x_m^{gt}, y_m^{gt}, w_m^{gt}, h_m^{gt}) | m \in 1 \ldots M\}$. Formulating a bipartite graph from Q^{gt} to $\mathcal{A}$, where we define edges by their IoU, we match optimal pairs using Equation 6.2:

$$q_m^{gt*} = \underset{q_m^{gt} \in M}{\operatorname{argmax}} \operatorname{IoU}(q_m^{gt}, a_{ijk}) \tag{6.2}$$

where q_m^{gt*} is the optimal pairing to a_{ijk}.

Simplifying, we let $q_m^{gt*} \rightarrow q_{ijk}^{gt}$ so that matched target objects and anchors share the same indices. We now denote the set of matched targets to cell-anchors $Q^{gt} = \{q_{ijk}^{gt} \supseteq (x_{ijk}^{gt}, y_{ijk}^{gt}, w_{ijk}^{gt}, h_{ijk}^{gt}) | \neg\forall(i,j,k)\exists q_{ijk}^{gt}\}$.

6.2.2 Network components

We define object proposal predictions for each row $i \in I$, column $j \in J$, and anchor $k \in K$ with six tensor variables (v): v_{ijk}^{x}, v_{ijk}^{y}, v_{ijk}^{w}, v_{ijk}^{h}, v_{ijk}^{obj}, and v_{ijk}^{iou}.

Tensors v_{ijk}^{x}, v_{ijk}^{y}, v_{ijk}^{w}, and v_{ijk}^{h} correspond to one bounding box location in cartesian coordinates (x position, y position, width, and height, respectively). Tensor $v_{ijk}^{obj} \in [0,1]$ corresponds to the likelihood that the object proposal contains an object. Finally, tensor $v_{ijk}^{iou} \in [0,1]$ corresponds to the predicted accuracy of the bounding box's overlap with an object.

Predicting object locations

RaccooNet makes an object proposal prediction for each row $i \in I$, column $j \in J$, and anchor $k \in K$. Intuitively, this formulation divides an image into a grid, where we can consider each i, j position a "cell". With respect to the bounding box positional

coordinates, RaccooNet makes the initial assumption that objects occupy the center of each cell, where tensor variables v^x_{ijk} and v^y_{ijk} predict positional offsets between a bounding box and the origin of its cell. Using a sigma function, we constrain the offsets so that they cannot extend beyond the boundaries of their cells. To convert the tensor predictions to normalized coordinates, we add the column and row offsets, $c_i = \frac{1}{I} i$ and $c_j = \frac{1}{J} j$, $x^{pred} \in [0,1]$ and $y^{pred} \in [0,1]$, shown in Equations 6.3 and 6.4:

$$x^{pred} = \frac{1}{I}\sigma(v^x_{ijk}) + c^x_{ij} \tag{6.3}$$

$$y^{pred} = \frac{1}{J}\sigma(v^y_{ijk}) + c^y_{ij} \tag{6.4}$$

Tensor variables v^w_{ijk} and v^h_{ijk} predict how to adjust the anchor boxes, $\mathcal{A} = \{a_{ijk} \supseteq (a^w_{ijk}, a^h_{ijk})\}$, so that they fit an object's width ($w^{pred} \in [0,1]$) and height ($h^{pred} \in [0,1]$). Consistent with ubiquitous bounding box regression techniques, we adapted w^{pred} and h^{pred} to the log space to improve their fitness to a wide range of generic objects, which we show in Equations 6.5 and 6.6:

$$w^{pred} = a^w_{ijk} e^{v^w_{ijk}} \tag{6.5}$$

$$h^{pred} = a^h_{ijk} e^{v^h_{ijk}} \tag{6.6}$$

From Equations 6.3-6.6, we denote the set of predicted bounding boxes $Q^{pred} = \{q_{ijk}^{pred} \supseteq (x_{ijk}^{pred}, y_{ijk}^{pred}, w_{ijk}^{pred}, h_{ijk}^{pred}) | \neg\forall(i,j,k)\exists q_{ijk}^{pred}\}$.

Next, we define a loss function that minimizes the positional (x and y) and dimensional (h and w) errors between predicted and target bounding boxes. Rather than employing conventional losses (e.g., $L1, L2, smooth_{L1}$) to train each term individually [71, 175], we adopt Complete IoU (CIoU) loss [246]. This enables our network to learn positional and dimensional terms jointly, which consolidates other important factors such as aspect ratio and IoU overlap [179, 246]. The general form of CIoU loss given two arbitrary bounding boxes is given by:

$$\begin{aligned} &\mathcal{L}_{CIoU}(b_1, b_2) = \\ &\big(1 - \text{IoU}(b_1, b_2)\big) + \frac{g(b_1, b_2)^2}{\mathcal{H}(b_1, b_2)^2} + \theta(b_1, b_2) f(b_1, b_2) \end{aligned} \tag{6.7}$$

where $g(b_1, b_2)^2$ is the squared Euclidean distance between centroids of bounding boxes b_1 and b_2, and $\mathcal{H}(b_1, b_2)$ is the diagonal of the rectangular convex hull of b_1 and b_2. $f(b_1, b_2)$ measures the aspect ratio consistency between b_1 and b_2 and is given by Equation 6.8:

$$f(b_1, b_2) = \frac{4}{\pi^2}(arctan\frac{b_1^w}{b_1^h} + arctan\frac{b_2^w}{b_2^h}) \tag{6.8}$$

θ scales down the weight of f to give higher priority to the other terms (i.e., overlap area and distance) and is given by Equation 6.9:

$$\theta(b_1, b_2) = \frac{g(b_1, b_2)}{\Big(1 - \text{IoU}(b_1, b_2)\Big) + f(b_1, b_2)} \tag{6.9}$$

Applying Q^{pred} and Q^{gt} to CIoU loss, we formulate our bounding box regression loss in Equation 6.10:

$$\mathcal{L}_{bbreg}(q^{gt}_{ijk}, q^{pred}_{ijk}) = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \mathbb{1}^{gt}_{ijk} \mathcal{L}_{CIoU}(q^{gt}_{ijk}, q^{pred}_{ijk}) \tag{6.10}$$

where

$$\mathbb{1}^{gt}_{ijk} = \begin{cases} 1 & \text{if } \exists q^{gt}_{ijk} \\ 0 & \text{else} \end{cases} \tag{6.11}$$

Predicting objectness confidence

For each anchor, RaccooNet outputs an objectness confidence interval, $v^{obj}_{ijk} \in [0, 1]$, to predict the likelihood that each bounding box contains an object. This is a critical component of our design, because the network learns bounding box parameters for each anchor, even though anchors do not necessarily contain objects. Thus, the network will output noisy bounding boxes which can be filtered out by thresholding those with low objectness confidence.

One common approach to designing a network to predict objectness is to minimize the cross-entropy (CE) loss between ground truth class s and predicted class u, defined by the expression: $\mathcal{L}_{CE}(s, u) = -s\log(u) - (1-s)\log(1-u)$, $s \in [0, 1]$ and $u \in [0, 1]$ [30, 133, 100]. However, CE loss can bias the network to falsely classify objects as background when the number of objects per image is disproportionately smaller than the number of background regions.

Here, we adopt focal loss [121] to rebalance the weight distribution across correctly and incorrectly classified objects. For convenience, we write CE loss using shorthand notation first defining z as a piecewise function of s and u in Equation 6.13:

$$z(s,u) = \begin{cases} u & \text{if } s = 1 \\ 1-u & \text{if } s \neq 1 \end{cases} \tag{6.12}$$

We can then write CE loss in the form:

$$\mathcal{L}_{CE}(s,u) = -\log\Big(z(s,u)\Big) \tag{6.13}$$

Focal loss is then given by Equation 6.14:

$$\mathcal{L}_{FL}(s,u) = -\alpha\Big(1-z(s,u)\Big)^{\gamma}\log\Big(z(s,u)\Big) \tag{6.14}$$

where $\alpha \in [0,1]$ and $\gamma > 0$ are hyperparameters that adjust the weights of the correctly/incorrectly classified training examples; we use the default values from [121], where $\alpha = 0.25$ and $\gamma = 2$.

Using the standard IoU threshold ($IoU_t = 0.5$) true-positive metric, we consider positive examples $q^{gt}_{ijk}, q^{pred}_{ijk}$ s.t. $\text{IoU}(q^{gt}_{ijk}, q^{pred}_{ijk}) \geq 0.5$. Objectness confidence loss is then given by Equation 6.15:

$$\mathcal{L}_{objconf}(s^{obj}_{ijk}, v^{obj}_{ijk}) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\mathcal{L}_{FL}(s^{obj}_{ijk}, v^{obj}_{ijk}) \tag{6.15}$$

where

$$s_{ijk}^{obj} = \begin{cases} 1 \text{ if } \exists q_{ijk}^{gt} \\ 0 \text{ else} \end{cases} \tag{6.16}$$

and

$$v_{ijk}^{obj} = \begin{cases} 1 \text{ if IoU}(q_{ijk}^{gt}, q_{ijk}^{pred}) \geq 0.5 \\ 0 \text{ else} \end{cases} \tag{6.17}$$

Predicting overlap accuracy

One unavoidable complication for using CNNs to generate object proposals is that they generate redundant predictions that negatively impact object recall performance. Within many object perception contexts, the de facto solution is to apply a greedy-class non-maximum suppression (NMS) strategy. Applied after forward propagation, NMS selectively outputs bounding boxes having the strongest classification score among overlapping proposals; other bounding boxes are otherwise removed. While greedy-class NMS is generally effective at removing outlier bounding boxes, Jiang et al. [98] showed that class confidence is suboptimal because it does not correlate to bounding box overlap accuracy.

To address this, we designed our network to directly predict bounding box overlap accuracy using an IoU confidence loss, shown in Equation 6.18. We formulate our network to learn a novel IoU confidence interval, $v_{ijk}^{iou} \in [0,1]$ for positive target objects. Using a weighted regression approach, our network accomplishes negative hard mining to suppress object proposals that have poor overlap, which we define as $q_{ijk}^{gt}, q_{ijk}^{pred}$ s.t. $\text{IoU}(q_{ijk}^{gt}, q_{ijk}^{pred}) < 0.4$.

$$\begin{aligned}\mathcal{L}_{iouconf}(\text{IoU}(q_{ijk}^{gt}, q_{ijk}^{pred}), v_{ijk}^{iou}) = \\ \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} u_{ijk}^{obj} smooth_{L1}\left(\text{IoU}(q_{ijk}^{gt}, q_{ijk}^{pred}), v_{ijk}^{iou}\right) \\ + \Psi \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} u_{ijk}^{noobj} smooth_{L1}\left(0, v_{ijk}^{iou}\right)\end{aligned} \tag{6.18}$$

where Ψ is a weighing factor to balance the number of objects with non-objects (we select a value of $\Psi = 0.001$ to approximate a 1:1000 ratio). $smooth_{L1}$ is a loss function that incorporates the benefits of $L1$ and $L2$ losses [71]. u_{ijk}^{noobj} is a threshold used to determine hard negatives, and is given by Equation 6.19:

$$u_{ijk}^{noobj} = \begin{cases} 1 & \text{if IoU}(q_{ijk}^{gt}, q_{ijk}^{pred}) < 0.4 \\ 0 & \text{else} \end{cases} \tag{6.19}$$

6.2.3 Loss function

Combining Equations 6.1, 6.10, and 6.15, the loss function which we use to optimize RaccooNet is given by Equation 6.20:

$$\begin{aligned}\mathcal{L}_{RaccooNet} = \lambda_1 \mathcal{L}_{bbreg}(q_{ijk}^{gt}, q_{ijk}^{pred}) \\ + \lambda_2 \mathcal{L}_{objconf}(s_{ijk}^{obj}, v_{ijk}^{obj}) \\ + \lambda_3 \mathcal{L}_{iouconf}(\text{IoU}(q_{ijk}^{gt}, q_{ijk}^{pred}), v_{ijk}^{iou})\end{aligned} \tag{6.20}$$

where $\lambda_1 = 1$ and $\lambda 2 = \lambda 3 = 2$ are weights to balance the individual losses.

