Depth enhancement in a robotic pick and place task

How does depth information enhance 6D pose estimation by varying camera distance with priori known objects

Literature Survey

Max Waterhout

TU Delft

Contents

Abstract

This study investigates how depth information affects 6D pose estimation using priori known objects at varying camera distances. The Linemod dataset was utilized to test the accuracy of three neural-network based models, and results indicate that as the camera distance increases, the performance of all algorithms decreases, with a more pronounced decline observed for complex objects. Although depth can improve pose estimation accuracy, it cannot fully compensate for poorly predicted poses. The study also suggests several potential directions for future research, such as testing more algorithms and datasets, developing new metrics for measuring difficulty of objects, and investigating the effects of depth enhancement in occluded scenarios.

1
Introduction

Robotic manipulation is a key area of research in the robotics community, particularly the pick and place problem. Many challenges, such as the Amazon Picking Challenge [1] and the DHL Robotics challenge [2] have been built around this task. The aim of these challenges is to enable robots to assist human workers with tedious tasks, leading to increased efficiency, improved throughput, and reduced costs.

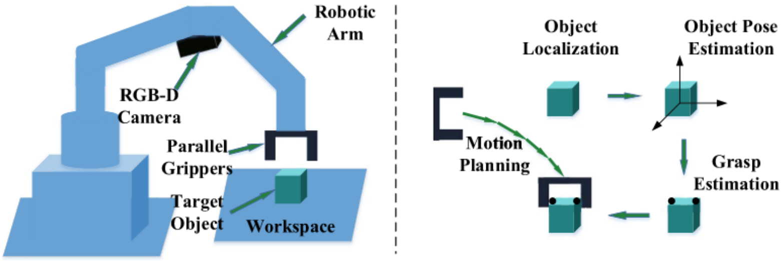


Figure 1.1: The pick and place problem pipeline

1.1. Problem statement

In many pick and place scenarios, it is essential to accurately estimate the placement of objects in the world. For example, consider a scenario where a robotic arm retrieves items from a moving assembly line, with objects situated in varying positions [3]. Similarly, in high-precision assembly tasks like the Siemens Innovation Challenge, the assembly of gears within industrial settings with millimeter-accurate placement [4]. The process of estimation the placement in the real world is known as 6D pose estimation. A process that transforms the object from its object coordinate system into the cameras coordinate system [5, 6]. The complete pick and place pipeline, which includes this critical step, is depicted in figure 1.1. The 6D pose is a 4x4 matrix that is composed of a 3D rotation and a 3D translation of a frame with respect to another frame, as shown in equation (1.1). In equation (1.1), the 6D pose is represented as the transformation matrix ${}^A_BT$, which describes the position and orientation of frame B relative to frame A. In real life we do not know the exact pose, but we can estimate it, mainly done with RGB or RGB-Depth images. The position and rotation of a known object model can be obtained using various algorithms. Once the 6D pose is obtained and the position and rotation of the camera is captured with respect to the gripper, it becomes possible to determine how to move the gripper towards the object for picking and placing.

$$ {}^A_BT = \begin{bmatrix} {}^A_BR & {}^BP \\ 0 & 1 \end{bmatrix} \tag{1.1} $$

Figure 1.2: An example 6D pose from frame B to frame A

1.2. Relevant work

Traditionally, 6D pose estimation methods based on images have relied on matching RGB feature points between 3D models and images [7, 8], dating back to 1999 with [9]. These methods involve extracting features from images, which can be RGB colors or more recent approaches that employ neural networks. The goal is to match keypoints of objects in the image with keypoints from the object stored in the database. By establishing these keypoint matches, 6D pose estimation can be performed. However, one major limitation of this method is that objects require rich texture for detection of feature points. With the introduction of affordable depth cameras, such as the Microsoft Kinect and Intel RealSense, several new methods have emerged that that add depth information to match features of less texture-rich objects [10, 11, 12].

The review [13] categorized the various pose estimation methods from RGB to RGB-D methods. Geometric pose estimation methods that extracted Pair Point Features (PPF) [14] from keypoints were the top-performing techniques from 2010 to 2019, according to the Benchmark for 6D Object Pose Estimation (BOP Challenge) [14, 15]. However, starting in 2020, Deep Neural Network (DNN) approaches that exclusively utilized RGB inputs, trained on extensive datasets, and did not rely on keypoints, emerged as contenders [16]. By 2022, DNN-based methods had surpassed PPF-based methods in both speed and accuracy [17, 18]. In 2022 the best methods based on RGB even surpassed the best RGB-D methods from 2020. Recent works [19, 20] still consider depth information crucial for the task of 6D pose estimation, despite RGB methods becoming more competitive in recent years. This is because the 2D projection of an object in an image may lack vital geometric details that are preserved in depth information. In essence, the 3D representation contains richer information compared to an 2D image. Also state-of-the-art results are still achieved with depth refinement looking at results from the BOP challenge from 2022.

Research focused on 6D pose estimation at varying distances can play a critical role as objects in real-world scenarios are often situated at different distances from the camera. Typically, the accuracy of 6D pose estimation is evaluated based on the mean accuracy across all objects, regardless of their distances. However, a study by [21] delves into the impact of varying camera distances specifically on object detection. The findings of this study indicate that accuracy tends to decrease at longer distances, although higher resolution can improve the accuracy. It is important to note that practical applications may not easily accommodate resolution enhancements due to the increased computational costs [22] and bandwidth demands associated with higher pixel-count images. Implementing such enhancements can strain computational resources and pose difficulties in transmitting or streaming high-resolution images over networks. The addition of depth information could compensate for this limitation.

1.3. Research question

Despite the existence of research on 6D pose estimation and a study exploring the effect of varying camera distances in object detection, no prior investigations have specifically examined the influence of different camera distances on the accuracy of 6D pose estimation. Furthermore, there is a lack of literature addressing how depth information provides improved performance. Therefore, a research gap exists regarding the impact of depth information on 6D pose estimation when the camera distance is changed.

The purpose of this study is to investigate how depth information can improve the accuracy of 6D pose estimation by varying the camera distance with known 3D objects. Specifically, the study will explore the impact of depth information on the performance of existing 6D pose estimation algorithms and determine how different camera distances affect the quality of 6D pose estimates. Ultimately the research question will be: **How does depth information enhance 6D pose estimation when varying the camera distance with known objects?**

2
Methods

As mentioned in chapter 1, significant progress has been made in RGB methods in recent years. The present study aims to evaluate the potential enhancement of depth in addition to pure RGB information in pose estimation. In this chapter, the methodology of the experiments and the analysis is presented. First, the selection of the Linemod dataset is discussed, including an analysis of the reasons behind its choice, followed by an explanation of the metrics used to assess the performance of the models. The three models used in the study are introduced, along with a justification for their selection. Finally, the depth refinement technique utilized in the study is explained.

2.1. Choosing the dataset

There are various open-source datasets available for testing 6D pose estimation algorithms. In 2022 the BOP challenge uses twelve different datasets [1]. The two most widely used datasets are YCB-Video [23] and Linemod [10], which have been cited 1405 and 1200 times, respectively. In comparison, the third most popular dataset has only been cited 250 times [24]. In this section, the comparison will focus on these two mostly used datasets. These datasets are selected because they are extensively employed in pose estimation studies, and many researchers openly share their model weights specifically for these datasets.

Despite the fact that the YCB-Video dataset comprises a much larger number of images (90K vs 15K) compared to Linemod, the focus of this study is to investigate the impact of depth in evaluation, wherein the predictions on diverse geometrical complexities of different objects can be an important feature. Therefore a comparison is conducted to determine which dataset to choose based on their geometrical complexities.

Measuring geometrical complexity

Capturing the complexity of the shape of an object cannot be done by relying on a single metric. One approach to gain insight into an object's complexity is to use Topological Data Analysis (TDA) [25, 26]. TDA is an analysis technique that leverages concepts from algebraic topology, geometry, and data analysis to extract meaningful information about the underlying structure of complex data. In this study, TDA is employed to analyze the complexity of the objects. Specifically, the number of vertices and edges is chosen as a metric to represent this complexity. Consequently, a lower count of vertices and edges indicates a reduced level of texture detail for an object, as also stated in [27]. The choice of choosing this as a metric is also motivated by the fact that many relevant studies [23, 10] discuss texture and texture-less objects in specific datasets without thoroughly substantiating their claims. Additionally, previous works such as [19, 23] have observed that estimating the pose of texture-less objects poses greater challenges.

