Imperial College London
Department of Electrical Electronic Engineering

Accelerating 6D Object Pose Estimation with Scalable Approaches

Pedro Vieira de Castro

Supervised by Dr. Tae-Kyun Kim

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Electrical Electronic Engineering and
the Diploma of Imperial College London, February 2025

Abstract

For real-world applications, extracting 6D object pose of objects requires high accuracy, throughput and scalability. In this thesis, we propose solutions to address these needs.

Firstly, we design a GraphCNN based architecture specifically built for a single object. In contrast to previous works, we explicitly exploit the distinct topological information, 3D dense meshes, of each object in the pose estimation model, with an automated process and prior to any post-processing refinement stage.

Previous methods overlook depth errors caused by the projective nature of the camera sensor. We propose Multi-Frustum Network, which uses a multi-frustum grid in allocentric space as an intermediate pose representation, simplifying the main task and enabling simultaneous detection and pose estimation within the same network for fast performance.

6D object pose projects typically train a network per object but optimizing a model for multiple objects sharply drops its accuracy. We propose applying object conditional parameterizations to a backbone network and distill knowledge from multiple pre-trained single-object teacher models to a multi-object student network.

6D object pose refinement methods rely on a slow *render-compare* pipeline. A rendering with a known object, camera and lighting represents its pose. We propose replacing the rendering step with a more efficient representation which is used in Cascaded Refinement Transformers. We achieve inference runtimes twice as fast as the nearest real-time state-of-the-art methods, supporting up to 30 objects in a single model without sacrificing accuracy.

Lastly, we introduce PoseMatcher, a one-shot object pose estimator that accurately estimates poses of unseen objects using few template images. At each training step, we emulate test-time scenarios by cheaply constructing an approximation of the full object point cloud with a three-view system: a query, positive and negative templates. Moreover, we redesign the transformer matching layer to accommodate different modalities and propose 3D based refinement module.

Acknowledgements

Dedicated to my father, whose love and sacrifices made this possible.

Statement of Originality

I hereby declare that this thesis and the work herein detailed, was composed and originated by myself, except where appropriately referenced and credited.

Copyright Declaration

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution-NonCommercial 4.0 International Licence (CC BY-NC).

Under this licence, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose.

When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes.

Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

Contents

Abstract **i**

Acknowledgements **iii**

1 Introduction **1**

1.1 The 6D object pose estimation task 2

1.2 The challenges of 6D object pose estimation 3

1.3 Thesis Overview 5

1.3.1 Pose Conditioned Mesh Reconstruction 6

1.3.2 Multi-Frustum Network 6

1.3.3 Scaling to multi-object pose estimation 7

1.3.4 CRT-6D: Cascaded Refinement Transformers 8

1.3.5 PoseMatcher: Cascaded Refinement Transformers 8

1.4 Publications 9

2 Background Theory **10**

2.1 Classical Methods 10

2.2 Deep learning based methods 11

2.2.1 2D Projection of 3D Keypoints 12

2.2.2 Dense 3D Keypoint Detection 13

2.2.3 Direct 6D Pose Estimation 15
2.2.4 Template Matching 18
2.2.5 Pose Refinement 19
2.2.6 Conclusion 21
2.3 Datasets and Evaluation Metrics 21

3 Pose Estimation by Mesh Reconstruction 25

3.1 Related Work 27
3.2 Methodology 27
3.2.1 Fine Object Localization 29
3.2.2 3D Mesh Reconstruction 32
3.2.3 Allocentric Pose Estimation & 6D Egocentric Pose Lifting 33
3.3 Experimental Evaluation 34
3.3.1 Experimental Setup 36
3.3.2 Evaluation Results 38
3.3.3 Aditional Discussions 39
3.4 Conclusion 41

4 Multi-Frustum Network 42

4.1 Related Work 44
4.2 Methodology 45
4.2.1 Detection Stage 45
4.2.2 Perspective Warping 46
4.2.3 Single-Frustum Generation 47
4.2.4 Multi-Frustum Generation 50
4.2.5 3D-3D NOCS correspondence 52

4.3 Experimental Evaluation 53
4.3.1 Evaluation Setup 53
4.3.2 Evaluation Results. 54
4.4 Conclusion 59

5 Scaling to multi-objects **60**
5.1 Related Work 61
5.2 Methodology 62
5.3 Experimental Evaluation 65
5.3.1 Evaluation Results 66
5.4 Conclusion 68