6.3 Evaluation and results

We evaluated RaccooNet (and RaccooNet Mobile) for its speed, efficiency, and overall effectiveness for generating object proposals for unseen objects in robot perception contexts (shown in Figures 6.4). We compare our method to four state-of-the-art deep learning object proposal algorithms (shown in Figure 6.1): DeepMask [165], Guided Anchoring [224], RPN [120], and SharpMask [166]. To investigate performance for a variety of robot vision contexts, we conducted our experiments on three challenging robotics datasets: RGBD-scenes [113], Autonomous Robot Indoor Dataset 40k Scene (ARID) [126], and the ETH Bahnhof Dataset [59]. We also conducted ablation experiments to show the effectiveness of RaccooNet's unique IoU confidence component, which predicts overlap accuracy to improve object proposal efficiency.

Table 6.1: Summary of methods used in our evaluation along with their attributes.

Method	Type	Backbone
DeepMask [165]	Segmentation	ResNet-50
Guided Anchoring [224]	Bounding Box	ResNet-50-FPN
RPN [120]	Bounding Box	ResNet-50-FPN
SharpMask [166]	Segmentation	ResNet-50
RaccooNet	Bounding Box	ResNet-50-FPN
RaccooNet Mobile	Bounding Box	MobileNetV2-FPN

6.3.1 Implementation details

To enable RaccooNet to achieve real-time object proposal generation suitable for robot vision, we designed a CNN that is both fast and expressive in its ability to represent complex objects. Our network is composed of a backbone network (ResNet-50 [86] pre-trained on ImageNet [108]), which we augmented with an FPN consisting of three levels, where each level is designed to predict small, medium, and large objects. To accommodate standard camera ratios (4:3), we modified the backbone to accept

$512 \times 384 \times 3$ input images.

We designed our FPN using three principal design components: upsampling, lateral connectivity, and 1×1 convolutional pooling. To combine the deeper, semantically-rich features [119] with the shallower spatial features from the backbone network, we recursively upsampled the output of the backbone by multiples of two, to create a three-layer FPN network that emulates the shape of the backbone. We then add 1×1 lateral connections from the backbone layer to their respective FPN levels. For each FPN level, we apply 1×1 convolution filters to create 1×512 channel necks, which serve to increase representational power and preserve fine-grained spatial information which is crucial for locating objects. Finally, from each neck we form a mixed regression-classification network that outputs object proposal prediction tensors.

RaccooNet Mobile

To make our method suitable for mobile robots with limited computational resources, we developed a smaller and faster variant, RaccooNet Mobile, which we built on top of a lightweight backbone optimized for mobile devices, i.e., (MobileNetV2 [191]). We used the default input resolution of the backbone network ($224 \times 224 \times 3$), which significantly reduced its number of learnable parameters compared to the baseline RaccooNet. However, this also means that RaccooNet Mobile predicts a much smaller number of object proposals. Nevertheless, this tradeoff allows RaccooNet to be considerably faster at 171 fps.

6.3.2 Training

We trained RaccooNet using the Microsoft COCO 2017 training and validation sets [122], which consists of candid objects in everyday scenes. Each image consists of a unique background that contains an arbitrary number of objects belonging to 91 common

object classes. In total, the dataset contains approximately 118k training images and 5k validation images. Due to its large size and for its high-quality annotations, COCO serves as the standard training and evaluation platform for a wide range of computer vision problems.

We optimize our network via gradient descent using the Adam algorithm [102] and its default decay parameters ($\beta 1 = 0.9, \beta 2 = 0.999$), with a sample size of 16 images per iteration for a total of 35 epochs, at an initial learning rate of $1e-4$ for 5 epochs. For the final 30 epochs, we trained the network using a scheduled learning rate of $5e-5$, dropping this value by a factor of two for every 10 epochs.

We regularize each of the samples with 50% probability ($Pr = 0.5$) of horizontal and vertical flip transformations. We also randomly apply Gaussian blur ($Pr = 0.1$, VAR $\in [1,2]$). These values were chosen by experimenting with our model on a small subset of the MS COCO training and validation splits until we achieved convergence and maximum recall.

6.3.3 Datasets

To investigate the performance of RaccooNet for a variety of mobile robot perception contexts, we selected two indoor datasets (RGBD-scenes and ARID) and one outdoor dataset (ETH Bahnhof) to use as evaluation testbeds (see Figure 6.6 for dataset examples and sample outputs from RaccooNet). We selected these datasets because they consider robot vision challenges such as variable lighting conditions, blur, occlusion, clutter, scale, and camera/object motion among diverse scenes. Additionally, none of the methods in our experiments were trained on these datasets, enabling them to be favorable platforms to evaluate their generalizability to recall unseen objects in real-world applications.

6.3.4 Metrics

To measure how object proposal algorithms can efficiently and accurately locate unseen objects, we employ the following metrics:

- Recall at $IoU_t = \{0.5, 0.7\}$ vs. number of proposals.
- Recall vs. IoU_t, fixing number of proposals to 1000.

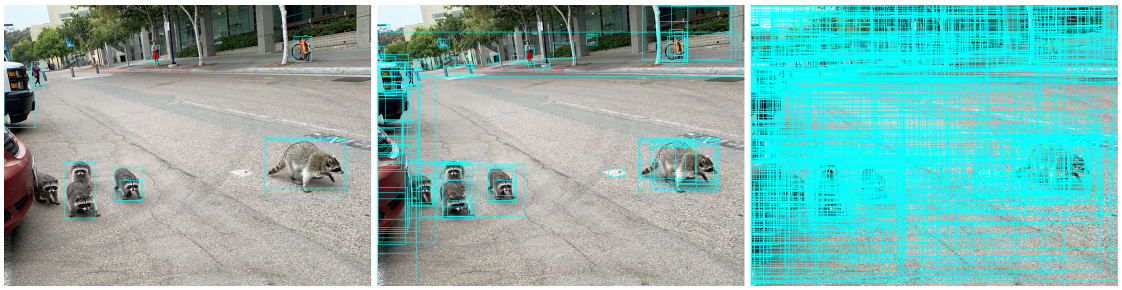

Figure 6.2: Object proposal algorithms can depend in input parameter τ, which directly controls the number of object proposals that they output: $\tau = 10$ (left), $\tau = 100$ (center), $\tau = 1000$ (right). If an object proposal algorithm is more efficient if it can recall more objects with a lower τ.

Object proposal algorithms are dependent on an input parameter that controls their number of object proposal predictions (τ); object proposals are typically ranked by their objectness confidence score to define top-τ predictions. Unique to RaccooNet's design, we take the top-τ predictions derived from the product of its objectness and IoU confidence scores, $v^{obj}v^{iou}$.

A larger τ increases the likelihood that a method will recall more objects, however it will also yield a greater number of noisy proposals (i.e., those containing no objects). When used as a component of vision pipelines, object proposal algorithms with larger τ values will require a proportional amount of filtering to remove noisy proposals, which adds computational complexity [29, 25]. Consequently, an object proposal algorithm is more efficient if it can recall more objects with a smaller τ value.

To compare the efficiency of each method, we compute recall varying $\tau = [1, 1000]$. Intuitively, $\tau = 1$ represents how well a method can recall the most salient objects [124, 13]. Conversely, $\tau = 1000$ represents how well a method can recall all objects without computational restrictions[3].

Varying τ, we computed recall via bipartite matching between ground truth labels and predictions, maximizing IoU overlap. True positives were determined using two commonly-used IoU thresholds (IoU_t): $IoU_t = 0.5$ and $IoU_t = 0.7$. $IoU_t = 0.5$ is widely accepted for general perception tasks, which represents an optimal compromise between recall and localization accuracy. $IoU_t = 0.7$ prioritizes localization accuracy over the number of recalled objects, and is often preferred in applications that require tight object boundaries (e.g., manipulation and grasping [215, 147]).

Recall vs. IoU_t, fixing number of proposals to 1000

To measure recall for the practical, computational upper limits of object proposal algorithms we fix τ so that each method outputs a maximum of 1000 proposals. We vary $IoU_t = \{0.5, 0.55, \ldots, 1.0\}$ to investigate how well each method can accurately localize objects as the IoU criterion becomes more challenging.

6.3.5 Comparison to the state-of-the-art

To compare RaccooNet to the state-of-the-art, we selected four top-performing methods that are widely used in a broad range of object perception and robot vision applications: DeepMask [165], Guided Anchoring [224], Region Proposal Network [120], and SharpMask [166] (see Figure 6.1 for a summary of their attributes). To conduct a fair evaluation, all methods were trained on the COCO 2017 training set. We show our results in Figures 6.5-6.4 and Table 6.2.

[3]Many sources do not report values for $\tau > 1000$ because it is computationally impractical.

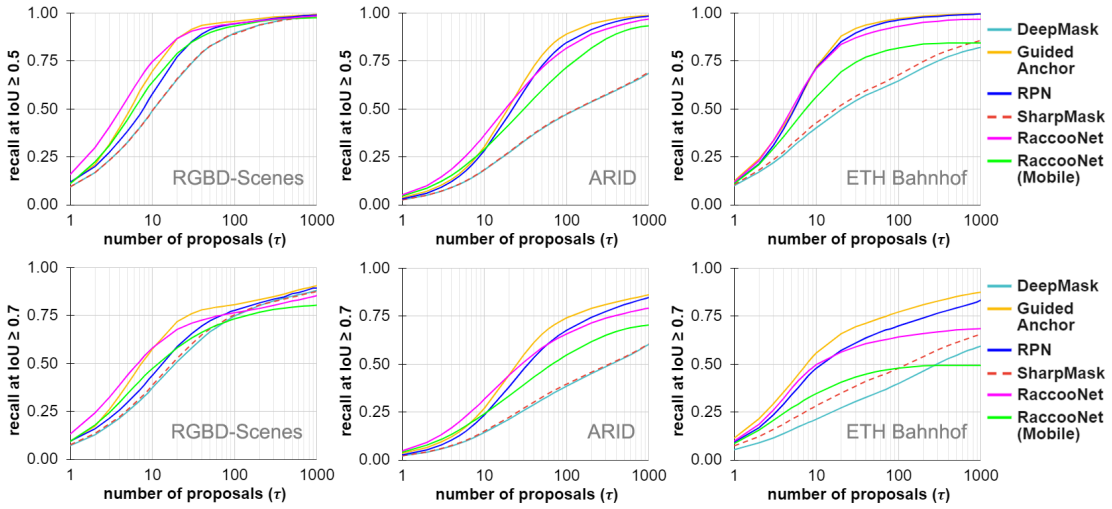

Figure 6.3: Recall at fixed thresholds $IoU_t = 0.5$ (top row) and $IoU_t = 0.7$ (bottom row) vs. number of proposals (τ) shown on logarithmic scale. Left column: RGBD-Scenes dataset (all scenes). Middle column: Autonomous Robot Indoor Dataset (all experimental sets). Right column: ETH Bahnhof. All methods were trained on the MS COCO 2017 training set. Higher recall at lower τ directly correlates to the efficiency of the method, while higher recall at $\tau = 1000$ represents the potential of a method without computational restrictions. Best viewed in color.

Table 6.2: Summary of results for robot vision datasets: computation time in frames per second (fps) and recall (R) for $IoU_t = 0.5$ w.r.t. top-τ proposals (R_τ).