Comparison Results for shape complexity

For the measurements of these metrics Meshlab [28] is used. Meshlab is a widely-used software program for processing and analyzing 3D meshes, making it suitable for extracting metrics such as the

[1] The datasets can be found at `https://bop.felk.cvut.cz/datasets/`

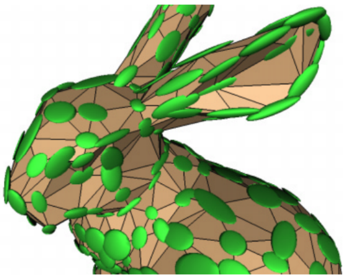


Figure 2.1: An example of vertices(green) and edges (black lines) of a bunny [27]

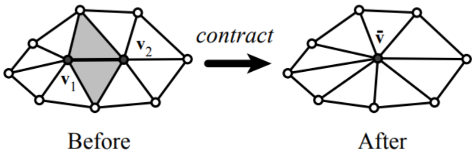


Figure 2.2: Example of quadratic edge collapse decimation [27]

number of vertices and edges in this study. The models are all extracted from the BOP site. Additionally, a simplification process called quadratic edge collapse decimation is done to reduce the number of vertices and faces in the mesh while preserving the overall shape, which may help to mitigate errors in the initial measurement of edges and vertices, as the shapes of the objects were hand-captured with point clouds. An example of this process can be seen in figure 2.2. Due to their size , the tables displaying the results of the complexity comparison for Linemod and YCB-V objects can be found in table A.1 and table A.2. The results reveal that the differences between the shapes of the objects from YCB-V are much lower compared to the objects from Linemod. This observation is consistent with the types of objects in figure 2.3, where YCB-V has a larger number of household products that share similar shapes, while Linemod features a diverse set of textures and shapes.

Selection of Dataset Based on Geometrical Complexity

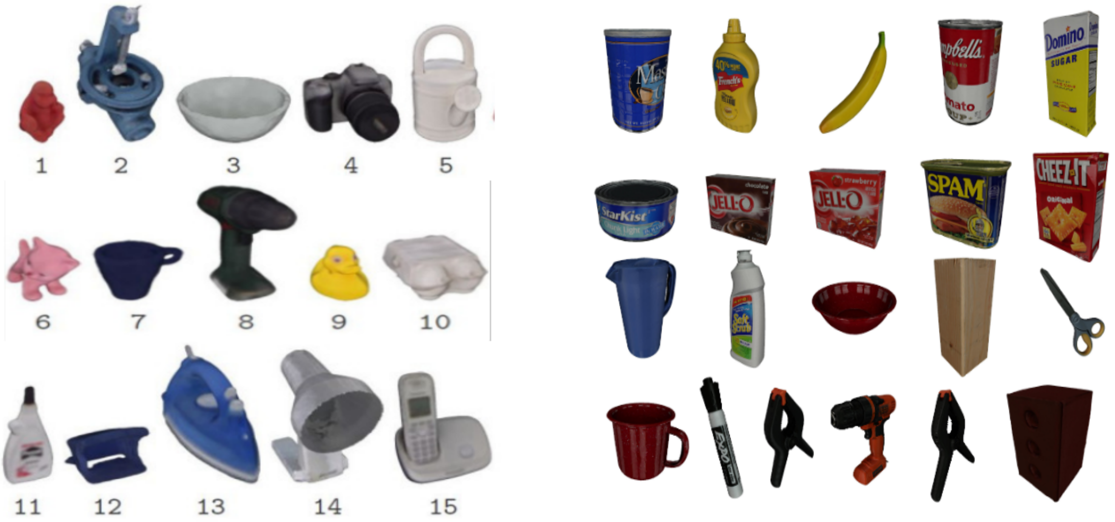
Based on the results that Linemod has a more diverse set of complexities in shapes, we will evaluate the Linemod dataset in this study. The Linemod dataset comprises of 15K images with approximately 1000 images per object. Specifically, each image in the dataset corresponds to one 6D pose estimation for a single object. A sample test image from the Linemod dataset is illustrated in figure 2.7a.

2.2. Evaluation metrics

Evaluation metrics play a crucial role in determining the performance of 6D pose estimation methods. In this study, these metrics are employed to compare and evaluate the performances at varying distances. For non-symmetric objects, the commonly used evaluation metric is the Average Distance of Model Points (ADD), introduced in [10], which measures the mean pairwise distance between transformed points from the estimated and ground truth poses, see equation (2.1). In this notation, the symbols **R** and **T** represent the ground truth rotation and translation, respectively. Similarly, the symbols $\tilde{\boldsymbol{R}}$ and $\tilde{\boldsymbol{T}}$ denote the estimated rotation and translation, respectively. $\mathcal{M}$ denotes the set of 3D points and m is the number of points. The metric computes the mean of pairwise distances between the transformed points from the ground truth and the estimated pose.

$$ADD = \frac{1}{m} \sum_{x \in \mathcal{M}} ||(\boldsymbol{R}x + \boldsymbol{T}) - (\tilde{\boldsymbol{R}}x + \tilde{\boldsymbol{T}})|| \tag{2.1}$$

When dealing with symmetric objects, the evaluation metric used is the Average Closest Point Distance (ADD-S), introduced in [23]. This metric selects the average distance because matching points between



(a) The linemod objects

(b) The YCB-video objects

Figure 2.3: Comparison of the linemod and YCB-video objects

symmetric objects can be ambiguous, see equation (2.2).

$$ADD-S = \frac{1}{m}\sum_{x_1\in\mathcal{M}} \min_{x_2\in\mathcal{M}} ||(Rx_1+T)-(\tilde{R}x_2+\tilde{T})|| \tag{2.2}$$

In the same vein, the ADD(-S) is mainly used to evaluate 3D object tracking, as it considers both symmetric and non-symmetric objects, see equation (2.3). For non-symmetric objects, it calculates the ADD distance metric, while for symmetric objects, it calculates the ADD-S distance metric.

$$ADD(-S) = \begin{cases} ADD & \textit{if asymmetric,} \\ ADD-S & \textit{if symmetric.} \end{cases} \tag{2.3}$$

The pose estimation, accuracy is determined by comparing the ADD(-S) metric with a pre-defined threshold. In [23, 16], the evaluation is considered correct if the ADD(-S) falls below the threshold of 10% of the object's diameter. However, [19, 20, 29] also vary the threshold values and plot an accuracy-threshold curve to visualize the performance of pose estimation over different thresholds relative to the objects diameter. Varying thresholds allows for assessing pose estimation methods across various difficulty levels, with lower thresholds used to evaluate more accurate estimations and higher thresholds for more tolerant evaluation. This approach provides a more comprehensive performance evaluation of pose estimation methods. The area under the accuracy-threshold curve is then computed as a metric for pose estimation accuracy. With this, a single metric can be obtained to quantify the performance of a model.

2.3. Models

For the purpose of evaluating how depth information can enhance pose estimation, the study includes three different models with every model an unique algorithm, which were not selected to determine which one performs best. The study focuses specifically on deep learning based methods because they are found to have the best performances in the benchmarks, thus the algorithms used must be neural-network based and include pre-trained models for practical purposes. Furthermore, the papers should report results on Linemod to enable comparison with the findings of this study. The selection of models is also based on their popularity like number of citations and also the documented performance on the Linemod dataset. By selecting models that have been widely cited and have demonstrated good performance on the Linemod dataset, the study ensures a basis for comparison with existing research. This allows for evaluating the proposed approach against established methods and understanding its relative performance.