6 CRT-6D: Cascaded Refinement Transformers **69**
6.1 Related Work 72
6.2 Methodology 72
6.2.1 Coarse Pose Estimation 73
6.2.2 **O**bject **S**urface **K**eypoint **F**eatures - OSKF 74
6.2.3 OSKF Pose Transformer (OSKF-PT) 75
6.2.4 Objective Functions 77
6.3 Experimental Evaluation 78
6.3.1 Experimental Setup 79
6.3.2 Evaluation Results 80
6.3.3 Ablation studies 84
6.3.4 Qualitative Results 86
6.4 Conclusion 86

7 PoseMatcher **88**

7.1 Related Work . . . 89

7.2 Method . . . 90

7.2.1 Template Based Training . . . 90

7.2.2 IO-Layer: An object-image attention layer . . . 92

7.2.3 Object Pruning . . . 94

7.2.4 Pose Refinement . . . 95

7.3 Experiments . . . 97

7.3.1 Experimental Setup . . . 97

7.3.2 Evaluation Results . . . 98

7.4 Conclusion . . . 103

8 Conclusion and Future Work **104**

8.1 Contributions Overview . . . 104

8.2 Future Directions . . . 107

8.3 Epilogue . . . 108

Bibliography **108**

List of Tables

3.1 **ADD-(S) results of our experiment on the LINEMOD dataset.** Our method outperforms all prior methods w.r.t. ADD-(S) accuracy falling short on the 2D projection metric. Symmetric objects are denoted with a *. 36

3.2 **ADD-(S) comparison of our approach on the OCCLUSION dataset with state-of-the-art.** Our method outperforms prior state of the art, specially when it comes to 2D projection accuracy. Symmetric objects are denoted with a *. The method by Oberweger et al. [86], denoted with a ‡, is evaluated on a combined dataset of OCCLUSION and synthetic data, which is not public. 37

3.3 **Comparison of YCB-Video dataset results with state-of-the-art approaches.** Our method outperforms prior state of the art by a significant margin in all metrics. While the method by Oberweger et al. [86], denoted with a *, is evaluated along with synthetic test data, we still outperform it on YCB-V. 38

4.1 **Comparing ADD-(S) results on Linemod.** Our method outperforms previous RGB only methods. Both CDPN [74] and GDR-Net [137] rely on an external object detector while MFNet detection and estimates pose with the same model. 55

4.2 **ADD-(S) lower threshold comparison.** Our translation precision is far superior than GDR-Net[137] however our accuracy for low threshold *AD* is lower. 55

4.3 **Translation error evaluation and comparison study.** Evaluate the impact of the design of **MFNet** in translation errors compared to previous state-of-the-art. AUC refers to the area under the curve of translation errors with a threshold up to 2cm. . . 57

4.4 **YCB-V study.** . We report results of MFNet on YCB-V and compare it with recent non-refinement methods. 57

4.5 **Runtime comparison.** Comparing with monocular RGB only methods (including detection) measured on a GTX 1080Ti, or equivalent, except GDR* [137] which is measured on a GTX 2080Ti. 59

5.1 **Linemod Ablation study using Sock et al. [118].** Showing the impact of the combined solutions using ADD metric on LINEMOD dataset. 66

5.2 **Ablation study:** Different model capacities on the T-LESS dataset, reported with the VSD metric. The numbers in the bracket indicate the number of teacher models compressed into each student model. 68

6.1 **Comparison study on LM-O.** The results for the LM-O dataset under the ADD(-S) metric are presented and compared to other state of the art methods. In terms of accuracy, **CRT-6D** is outperformed only by ZebraPose [120], a method with an inference time of ~ 191ms for a single object, while **CRT-6D** estimates the pose for all objects in a single LM-O image (~ 8 objects) in 36ms. Best results are bolded while second best are underlined. *N.S.O.* refers to the number of objects supported by a model and * denotes symmetric objects. 81

6.2 **LM-O and YCB-V datasets under the BOP standards [49].** We present the results for all metrics used on the challenge with *Mean AR* referring to average *AR* over both datasets. Best results are **bolded** while second best are underlined. 82