			RGBD-Scenes				ARID				ETH Bahnhof			
Method	fps	fps_{TX2}	R_1	R_{10}	R_{100}	R_{1000}	R_1	R_{10}	R_{100}	R_{1000}	R_1	R_{10}	R_{100}	R_{1000}
DeepMask	3.33	-	.094	.492	.896	.984	.026	.182	.472	.686	.100	.402	.647	.822
Guided Anchoring	12.8	.766	.112	.698	**.957**	**.993**	.037	.303	**.891**	**.987**	**.124**	**.720**	**.971**	**.997**
RPN	16.8	1.09	.118	.578	.944	.990	.031	.283	.847	.984	.113	.715	.964	.996
SharpMask	2.77	-	.094	.492	.890	.985	.026	.184	.475	.692	.105	.428	.678	.858
RaccooNet	47.9	3.57	**.158**	**.747**	.946	.985	**.052**	**.364**	.818	.969	.119	.711	.931	.968
RaccooNet Mobile	**171**	**12.6**	.111	.64	.933	.976	.046	.289	.718	.934	.111	.564	.818	.844

Comparing recall performance

Comparing against the other methods, we found that RaccooNet consistently scored higher recall on the indoor datasets for $\tau \leq 10$, while scoring similarly to RPN and Guided Anchoring on the ETH Bahnhof dataset. DeepMask and SharpMask consistently scored the lowest, except in region $IoU_t = 0.7$.

We found that for the standard threshold at $IoU_t = 0.5$, RaccooNet, RPN, and Guided Anchoring scored similarly across all datasets. Despite being a much smaller

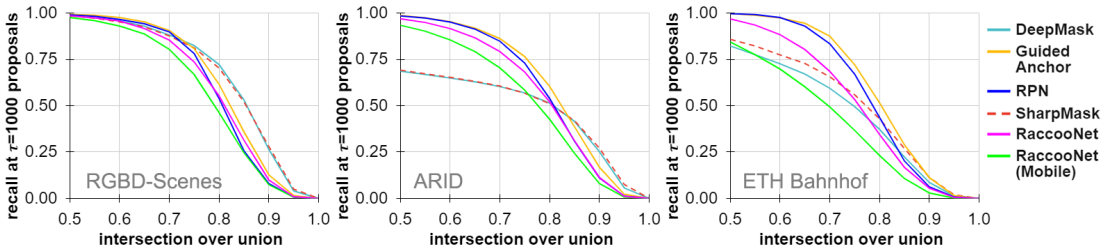

Figure 6.4: Recall at fixed number of proposals, $\tau = 1000$ vs. intersection over union threshold (IoU_t). Left: RGBD-Scenes dataset (all scenes). Middle: Autonomous Robot Indoor Dataset (all experimental sets). Right: ETH Bahnhof. Higher recall at a higher intersection over union threshold correlates to higher overlap accuracy. Best viewed in color.

model than our full-sized implementation, we found that RaccooNet Mobile achieved higher recall than DeepMask and SharpMask.

At $IoU_t = 0.7$, we found that Guided Anchoring consistently performed the best. Comparing recall measured at $IoU_t = 0.5$, we found that RaccooNet's performance suffered at $IoU_t \geq 0.7$, particularly on the ETH Bahnhof dataset. For extremely high $IoU_t \geq 0.8$ at $\tau = 1000$, we found that DeepMask and SharpMask scored the highest.

Comparing algorithm speed

We evaluated the computation time of all methods using a desktop computer with the following specifications: Intel I9 9900K CPU, 64GB RAM, and an Nvidia RTX 2080TI GPU. Comparing computation time across all experiments, we found that RaccooNet and RaccooNet mobile were the only methods capable of real-time.

To gain insight into how our method would perform on a mobile robot platform, we measured computation time on a portable embedded computing device, the Nvidia TX2, which is ubiquitous in many robotics domains [36, 232, 169]. RaccooNet Mobile achieved the highest frame rate at 12.6 fps.

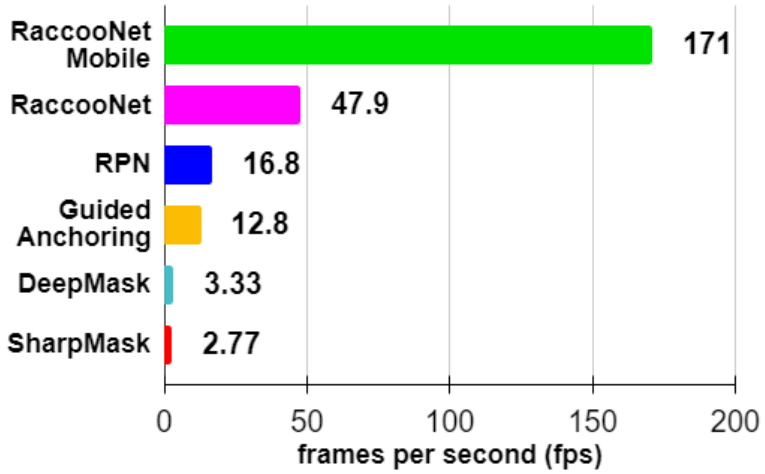

Figure 6.5: Computation time in frames per second on 640×480 RGB image, measured using a desktop computer with an I9 9900K CPU and RTX 2080Ti GPU (fps).

6.3.6 Ablation experiments

To validate the effectiveness of the IoU confidence prediction module, which is unique to RaccooNet's design, we conducted ablation experiments (results shown in Table 6.3). Specifically, we investigated the effects on recall at $IoU_t = \{0.5, 0.7\}$, removing the IoU and objectness confidence modules. To conduct our experiments, we aggregated all evaluation datasets.

Comparing the baseline to RaccooNet omitting the IoU prediction module, the baseline scores approximately 8% higher recall for $10 \geq \tau \geq 100$ at $IoU \geq 0.5$. The baseline also scores 7% higher recall for $\tau = 10$ at $IoU_t = 0.7$.

All versions of RaccooNet achieved similar recall at $\tau = 1000$. However, comparing the baseline to RaccooNet omitting the objectness confidence module, the baseline scored markedly higher for $\tau < 1000$. In summary, when either the objectness and IoU prediction modules are used alone to compute the top-τ proposals, there is notable recall degradation compared to the baseline. Thus, our ablation experiments show that both objectness and IoU confidence are critical to improving the recall performance and efficiency of RaccooNet.

Table 6.3: We aggregated all of the evaluation datasets to conduct ablation experiments on RaccooNet to study effects on recall w.r.t. objectness (Obj) and IoU confidence (IoU). Rows with Obj + IoU components represent our baseline version of RaccooNet.

Obj	IoU	*IoU* threshold	R_1	R_{10}	R_{100}	R_{1000}
x		0.5	**.089**	.468	.823	.951
	x	0.5	.054	.368	.816	**.953**
x	x	0.5	.088	**.504**	**.850**	.952
x		0.7	.075	.374	.666	**.794**
	x	0.7	.048	.313	.663	.775
x	x	0.7	**.076**	**.404**	**.668**	.774

6.4 Discussion

Among recent state-of-the-art methods, RaccooNet is approximately three times faster, being the only method capable of achieving real-time at 47.9 fps, while also achieving comparable recall at the standard threshold metric of $IoU_t = 0.5$. Compared to the other methods, we also show that in many cases, RaccooNet has superior efficiency, which could recall more objects using a smaller number of proposals ($\tau \leq 10$). This result suggests that RaccooNet may be more adept at recalling conspicuous objects, including salient objects [103, 27].

We also designed RaccooNet Mobile, running at 171 fps, is over 10 times faster than the leading object proposal algorithms. While RaccooNet Mobile makes compromises to recall performance to be substantially faster than the baseline, it still outperforms DeepMask and SharpMask across most practical metrics. It is also worth mentioning that RaccooNet Mobile was the only method that was capable of achieving near real-time performance on the Nvidia TX2, making it the only viable object proposal algorithm for resource-limited robotic platforms.

In our ablation experiments, we found that RaccooNet's IoU confidence prediction module allowed it to attain higher recall for $\tau \leq 10$, which is the most common use-case for object proposal algorithms. Our results suggest that while the objectness

score is important for ranking the top-τ proposals, IoU confidence adds considerable refinement. We believe that designing a similar module can benefit other researchers who are investigating how to build more efficient object proposal algorithms.

When designing RaccooNet, it was our intention to build an object proposal algorithm for more general robot vision applications for the commonly-accepted $IoU_t = 0.5$ metric, prioritizing speed. Thus, the primary limitation of RaccooNet was that its recall performance suffered at $IoU_t \geq 0.7$ for $\tau \geq 10$, where we did not expect our algorithm to perform as well as Guided Anchoring; however, Guided Anchoring was also the slowest among the bounding box-type methods. Nevertheless, RaccooNet was still able to achieve superior efficiency on the indoor robot datasets, scoring higher recall at $\tau \leq 10$, even at threshold $IoU_t = 0.7$. In contrast, DeepMask and SharpMask scored the highest for $\tau = 1000$ at $IoU_t \geq 0.8$, which can make them favorable for robot vision applications that require extremely precise object boundaries (e.g., object grasping and manipulation [236, 11, 42, 130]). This result was not surprising, given that DeepMask and SharpMask are segmentation algorithms, allowing them to achieve more precise object boundaries, which directly correlates to higher IoU accuracy.

To make our method practical for roboticists, we designed RaccooNet to serve as a drop-in module for existing robot vision pipelines that use object proposal algorithms [91, 44]. RaccooNet can be a powerful component for a broad range of real-time object perception tasks including discovery [153, 27, 237] and detection [245], as well as for other problems such as simultaneous localization and mapping (SLAM) [146, 115].

In our future work, we plan to use RaccooNet to solve real-world, real-time robot perception problems. Specifically, our goal is to build scalable vision systems that will enable robots to learn about unseen objects "in the wild", in an online and holistic manner similar to how humans learn. We are also interested in using our method on low-powered, computationally-limited robots for autonomous exploration in unfamiliar environments,

such space or deep sea contexts [66, 134].

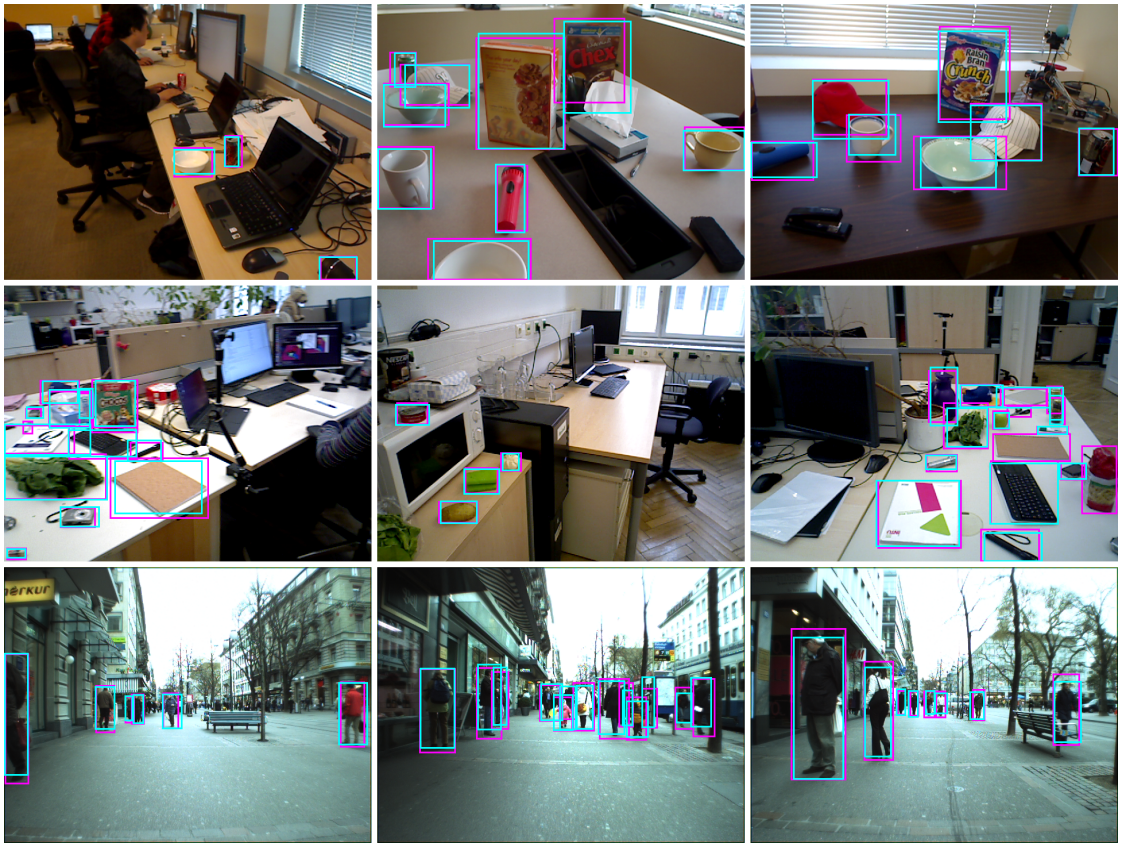

Figure 6.6: We show that RaccooNet can locate arbitrary objects despite not being explicitly trained to detect them. Sample outputs computed by RaccooNet ($\tau = 1000$) on various datasets, from top to bottom row: RGBD-scenes, ARID, and ETH Bahnhof. Magenta boxes denote ground truth objects and cyan boxes denote object proposals that best overlap with ground truth objects.