1. **PVNet [19]** introduces a model that employs object keypoints, where each pixel learns a vector field and uses it to vote towards a keypoint location. The vectors learned by each pixel indicate the direction in which they believe the corresponding keypoint is located. This voting process is performed for each pixel in the image, allowing the model to gather information from multiple pixels to collectively determine the keypoint locations. The vector field is shown in figure 2.4 (b). After the process of gathering vectors for each pixel, the model utilizes a RANSAC-based voting scheme to localize the keypoints, effectively handling outliers and enhancing the accuracy of keypoint localization (c). The approach utilizes the keypoints and 2D to 3D correspondences to estimate the 6D pose. This method was considered as one of the state-of-the-art methods in 2019 as stated in the paper and also serves as a valuable baseline for future research as it has been extensively studied and compared with other methods with more than 700 citations.

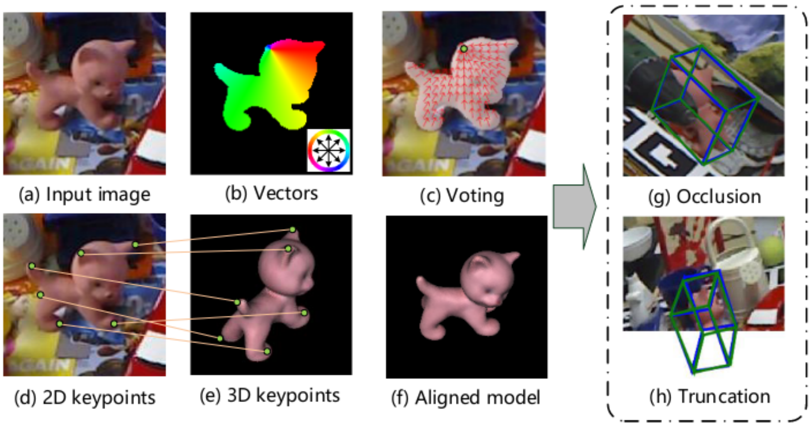


Figure 2.4: The PVNet algorithm visualized

2. **GDR-Net [17]** is a model that does not, unlike PVNet, rely on specific keypoints. The proposed GDR-Net model aims to directly regress the 6D pose from geometric guidance without the need for intermediate steps. This is done by exploiting intermediate geometric features. Given an RGB object crop as input, the network predicts various outputs including 2D-3D correspondences, a mask indicating the visible object, and surface fragment identities that represent specific parts or regions of the object's surface. The 2D-3D correspondences and surface fragments are concatenated and fed in a CNN that directly regresses a translation and 3D rotation. The idea of this network is to leverage both the image features and geometric information. The selection of this model is based on its performance in the BOP challenge of 2022, where it emerged as the winner in most categories. The whole competition can be found at [18].

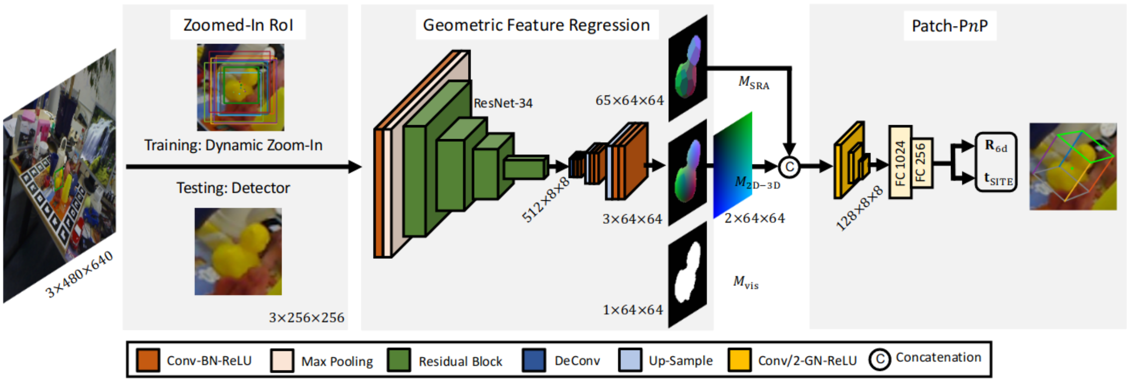


Figure 2.5: The GDR-net framework

3. **EfficientPose [30]** is a model that proposed a scalable model based on the popular EfficientDet [31]. The network utilized a convolutional neural network (CNN) architecture that takes RGB images as input and directly predicts the 6D poses of the objects in the scene. The model is designed to be efficient making it suitable for training on large datasets but also suitable for real time applications. This model had the highest score on the Linemod dataset, making it interesting to see how depth refinement can help with the best model.

The pretrained models for the bowl and cup objects were not provided in many papers, including the ones used for this study. This issue has been noted in other research papers as well, but the reason behind it remains unclear. Additionally, PVNet did not provide additional models for the eggbox, glue, holepuncher, iron, and phone objects.

2.4. Depth refinement

For the depth refinement the Iterative Closest Point (ICP) [32] is chosen. This classic method is still the most used method for depth refinement in the BOP challenge. ICP is a method for point cloud registration, which is a process that tries to allign two (or more) point clouds together. The idea is to refine the initial guess of the transformation that maps one point cloud to another by minimizing the distance between all corresponding points in the two clouds. An example can be seen in figure 2.6 where the blue point cloud would be the (partial) depth map and the green point cloud the initial 6D pose estimation [33].

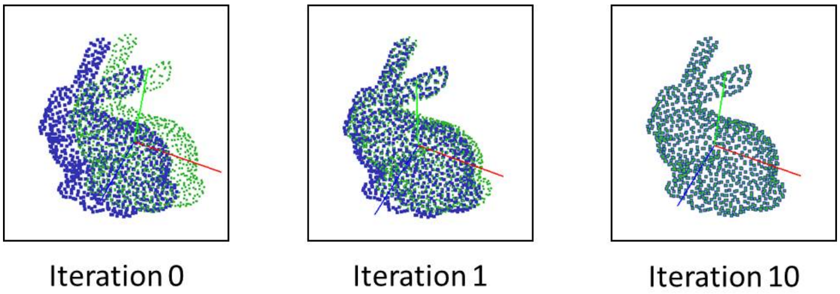
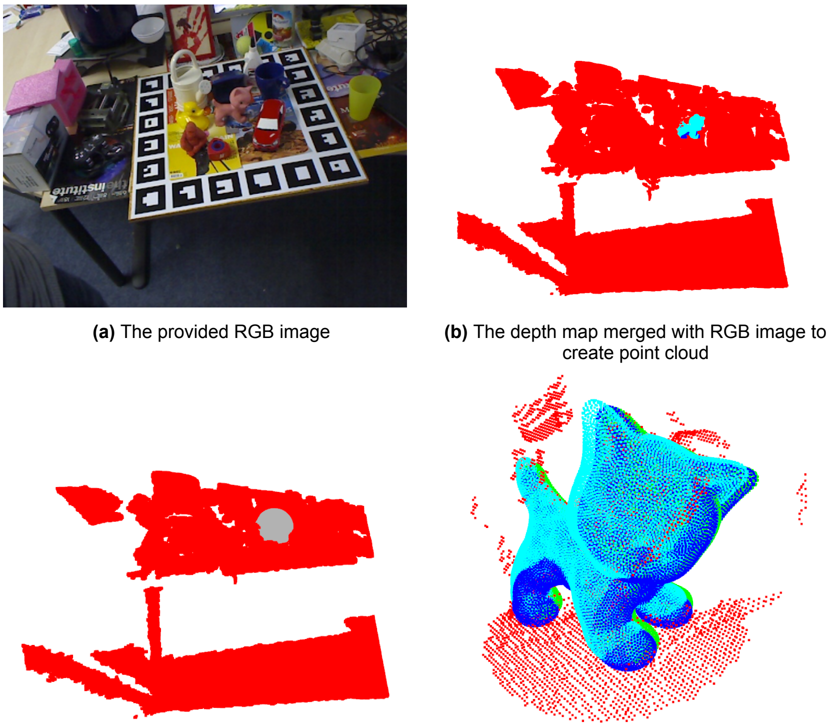


Figure 2.6: The Iterative Closest Point algorithm visualized

For the evaluation, the depth information is enhanced by merging the provided depth map with the RGB image, see figure 2.7a. The depth map specifies the depth value for each pixel, which is used to map each pixel in the RGB image to a corresponding point in space. This merging process results in a point cloud representation of the image, but only for the visible portions of the scene. In figure 2.7b, the points of the scene are highlighted in red. To limit the ICP algorithm to points close to the object, a sphere is created around the initial pose estimation with a diameter relevant to the object. Points outside this sphere are rejected. The initial estimation is shown in cyan in figure 2.7b, the sphere is shown in grey in figure 2.7c. Finally, ICP is performed with a maximum iteration of 30 and the resulting blue points represent the final pose estimation and the ground truth is shown in green in figure 2.7d.



