Geometry-Based Region Proposals for Accelerated Image-Based Detection of 3D Objects

Alexander Broad*§ and Brenna Argall*†‡§
*Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, Illinois 60628
Email: alex.broad@u.northwestern.edu
†Department of Mechanical Engineering
‡Department of Physical Medicine & Rehabilitation
§Rehabilitation Institute of Chicago, Chicago, Illinois, USA
Email: brenna.argall@northwestern.edu

***Abstract*—We present a novel object detection pipeline for localization and recognition in three dimensional environments. Our approach makes use of an RGB-D sensor and combines state-of-the-art techniques from the robotics and computer vision communities to create a robust, real-time detection system. We focus specifically on solving the object detection problem for table top scenes, a common environment for assistive manipulators, especially in laboratory settings. Our detection pipeline locates objects in a point cloud representation of the scene by exploiting known geometric relationships through an unsupervised clustering technique. From these clusters we compute the three dimensional position of each object and derive the coordinates of a bounding box, which is then translated into the corresponding points in RGB space. The defined patch is then fed into a Convolutional Neural Network (CNN) for object recognition. We also demonstrate that our region proposal method can be used to develop novel datasets that are both large and diverse enough to train deep learning models, and easy enough to collect that users can develop their own datasets. Lastly, we validate the resulting system through a preliminary analysis of the accuracy and run-time of the full pipeline. All source code is available online.**

I. Introduction

As the field of robotics advances, and personal robots that assist users in their home and work environments become more prevalent, it will be necessary to extend a robot's autonomy to include more advanced cognitive reasoning and improve abilities in highly dynamic environments. To do so, the robot will need to have knowledge of many of the same physical attributes of the world that a human does. One such important aspect is the ability to recognize and localize objects. Through this knowledge, a robot can make informed decisions in achieving tasks like intelligently searching for an object in a novel environment, cleaning a room or retrieving an object for a human partner.

The problem of object detection is not unique to robotics. In computer vision it is used to solve problems such as automatic caption generation [10] and automatic tagging of shared social pictures. However, it is often the case that techniques used in the two communities are distinct from one another. One main reason for this is that the desired and available sensor information is frequently different–in computer vision, systems are usually limited to the RGB space while solving problems in robotics generally requires depth as an additional, or primary, modality. The requirement of depth information often necessitates an additional sensor (with a few notable exceptions [14, 19]), which in turn requires a potentially difficult calibration and sensor fusion problem. For this reason, we frequently see methodologies in the two communities that parallel each other in purpose, such as object recognition, but are divergent in technique. However, due to the rise of RGB-D cameras like the Microsoft Kinect [24], robotics researchers have access to sensors that provide both color and depth information in a single device. These cameras can be easily calibrated (up to a level of tolerance) and aligned through a single transformation defined by the static configuration of the two integrated sensors.

In this paper, we propose a novel method for object detection that makes use of both the depth and color modalities of RGB-D sensors to recognize and localize objects in real-time. We focus specifically on table top environments as many manipulation tasks take place in this type of configuration. Our approach solves the localization and recognition tasks independently–the former through an exploitation of the geometry of the scene, and the latter with state-of-the-art deep learning methods.

We begin by discussing related work in Section II and then present our approach in detail in Section III. We also discuss how our region proposal method can be used for data acquisition and developing novel datasets in Section IV. Finally, we describe an experimental validation of our system in Section V and discuss the success of our approach and future direction in Section VI. We conclude in Section VII.

II. Related Work

From a computational perspective, object detection is a two part problem. (1) Where is the object? and (2) What is the object? The long-standing baseline approach in computer vision is known as a sliding window. In this technique, each patch of size (m, n), from an image of size (M, N), is fed through an object recognition model. The concept is that, while this approach may be inefficient, it maximizes recall by