6.3 **BOP challenge results.** At the time of writing, CRT-6D is the fastest method on the BOP challenge website achieving state-of-the-art results, with its average runtime being than 60ms for all objects in an image. 82

6.4 **Effects of the Cascaded Pose Refinement**. We present the improvement with each refinement step. We can observe the diminishing returns where the first refinement improvement is larger than all other iterations combined. 84

6.5 **Initial Pose ablation study.** The experimental results are presented, demonstrating the effects of applying the proposed refinement on top of prior art. The initial poses are shown in the top three rows, while the refined poses can be found in the bottom three rows. An * indicates the utilization of the refinement technique from CRT-6D on the estimated pose of the respective method. 86

7.1 **Comparison study on Linemod.** We present the results for ADD(-S) metric and compare them to state of the art. While Linemod is close to saturated for fully supervised methods, one-shot pose estimation is still challenging. PoseMatcher achieves the best results for all objects for the one-shot category, surpassing self-supervised methods and close to fully supervised methods. Best results for one-shot are bolded. * denotes symmetric objects. 98

7.2 **Ablation studies for each component.** We present the results on Linemod and YCB-V measured by ADD-(S). The first line are the results gathered from OnePose++ [122] The model used for the second and third rows refer is identical to the OnePose++ [44] model trained with our three-view based pipeline. 99

7.3 **Number of templates.** We present the results on Linemod for $\%$ of training images used. Each object has around 180 images therefore 10% is only 18 templates to use. . 103

List of Figures

2.1 **Hough Voting by Doumanoglou et al. [28].** RGB-D patches are extracted and fed into a Sparse Autoencoder. The bottlenecked features produced by the autoencoder are forwarded to a Hough Forest, which then generates the final hypotheses through Hough voting space. Joint registration is used to refine the hypotheses which can then be used to we estimate the next-best-view. 11

2.2 **NOCS representation by [90].** This dense representation extends 3D keypoint detection to every visible surface area of an object in an image. With the 2D-3D correspondence map, the 6D pose can be extracted by solving the PnP problem, usually assisted by RANSAC based outlier removal. 14

2.3 **Dense binary encoding in ZebraPose [120].** Su et al. [120] generates a hierarchical encoding that is efficiently densely estimated and for each encoding a 3D keypoint correspondence can be retrieved from a lookup table. 15

2.4 **Allocentric vs. egocentric orientation differences by Kundu et al. [66].** (a) illustrates the difference between cars in an image that share the same egocentric orientation with respect to the camera but exhibit significant changes in appearance, versus cars that share the same allocentric orientation and exhibit no appearance changes. (b) refers to the representation used and transformation applied to recover the egocentric pose.) 16

2.5 **Direct pose estimation through intermediate representations by Wang et al. [137].** The pipeline of GDR-Net generates the most widely used dense intermediate representations and uses them to generate a final pose, which means that the pipeline can be trained end-to-end, along with intermediate representation supervisions. 17

2.6 **Refinement diagram by Zakharov et al. [153].** Taking as input the query image and a rendering of an approximate pose, the offset rotation and translation are extracted. This process can be repeated multiple times until satisfied. 20

2.7 **Illustration of several datasets contained by the BOP challenge Hodaň et al. [49].** The first row refers to 2D detections while the second one illustrates pose estimation. From left to right, the examples belong to the following datasets: YCB-V[13], HomeBrewed6D[61], MVTec ITODD [31] and T-Less [48]. 23

3.1 This method recovers the 6D pose of an object from a single image by means of an intermediate object reconstruction. From an RGB image of a detected object (a), a pose conditioned mesh (b) is reconstructed from which 6D pose can be extracted (c). (d) and (e) present alternative viewpoints of reconstruction (b) while (f) displays the groundtruth mesh. 26