6.5 Chapter summary

In this chapter, I discussed RaccooNet, a new object proposal algorithm designed for real-time robot perception. I also introduced a novel IoU overlap confidence prediction module, which increased the efficiency of object proposal generation. I validated RaccooNet on several robot vision datasets, and on resource-constrained hardware to show that it is a viable algorithm for mobile robot applications. This algorithm has

the potential to propel a new class of real-time object perception algorithms that will further the ability for robots to solve open-world recognition challenges in unstructured environments.

6.6 Acknowledgements

This chapter contains material from "RaccooNet: Real-time Object Proposal Generation for Robot Vision", by D. M. Chan and L. D. Riek, which is currently in review for publication. The dissertation author was the primary investigator and author of this paper.

Chapter 7

Conclusion

This chapter discusses the main contributions of my doctoral research toward unseen object discovery for robot perception. It then introduces future avenues of research related to my work, followed by broader open questions in the field. Finally, this chapter closes with concluding remarks.

7.1 Contributions

7.1.1 Developed SDP, a new computationally efficient region of interest extraction algorithm for RGB-D images

Deep learning-based object detectors can be incompatible for mobile robots, particularly those that lack the specialized hardware (i.e., GPUs) to support them. In contrast, sliding window-based object detectors do not require as much computational bandwidth, making them more suitable for a broader range of mobile robot computing platforms. However, sliding window-based algorithms can still be computationally expensive, since they essentially operate using a brute-force paradigm to search each pixel in an image for potential object candidates. To address these challenges, I developed

Salient Depth Partitioning (SDP), an RGB-D-based region cropping algorithm devised to be easily adapted to existing object detection algorithms.

I designed SDP to give robots a better sense of visual attention, and to reduce the processing time of object detectors (see Chapter 3). In contrast to object proposal algorithms, SDP extracts sparse image regions, which are more robust to image degradation caused by robot motion, making them more suitable for real-world operation. Moreover, SDP requires no training, and is designed to work with any detection algorithm, provided that the input is in the form of a calibrated RGB-D image.

I validated SDP by applying it to four state-of-the-art pedestrian detectors (HOG and SVM [39], Aggregate Channel Features [50], Checkerboards [243], and R-CNN [73]), and showed that it improved computation time by up to 30%, with no discernible change in accuracy. Furthermore, I showed that SDP is compatible with smaller computing platforms commonly found on mobile robots, and was able to achieve real-time performance (77 frames per second) on a gaming laptop with a single CPU core without a GPU.

These results show that SDP can serve as a preprocessing step to alleviate the computational burden of RGB-D object detectors, which addresses a vital need to increase the efficiency of low-power mobile robots.

7.1.2 Investigated object proposal algorithms for robot perception contexts

The recent emergence of object proposal algorithms in computer vision shows great promise toward addressing difficult problems in robotics such as object discovery and salient object detection. However, it is difficult to determine how these algorithms actually perform in robot vision applications, because the standard evaluation protocol validates them on datasets which do not adequately account for real-world vision

challenges (motion blur, noise, occlusion, etc.). Thus, to gain insight into how object proposal algorithms perform in more realistic robot vision contexts, I evaluated several state-of-the-art object proposal algorithms using naturalistic datasets from the robotics community (discussed in Chapter 4).

This study highlighted substantial performance differences between the standard evaluation dataset and the robot vision datasets across all object proposal algorithms. In general, I found that object proposal algorithms recalled a larger number of objects on the standard evaluation dataset than on the robot vision datasets. This suggested that object proposal algorithms are not as generalizable as the computer vision literature purports, which can have a significant impact on how they perform in robotics domains.

I also conducted a study on how object proposal algorithms can be influenced by specific kinds of real-world robot vision challenges. I achieved this by evaluating the algorithms on a modified version of the PASCAL VOC [60] dataset, which I augmented by systematically adding variable brightness, gamma correction, Gaussian blur, and Gaussian noise. The results provided insights into the strengths and weaknesses of object proposal algorithms, which can affect how robot vision practitioners choose to deploy them.

This work enables roboticists to be better informed regarding algorithm trade-offs between computation time and recall. Moreover, this work will motivate future research about how to design more flexible and robust object proposal algorithms for the robotics community.

7.1.3 Developed a new salient object discovery method for monocular robot vision

Many state-of-the-art salient object discovery methods are limited to post-processing (i.e., offline) large video segments before discovering objects, which can

constrain how quickly robots perceive their surroundings. To that end, I designed Unsupervised Foraging of Objects (UFO), a new online approach toward unsupervised salient object discovery (discussed in Chapter 5).

In contrast to existing methods, UFO leverages object proposals to discover arbitrary objects in a frame-by-frame manner. In this way, UFO can infer arbitrary objects using only a few observation samples, making it faster than similar, competitive methods. For context, UFO can discover objects in a matter of seconds, while prior methods can take on the order of *minutes*.

UFO can simultaneously predict and track object proposals over unbounded spatiotemporal sequences to accommodate robot perception tasks. Leveraging a sliding window graph, UFO also keeps a history of prior discovered objects to make self-correcting predictions as objects change appearance over time.

I validated UFO by comparing it to the two fastest and most accurate methods for unsupervised salient object discovery (i.e., Fast Segmentation and Saliency-Aware Geodesic), and showed that UFO is 6.5 times faster, achieving state-of-the-art precision, recall, and accuracy. The results suggest that UFO is robust to real-world perception challenges encountered by robots, including moving cameras and moving objects, motion blur, and occlusion.

As a method that can be readily deployed on robots, UFO can provide a step forward toward scalable object detection frameworks that can learn on-the-fly. In particular, this work can be used to address difficult challenges in building scalable object detection frameworks for robots that can learn to recognize new objects in real-time.

7.1.4 Developed a new real-time deep learning-based object proposal algorithm to support real-time mobile robot perception

Existing object proposals are computationally expensive, making them unsuitable to deploy on resource-limited robots. Thus, I designed RaccooNet, a new efficient, dense-sampling approach toward real-time object proposals for robots (discussed in Chapter 6). RaccooNet runs at 47.9 fps, which to my knowledge, makes it the only object proposal algorithm capable of real-time performance (at time of writing) compared to state-of-the-art methods (e.g., RPN, Guided Anchoring), while also achieving comparable recall performance.

I designed a robot-optimized variant of RaccooNet, RaccooNet Mobile, which is over ten times faster than the state-of-the-art (171 fps) and is also well-suited for computationally-limited robotics applications. When deployed on the Nvidia TX2, RaccooNet Mobile was the only algorithm that was capable of near real-time performance at 12.6 fps; for context, the second fastest method was the baseline version of RaccooNet, which ran at 3.57 fps.

Unique to RaccooNet and RaccooNet Mobile, I developed a novel intersection over union overlap confidence module, which I demonstrated to improve object recall efficiency. I validated this method using a series of ablation studies, to show that it enables RaccooNet to recall more objects using a smaller number of proposals. In several cases, I show that RaccooNet can recall more objects than the leading object proposal algorithms in the literature.

I validated RaccooNet and RaccooNet Mobile on three real-world robot vision datasets to simulate indoor and outdoor environments, including RGB-D-scenes, ARID, and ETH Bahnhof, and showed that our method is robust to a variety of robot vision contexts and challenges (e.g., blur, motion, lighting, object scale).

By introducing this new object proposal algorithm that is both fast and robust, it is my intent that this research will propel a new class of real-time vision algorithms that will further the ability for robots to solve open-world recognition challenges in unstructured environments.

7.1.5 Evaluation of object perception methods on resource-constrained hardware

Prior to my work, few researchers reported the computation time of object perception methods which made it difficult to gauge tradeoffs between speed and performance for robotics applications. Moreover, computation time is often reported for high-performance workstations or GPU clusters, which are not representative of typical robotics hardware. To address this research gap, I performed runtime analysis of various object perception methods on mobile computing platforms to simulate their performance on mobile robots.

In Chapters 3-5, all of my experiments were conducted on a portable laptop computer to conduct runtime analysis. To my knowledge, I am also the first to evaluate object proposal algorithms on mobile hardware. In Chapter 6, I conducted experiments on a small embedded device, i.e., Nvidia TX2, and demonstrated that RaccooNet is a suitable algorithm for mobile robots.

It is my ambition that my approach toward evaluating object perception methods on resource-constrained computing platforms will be foundational to how new methods will be designed and validated for real-world, real-time robotic platforms.

7.2 Future work

7.2.1 End-to-end deep-learning for object discovery

In Chapter 5, I introduced UFO, a new method that enables robots to automatically discover arbitrary, salient objects in unbounded image sequences. UFO is composed of a mixture of deep learning and traditional algorithmic components which consist of: an object proposal algorithm, a non-maximum suppression algorithm, a saliency algorithm, a convolutional neural network feature extractor, and a sliding window graph. One limitation of this approach was that object proposal generation followed by feature extraction can be computationally prohibitive for real-time robot vision. In contrast, an end-to-end deep learning approach can perform these algorithmic processes in a unified neural network, where features can be shared to eliminate computational redundancy. This research motivated my current and future exploration into designing new object discovery methods using an end-to-end deep learning approach.

In Chapter 6, I introduced RaccooNet, a new deep learning-based object proposal algorithm, which is significantly faster than previous methods, and, to my knowledge, is the only method capable of real-time performance on mobile computing platforms. Combining ideas from UFO, this work lays the foundation for formulating temporal models (e.g., recurrent neural network) that can perform salient object discovery in an online and real-time manner. Moreover, twin neural networks [79, 84] can be a promising approach to build on top of this work, which can be used to build more robust spatiotemporal object proposals. These expansions to my current work can enable multiple objects to be simultaneously discovered in a manner that is suitable for real-time robot vision.

7.2.2 Synthesizing known and unseen objects

Before robots can become intelligent agents in open-world environments, they need to understand the notion of known and unknown objects. Specifically, robots need the ability to differentiate recognizable objects from those that are novel, so that they can begin to understand how to learn from their surroundings. For example, if a robot were to have this ability, their perception can be used to bootstrap their curiosity and sentience, so that they can learn in a holistic way similar to how humans learn.