(a) The provided RGB image

(b) The depth map merged with RGB image to create point cloud

(c) A sphere is created to reject points that are not relevant to ICP

(d) The final estimations in cyan and blue, the ground truth in green

Figure 2.7: The depth refinement pipeline

3

Results

In this chapter, the camera distance distribution in the Linemod dataset is analyzed. The results for the three models are then presented using different metrics, without varying the camera distance but with the inclusion of depth refinement. The aim of this analysis is to assess the consistency of the results with prior literature and evaluate the overall impact of depth refinement. Additionally, we conduct an analysis of the results obtained under varying camera distances, specifically focusing on the effects of depth refinement.

3.1. Dataset analysis

The Linemod dataset was designed to ensure that test images are uniformly distributed around the objects within a hemisphere. The camera distances of the test images range from 650mm to 1150mm. figure 3.1 shows the distribution of sample numbers for each distance. The distances in this distribution are calculated based on the ground truth estimations, using the Euclidean distance computed from the x, y, and z translations. These distances are rounded to the nearest 20th value. Additionally, the individual distributions for each object are consistent with the overall distribution, see the distribution of the first two objects at figure A.1.

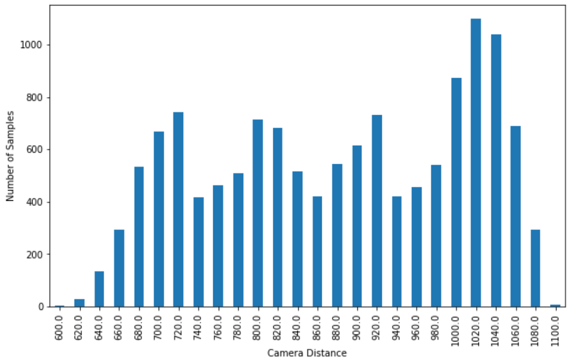
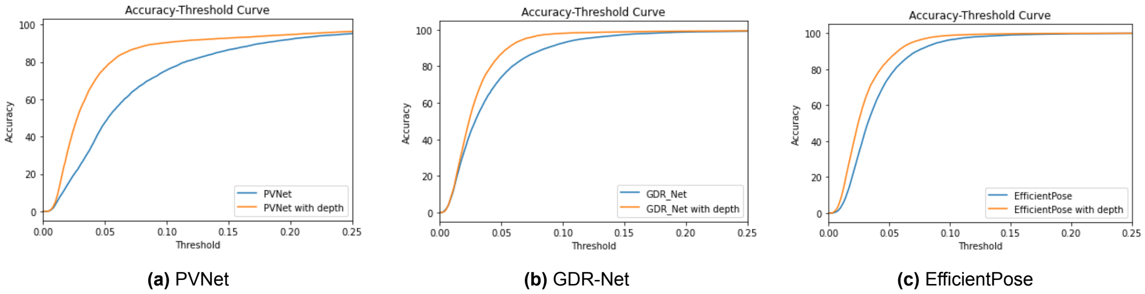


Figure 3.1: The distance distribution of the Linemod dataset at nearest 20th value

3.2. Results

3.2.1. Without varying the camera distance

For the results the classical 10% diameter threshold for each model is calculated. Also to obtain a more precise metric for prediction, the threshold is varied from 0 to 100% with steps of 0.1% and the area under the curve is measured for each object. The accuracy threshold curves for each model for the average of all objects is plotted in figure 3.2. A more detailed table of the results of each individual object can be found at table A.3 and table A.4.



(a) PVNet

(b) GDR-Net

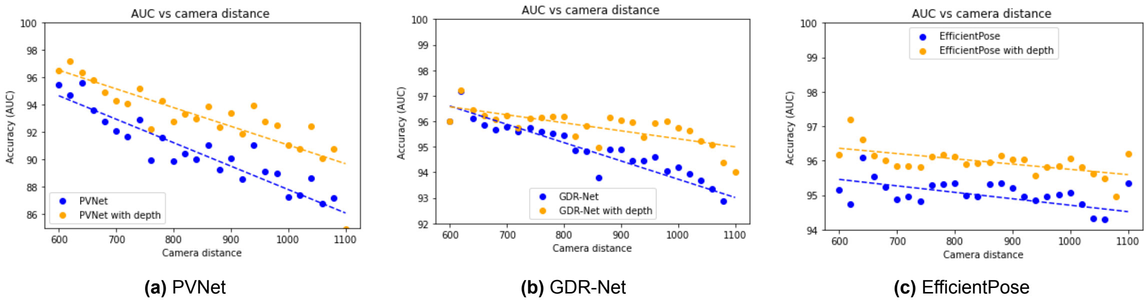
(c) EfficientPose

Figure 3.2: Accuracy-threshold comparison of RGB vs RGB-D for different models

Based on the results obtained using the 10% diameter threshold, it can be concluded that the performance of the models is reliable and consistent with the outcomes reported in the literature of the individual models [19, 17, 30]. The ape and duck were the most challenging objects for the models to estimate, indicating difficulty with objects that have less structure. This is evidenced by the AUC scores shown in table A.4, where the ape had the lowest score, followed by the duck (with PVNet scoring a close third lowest), both with and without depth refinement. It is worth noting that the ape and duck had fewer vertices and edges than the other objects but also lack distinctive color changes, which contributes to their complexity. It can be concluded that adding depth refinement improves the accuracy of the predictions, with the biggest improvements for smaller thresholds. While the newer models achieve near-perfect results with RGB only for a 10% threshold, the addition of depth refinement results in even better performance. Among the three models, PVNet benefits the most from depth refinement, as its initial predictions without depth are worse than the other two models. However, it is interesting to note that the RGB-only methods of the other two models still outperform PVNet with depth refinement, indicating that depth alone cannot fix a bad prediction in this case.

3.2.2. With varying the camera distance

To analyze the results obtained at varying camera distances, plots were created to visualize the relationship between performance and camera distance using the Area Under the Curve (AUC) values. In each plot, a linear dashed line was fitted for further analysis. figure 3.3 displays the AUC/varying distance plots for the average of all objects together. It is important to note that the Y-axis scales differ between the plots. For a more results, individual AUC/varying distance plots for each object can be found in figures A.2 to A.4.