3.2 **Overview of the pipeline the object topology is exploited for 6D pose estimation.** Given an RGB input image $\mathcal{I}$, an object region of interest (ROI), denoted as $\mathcal{I}_{ROI}$, is obtained by cropping with a fine-tuned 2D object detector (depicted in orange). This cropped region is employed to precisely localize the object by estimating a binary object mask, $\hat{\mathcal{H}}_m$, and a heatmap, $\hat{\mathcal{H}}_c$, for the object centroid (depicted in pink), which are subsequently used to reconstruct a pose-conditioned 3D mesh, $\hat{\mathcal{M}}$, of the object (depicted in green). An allocentric orientation, $\mathcal{R}_a$, of $\hat{\mathcal{M}}$ with respect to a corresponding canonical object mesh, $\mathcal{M}$, can be estimated in a differentiable manner (indicated in purple) by determining the relative rotation of $\hat{\mathcal{M}}$ to the corresponding canonical object mesh, $\mathcal{M}$, through Procrustes alignment. The cropped image locations, $\hat{\mathcal{H}}_m$, $\hat{\mathcal{H}}_c$, and $\mathcal{R}_a$ are later utilized to estimate the complete 6D pose of the object. The symbol $\bigodot$ signifies the channel-wise concatenation operation. 30

3.3 **Qualitative results obtained utilizing the reconstruction intermediate representation.** For every example: (a) estimated mask and centroid projection, (b) reconstructed pose-conditioned mesh, and (c) groundtruth (in green) and our estimated (in blue) pose visualized on the input image. The last column features failure examples, which typically occur due to poor $\hat{\mathcal{H}}_m$ and $\hat{\mathcal{H}}_c$ estimations or reconstruction errors caused by occlusion, extreme illumination conditions, or ambiguous viewpoints. It is important to note that the mask and centroid projection are blended on the same image, and test images are zoomed in for improved visualization. 35

3.4 ADD-(S) comparison of our approach by using groundtruth priors for the centroid and the mask estimation intermediate steps on LINEMOD. 39

3.5 2D Projection accuracy using different priors on LINEMOD. 39

3.6 Histogram of mean pose losses L_a over standardized mesh reconstruction error L_v thresholds. This measurement reveals a directly proportional relationship between the mesh reconstruction error, which can be evaluated at inference time, and the predicted pose error, which is unknown at that stage. 40

4.1 **MFNet motivation.** The motivation for this work is illustrated using a 2D NOCS representation in a crop, similar to the approaches in [138, 137, 25, 58]. A ground truth pose (a) is taken and perturbed in terms of translation parallel to the camera plane (b), distance to the camera (c), and rotation (d). All perturbations result in a final ADD-(S) error of 17mm (the standard threshold error for the object). When applying the common mean squared error(MSE) loss for optimizing NOCS-based methods, it can be observed that the magnitude of the loss varies for all perturbations. The priority becomes evident: the network first learns 2D detection, followed by object rotation, and only then the distance to the camera. 43

4.2 **Detection Pipeline.** Diagram of the first stage of **MFNet**, where the object is found and cropped using PCL [151]. 47

4.3 **Choosing frustum focal lengths.** This figure illustrates the need to carefully consider the focal lengths MFNet should use to generate. We illustrate in 2D for ease of understanding. a shows that high focal length does not cover the full search space, while b creates a sparse grid. Instead, we chose c and d, a structure where multiple high resolution short frustums are generated at different positions, therefore maintaining a large search scope with high resolution. 49

4.4 **Multi-Frustum inference illustration.** Our method starts by generating, from an allocentric virtual camera, a 3D NOCS representation on a Multi-Frustum grid (in red). We then project the Multi-Frustum grid back to egocentric space where the Kabsch algorithm can be applied to retrieve full 6D pose (in blue). 51

4.5 Translation accuracy curve on Linemod at multiple thresholds. 56

4.6 **Qualitative Results.** We show some qualitative results of MFNet on the Linemod dataset and YCB-V dataset. Predicted poses are shown by their light blue bounding boxes while groundtruth are color-coded with red. 58

5.1 **Diagram of CIN conditioning.** This diagram shows our compression pipeline from multiple networks to a single network by Conditional Instance Normalization (CIN) [32] and Knowledge Distillation (KD) [47]. 63

5.2 **BB8 Ablation Study.** (a,c) Relative performance comparing the single model (upper bound, UB) to a naive extension to multi object (lower bound, LB) and our multi-object proposed methods (b,d) Relative performance of BB8 combined with self-supervision, trained with self-supervised single-object as teachers and synthetic data. (a,b) refer to the 13 LINEMOD dataset, while (c,d) refer to T-LESS. Best viewed in colour. 67