While object detection addresses perception of known objects, and object discovery addresses perception of unknown objects, they are treated as separate research problems. Consequently, there is very little work that attempts to unify them.

My work lays the foundation toward a new system that can simultaneously detect and discover objects. In particular, the methods in Chapters 5 and 6 can be modified to include a classification network to perform object detection. This is the logical next step in my line of research because the methods that I designed can be readily adapted to detect from a database of known objects using transfer learning techniques [157, 217]

My research can also be used to develop a system that can automatically append newly discovered objects to a database of known objects. Such a system can automate the process of annotating training datasets for vision algorithms, which is otherwise time consuming and expensive for humans to manually implement. By extension, this can have the potential to enable robots and intelligent systems to learn from observation, which will allow them to learn as they are deployed in real-world environments.

7.2.3 Spatiotemporal non-maximum suppression

Due to their feed-forward nature, object perception algorithms generate redundant false positives that can lead to adverse ramifications for intelligent systems. For

example, false positives can cause a robot to perceive objects that are not actually present. Conversely, removing false positives in a naive manner (e.g., thresholding) can inadvertently remove true positives to yield detrimental effects. This can cause a robot to not perceive objects that are actually present. Consequently, false positives can ultimately yield negative effects on robot decision-making and behavior.

Object perception methods typically employ a non-maximum suppression (NMS) algorithm to remove redundant false positives. Briefly described in Chapter 5, the de facto NMS algorithm follows a greedy approach (i.e., greedy-NMS [148]), which seeks to select the highest scoring sample among distributions of neighbors. While NMS is a critical component for vision pipelines, there has been very little focus to make improvements to the greedy-NMS paradigm [187, 89, 90].

In Chapter 5, I showed that greedy-NMS can be a suboptimal approach to mitigating redundant object proposals. Thus, I augmented the greedy-NMS algorithm with the notion of saliency density, which improved recall performance. However, one limitation to this approach is that it can still add considerable computational complexity to hinder its adoption in real-time object discovery applications.

Leveraging spatiotemporal constraints can possibly lead to a faster NMS algorithm for object discovery. If a system were to have a way of tracking object proposals over time, it could more efficiently perform NMS using data association to determine a more optimal distribution of neighbors. This could serve to narrow the search margin for NMS neighbors, which will improve computation time. Additionally, the *IoU* confidence prediction method that I introduced in Chapter 6 could be used in tandem with this approach to rescore proposals, leading to higher recall for object discovery.

7.3 Open questions

7.3.1 How can we redefine objects?

My current research explores how to develop computational models for object discovery using the following three axioms from Chapter 2:

Axiom 1 An object has closed boundaries [248, 114].

Axiom 2 An object must differ in appearance from its surroundings [124, 131].

Axiom 3 An object sometimes has a unique appearance that is salient [19, 65].

While these axioms can be used to solve today's research challenges, they will eventually need to evolve as the field progresses. In particular, more research is needed to address perceiving objects that lie outside of these axioms, for example, partially occluded objects, reflective objects, liquids, and fabric materials. In computer vision research, there is also little consensus about whether or not *whole objects* should be regarded as the sum of its individual object parts. Because there are no precise definitions to describe a broader definition of objects, object perception has many uncharted areas of research.

Despite recent efforts to define objects, many researchers are still designing object detection methods around the notion of predefined objects, which limits them from reaching broader object perception domains. Thus, it is important to develop a more standardized, broader definition of objects that will allow researchers to work toward a common goal. If researchers were to adopt a universal definition of objects, researchers can develop object perception methods that have standardized specifications so that they can be more seamlessly used together in robotics applications.

7.3.2 How can we develop context-aware methods for object discovery?

Context dictates how robots understand their surroundings, and ultimately controls how they function as intelligent agents. With respect to object perception, context can provide robots with cues about how to navigate their environments safely around people, while also giving them a sense of where to find objects relevant to their tasks.

However, one limitation with existing computational models is that context is treated with *a priori* assumptions, i.e., that their environments are deterministic, rigid, and static [180, 149]. Moreover, the primary focus of current object perception research is to infer low-level context from the viewpoint of traditional computer vision challenges (i.e., saliency and scale from single images). In reality, context is a much broader problem, where environments are prone to constantly evolve as the people, objects, activities, and tasks change over time. Many object perception methods are unable to adapt to these kinds of dynamic environments, which ultimately constrains robot intelligence.

As an emerging research topic, questions about how to incorporate context into object perception processes will become increasingly important for robotics. For example, computer vision models can address bottom-up (e.g., low-level visual cues) and top-down context (e.g., tasks and activities) as separate problems, but there is little work to develop new computational models that can synthesize them. Moreover, it is also unclear how we can develop computational mechanisms that can decide whether bottom-up or top-level processes should take higher priority when making decisions. To highlight these differences, bottom-up perception is important when robots are engaging in exploratory tasks; for example, with the absence of information, a robot might use saliency cues to determine what is important in a new environment. In contrast, top-down perception is important when robots need to make decisions about how to solve well-defined tasks; for

example, they might use object detection to find a task-specific object. However, in many situations, a robot might need to use both top-down and bottom-up processes in tandem; for example, they might need to identify an object based on some vague description.

Understanding how bottom-up and top-down perception processes work together is still largely a challenge in the neuroscience literature [46, 4]. Thus, extensive research still needs to be conducted about how neurological processes distribute priority between bottom-up and top-down perception processes before we can model this kind of behavior for robots.

7.3.3 How can we design robots to learn from curiosity?

While the goal of my dissertation is to develop computer vision methods that can enable robots to discover unseen objects, there are many areas of research that can stem from my work. For example, object discovery can bootstrap curiosity for robots, which can be used as visual feedback for active vision control systems [3, 57]. Object discovery can also be used in tandem with haptics, to enable robots to interact with novel objects and learn about their properties (i.e., affordances [70]).

If robots had the ability to be curious, to seek novel objects and interact with them, they can be used to gather data in unconstrained environments. This can enable their object recognition models and perception systems to learn on-the-fly, and in real-time. This will ultimately enable robots to become more seamlessly integrated into real-world and open-world environments.

However, enabling robots to be curious has major challenges, which can closely relate to the problem of context. For example, more research needs to be conducted to determine when a robot should be doing something productive (i.e., completing a set of assigned tasks), and when it should be engaging in more open-ended curiosity-driven tasks (i.e., exploration and learning).

Moreover, it is not only important to understand *when* robots should be curious, but also *how* they should be curious. In particular, robots need to understand how to interact and learn about new objects in a way that is appropriate to humans. For example, robots needs to consider when objects are currently in use by a human, which should either be avoided (i.e., to prevent interrupting the human) or encouraged (i.e., the human is trying to teach the robot [83, 192]).

7.4 Closing remarks

Summarizing this dissertation, the central theme of my doctoral research explores how to design computational models for object perception, toward more intelligent robots that can operate in the wild. The ultimate goal of my work is to design machines that will eventually learn from their surroundings in an automatic and holistic manner, similar to how humans learn.

Before robots are ready to transition to open-world environments where they will be expected to operate autonomously among humans, their perception will need to be robust to uncertainty. While there are many facets to this problem, my work lays the foundation for enabling robots to perceive novel or ambiguous objects. Specifically, the contributions of my work are the introduction of new algorithms and systems that enable robots to discover unseen objects in a manner that is both fast and robust to real-world vision challenges. My research also highlights several future research directions that will need to be addressed before robust perception can be truly attainable.

It is my hope that this research will lead to new algorithms that will allow robots to solve open-world challenges in unstructured environments, so that they will become the intelligent machines that we have long envisioned.

Appendix A

Glossary

activation function A mathematical function of a perceptron that maps numerical inputs to a new numerical output space. 23

affordance The possible ways that an object can be interacted with. 13

back propegation A process of training a convolutional neural network, where its parameters are iteratively adjusted to optimize a loss function. 26

backbone A pre-built convolutional neural network that is re-purposed to solve another computer vision problem. 91

background Entities in an image that do not provide significant meaning, including those that do not correlate to objects. This can also mean entities that are not foreground elements. 12

bounding box A box that forms the rectangular perimeter of an object in an image. 18

bounding box regression A computer vision technique where regression is employed to predict a set of rectangular coordinates. 24

class In object perception, this pertains to a set of objects that are grouped by similar appearance or function. More generically, it can also mean a category of entities that share one or more properties or attributes. 14

classification A computer vision process that determines how to associate data (e.g., object features) with classes. More generically, a machine learning process where an algorithm makes a discrete variable prediction to assign a piece of data to a known set. 14, 24

context How computer and robot vision systems should perceive objects with respect to their surroundings. 12

convolutional layer A topological structure of convolutional neural networks that consists of one or more learned convolutional kernels. 24

convolutional neural network A type of neural network architecture that consists of one or more learned convolutional layers. 22

dataset A collection of images that are used to train and evaluate object perception algorithms. 26

feature Quantized patterns extracted from images that are used to solve machine learning problems. 20

feature extraction A computer vision task that involves converting images or image regions to features. 20

feature pyramid network A network extension of a CNN that concatenates features from various depths. 25

feature template A type of convolution kernel that contains appearance-based features that correspond to a specific kind of object. 21

flattened feature A one-dimensional feature that has been converted from a multi-dimensional space. 24

foreground Entities in an image that potentially provide significant meaning, including those that correlate to objects. This can also refer to entities that are not background elements. 13

fully convolutional network A type of convolutional neural network that does not contain any fully-connected layers. 24

fully-connected layer A topological structure of neural networks in which every perceptron from one layer connects to every perceptron in another layer. 23

gradient descent An iterative optimization algorithm that computes the first-order derivative of a loss function to estimate its local minimum as the parameters of a machine learning algorithm are updated. 26

hand-crafted feature An image feature derived explicitly from mathematical formulation, rather than from data. 21

hyperparameter An independent variable that influences how a machine learning algorithm performs and learns. 26

image patch Any small rectangular region in an image. 21

image pyramid A multi-scale image representation of an image that involves a linear combination of upsampling and/or downsampling. 22

inference A machine learning process where an algorithm makes a prediction on a new piece of data. 24

instance Referring to any one individual object or entity. 69

intersection over union A qualitative metric that describes the quality of overlap between two bounding boxes; also called the Jaccard Index. 27

kernel A matrix consisting of specialized values that are used in conjunction with convolution to extract atomic features (e.g., edges, corners) from images. A kernel is synonymous with a convolution filter. 20

keypoint Small image regions that are robust to illumination, scale, and rotation (e.g., corner points). 21

label Information given to images that characterizes what an object perception algorithm should learn. 26

localization A computer vision process that determines the locations of objects in images. 14

loss A quantitative representation of error that represents cost. 26

loss function A function that quantifies the error between the actual output of a machine learning algorithm and its expected output. 26

object detection A computer vision process that performs localization and classification to determine the presence of class-specific objects in images; a computer vision algorithm or system that performs object detection is sometimes called an object detection algorithm or object detector. 14

object discovery A computer vision process that determines the presence and locations of unseen objects. 16

object proposal algorithm A computer vision algorithm that extracts image regions containing possible objects. 18

one-shot learning A machine learning task that describes an algorithm's ability to make future predictions given only one ground truth example. 16

one-stage detector An object detector that performs localization and classification simultaneously. One-stage detectors typically consist of convolutional neural networks that perform bounding box regression and classification. 15

optimizer An algorithm that employs a learning strategy to train neural networks. 26

perceptron An atomic unit of a neural network that behaves like a binary classifier. Perceptrons consist of weights to control the sensitivity of inputs, and an activation function that maps its inputs to a new output space. 23

receptive field The spatial relationship between the size of an image and the input size of a convolutional kernel. 24

regression A machine learning process where an algorithm makes a continuous variable prediction. 24

RGB-D An imaging modality that includes color (i.e., red, green, and blue) and depth. 6

saliency A confidence interval that correlates to the likelihood that regions or pixels in an image attract human attention; the process by which a computer or robot vision algorithm computes saliency is called saliency estimation. 14

saliency estimation A computer vision process that determines the saliency of individual pixels in an image. The output of a saliency estimation algorithm is sometimes called a saliency map. 14

salient object discovery A computer vision process that determines the locations of objects that are likely to attract human attention. 68

scale How large an object appears in an image relative to its distance from the camera. In the computer vision literature this can also relate to the problem of how objects should be perceived relative to their size in the image. 12

segmentation mask A set of pixels that intersect with the area or volume of an object in an image. 18

sliding window method An algorithmic approach in computer vision that involves convolving a feature template across an image and performing classification across all positions and scales. 22

training The process by which a machine learning algorithm adjusts its internal parameters to build a model that maps inputs to outputs based on exemplified data. 26

training example One image-label pair from training data. 26

two-stage detector An object detector that performs localization and classification as two separate processes. In two-stage detectors, classification is performed after localization. 15

unseen A property that describes objects that are unknown to a computer or robot vision system. Unseen objects are those that have not been explicitly seen in training, or does not belong to any predefined object class. 16

Bibliography

[1] G. Alenyà, S. Foix, and C. Torras. Using tof and rgbd cameras for 3d robot perception and manipulation in human environments. *Intelligent Service Robotics*, 7(4):211–220, 2014.