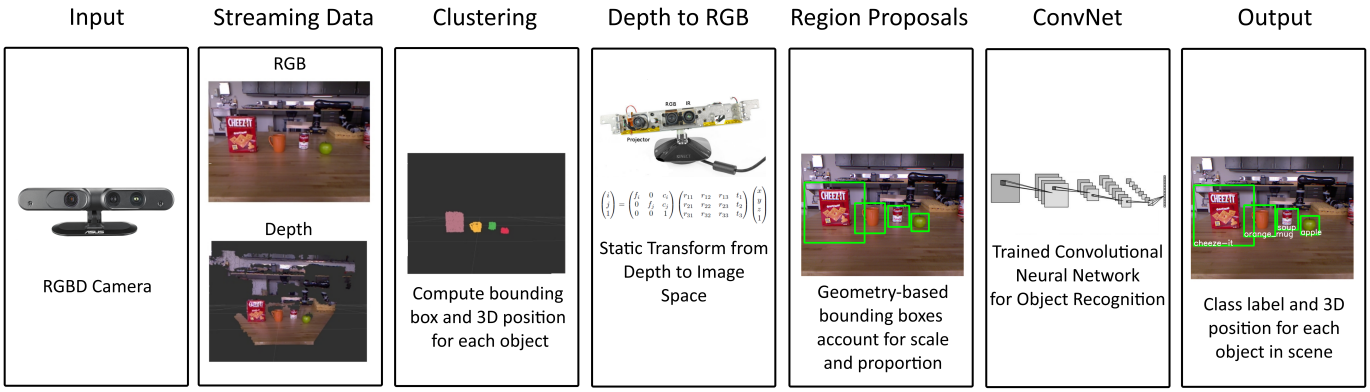

Fig. 1. High-level step-through of our object detection pipeline. From left to right : We use an RGBD camera to capture color and depth information about our scene. We then exploit known geometric properties to compute a bounding box and 3D position for each visible object. The bounding boxes are then translated to RGB space using the static transform defined by the position of the two sensors in the camera. These patches are then fed through a trained CNN for object recognition. The output is a class label and 3D position for each object in the scene.

ensuring not to skip any possible object locations. To account for scale the same process is repeated over an image pyramid.

More recently, as interactive systems have become more popular [2] there has been an increased focus on the efficiency of object detection systems. The most common way to improve the run-time of the system is to intelligently reduce the number of candidate regions that are run through the object recognition model. New approaches focus on novel techniques and methods for producing *region proposals*. For example, Girshick [6] use multi-layer segmentation to produce region proposals at different positions and scales, while Szegedy et al. [21] train a neural network to predict segmentation masks and Zitnick and Dollár [25] use edge detections. Hosang et al. [8] provide a comprehensive comparison of region proposal techniques in the computer vision community. By reducing the number of candidate regions, one is able to perform a more efficient search through position, scale and orientation. These techniques greatly reduce the computational burden when compared to an exhaustive sliding window approach, however they often still require expensive systems and GPUs to train and run. For example, Fast R-CNN [6] has proven very successful, yet using this approach on a 640 x 480 image with a Core i7 laptop with a mid-tier GPU (nVidia GeForce 860M), the full pipeline takes about 0.75 seconds per image. One reason for this is that the segmentation algorithm produces between 1k and 10k proposals per image depending on the *quality mode* parameter. To increase the speed of this system, Ren et al. [16] extend Girshick's method to a model called Faster R-CNN, which uses a separate neural network to produce object proposals, decreasing the run-time to about 0.2 seconds per image. Of note this approach still requires significantly more training data than our proposed method. Additionally localizing an object in the 2D plane does not fully solve the problem for robotics applications when the three dimensional location is equally as important as correctly recognizing the object.

There is also related work from the robotics community in 3D object detection. Early approaches are similar to pre-deep learning methods in the vision community; namely, they focus on developing hand-crafted features in the point cloud space. Some examples include local features such as the histogram-based Fast Point Feature Histogram [17] and Signature of Histogram of Orientations [23], as well as global features such as the Viewpoint Feature Histogram [18] which also takes the viewpoint into account. Tang et al. [22] describe a similar segmentation approach to our own, however the recognition is again done in the point cloud space by comparing features to learned object models. The approaches mentioned here work relatively well, however they rely on hand-crafted features and no single approach has found the type of success or wide-spread adoption that Convolutional Neural Networks have found in the image space [13].

More closely related to our own work, researchers have begun to look at other methods for combining the RGB and depth modalities in solving the object detection problem. Song and Xiao [20] describe a three dimensional version of the sliding window approach in which they fit 3D regions to learned CAD-based object models. Dahan et al. [4] describe a method for computing region proposals by segmenting the input image using information from both the color and depth channels. Gupta et al. [7] and Couprie et al. [3] similarly describe methods for including depth during segmentation and they also augment the standard vision approach by including the depth map as another input channel in training CNN model. The main difference between these works and our own is that we explicitly generate region proposals in the point cloud space based on geometric constraints which produces significantly fewer candidate regions.