6.1 **Illustration of CRT-6D.** The feature pyramid is initially generated, and the last feature level is utilized to produce a pose estimation P^0, which serves as the initial reference for the refinement module. **(a)** With P^0, a set of surface keypoints $\mathcal{S}$ is projected into the image plane. **(b)** The deformable attention mechanism employs the location of the set of projected keypoints to sample the feature pyramid. **(c)** Self-attention is then performed over the sampled feature set, and an offset pose ΔP is estimated, such that $P^t = \hat{P}^{t-1} + \Delta P^t$, where t represents the refinement step. 71

6.2 **Illustration of OSKF-PT.** OSKF-PT receives as an input the previous Q^{t-1}, P_{6D}^{t-1} and $\mathcal{F}$, the last one being the same on every iteration. It outputs a new refined pose P_{6D}^t and Q^t, an initial query to guide the next iteration. 75

6.3 **Comparing the inference time between CRT-6D and other state-of-the-art methods.** CRT-6D is the fastest method, even after three refinement steps, compared to previous approaches while maintaining highly competitive results relative to methods that are 10 times slower. The average time required to estimate the pose for all objects in an image (approximately 4.75 objects per image on YCB-V) is measured. It should be noted that ZebraPose [120] and SurfEmb [41] have been excluded from the comparison due to their significantly longer runtimes, as the pose estimation takes over 250ms and 2000ms per crop, respectively. 83

6.4 **Experiments over number of OSKF used.** While there is a noticeable decline in performance when using only 8 keypoints, the differences become minor as the number of keypoints increases. The optimal configuration employs 64 keypoints for both datasets, striking the right balance between performance and complexity. CRT-6D is not reliant on a large number of keypoints. 84

6.5 **Qualitative results on refinement steps.** The progress in pose accuracy over iterative refinements is demonstrated, with the initial coarse pose in the first row and the final results in the last row. Despite the initial pose being of low quality according to evaluation standards, it is found to be sufficient as a reference for CRT-6D. 85

6.6 **Deformable attention sampling locations.** The red circle indicates the reference point while the white circles indicate deformed sample positions. On the last column, we show attention sampling for the same object and keypoint, with the first row having the keypoint visible whereas on the bottom two the keypoint is self or externally occluded. When the keypoint is occluded, attention learns to sample clues elsewhere on the object silhouette. 87

7.1 **Diagram of PoseMatcher.** At training time we sample from the synthetic training dataset and from two opposing views templates we construct a partial point cloud. At test time, a novel unseen object is used instead. We extract local features from both the query and the templates. Using our novel IO-Layer, we compute dense correspondences between the image and the object. We then take the matches (yellow star) and further refine them using 2D and 3D techniques. 90

7.2 **IO-Layer architecture.** We improve on the attention modules used by LoFTR and OnePose++ [121, 122]. We have modality specific projections for image features embedded with 2D positional encoding and for object template features with its 3D positional encoding. We found that separating the two modalities improves the final pose accuracy. 93

7.3 **Difference between refinement operations.** We can see that for all objects in Linemod, the 3D-refinement has a big impact on lower thresholds. 100

7.4 **Visualization of matching keypoints.** In this figure, we show PoseMatcher output $\mathcal{C}$, where we assigned a normalized coordinate of the highest confidence keypoint to the corresponding pixel. The last two rows are examples from the YCB-V dataset. . . 101

7.5 **Pruning ablations.** On this ablation study, we show that PoseMatcher does not degenerate when aggressively pruned. Since keypoints have very distinct descriptors, PoseMatcher can early on remove self or externally occluded keypoints or keypoints that fall on a textureless area. 102

Chapter 1

Introduction

The world is increasingly becoming automated as humans shift their focus from replacing themselves in dull and repetitive tasks to now replacing themselves in the automation of more complex and dynamic processes. Services such as banking, accounting, and information search have been at the forefront of innovation, with automation enabling these services to be delivered through virtual platforms that can be both deployed and accessible from anywhere in the world. However, humans require fundamental services and products that cannot be abstracted through software, such as food, transportation or housing, and therefore require a physical machine that can dynamical interact with us and our world, which we commonly call a *robot*. In order to successfully build a *robot*, it needs to absorb information about the current state of the its surrounding environment and adjust its behaviour accordingly. One way we can feed environmental information is with images, which captures a 2D representation of the world. We as humans know that navigating our 3D world with just 2D images is possible and therefore aim to build machines that can mimic our performance. Human beings take for granted the ability to extract information from images but it poses significant challenges when designing an automated system. Computer vision is the field that studies the ability for a computer to perceive the world from an image. This thesis will focus on solutions to the 6D object pose estimation problem.