[2] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *IEEE computer society conference on computer vision and pattern recognition (CVPR)*, pages 73–80. IEEE, 2010.

[3] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[4] G. Baghdadi, F. Towhidkhah, and R. Rostami. A mathematical model of the interaction between bottom-up and top-down attention controllers in response to a target and a distractor in human beings. *Cognitive Systems Research*, 58:234–252, 2019.

[5] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[6] M. Bar. Visual objects in context. *Nature Reviews Neuroscience*, 5(8):617–629, 2004.

[7] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European conference on computer vision (ECCV)*, pages 404–417. Springer, 2006.

[8] A. Bieniek and A. Moga. A connected component approach to the watershed segmentation. *Computational Imaging and Vision*, 12:215–222, 1998.

[9] Bikingdog. Samples of object co-segmentation. `https://upload.wikimedia.org/wikipedia/commons/6/6b/Samples_of_object_co-segmentation.jpg`.

[10] H. Blum, A. Gawel, R. Siegwart, and C. Cadena. Modular sensor fusion for semantic segmentation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3670–3677. IEEE, 2018.

[11] C. Bodnar, A. Li, K. Hausman, P. Pastor, and M. Kalakrishnan. Quantile qt-opt for risk-aware vision-based robotic grasping. Robotics: science and systems (RSS), 2019.

[12] N. Bore, R. Ambrus, P. Jensfelt, and J. Folkesson. Efficient retrieval of arbitrary objects from long-term robot observations. *Robotics and Autonomous Systems*, 91:139–150, 2017.

[13] A. Borji, M.-M. Cheng, H. Jiang, and J. Li. Salient object detection: A benchmark. *IEEE transactions on image processing*, 24(12):5706–5722, 2015.

[14] L. Bose and A. Richards. Fast depth edge detection and edge based rgb-d slam. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1323–1330. IEEE, 2016.

[15] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT*, pages 177–186. Springer, 2010.

[16] M. Braham, S. Piérard, and M. Van Droogenbroeck. Semantic background subtraction. In *IEEE International Conference on Image Processing (ICIP)*, pages 4552–4556. IEEE, 2017.

[17] C. Breazeal. Emotion and sociable humanoid robots. *International journal of human-computer studies*, 59(1-2):119–155, 2003.

[18] R. Brehar and S. Nedevschi. Scan window based pedestrian recognition methods improvement by search space and scale reduction. In *IEEE Intelligent Vehicles Symposium Proceedings*, pages 529–534. IEEE, 2014.

[19] N. Bruce and J. Tsotsos. Saliency based on information maximization. In *Advances in neural information processing systems (NIPS)*, pages 155–162, 2005.

[20] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 221–230, 2017.

[21] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *European conference on computer vision (ECCV)*, pages 778–792. Springer, 2010.

[22] M. Camplani, T. Mantecon, and L. Salgado. Depth-color fusion strategy for 3-d scene modeling with kinect. *IEEE transactions on cybernetics*, 43(6):1560–1571, 2013.

[23] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, (6):679–698, 1986.

[24] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3241–3248. IEEE, 2010.

[25] D. M. Chan and L. D. Riek. Object proposal algorithms in the wild: Are they generalizable to robot perception? In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2601–2607. IEEE, 2019.

[26] D. M. Chan and L. D. Riek. Unsupervised salient object discovery for robots. In *Robotics: Science and Systems (RSS) Pioneers*, 2019.

[27] D. M. Chan and L. D. Riek. Unseen salient object discovery for monocular robot vision. *IEEE Robotics and Automation Letters (RAL)*, 5(2):1484–1491, 2020.

[28] D. M. Chan, A. Taylor, and L. D. Riek. Faster robot perception using salient depth partitioning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4152–4158. IEEE, 2017.

[29] N. Chavali, H. Agrawal, A. Mahendru, and D. Batra. Object-proposal evaluation protocol is 'gameable'. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 835–844, 2016.

[30] G. Chen, W. Choi, X. Yu, T. Han, and M. Chandraker. Learning efficient object detection models with knowledge distillation. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 742–751, 2017.

[31] K. Chen, H. Song, C. Change Loy, and D. Lin. Discover and learn new objects from documentaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3087–3096, 2017.

[32] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1907–1915, 2017.

[33] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford. Deep learning features at scale for visual place recognition. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3223–3230. IEEE, 2017.

[34] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, Z. Liu, H. I. Christensen, and F. Dellaert. Multi robot object-based slam. In *International Symposium on Experimental Robotics*, pages 729–741. Springer, 2016.

[35] R. G. Cinbis, J. Verbeek, and C. Schmid. Weakly supervised object localization with multi-fold multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 39(1):189–203, 2016.

[36] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy. End-to-end driving via conditional imitation learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4693–4700. IEEE, 2018.

[37] G. B. Coleman and H. C. Andrews. Image segmentation by clustering. *Proceedings of the IEEE*, 67(5):773–785, 1979.

[38] R. T. Collins, A. J. Lipton, and T. Kanade. Introduction to the special section on video surveillance. *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, 22(8):745–746, 2000.

[39] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE computer society conference on computer vision and pattern recognition (CVPR)*, volume 1, pages 886–893. Ieee, 2005.

[40] T. Dang, C. Papachristos, and K. Alexis. Visual saliency-aware receding horizon autonomous exploration with application to aerial robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2526–2533. IEEE, 2018.

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on computer vision and pattern recognition (CVPR)*, pages 248–255. IEEE, 2009.

[42] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox. Self-supervised 6d object pose estimation for robot manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3665–3671. IEEE, 2020.

[43] M. Denninger and R. Triebel. Persistent anytime learning of objects from unseen classes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4075–4082. IEEE, 2018.

[44] C. Devin, P. Abbeel, T. Darrell, and S. Levine. Deep object-centric representations for generalizable robot learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7111–7118. IEEE, 2018.

[45] M.-W. Dictionary. Merriam-webster. *On-line at http://www. mw. com/home. htm*, 2002.

[46] N. Dijkstra, P. Zeidman, S. Ondobaka, M. A. van Gerven, and K. Friston. Distinct top-down and bottom-up brain connectivity during visual perception and imagery. *Scientific reports*, 7(1):1–9, 2017.

[47] T.-T. Do, A. Nguyen, and I. Reid. Affordancenet: An end-to-end deep learning approach for object affordance detection. In *IEEE international conference on robotics and automation (ICRA)*, pages 5882–5889. IEEE, 2018.

[48] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 304–311. IEEE, 2009.

[49] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 1841–1848, 2013.

[50] P. Dollár and C. L. Zitnick. Fast edge detection using structured forests. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 37(8):1558–1570, 2014.

[51] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 370–386, 2018.

[52] I. Endres and D. Hoiem. Category independent object proposals. In *European Conference on Computer Vision (ECCV)*, pages 575–588. Springer, 2010.

[53] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision (ECCV)*, pages 834–849. Springer, 2014.

[54] M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 31(12):2179–2195, 2008.

[55] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[56] B. A. Erol, A. Majumdar, J. Lwowski, P. Benavidez, P. Rad, and M. Jamshidi. Improved deep neural network object tracking system for applications in home robotics. In *Computational Intelligence for Pattern Recognition*, pages 369–395. Springer, 2018.

[57] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, 1992.

[58] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[59] A. Ess, B. Leibe, and L. Van Gool. Depth and appearance for mobile scene analysis. In *International conference on computer vision (ICCV)*, pages 1–8. IEEE, 2007.

[60] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[61] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 32(9):1627–1645, 2009.

[62] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[63] M. Gabb, O. Löhlein, R. Wagner, A. Westenberger, M. Fritzsche, and K. Dietmayer. High-performance on-road vehicle detection in monocular images. In *International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 336–341. IEEE, 2013.

[64] C. Galleguillos and S. Belongie. Context based object categorization: A critical survey. *Computer vision and image understanding*, 114(6):712–722, 2010.

[65] D. Gao and N. Vasconcelos. Bottom-up saliency is a discriminant process. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–6. IEEE, 2007.

[66] Y. Gao and S. Chien. Review on space robotics: Toward top-level science through space exploration. *Science Robotics*, 2(7), 2017.

[67] S. Garg, N. Suenderhauf, and M. Milford. Lost? appearance-invariant place recognition for opposite viewpoints using visual semantics. In *Robotics: Science and Systems (RSS)*, 2018.

[68] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 32(7):1239–1258, 2009.

[69] M. Ghafarianzadeh, M. B. Blaschko, and G. Sibley. Efficient, dense, object-based segmentation from rgbd video. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2310–2317. IEEE, 2016.

[70] J. J. Gibson. The theory of affordances. *Hilldale, USA*, 1(2):67–82, 1977.

[71] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 1440–1448, 2015.

[72] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 580–587, 2014.

[73] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 38(1):142–158, 2015.

[74] G. Gkioxari, R. Girshick, P. Dollár, and K. He. Detecting and recognizing human-object interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8359–8367, 2018.

[75] D. Gordon, A. Farhadi, and D. Fox. Re^3: Real-time recurrent regression networks for visual tracking of generic objects. *IEEE Robotics and Automation Letters (RAL)*, 3(2):788–795, 2018.

[76] H.-M. Gross, S. Meyer, A. Scheidig, M. Eisenbach, S. Mueller, T. Q. Trinh, T. Wengefeld, A. Bley, C. Martin, and C. Fricke. Mobile robot companion for walking training of stroke patients in clinical post-stroke rehabilitation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1028–1035. IEEE, 2017.

[77] Z. Gu, J. Cheng, H. Fu, K. Zhou, H. Hao, Y. Zhao, T. Zhang, S. Gao, and J. Liu. Ce-net: Context encoder network for 2d medical image segmentation. *IEEE transactions on medical imaging*, 38(10):2281–2292, 2019.

[78] C. Guan, A. Bouzida, R. Oncy-Avila, S. Moharana, and L. D. Riek. Taking an (embodied) cue from community health: Designing dementia caregiver support technology to advance health equity. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–24, 2021.

[79] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang. Learning dynamic siamese network for visual object tracking. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 1763–1771, 2017.

[80] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision*, 112(2):133–149, 2015.

[81] R. M. Haralick and L. G. Shapiro. Image segmentation techniques. *Computer vision, graphics, and image processing*, 29(1):100–132, 1985.