Lastly, Pillai and Leonard [15] present a robotic recognition system that incorporates multi-view object proposals and efficient feature encoding methods to solve a similar problem. In particular the researchers develop a SLAM-aware system

that incorporates a detection model to improve robotic object recognition. However, again, this work is distinct from our own as it performs the recognition in the point cloud space.

III. Our Approach

Our approach leverages both RGB and depth sensing modalities in a single object detection pipeline. In this work we focus specifically on detection in a table top setting, a common environment for assistive manipulators and particularly useful to researchers in laboratory settings. We take inspiration from the computer vision community and develop a novel region proposal method, however, our technique is rooted in robotic perception and makes use of three dimensional point cloud data. To do so, we exploit the geometry of the environment to produce a minimum set of region proposals as described in Section III-A. We then translate our candidate regions from three dimensional bounding boxes into their two dimensional representation in the image plane as described in Section III-B. This image patch is then fed into a CNN for classification as described in Section III-C.

A. Object Localization

Our object localization method is detailed in Algorithm 1. This algorithm simultaneously computes a bounding box, $\mathbb{B}$, and the three dimensional position, $\mathbb{P}$, of each object in the table top scene. The localization method capitalizes on known geometric properties of the table to reduce the computational burden and produce highly reproducible results.

The input to the algorithm is a point cloud, C, which we then downsample (Line 2). We choose a reasonable sampling parameter, $\alpha = 0.1$, to ensure coverage and speed. An optional step to further reduce the computational burden is to also run the point cloud through pass-through filters parameterized by the geometry of the table top (Line 3). Our experimental results in Section V use this step. We can further remove any points belonging to the table top itself through the use of a RANSAC filter (Lines 4–5), where T is the point cloud representation of the table. A Euclidean clustering algorithm is run on the remaining points to discover continuous objects in the scene (Line 6). We can then compute a bounding box $\mathbb{B}$ around each object in the set of clusters $o \in \mathbb{O}$ by finding the upper left, U, (Line 9) and lower right, L, (Line 10) corners of the cluster o. We also compute the three dimensional position $\mathbb{P}$ of an object by computing its centroid (Line 12).

Similar to other model-free segmentation approaches, a benefit of our method is that the region proposal algorithm does not require learning a model from a large dataset. Instead, we make use of the geometry of the scene to develop candidate object locations. This approach has the added benefit of significantly decreasing the number of region proposals when compared to other methods. That is, for each object in the scene we propose only a single region by using the physical properties of the object to account for both position and scale. Our method is particularly well suited for our problem domain as a vast number of manipulation objects are easily clustered due to their shape and size, particularly in uncluttered environments. Another benefit to computing the region proposals in the depth modality is that our localization of the object is very accurate [11]. For example, in our experiments the table was one meter wide, indicating a maximum error of approximately 6mm.

Algorithm 1 Geometric Region Proposal

1: **Given** Point Cloud C, *optional : table dimensions*
2: $C \leftarrow downsample(C, \alpha)$
3: *optional:* $C \leftarrow passthrough(C,$ table dimensions$)$
4: $T_{inliers} \leftarrow RANSAC(C)$
5: $T_{outliers} \leftarrow C - T$
6: $\mathbb{O} \leftarrow Cluster(T_{outliers})$
7: **Init** $\mathbb{B} \leftarrow \varnothing, \mathbb{P} \leftarrow \varnothing$
8: **for** $o \in \mathbb{O}$ **do**
9: $\quad U \leftarrow (x_{min}, y_{max}, z_{max})$
10: $\quad L \leftarrow (x_{max}, y_{min}, z_{max})$
11: $\quad \mathbb{B} \cup (U, L)$
12: $\quad \mathbb{P} \cup centroid(o)$
13: **return** $\mathbb{B}, \mathbb{P}$

B. Translation between Depth and RGB space

The next step in our object detection pipeline is classifying each proposal region. We choose to perform the classification in the image space due to the demonstrated accuracy and expressivity of deep learning methods. Therefore, we must translate the bounding box from the depth frame into the image frame. The coordinates of the bounding boxes in these two modalities are not directly aligned due to a physical offset in the sensor, however we can compute the transformation between the two [9].