In this thesis, we particularly aim to estimate the pose of rigid and static targets. A rigid object is defined as an object which is not subject to deformations or joint displacement. Static objects are those whose appearance do not dynamically change. The changes in visual appearance may arise

from various textural attributes, in a television screen, internal light sources, in a lamp, or reflective surfaces, such as a mirror. This complex task is an extension of wider object recognition problems such as the more common image classification [43, 126] where the category of the subject of the image is retrieved and object detection [42, 78, 102] where the 2D location, i.e. the position within the image, of the target is predicted. The latter can be further extended to object segmentation [42] where a pixel-wise silhouette of the target is determined. Solving this task is imperative as this step is an indispensable component of the infrastructure of an intelligent software system applied to a plethora of different fields from autonomous driving and manufacturing automation to healthcare. With Augmented Reality (AR) becoming popular in both entertainment and industry work, the increasing need to correctly identify the pose of objects in the real world with accuracy, fast throughput and on wearable devices' hardware, is pushing development of new 6D object pose estimation techniques.

1.1 The 6D object pose estimation task

The 6D object pose estimation task is a narrow computer vision problem where the orientation and translation of an object must be retrieved from a single image. Each component has three degrees of freedom, hence the name. Translation refers to the 3D position of the object in world or camera coordinates reference frame. Recovering 3D translation differs from 2D object detection as the former requires the estimation of the distance from the camera to the target object while the latter simply recovers the region where the projection of the object is located in pixel space. Most 6D pose estimation methods [137, 25, 118, 98, 67, 12, 124, 148], including most of the ones described in this thesis, make use of an initial 2D detection and crop in order to narrow the information consumed by the pose estimator in order to reduce complexity and background noise. However, while 2D detection for object pose is an active area of research, it is outside the scope of this thesis.

We define three different 6D object pose estimation problems: instance-level, category-level and one-shot. Instance-level 6D object pose estimation tackles estimating the pose of known static and rigid objects. We design the pose estimator with full information about the target objects which can be provided in the form of 6D pose labeled training images, in RGB (color) or RGB-D (color with aligned depth information) format, object 3D models from which one can recreate synthetic

examples. On the other hand, category-level allows for the target object to be subject to deformation and texture change within a narrow scope [154, 34, 138, 145]. All objects belonging to a category can be reconstructed by slightly deforming a canonical object [154]. These objects share intra-class semantic similarities which allows a pose estimator to identify these regions and subsequently the pose of the object. Regardless of the paradigm, the intrinsics of the camera must be known at test-time, however it does not have to match the intrinsics of the sensor used to generate training data. In this thesis, we pay specific attention to the problem within the instance-level paradigm with exception to of last chapter, where we tackle the problem in the one-shot paradigm.

The pipeline of 6D object pose estimator is usually made up of 2 stages: detection and pose estimation. This pipeline brings several advantages. Firstly, we can leverage fine-tuned off-the-shelf detectors [78, 103, 102] which have been extensively proven and solely focus the research efforts on 6D object pose estimation. Secondly, a rough detection of the object can be used to crop a new image around the position of the target object thereby removing any unwanted background noise. Not only will the new image will be centered around the object, within a degree of acceptable error, but most of the image region will be occupied by the projection of the object . Moreover, detection methods do not require high precision due to the fact that the following pose estimator will output a pose relative to the detection output. As a peripheral advantage to region of interest cropping, it becomes cheaper to train multiple pose networks and perform ablations studies, seeing as the spatial size of the input can be considerably smaller [74, 137, 15, 136, 98, 124, 118].

1.2 The challenges of 6D object pose estimation

On top of having to retrieve the pose given only the projected appearance of an object, a successful 6D object pose estimation method must be robust to additional challenges that arise from its own problem nature. Without tackling all these complex problems, the performance of any method will be degraded. We will split these issues into 2 distinct classes:

- Classical challenges: arise from the environment or from the object itself.
- Practical challenges: arise from the consequences of the pose estimator design.