(a) PVNet

(b) GDR-Net

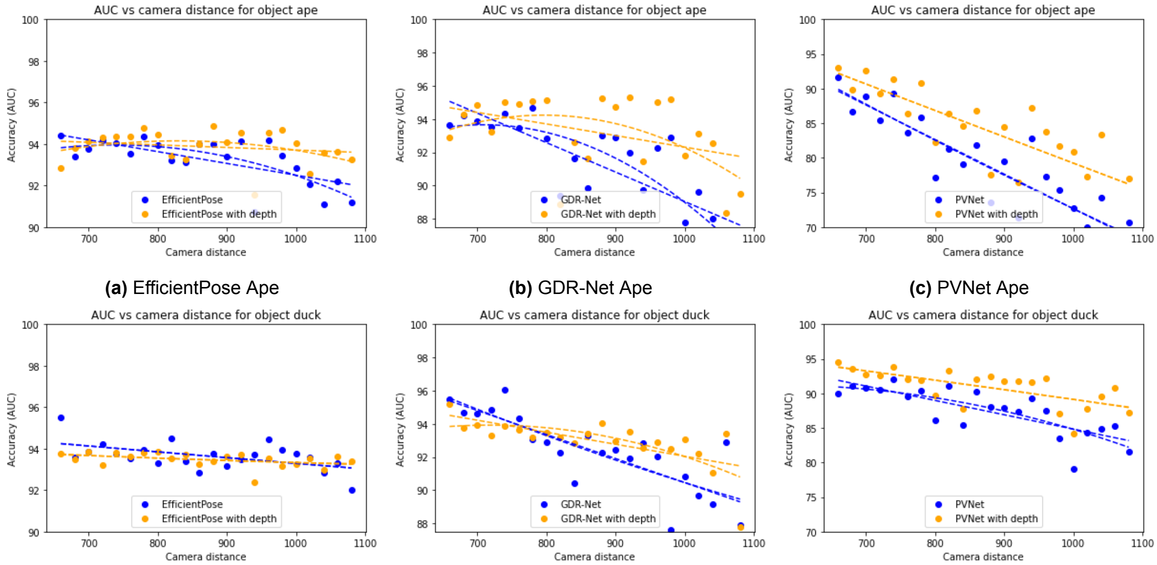
(c) EfficientPose

Figure 3.3: AUCs of different models on varying distances

After analyzing the plots in figure 3.3, two observations were made. Firstly, the estimation becomes progressively worse with increasing depth, which is a logical outcome. This is evidenced by the declining trend line for all three models. Secondly, it was surprising to find that the AUC score for RGB and depth remained linearly dependent for PVNet and EfficientPose as the lines were even parallel. This outcome further indicates that depth alone cannot improve a weaker prediction.

Looking closer at the plots of individual objects in figures A.2 to A.4, it is evident that not every model behaves in the same way. But in general, the more complex objects have steeper lines with only RGB, and the depth line is less parallel in comparison with other objects. This suggests that as the distance

between the camera and the object increases and the object becomes more complex, the estimation using RGB alone becomes significantly worse. However, incorporating depth information can help reduce the steepness of this slope and improve the estimation.
To further analyze the observation that depth can help reduce the steepness of the slope and improve estimation for more complex objects, a second polynomial line dotted line was fitted. This line aims to examine whether the slope increases with larger distances when using RGB alone. The plots illustrating the fitted second polynomial lines for the two most complex objects of the dataset, the ape and the duck, can be observed in figure 3.4. From these plots, it can be concluded that although the slope is generally steeper with RGB alone, there is no significant downturn in a second polynomial trend.



(a) EfficientPose Ape

(b) GDR-Net Ape

(c) PVNet Ape

(d) EfficientPose Duck

(e) GDR-Net Duck

(f) PVNet Duck

Figure 3.4: AUCs on varying distances of ape and duck with 2nd polynomial line

4 Conclusion

In recent years, significant progress has been made in RGB-only pose estimation methods. However, the importance of depth information has not been fully explored, especially concerning varying camera distances. Although some studies suggest the significance of depth information, no prior work has thoroughly investigated this aspect. Studies conducted in the field of object detection recognize the correlation between decreasing accuracy and increasing distance between the object and the detector. These studies indicate that employing higher resolutions can enhance the accuracy in such scenarios. However, it remains to be investigated whether depth cameras, instead of higher resolution cameras, also leads to improved performance in pose estimation tasks with increasing camera distance. To address this, this study aims to answer the following research question: **How does depth information enhance 6D pose estimation when varying the camera distance with known objects?**.

4.1. Conclusion

To answer the research question, three models were evaluated using the Linemod dataset and the results were analyzed based on different evaluation metrics. The findings of this study suggest that performance generally decreases as the distance between the camera and the objects increases. This decline in performance aligns with the literature findings of [21], where a similar decrease was observed. Notably, the performance decline is more pronounced for complex objects, which are objects with fewer vertices and lacking distinctive colors. While relying solely on RGB information shows a more significant decrease in performance for complex objects, no secondary polynomial trend indicating a downturn is observed. Furthermore, incorporating depth information improves the accuracy of pose estimation, as indicated by all the evaluation metrics in the results. However, depth information alone cannot fully compensate for inaccurately predicted poses as the performance of depth also drops at greater distances.

4.2. Discussion and future work

The results of this study present various potential directions for future research, which could enhance and broaden the current understanding of the topic of difference in accuracy at different distances. At first, this study tested several models on a specific dataset, it may be valuable to test these models on other datasets like YCB-V [23] to verify the generalizability of the results on other objects. Additionally, this study only evaluated three models, and future research could involve testing more models to determine if the conclusions drawn are applicable to a broader range of models. Generating more images through simulation for longer distances could also contribute to more robust conclusions with Blenderproc [34]. Secondly, although this study only used topological measures to estimate the complexity of objects, there are other factors that can affect the accuracy of 6D pose estimation, such as the way objects reflect light and color. There exists a framework, as described in Hoang et al. (2005) [35], which facilitates the analysis of color and texture in wavelength-Fourier space. This framework can be utilized for this purpose. Future research could explore the development of metrics that measure the difficulty of estimating 6D pose for objects. This metric would enable researchers to compare the performance of various 6D pose estimation models more accurately and identify areas for improvement. Finally, this study only explored how depth information can enhance 6D pose estimation in scenarios with little or no occlusion. Future work could investigate the effects of depth enhancement in occluded scenarios, where objects may be partially or completely obscured from view like in the Linemod Occlusion [36] dataset.

References

[1] Nikolaus Correll et al. *Analysis and Observations from the First Amazon Picking Challenge*. arXiv:1601.05484 [cs]. Sept. 2017. URL: http://arxiv.org/abs/1601.05484 (visited on 04/04/2023).

[2] Post & Parcel. *Effidence wins DHL Robotics Challenge*. https://postandparcel.info/76645/news/effidence-wins-dhl-robotics-challenge/. Nov. 2016.

[3] Kinemetrix. *How to Make a Robot Pick Up a Part - Introduction to Pick and Place*. Nov. 2019. URL: https://www.youtube.com/watch?v=ZrquLlVzClo.

[4] Yuval Litvak, Armin Biess, and Aharon Bar-Hillel. *Learning Pose Estimation for High-Precision Robotic Assembly Using Simulated Depth Images*. arXiv:1809.10699 [cs]. Mar. 2019. URL: http://arxiv.org/abs/1809.10699 (visited on 05/24/2023).

[5] TUDelft, Dariu M. Gavrila. *3D Machine Vision*. PowerPoint slides. 2022. URL: https://brightspace.tudelft.nl/d2l/le/content/401407/viewContent/2537226/View.

[6] Yingzhao Zhu et al. “A Review of 6D Object Pose Estimation”. In: *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*. Vol. 10. ISSN: 2693-2865. June 2022, pp. 1647–1655. DOI: 10.1109/ITAIC54216.2022.9836663.

[7] Fred Rothganger et al. “3D Object Modeling and Recognition Using Local Affine-Invariant Image Descriptors and Multi-View Spatial Constraints”. en. In: *International Journal of Computer Vision* 66.3 (Mar. 2006), pp. 231–259. ISSN: 1573-1405. DOI: 10.1007/s11263-005-3674-1. URL: https://doi.org/10.1007/s11263-005-3674-1 (visited on 04/04/2023).

[8] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. “The MOPED framework: Object recognition and pose estimation for manipulation”. en. In: *The International Journal of Robotics Research* 30.10 (Sept. 2011), pp. 1284–1306. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364911401765. URL: http://journals.sagepub.com/doi/10.1177/0278364911401765 (visited on 04/04/2023).

[9] D.G. Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2. Sept. 1999, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

[10] Stefan Hinterstoisser et al. “Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes”. en. In: *Computer Vision – ACCV 2012*. Ed. by David Hutchison et al. Vol. 7724. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 548–562. ISBN: 978-3-642-37330-5 978-3-642-37331-2. DOI: 10.1007/978-3-642-37331-2_42. URL: http://link.springer.com/10.1007/978-3-642-37331-2_42 (visited on 04/04/2023).