To transform the bounding box in depth space to its representation in RGB space, we can begin by representing the RGB-D sensor as a pinhole camera. Under this assumption, each point in the depth space $(x, y, z) \in \mathbb{R}^3$ and each point in the image space $(i, j) \in \mathbb{R}^2$ are mapped into their homogeneous coordinate definitions, $(x, y, z, 1)$ and $(i, j, 1)$, respectively. We can then define a projective relationship between the two representations based on the intrinsic and extrinsic parameters of the camera as seen in Equation 1.

$$\begin{pmatrix} i \\ j \\ 1 \end{pmatrix} = \begin{pmatrix} f_i & 0 & c_i \\ 0 & f_j & c_j \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (1)$$

In this equation the first matrix represents the camera's intrinsic parameters and describes a transformation between the optical center of the camera and a given point in the image frame. Specifically, f_i, f_j represent the focal length in pixel space and c_i, c_j represent the physical offset between the origins of each frame in pixel space. The second matrix represents the camera's extrinsic parameters and describes a transformation between the position and orientation of the depth- and RGB-cameras. These are defined by the sensor

Fig. 2. Top row : Example data captured using our object localization procedure with multiple orientations. Bottom row : Example output of object detection pipeline with multiple positions.

hardware and are often both readable and tunable using the associated driver (the values used in our implementation are included in the open source code). Through the use of Equation 1, we can therefore translate each bounding box in the point cloud to a bounding box in the image space.

Until this point, the bounding box we have computed tightly constrains each object in the scene, however, for the recognition portion of our pipeline it is useful to have a border around the object itself. This is because most image based recognition networks are trained with patches that include a border around the object of interest. For this reason, we slightly expand the bounding box associated with each object. The size of the border can be tuned (in our work, we expanded the border by 40%), however the same parameters should be used during training and testing.

C. Object Recognition

The final step in our object detection pipeline is recognition which we solve using a Convolutional Neural Network. Each region proposal is extracted from the full image, scaled to the input size of our trained neural network and classified.

The network architecture we used in this work is as follows. The input layer is connected to 3 sequences of Convolutional filters with max pooling and ReLu activation functions. This sequence is followed by a final hidden layer of non-Convolutional filters. The output layer is a learned soft-max classifier.

Of note, at this point it is possible to use any pre-trained model that includes the classes of interest. However, in these experiments (Section V), we opt to train own network for two reasons. First, our object set is not fully encompassed by any widely circulated pre-trained network; and second, we opt to demonstrate a secondary capability of our object localization method; namely, data acquisition (discussed further in Section IV).

IV. Dataset Creation

A secondary application of our region proposal method is data acquisition. Developing new datasets suitable for training deep learning models is normally a heavily human-time intensive process. This is particularly important for robotics applications, where there is a dearth of pre-trained recognition models. Using our object localization method, researchers can quickly and easily create labeled data for objects not commonly found in circulated datasets. Our approach is described in Algorithm 2. This algorithm works by employing the use of Algorithm 1 on the set of object classes important to a researcher. By placing an instance of a known object class in the view of the RGB-D sensor (Line 4), we can store the streaming output of Algorithm 1 along with the user provided label (Lines 5-7) in a supervised learning dataset. This process is repeated for the full set of objects that a user is interested in at multiple locations throughout the scene. As Algorithm 1 is very fast, it is possible to store a large quantity of data very quickly.

Algorithm 2 Dataset Acquisition

1: **Given** object class labels, $\mathbb{Y}_{labels}$
2: **Init** $\mathbb{X}_{train} \leftarrow \varnothing, \mathbb{Y}_{train} \leftarrow \varnothing$
3: **for** $y_{label} \in \mathbb{Y}_{labels}$ **do**
4: place object of type y_{label} in the scene
5: $\mathbb{B}, \mathbb{P} \leftarrow$ Algorithm 1(object point cloud)
6: $b_{rgb} \leftarrow$ Convert $\mathbb{B}$ to RGB space
7: $x \leftarrow$ image patch defined by b_{rgb}
8: $\mathbb{X}_{train} \cup x$
9: $\mathbb{Y}_{train} \cup y_{label}$
10: **return** $\mathbb{X}_{train}, \mathbb{Y}_{train}$

While capturing example images, it helps generalize the model to alter the position and orientation of object(s) thereby providing multiple views of each class. It can also help to alter aspects like lighting conditions and orientation with respect to the camera. An example of the types of data collected via this method can be seen in Figure 2. Source code for the dataset acquisition process is also a part of the released package.

V. Experimental Validation

A preliminary analysis of our system shows promising results in the accuracy of both the localization and recognition modules as well as in the speed of the system. Our recognition model is trained on a dataset collected using the method described in Section IV. The dataset consists of 11 different object classes and 2180 total images, evenly split by class.