[82] C. G. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[83] C. J. Hayes, M. Moosaei, and L. D. Riek. Exploring implicit human responses to robot mistakes in a learning from demonstration task. In *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 246–252. IEEE, 2016.

[84] A. He, C. Luo, X. Tian, and W. Zeng. A twofold siamese network for real-time object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4834–4843, 2018.

[85] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision (CVPR)*, pages 2961–2969, 2017.

[86] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 770–778, 2016.

[87] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 38(4):814–830, 2015.

[88] J. Hosang, R. Benenson, and B. Schiele. How good are detection proposals, really? *arXiv preprint arXiv:1406.6962*, 2014.

[89] J. Hosang, R. Benenson, and B. Schiele. A convnet for non-maximum suppression. In *German Conference on Pattern Recognition*, pages 192–204. Springer, 2016.

[90] J. Hosang, R. Benenson, and B. Schiele. Learning non-maximum suppression. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 4507–4515, 2017.

[91] Y. Hou, H. Zhang, and S. Zhou. Evaluation of object proposals and convnet features for landmark-based visual place recognition. *Journal of Intelligent & Robotic Systems*, 92(3):505–520, 2018.

[92] F. Huang, J. Qi, H. Lu, L. Zhang, and X. Ruan. Salient object detection via multiple instance learning. *IEEE Transactions on Image Processing*, 26(4):1911–1922, 2017.

[93] IBM. Ibm thinklab aging in place environment: Softbank pepper robots. `https://www.flickr.com/photos/40748696@N07/30654715923`.

[94] T. Iqbal, S. Rack, and L. D. Riek. Movement coordination in human–robot teams: a dynamical systems approach. *IEEE Transactions on Robotics (TRO)*, 32(4):909–919, 2016.

[95] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, 20(11):1254–1259, 1998.

[96] V. Jampani, R. Gadde, and P. V. Gehler. Video propagation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 451–461, 2017.

[97] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3d object dataset: Putting the kinect to work. In *Consumer depth cameras for computer vision*, pages 141–165. Springer, 2013.

[98] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018.

[99] R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987.

[100] M. Khodabandeh, A. Vahdat, M. Ranjbar, and W. G. Macready. A robust learning approach to domain adaptive object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 480–490, 2019.

[101] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for object tracking. In *The DAVIS challenge on video object segmentation*, 2017.

[102] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[103] D. A. Klein, B. Illing, B. Gaspers, D. Schulz, and A. B. Cremers. Hierarchical salient object detection for assisted grasping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2230–2237. IEEE, 2017.

[104] D. Kochanov, A. Ošep, J. Stückler, and B. Leibe. Scene flow propagation for semantic mapping and object discovery in dynamic street scenes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1785–1792. IEEE, 2016.

[105] H. S. Koppula and A. Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 38(1):14–29, 2015.

[106] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *European conference on computer vision (ECCV)*, pages 725–739. Springer, 2014.

[107] P. Krahenbuhl and V. Koltun. Learning to propose objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1574–1582, 2015.

[108] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems (NIPS)*, 25:1097–1105, 2012.

[109] A. Kubota, E. I. Peterson, V. Rajendren, H. Kress-Gazit, and L. D. Riek. Jessie: Synthesizing social robot behaviors for personalized neurorehabilitation and beyond. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 121–130, 2020.

[110] A. Kubota and L. D. Riek. Methods for robot behavior adaptation for cognitive neurorehabilitation. *Annual Review of Control, Robotics, and Autonomous Systems*, 5, 2021.

[111] W. Kuo, B. Hariharan, and J. Malik. Deepbox: Learning objectness with convolutional networks. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2479–2487, 2015.

[112] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3050–3057. IEEE, 2014.

[113] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *IEEE international conference on robotics and automation (ICRA)*, pages 1817–1824. IEEE, 2011.

[114] L. J. Latecki, R. Lakamper, and T. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 424–429. IEEE, 2000.

[115] F. Leeb, A. Byravan, and D. Fox. Motion-nets: 6d tracking of unknown objects in unseen environments using rgb. *arXiv preprint arXiv:1910.13942*, 2019.

[116] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research (IJRR)*, 37(4-5):421–436, 2018.

[117] C. Li, H. Xiao, K. Tateno, F. Tombari, N. Navab, and G. D. Hager. Incremental scene understanding on dense slam. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 574–581. IEEE, 2016.

[118] L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2036–2043. IEEE, 2009.

[119] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[120] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[121] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2980–2988, 2017.

[122] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV)*, pages 740–755. Springer, 2014.

[123] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[124] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H. Shum. Learning to detect a salient object. *IEEE Transactions on Pattern analysis and machine intelligence (TPAMI)*, 33(2):353–367, 2010.

[125] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision (ECCV)*, pages 21–37. Springer, 2016.

[126] M. R. Loghmani, B. Caputo, and M. Vincze. Recognizing objects in-the-wild: Where do we stand? In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2170–2177. IEEE, 2018.

[127] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 3431–3440, 2015.

[128] D. G. Lowe. Object recognition from local scale-invariant features. In *IEEE international conference on computer vision (ICCV)*, volume 2, pages 1150–1157. IEEE, 1999.

[129] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[130] Q. Lu, N. Baron, A. Clark, and N. Rojas. The ruth gripper: systematic object-invariant prehensile in-hand manipulation via reconfigurable underactuation. Robotics: science and systems (RSS).

[131] Y.-F. Ma and H.-J. Zhang. Contrast-based image attention analysis by using fuzzy growing. In *Proceedings of the ACM international conference on Multimedia*, pages 374–381. ACM, 2003.

[132] L. Maddalena and A. Petrosino. Background subtraction for moving object detection in rgbd data: A survey. *Journal of Imaging*, 4(5):71, 2018.

[133] J. Mahler and K. Goldberg. Learning deep policies for robot bin picking by simulating robust grasping sequences. In *Conference on robot learning (CoRL)*, pages 515–524. PMLR, 2017.

[134] T. Manderson, J. C. G. Higuera, S. Wapnick, J. Tremblay, F. Shkurti, D. Meger, and G. Dudek. Vision-based goal-conditioned policies for underwater navigation in the presence of obstacles. Robotics: science and systems (RSS), 2020.

[135] S. Manen, M. Guillaumin, and L. Van Gool. Prime object proposals with randomized prim's algorithm. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2536–2543, 2013.

[136] O. Mendez, S. Hadfield, N. Pugeault, and R. Bowden. Sedar-semantic detection and ranging: Humans can localise without lidar, can robots? In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6053–6060. IEEE, 2018.

[137] R.-G. Mihalyi, K. Pathak, N. Vaskevicius, and A. Birk. Uncertainty estimation of ar-marker poses for graph-slam optimization in 3d object model generation with rgbd data. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1807–1813. IEEE, 2013.

[138] A. Milan, T. Pham, K. Vijay, D. Morrison, A. W. Tow, L. Liu, J. Erskine, R. Grinover, A. Gurman, T. Hunn, N. Kelly-Boxall, D. Lee, M. McTaggart, G. Rallos, A. Razjigaev, T. Rowntree, T. Shen, R. Smith, S. Wade-McCue, Z. Zhuang, C. Lehnert, G. Lin, I. Reid, P. Corke, and J. Leitner. Semantic segmentation from limited training data. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1908–1915. IEEE, 2018.

[139] M. J. Milford and G. F. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *IEEE international conference on robotics and automation (ICRA)*, pages 1643–1649. IEEE, 2012.

[140] M. J. Milford, G. F. Wyeth, and D. Prasser. Ratslam: a hippocampal model for simultaneous localization and mapping. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings (ICRA)*, volume 1, pages 403–408. IEEE, 2004.

[141] D. Miller, L. Nicholson, F. Dayoub, and N. Sünderhauf. Dropout sampling for robust object detection in open-set conditions. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3243–3249. IEEE, 2018.

[142] J. Mišeikis, P. Caroni, P. Duchamp, A. Gasser, R. Marko, N. Mišeikienė, F. Zwilling, C. de Castelbajac, L. Eicher, M. Früh, and H. Früh. Lio-a personal robot assistant for human-robot interaction and care applications. *IEEE Robotics and Automation Letters (RAL)*, 5(4):5339–5346, 2020.

[143] S. Moharana, A. E. Panduro, H. R. Lee, and L. D. Riek. Robots for joy, robots for sorrow: community based robot design for dementia caregivers. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 458–467. IEEE, 2019.

[144] M. Moosaei, M. Pourebadi, and L. D. Riek. Modeling and synthesizing idiopathic facial paralysis. In *IEEE International Conference on Automatic Face & Gesture Recognition (FG)*, pages 1–8. IEEE, 2019.

[145] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics (TRO)*, 31(5):1147–1163, 2015.

[146] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics (TRO)*, 33(5):1255–1262, 2017.

[147] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning (CoRL)*, pages 1101–1112. PMLR, 2020.

[148] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *International Conference on Pattern Recognition (ICPR)*, volume 3, pages 850–855. IEEE, 2006.

[149] A. Nigam and L. D. Riek. Social context perception for mobile robots. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 3621–3627. IEEE, 2015.

[150] nilexuk. Living room clutter. `https://www.flickr.com/photos/28763193@N00/87127358`.

[151] M. F. O'Connor and L. D. Riek. Detecting social context: A method for social event classification using naturalistic multimodal data. In *IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 3, pages 1–7. IEEE, 2015.

[152] A. Ošep, W. Mehner, P. Voigtlaender, and B. Leibe. Track, then decide: Category-agnostic vision-based multi-object tracking. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3494–3501. IEEE, 2018.

[153] A. Ošep, P. Voigtlaender, J. Luiten, S. Breuers, and B. Leibe. Large-scale object mining for object discovery from unlabeled video. In *International Conference on Robotics and Automation (ICRA)*, pages 5502–5508. IEEE, 2019.

[154] A. Ošep, P. Voigtlaender, M. Weber, J. Luiten, and B. Leibe. 4d generic video object proposals. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 10031–10037. IEEE, 2020.

[155] N. Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[156] p_a_h. Microphone. `https://www.flickr.com/photos/64654599@N00/2217562952`.

[157] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[158] A. Papazoglou and V. Ferrari. Fast object segmentation in unconstrained video. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 1777–1784, 2013.

[159] H. W. Park, I. Grover, S. Spaulding, L. Gomez, and C. Breazeal. A model-free affective reinforcement learning approach to personalization of an autonomous social robot companion for early literacy education. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 687–694, 2019.

[160] Y. Park, V. Lepetit, and W. Woo. Multiple 3d object tracking for augmented reality. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 117–120. IEEE, 2008.

[161] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung. Learning video object segmentation from static images. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 2663–2672, 2017.

[162] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR))*, pages 724–732, 2016.

[163] T.-H. Pham, A. Kheddar, A. Qammaz, and A. A. Argyros. Towards force sensing from vision: Observing hand-object interactions to infer manipulation forces. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 2810–2819, 2015.

[164] S. Pillai and J. Leonard. Monocular slam supported object recognition. *arXiv preprint arXiv:1506.01732*, 2015.

[165] P. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *Advances in neural information processing systems (NIPS)*, 2015.

[166] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. *arXiv*, 2016.

[167] R. Plamondon and S. N. Srihari. Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, 22(1):63–84, 2000.

[168] M. Pourebadi and L. D. Riek. Stroke modeling and synthesis for robotic and virtual patient simulators. In *AAAI Fall Symposium on Artificial Intelligence in Human Robot Interaction (AI-HRI)*, 2020.

[169] M. Quigley, K. Mohta, S. S. Shivakumar, M. Watterson, Y. Mulgaonkar, M. Arguedas, K. Sun, S. Liu, B. Pfrommer, V. Kumar, and C. J. Taylor. The open vision computer: An integrated sensing and compute system for mobile robots. In *International Conference on Robotics and Automation (ICRA)*, pages 1834–1840. IEEE, 2019.