[11] Eric Brachmann et al. “Learning 6D Object Pose Estimation Using 3D Object Coordinates”. en. In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Vol. 8690. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 536–551. ISBN: 978-3-319-10604-5 978-3-319-10605-2. DOI: 10.1007/978-3-319-10605-2_35. URL: http://link.springer.com/10.1007/978-3-319-10605-2_35 (visited on 04/04/2023).

[12] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. “Fast Point Feature Histograms (FPFH) for 3D registration”. In: *2009 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729. May 2009, pp. 3212–3217. DOI: 10.1109/ROBOT.2009.5152473.

[13] Guoguang Du et al. “Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review”. en. In: *Artificial Intelligence Review* 54.3 (Mar. 2021). Number: 3, pp. 1677–1734. ISSN: 1573-7462. DOI: 10.1007/s10462-020-09888-5. URL: https://doi.org/10.1007/s10462-020-09888-5 (visited on 02/06/2023).

[14] Bertram Drost et al. “Model globally, match locally: Efficient and robust 3D object recognition”. en. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA: IEEE, June 2010, pp. 998–1005. ISBN: 978-1-4244-6984-0. DOI: 10.1109/CVPR.2010.5540108. URL: http://ieeexplore.ieee.org/document/5540108/ (visited on 04/11/2023).

[15] Joel Vidal, Chyi-Yeu Lin, and Robert Martí. “6D pose estimation using an improved method based on point pair features”. In: *2018 4th International Conference on Control, Automation and Robotics (ICCAR)*. Apr. 2018, pp. 405–409. DOI: 10.1109/ICCAR.2018.8384709.

[16] Kiru Park, Timothy Patten, and Markus Vincze. “Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. arXiv:1908.07433 [cs]. Oct. 2019, pp. 7667–7676. DOI: 10.1109/ICCV.2019.00776. URL: http://arxiv.org/abs/1908.07433.

[17] Gu Wang et al. *GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation*. arXiv:2102.12145 [cs]. Mar. 2021. URL: http://arxiv.org/abs/2102.12145 (visited on 03/13/2023).

[18] Martin Sundermeyer et al. *BOP Challenge 2022 on Detection, Segmentation and Pose Estimation of Specific Rigid Objects*. arXiv:2302.13075 [cs]. Feb. 2023. URL: http://arxiv.org/abs/2302.13075 (visited on 04/11/2023).

[19] Yisheng He et al. *PVN3D: A Deep Point-wise 3D Keypoints Voting Network for 6DoF Pose Estimation*. arXiv:1911.04231 [cs]. Mar. 2020. URL: http://arxiv.org/abs/1911.04231.

[20] Chen Wang et al. *DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion*. arXiv:1901.04780 [cs]. Jan. 2019. DOI: 10.48550/arXiv.1901.04780. URL: http://arxiv.org/abs/1901.04780 (visited on 03/02/2023).

[21] Yu Hao et al. *Understanding the Impact of Image Quality and Distance of Objects to Object Detection Performance*. arXiv:2209.08237 [cs]. Sept. 2022. URL: http://arxiv.org/abs/2209.08237 (visited on 05/22/2023).

[22] D.J. Brady et al. “Multiscale gigapixel photography”. In: *Nature* 486 (June 2012), pp. 386–9. DOI: 10.1038/nature11150.

[23] Yu Xiang et al. *PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes*. arXiv:1711.00199 [cs]. May 2018. URL: http://arxiv.org/abs/1711.00199 (visited on 02/27/2023).

[24] Tomas Hodan et al. *BOP: Benchmark for 6D Object Pose Estimation*. arXiv:1808.08319 [cs]. Aug. 2018. URL: http://arxiv.org/abs/1808.08319 (visited on 04/05/2023).

[25] Alexander Bernstein et al. “Topological data analysis in computer vision”. In: Jan. 2020, p. 140. DOI: 10.1117/12.2562501.

[26] Nina Otter et al. “A roadmap for the computation of persistent homology”. en. In: *EPJ Data Science* 6.1 (Dec. 2017). Number: 1 Publisher: SpringerOpen, pp. 1–38. ISSN: 2193-1127. DOI: 10.1140/epjds/s13688-017-0109-5. URL: https://epjdatascience.springeropen.com/articles/10.1140/epjds/s13688-017-0109-5 (visited on 04/16/2023).

[27] M. Garland and P.S. Heckbert. “Simplifying surfaces with color and texture using quadric error metrics”. en. In: *Proceedings Visualization ’98 (Cat. No.98CB36276)*. Research Triangle Park, NC, USA: IEEE, 1998, pp. 263–269. ISBN: 978-0-8186-9176-8. DOI: 10.1109/VISUAL.1998.745312. URL: http://ieeexplore.ieee.org/document/745312/ (visited on 07/05/2023).

[28] Paolo Cignoni et al. “MeshLab: an Open-Source Mesh Processing Tool.” In: vol. 1. Jan. 2008, pp. 129–136. DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[29] Yann Labbé et al. *CosyPose: Consistent multi-view multi-object 6D pose estimation*. arXiv:2008.08465 [cs]. Aug. 2020. DOI: 10.48550/arXiv.2008.08465. URL: http://arxiv.org/abs/2008.08465 (visited on 04/11/2023).

[30] Yannick Bukschat and Marcus Vetter. *EfficientPose: An efficient, accurate and scalable end-to-end 6D multi object pose estimation approach*. arXiv:2011.04307 [cs]. Nov. 2020. URL: http://arxiv.org/abs/2011.04307 (visited on 03/08/2023).

[31] Mingxing Tan, Ruoming Pang, and Quoc V. Le. *EfficientDet: Scalable and Efficient Object Detection*. arXiv:1911.09070 [cs, eess]. July 2020. DOI: 10.48550/arXiv.1911.09070. URL: http://arxiv.org/abs/1911.09070 (visited on 04/12/2023).

[32] “An Iterative Image Registration Technique with an Application to Stereo Vision (IJCAI)”. In: vol. 81. Apr. 1981.

[33] URL: http://www.sanko-shoko.net/note.php?id=3c5m.

[34] Maximilian Denninger et al. *BlenderProc*. arXiv:1911.01911 [cs]. Oct. 2019. URL: http://arxiv.org/abs/1911.01911 (visited on 07/10/2023).

[35] Minh A. Hoang, Jan-Mark Geusebroek, and Arnold W.M. Smeulders. “Color texture measurement and segmentation”. en. In: *Signal Processing* 85.2 (Feb. 2005), pp. 265–275. ISSN: 01651684. DOI: 10.1016/j.sigpro.2004.10.009. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165168404002610 (visited on 07/10/2023).

[36] Eric Brachmann. *6D Object Pose Estimation using 3D Object Coordinates [Data]*. Version V1. 2020. DOI: 10.11588/data/V4MUMX. URL: https://doi.org/10.11588/data/V4MUMX.

A

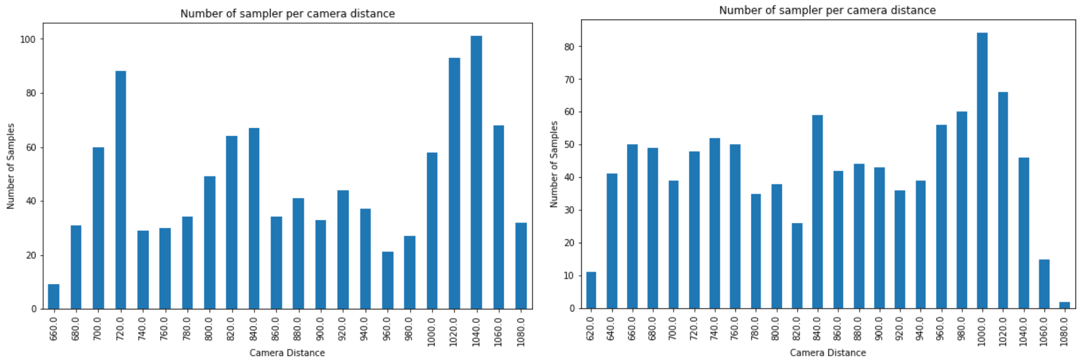
Appendix

Name	Vertices (Orig.)	Edges (Orig.)	Vertices (Simpl.)	Edges (Simpl.)	Diameter
Master Chef Can	9951	25487	4209	12074	172
Cracker Box	8291	24018	4132	12034	269
Sugar Box	8300	24027	4137	12056	198
Tomato Soup Can	8404	24130	4152	12031	120
Mustard Bottle	10983	26504	4926	12860	196
Tuna Fish Can	8227	23954	4097	11975	89
Pudding Box	8331	24058	4135	12050	142
Gelatin Box	8268	23995	4085	12015	114
Potted Meat Can	8342	24067	4144	12030	129
Banana	10710	26334	5050	13031	197
Pitcher Base	11912	27462	5300	13237	259
Bleach Cleanser	8251	23978	4098	11973	259
Bowl	8323	24050	4134	12003	161
Mug	16763	31464	6445	14375	124
Power Drill	9174	24873	4440	12295	226
Wood Block	8920	24642	4397	12347	237
Scissors	8628	24351	4243	12126	203
Large Marker	14043	29330	5772	13736	121
Large Clamp	9825	25511	4761	12657	174
Extra Large Clamp	10064	25742	4834	12733	217
Foam Brick	8455	24179	4198	12124	102