Fig. 3. Object set used to test detection pipeline. YCB Food: coffee can, cheeze-it box, pringles can, tomato soup can, apple, orange. YCB Kitchen: bowl, red mug. Other objects: orange bowl, orange mug, translucent bowl.

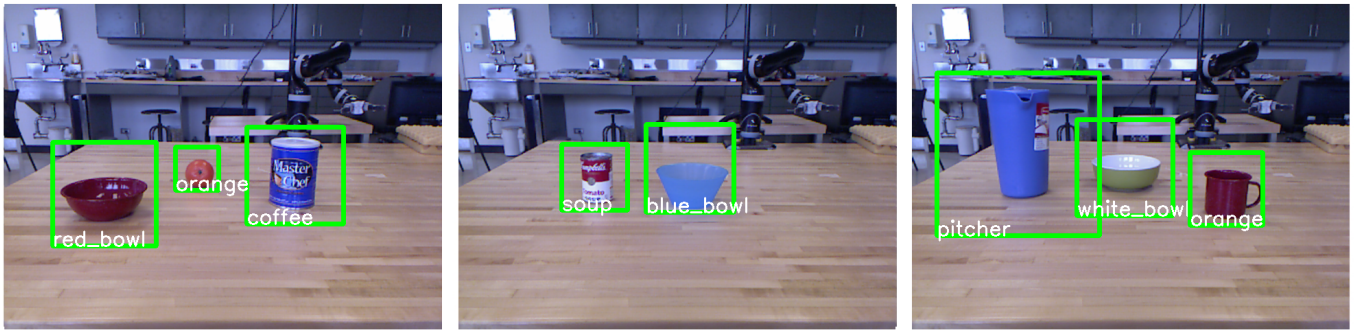

Fig. 4. Three example scenes with multiple objects. All objects correctly localized and classified, with the exception of the red mug in the far left image. This object is not in our model. Future work will include training a large model with more classes including a dustbin class for objects that are not in the dataset.

Eight of the objects are a selection from the YCB Object Set [1] that are relevant to our interests, while the remaining three objects are similar common household goods. We use an $80/20\%$ train/test split to train our model. The full object set can be seen in Figure 3. The red mug was not included in the training set and is only used to demonstrate a need for improved handling of objects not in the training set.

Object Class	Recognition Accuracy
Orange	100/100
Apple	100/100
Red Bowl	100/100
White Bowl	100/100
Blue Bowl	100/100
Soup	100/100
Pitcher	100/100
Pringles	100/100
Orange Mug	100/100
Coffee	100/100
Cheeze-it	100/100

TABLE I
RECOGNITION ACCURACY

We validate our approach with a pair of experiments that examine how our method performs on streaming RGB-D data unique from our training set. In the first experiment, the objects themselves are the same physical objects as those used to train our model, however we randomize each object's position and orientation in the scene. We placed one object in the view of the RGB-D sensor at a time and collected count data representing how many patches our model was able to label correctly in the 100 frames. As seen in Table I, our model was able to perfectly identify each class in our dataset. Example images of the variation in position and orienation in our testing configurations can be seen in Figure 5. During each test, the object in the scene was stationary to avoid the sensor detecting our arms as objects.

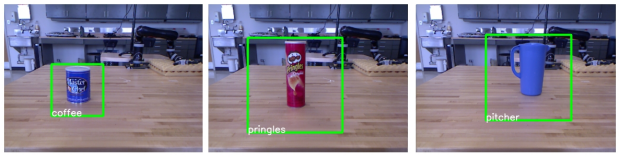

Fig. 5. A coffee can, pringles can and pitcher, all correctly detected.

Objects in Scene	Recognition Accuracy
Red Bowl, Orange, Coffee	300/300
Soup, Translucent Blue Bowl	300/300
Pitcher, White Bowl, Red Mug	200/300

TABLE II
SCENE RECOGNITION ACCURACY

In addition to examining how well our pipeline worked on single instances of each class in our dataset, we also developed three scenes with multiple objects in different arrangements. Using the same methodology described above, we collected count data representing the number of correctly labeled patches in the first 100 frames of each of the three scenes. The results are displayed in Table II. In two of the three scenes, our pipeline was again able to correctly locate and identify all objects in the scene. In the last scene, we accidentally included an object not in our training set, and as such, the model was unable to correctly label that object (the red mug). This does indicate the need for improved methodology in handling classes not in our dataset. Example output can be seen in Figure 4.