[170] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *IEEE 11th International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.

[171] radkuch.13. Bangkok crowded street. `https://www.flickr.com/photos/137294100@N08/50045987868`.

[172] E. Rahtu, J. Kannala, and M. Blaschko. Learning a category independent object detection cascade. In *International conference on Computer Vision (ICCV)*, pages 1052–1059. IEEE, 2011.

[173] K. Rakelly, E. Shelhamer, T. Darrell, A. A. Efros, and S. Levine. Few-shot segmentation propagation with guided networks. *arXiv*, 2018.

[174] A. Ramisa, G. Alenya, F. Moreno-Noguer, and C. Torras. Using depth and appearance features for informed robot grasping of highly wrinkled clothes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1703–1708. IEEE, 2012.

[175] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 779–788, 2016.

[176] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 7263–7271, 2017.

[177] R. Reis. Werke dr1 carbon microphone on a stand. `https://commons.wikimedia.org/w/index.php?curid=23298820`.

[178] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems (NIPS)*, pages 91–99, 2015.

[179] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 658–666, 2019.

[180] L. D. Riek. The social co-robotics problem space: Six key challenges. In *Robotics: Science and Systems (RSS), Robotics Challenges and Visions.*, 2013.

[181] L. D. Riek. Healthcare robotics. *Communications of the ACM*, 60(11):68–78, 2017.

[182] L. Robertson, A. Treisman, S. Friedman-Hill, and M. Grabowecky. The interaction of spatial and object pathways: Evidence from balint's syndrome. *Journal of Cognitive Neuroscience*, 9(3):295–317, 1997.

[183] robpegoraro. Waymo self-driving van. `https://www.flickr.com/photos/45975345@N00/42019354871`.

[184] P. Rochat. Object manipulation and exploration in 2-to 5-month-old infants. *Developmental Psychology*, 25(6):871, 1989.

[185] G. Ros, A. Sappa, D. Ponsa, and A. M. Lopez. Visual slam for driverless cars: A brief survey. In *Intelligent Vehicles Symposium (IV) Workshops*, volume 2, 2012.

[186] A. Rosebrock. Iou for bounding boxes showing a poor, good and excellent prediction. `http://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/`.

[187] R. Rothe, M. Guillaumin, and L. Van Gool. Non-maximum suppression for object detection by passing messages between windows. In *Asian conference on computer vision*, pages 290–306. Springer, 2014.

[188] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *International conference on computer vision (ICCV)*, pages 2564–2571. IEEE, 2011.

[189] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[190] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[191] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 4510–4520, 2018.

[192] T. R. Savarimuthu, A. G. Buch, C. Schlette, N. Wantia, J. Roßmann, D. Martínez, G. Alenyà, C. Torras, A. Ude, B. Nemec, A. Kramberger, F. Worgotter, E. A. Erdal, P. Jeremie, S. Haller, J. Piater, and N. Kruger. Teaching a robot the semantics of assembly tasks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(5):670–692, 2017.

[193] M. Schwarz, A. Milan, A. S. Periyasamy, and S. Behnke. Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter. *The International Journal of Robotics Research (IJRR)*, 37(4-5):437–451, 2018.

[194] I. O. Sebe, J. Hu, S. You, and U. Neumann. 3d video surveillance with augmented virtual environments. In *ACM SIGMM international workshop on Video surveillance*, pages 107–112, 2003.

[195] A. Shaban, A. Firl, A. Humayun, J. Yuan, X. Wang, P. Lei, N. Dhanda, B. Boots, J. M. Rehg, and F. Li. Multiple-instance video segmentation with sequence-specific object proposals. In *CVPR Workshop*, volume 1, 2017.

[196] J. Shin, R. Triebel, and R. Siegwart. Unsupervised 3d object discovery and categorization for mobile robots. In *Robotics Research*, pages 61–76. Springer, 2017.

[197] E. S. Short, A. Allevato, and A. L. Thomaz. Sail: simulation-informed active in-the-wild learning. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 468–477. IEEE, 2019.

[198] P. D. Siakaluk, P. M. Pexman, C. R. Sears, K. Wilson, K. Locheed, and W. J. Owen. The benefits of sensorimotor knowledge: Body–object interaction facilitates semantic processing. *Cognitive Science*, 32(3):591–605, 2008.

[199] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[200] Stereolabs. Zed - stereo camera for depth sensing. `https://www.stereolabs.com/`. Accessed: 2016-1-12.

[201] T. Stone, M. Mangan, P. Ardin, and B. Webb. Sky segmentation with ultraviolet images can be used for navigation. In *Robotics: Science and Systems (RSS)*. Robotics: Science and Systems, 2014.

[202] J. Su. Comparison of edge detectors. *Department of Civil and Environmental Engineering and Geodetic Science*, 2001.

[203] E. Sucar and J.-B. Hayet. Bayesian scale estimation for monocular slam based on generic object detection for correcting scale drift. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5152–5158. IEEE, 2018.

[204] N. Suenderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford. Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free. In *Robotics: Science and Systems (RSS)*, 2015.

[205] N. Sünderhauf, F. Dayoub, S. McMahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft, and M. Milford. Place categorization and semantic mapping on a mobile robot. In *IEEE international conference on robotics and automation (ICRA)*, pages 5729–5736. IEEE, 2016.

[206] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid. Meaningful maps with object-oriented semantic mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5079–5085. IEEE, 2017.

[207] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1–9, 2015.

[208] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov. Scalable, high-quality object detection. *arXiv*, 2014.

[209] K. Tang, A. Joulin, L.-J. Li, and L. Fei-Fei. Co-localization in real-world images. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1464–1471, 2014.

[210] A. Taylor, D. M. Chan, and L. D. Riek. Robot-centric perception of human groups. *ACM Transactions on Human-Robot Interaction (THRI)*, 9(3):1–21, 2020.

[211] A. Taylor, H. R. Lee, A. Kubota, and L. D. Riek. Coordinating clinical teams: Using robots to empower nurses to stop the line. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–30, 2019.

[212] A. Taylor, S. Matsumoto, and L. D. Riek. Situating robots in the emergency department. In *AAAI Spring Symposium on Applied AI in Healthcare: Safety, Community, and the Environment*, 2020.

[213] A. M. Taylor, S. Matsumoto, W. Xiao, and L. D. Riek. Social navigation for mobile robots in the emergency department. 2021.

[214] S. Tian, F. Ebert, D. Jayaraman, M. Mudigonda, C. Finn, R. Calandra, and S. Levine. Manipulation by feel: Touch-based control with deep predictive models. In *International Conference on Robotics and Automation (ICRA)*, pages 818–824. IEEE, 2019.

[215] J. Tobin, L. Biewald, R. Duan, M. Andrychowicz, A. Handa, V. Kumar, B. McGrew, A. Ray, J. Schneider, P. Welinder, W. Zaremba, and P. Abbeel. Domain randomization and generative models for robotic grasping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3482–3489. IEEE, 2018.

[216] D. Tomè, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi, and S. Tubaro. Deep convolutional neural networks for pedestrian detection. *Signal processing: image communication*, 47:482–489, 2016.

[217] L. Torrey and J. Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[218] J. K. Tsotsos, S. M. Culhane, W. Y. K. Wai, Y. Lai, N. Davis, and F. Nuflo. Modeling visual attention via selective tuning. *Artificial intelligence*, 78(1-2):507–545, 1995.

[219] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[220] V. V. Unhelkar, P. A. Lasota, Q. Tyroller, R.-D. Buhai, L. Marceau, B. Deml, and J. A. Shah. Human-aware robotic assistant for collaborative assembly: Integrating human motion prediction with planning in time. *IEEE Robotics and Automation Letters (RAL)*, 3(3):2394–2401, 2018.

[221] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders. Segmentation as selective search for object recognition. In *International Conference on Computer Vision (ICCV)*, pages 1879–1886. IEEE, 2011.

[222] D. Walther and C. Koch. Modeling attention to salient proto-objects. *Neural networks*, 19(9):1395–1407, 2006.

[223] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *The International Journal of Robotics Research (IJRR)*, 26(9):889–916, 2007.

[224] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. L. Region proposal by guided anchoring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2965–2974, 2019.

[225] L. Wang, G. Hua, R. Sukthankar, J. Xue, Z. Niu, and N. Zheng. Video object discovery and co-segmentation with extremely weak supervision. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 39(10):2074–2088, 2016.

[226] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1328–1338, 2019.

[227] W. Wang, J. Shen, R. Yang, and F. Porikli. Saliency-aware video object segmentation. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 40(1):20–33, 2017.

[228] Z. Wang, C. Zhang, M. K. Singh, L. Riek, and K. Chaudhuri. Multitask bandit learning through heterogeneous feedback aggregation. In *International Conference on Artificial Intelligence and Statistics*, pages 1531–1539. PMLR, 2021.

[229] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 101–108. IEEE, 2000.

[230] K. S. Welfare, M. R. Hallowell, J. A. Shah, and L. D. Riek. Consider the human work experience when integrating robotics in the workplace. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 75–84. IEEE, 2019.

[231] J. Wilson and M. C. Lin. Avot: Audio-visual object tracking of multiple objects for robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 10045–10051. IEEE, 2020.

[232] D. Wofk, F. Ma, T. Yang, S. Karaman, and V. Sze. Fastdepth: Fast monocular depth estimation on embedded systems. In *International Conference on Robotics and Automation (ICRA)*, pages 6101–6108. IEEE, 2019.

[233] C. Wojek, S. Walk, and B. Schiele. Multi-cue onboard pedestrian detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 794–801. IEEE, 2009.

[234] B. Woodworth, F. Ferrari, T. E. Zosa, and L. D. Riek. Preference learning in assistive robotics: Observational repeated inverse reinforcement learning. In *Machine Learning for Healthcare Conference*, pages 420–439. PMLR, 2018.

[235] B. Wu, F. Iandola, P. H. Jin, and K. Keutzer. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 129–137, 2017.

[236] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel. Learning to manipulate deformable objects without demonstrations. In *Robotics: science and systems (RSS)*, 2019.

[237] Y. Xiang, C. Xie, A. Mousavian, and D. Fox. Learning rgb-d feature embeddings for unseen object instance segmentation. *arXiv preprint arXiv:2007.15157*, 2020.

[238] S. Yang and S. Scherer. Monocular object and plane slam in structured environments. *IEEE Robotics and Automation Letters (RAL)*, 4(4):3145–3152, 2019.

[239] X. Ye, Z. Lin, H. Li, S. Zheng, and Y. Yang. Active object perceiver: Recognition-guided policy learning for object searching on mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6857–6863. IEEE, 2018.

[240] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morona, P. Q. Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser, and A. Rodriguez. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *IEEE international conference on robotics and automation (ICRA)*, pages 3750–3757. IEEE, 2018.

[241] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech. Minimum barrier salient object detection at 80 fps. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 1404–1412, 2015.

[242] J. Zhang, Y. Xia, H. Cui, and Y. Zhang. Pulmonary nodule detection in medical images: A survey. *Biomedical Signal Processing and Control*, 43:138–147, 2018.

[243] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele. How far are we from solving pedestrian detection? *arXiv preprint arXiv:1602.01237*, 2016.

[244] S. Zhang, R. Benenson, and B. Schiele. Filtered channel features for pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2015.

[245] Z. Zhao, P. Zheng, S. Xu, and X. Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[246] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *AAAI*, pages 12993–13000, 2020.

[247] X. Zhou, X. Lan, H. Zhang, Z. Tian, Y. Zhang, and N. Zheng. Fully convolutional grasp detection network with oriented anchor box. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7223–7230. IEEE, 2018.

[248] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European conference on computer vision (ECCV)*, pages 391–405. Springer, 2014.