Table A.1: Complexity of original and simplified YCB-V objects

Name	Vertices (Orig.)	Edges (Orig.)	Vertices (Simpl.)	Edges (Simpl.)	Diameter
Ape	5841	17517	2921	8757	102
Benchvise	38325	115000	19162	57486	248
Bowl	40759	122277	20377	61131	167
Camera	18995	56979	9498	28488	172
Can	22831	68499	11414	34248	201
Cat	15736	47202	7869	23643	154
Cup	16573	49731	8284	24864	124
Driller	12655	37959	6328	18978	261
Duck	7912	23730	3957	11865	109
Eggbox	18474	55413	9237	27705	164
Glue	7479	22431	3740	11220	176
Holepuncher	15972	47928	7984	23964	145
Iron	18216	54648	9108	27324	278
Lamp	27435	73596	12340	36844	282
Phone	16559	49671	8280	24834	212

Table A.2: Complexity of original and simplified Linemod objects



(a) Distribution ape

(b) Distribution benchvise

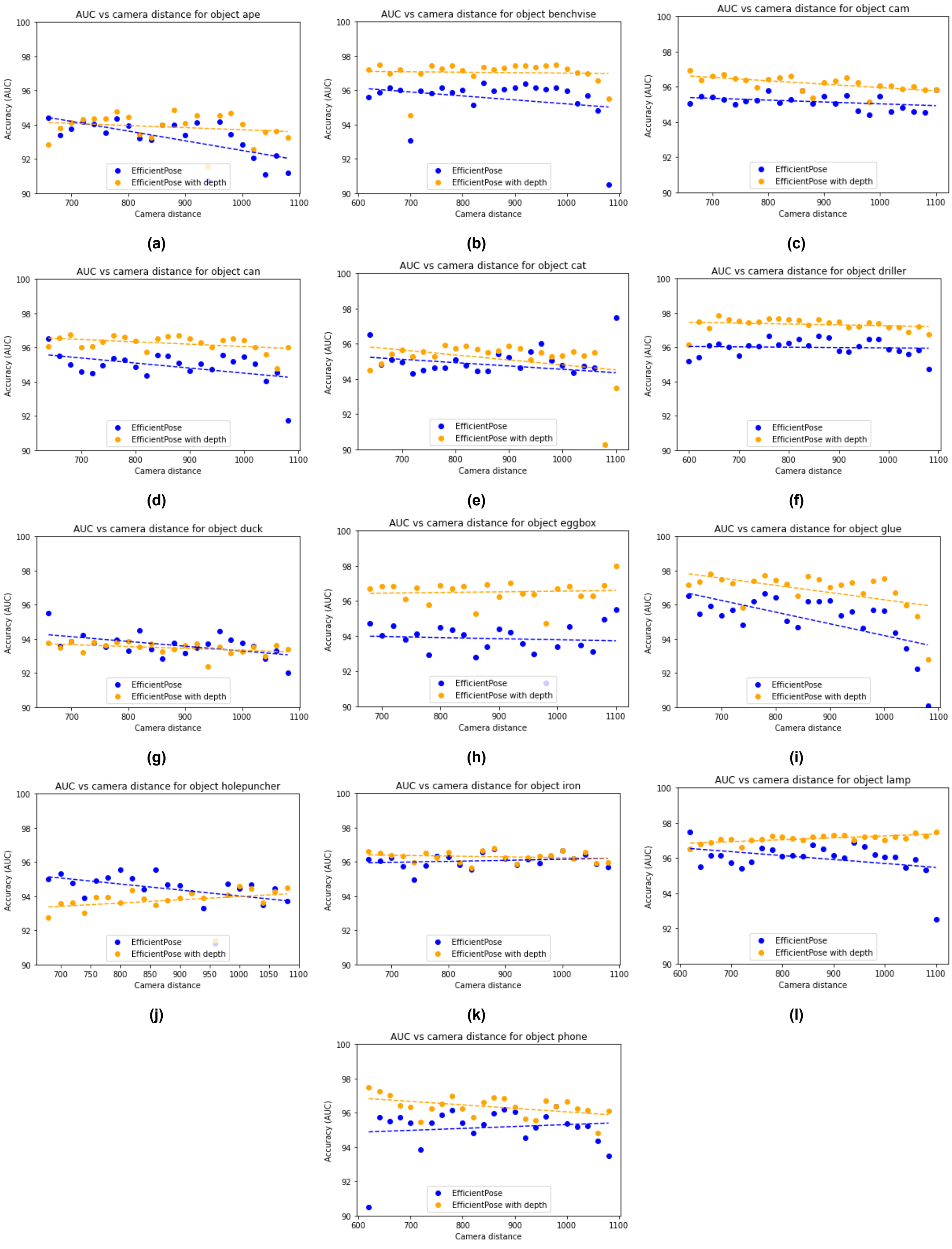
Figure A.1: Distribution of first two objects

Object	PVNet		EfficientPose		GDR-Net	
	RGB	$RGB + Depth$	RGB	$RGB + Depth$	RGB	$RGB + Depth$
ape	45	77	88	96	75	95
benchvise	100	100	100	100	98	99
cam	42	68	98	100	97	99
can	94	100	99	100	98	100
cat	75	94	98	100	93	99
driller	96	97	100	100	98	99
duck	57	86	91	98	81	94
eggbox*	-	-	97	99	99	100
glue*	-	-	98	99	96	99
holepuncher	-	-	95	98	92	99
iron	-	-	100	99	98	99
lamp	95	100	100	100	99	100
phone	-	-	98	98	92	95

Table A.3: Accuracy comparison of different models on 10% of diameter, symmetric objects are marked with *

Object	PVNet		EfficientPose		GDR-Net	
	RGB	$RGB + Depth$	RGB	$RGB + Depth$	RGB	$RGB + Depth$
ape	78.62	84.15	93.91	94.69	91.44	93.72
benchvise	96.81	98.15	96.73	98.06	96.79	97.89
cam	83.37	87.47	96.03	97.12	96	97.06
can	95.36	97.18	95.93	97.21	96.57	97.07
cat	92.48	95.43	95.61	96.34	95.46	96.31
driller	95.55	97.38	97.01	98.32	96.87	98.08
duck	88.12	91.57	94.44	94.37	92.93	93.69
eggbox*	-	-	94.85	97.48	97.01	98.31
glue*	-	-	96.12	97.92	94.66	97.93
holepuncher	-	-	95.29	94.77	95.2	94.9
iron	-	-	97.07	97.24	96.97	97.24
lamp	94.32	97.76	97.03	98.03	97.28	98
phone	-	-	96.29	97.18	95.4	96.63

Table A.4: Area Under the Curve of different models on all objects, symmetric objects are marked with *



(a) (b) (c)