Our experiments were run on a Core i7 laptop with a mid-tier mobile GPU (nVidia GeForce 860M). The system runs at an average of 12Hz. On the same computer, with the same size image (640 x 480), R-CNN [6], a competitive approach in speed and accuracy, is able to run at an average of 1.33Hz. This is a speed up factor of about 9x.

VI. DISCUSSION

We believe the presented paradigm is a promising direction for practical robotic perception systems. The marriage of state-of-the-art techniques from robotics and computer vision help produce a fast and accurate object detection framework that can be easily incorporated into any table top manipulation task. It is a real-time system that does not require top of

the line hardware and produces competitive results. This methodology is additionally useful for creating novel datasets which suggests that this approach could be ideal for other researchers and advanced users alike.

In this initial work, we do not provide a direct comparison to related methods using standard detection metrics (e.g. mean average precision (mAP), repeatability of results, scale changes and illumination changes Hosang et al. [8]) as the datasets used for these comparisons often only include RGB data [5]. In future work we hope to expand upon our preliminary results by providing a comparative analysis on an RGB-D detection dataset, such as the UW RGB-D Scene Dataset provided by Lai et al. [12]. Additionally, we hope to demonstrate how this approach can be used to solve the object detection problem in cluttered environments and non-table top scenes.

VII. Conclusion

In this paper, we describe and demonstrate a simple, and fast object detection pipeline for table top manipulation tasks using robot vision. The described system owes its speed and computational efficiency to the minimal set of regions proposed through unsupervised methods of analysis in the point cloud space, and it's accuracy and generalizability in the recognition space to Convolutional Neural Networks. The code is available online with an open-source MIT license at $https://github.com/asbroad/geom\_rcnn$.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant CNS 1329891. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the aforementioned institutions.

References

[1] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron Dollar. Benchmarking in Manipulation Research: The YCB Object and Model Setand Benchmarking Protocols. In *IEEE Robotics and Automation Magazine*, August 2015.

[2] Xiangrong Chen and Alan L Yuille. A Time-Efficient Cascade for Real-Time Object Detection: With Applications for the Visually Impaired. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2005.

[3] Camille Couprie, Clement Farabet, Laurent Najman, and Yann LeCun. Toward Real-time Indoor Semantic Segmentation Using Depth Information. *JMLR*, 2014.

[4] Meir Johnathan Dahan, Nir Chen, Ariel Shamir, and Daniel Cohen-Or. Combining Color and Depth for Enhanced Image Segmentation and Retargeting. *The Visual Computer*, 28(12): 1181–1193, 2012.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[6] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2015.

[7] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[8] J. Hosang, R. Benenson, and B. Schiele. How Good are Detection Proposals, Really? In *BMVC*, 2014.

[9] Branko Karan. Calibration of Kinect-type RGB-D sensors for robotic applications. *FME Transactions*, 43(1):47–54, 2015.

[10] Andrej Karpathy and Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015.

[11] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. *Sensors*, 12(2):1437–1454, 2012.

[12] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[14] Raul Mur-Artal, JMM Montiel, and Juan D Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[15] Sudeep Pillai and John Leonard. Monocular SLAM Supported Object Recognition. In *Proceedings of Robotics: Science and Systems (RSS)*, Rome, Italy, July 2015.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[17] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast Point Feature Histograms (FPFH) for 3D Registration. In *Proceedings of the International Conference on Robotics and Automation*. IEEE, 2009.

[18] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, and John Hsu. Fast 3d Recognition and Pose using the Viewpoint Feature Histogram. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[19] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. Make3D: Learning 3D Scene Structure from a Single Still Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 (5):824–840, 2009.

[20] Shuran Song and Jianxiong Xiao. Sliding Shapes for 3D Object Detection in Depth Images. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 2014.

[21] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep Neural Networks for Object Detection. In *Advances in Neural Information Processing Systems*, pages 2553–2561, 2013.

[22] Jie Tang, Stephen Miller, Arjun Singh, and Pieter Abbeel. A Textured Object Recognition Pipeline for Color and Depth Image Data. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[23] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique Signatures of Histograms for Local Surface Description. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[24] Zhengyou Zhang. Microsoft Kinect Sensor and its Effect. *MultiMedia, IEEE*, 19(2):4–10, 2012.

[25] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating Object Proposals from Edges. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 391–405. Springer, 2014.