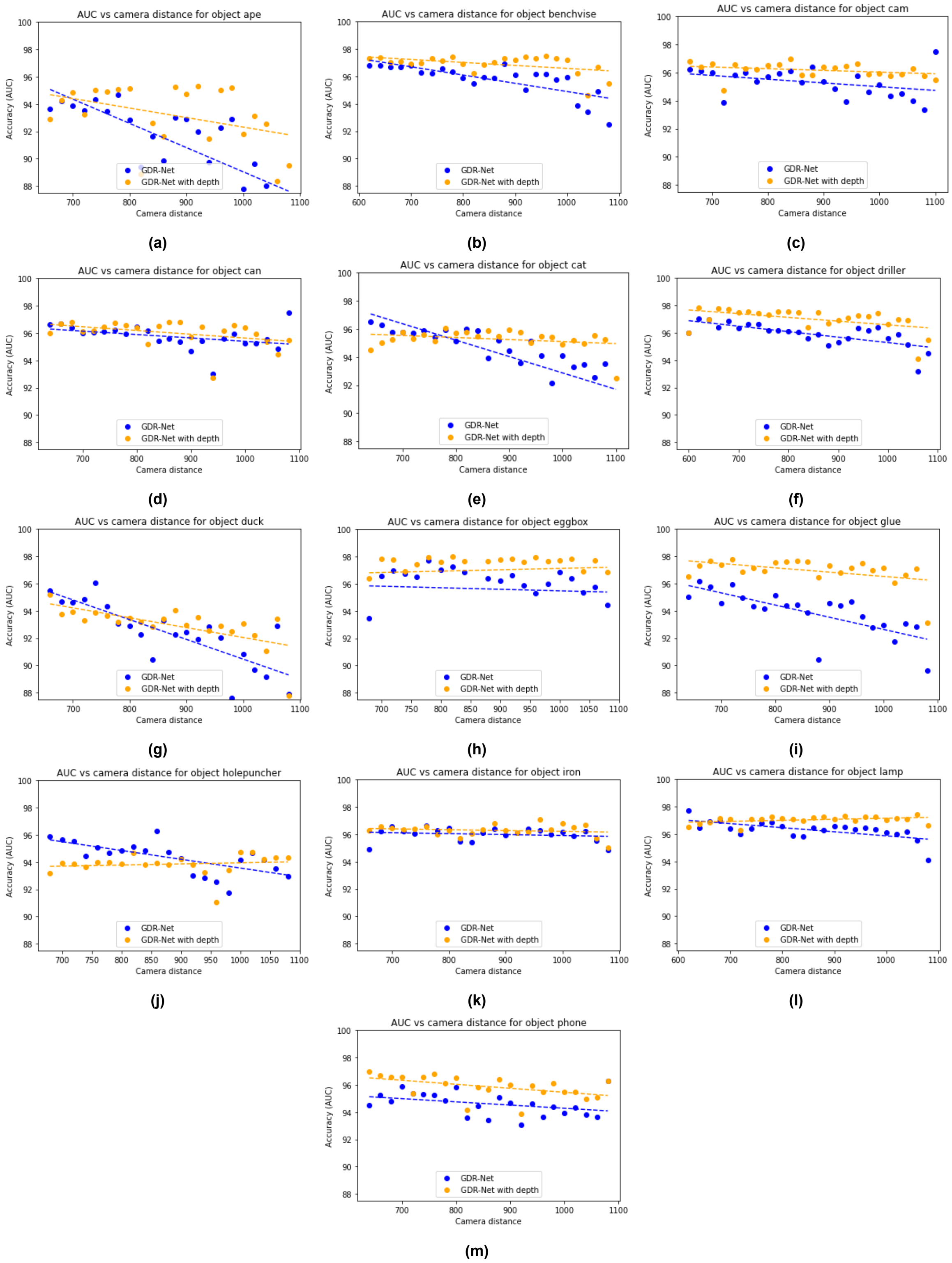
(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m)

Figure A.2: Grid of AUCs for every object at varying distances for EfficientPose



AUC vs camera distance for object ape

(a)

AUC vs camera distance for object benchvise

(b)

AUC vs camera distance for object cam

(c)

AUC vs camera distance for object can

(d)

AUC vs camera distance for object cat

(e)

AUC vs camera distance for object driller

(f)

AUC vs camera distance for object duck

(g)

AUC vs camera distance for object eggbox

(h)

AUC vs camera distance for object glue

(i)

AUC vs camera distance for object holepuncher

(j)

AUC vs camera distance for object iron

(k)

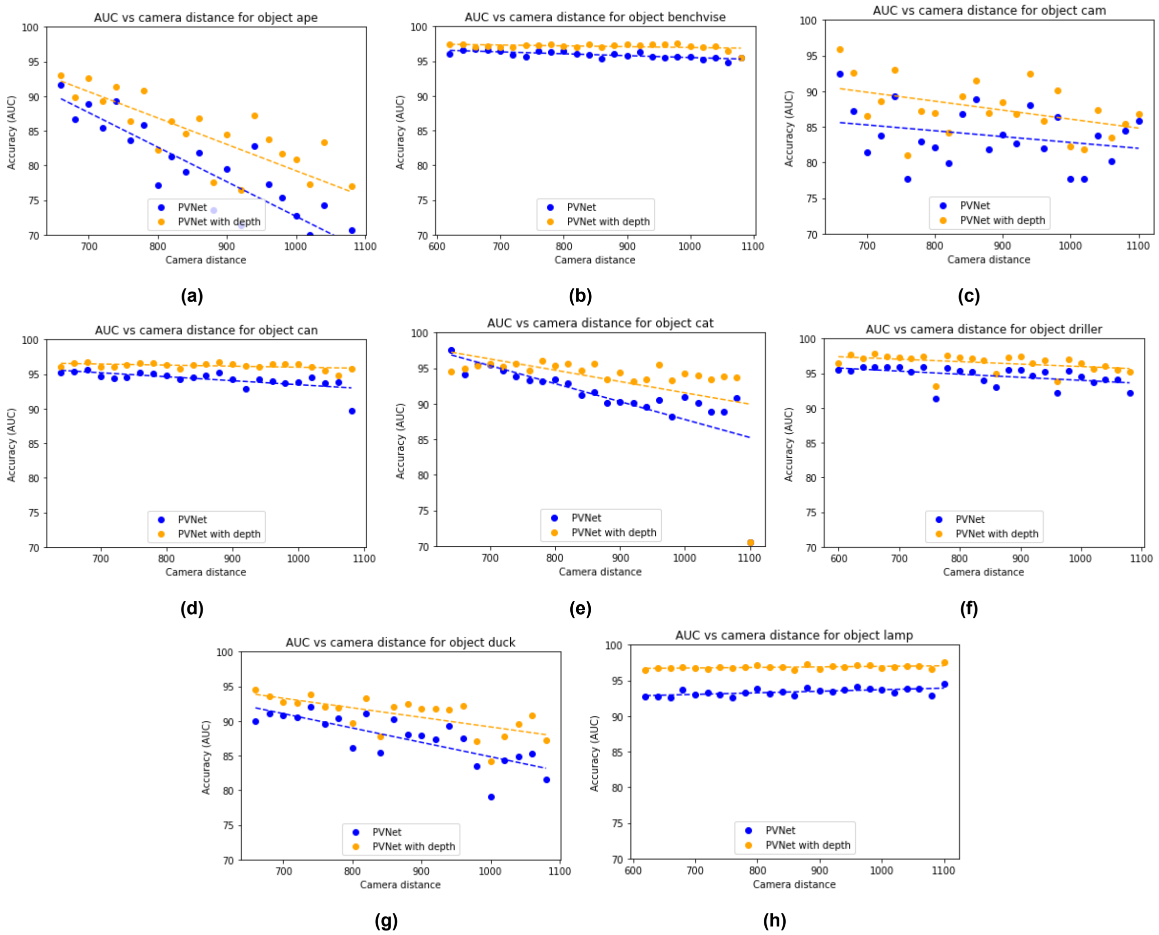
AUC vs camera distance for object lamp

(l)

AUC vs camera distance for object phone

(m)

Figure A.3: Grid of AUCs for every object at varying distances for GDR-Net



(a) (b) (c) (d) (e) (f) (g) (h)

Figure A.4: Grid of AUCs for every object at varying distances for PVNet