Classical Challenges. When taking a picture of an object, camera related factors can result in the same object being captured with difference appearances. Assuming that the camera characteristics do not change (same aperture, focal length, image resolution, etc.), multiple captures might have different forms of blurriness due to movement, or a dirty lens, or sensor noise in the form Gaussian or salt-and-pepper noise. An image can also be subject to external environmental factors that might alter the aspect of an image. The lighting of the scene can change and so can its background, due to movement of the camera or of the object itself. However, arguably the most difficult one, which has been the subject of extensive research works [86, 102, 90, 25], is occlusion. Occlusion occurs when a foreground obstacle partly blocks the target object or when a region of the target object lies outside the camera projection frustum, which is commonly referred as "truncated" and it can also happen when inaccurate cropped regions of interest are used. By removing regions of the target object, it is harder to recover textures as well as a holistic representation of the object, resulting in deteriorated estimations. An extreme extension of the occlusion problem occurs when multiple objects of the same category or instance are the foreground obstacles, although this is only a concern in a set of limited scenarios, such as industry pick and place operations [116, 28]. On the other hand, the shape and texture of an object can also impact the difficulty of estimating its pose. Objects displaying symmetric properties can create problems by displaying similar properties in two or more image regions. Even with a holistic representation, regressing the pose becomes an ill-pose problem as the image to pose correspondence becomes a one-to-many mapping. Some methods have proposed surpassing this issue by defining the problem as a classification problem [153, 63, 51]. Another way some have dealt with this problem is by designing a symmetric aware per-object loss function [138, 137, 74], however this requires the symmetry properties to be hand labeled. Additionally, since data-driven approaches rule state-of-the-art, the quality of the data is important. Capturing high quality annotated data is not an easy task and its acquisition requires human effort and a specialized setup [48, 45]. Instead, methods might rely on generating realistic synthetic data to replace the expensive real-data [23, 50]. However, these synthesized cannot match the exact appearance of real data, whether it is due to object material differences, sensor noise, resolution or another artifact, resulting in a degradation of pose accuracy. Data augmentation is usually used to reduce this domain gap however some methods have developed self-supervising techniques that allow data collection from real images without annotations [118, 136,

20].

Practical Challenges. The prior challenges are standard problems that all successful pose estimators must overcome. However, the trade-off for improving accuracy and robustness to the previous challenges usually comes at a substantial computational cost, which is not reflected in the evaluation metrics. In order to be robust to occlusion problems, state-of-the-art methods use high resolution object coordinate correspondence maps [137, 138, 90, 25, 41] as intermediate representations which not only increases memory usage but also slows down pose retrieval [68, 92, 18]. Others have proposed the use of learned refiners [73, 153, 67, 88] which rely on an slow and iterative process of feature extraction after rendering an object at high resolutions. While achieving impressive results most methods are not suitable for real-world applications that cannot dispense the necessary compute resources due to cost, hardware size, computational availability or environmental concerns. An autonomous car cannot have its battery life impaired by a large pose estimator nor can it wait idly for milliseconds for an obstacle to be detected when it is travelling at high velocity. Wearable smart devices have similar requirements. Due to their size and weight restrictions, the available usable hardware is limited. Several methods optimize a single network in order to estimate the pose of a single object [92, 90, 98, 118, 86]. Scaling to multiple objects should require training multiple networks however it has been found that multi-object models degenerate results [61]. There is a real-world need for efficient 6D object pose estimation techniques that can run with high throughput on budget hardware, that are lightweight and that can easily scale to multiple objects. In this thesis we mainly focus on solving practical problems in order to improve 6D object pose estimators real world applicability.

1.3 Thesis Overview

In this thesis we introduce novel contributions to solving the 6D object pose estimation problem, in an instance-level paradigm, with a special focus on high throughput and scalability. Each of the following subsections will summarize the motivation behind each chapter contribution along with a summary of our proposed solution. The first chapter, summarized in subsection 1.3.1 aims at introducing an object specific bias into the pose estimator by incorporating the object mesh within its structure in the form of a undirected graph. The second and fourth chapter, summarized in subsections 1.3.2 and 1.3.4,

respectively, aim at solving the throughput problem by proposing a faster and more efficient pose estimation methodology, regardless of hardware used. Chapters 5 and 7, summarized in subsections 1.3.3 and 1.3.5, respectively, aim at solving the object scalability problem first by introducing a novel compression algorithm that recycles previously trained one-object models and then by proposing a one-shot pose estimation, that can generalize to any object shape and texture.

1.3.1 Pose Conditioned Mesh Reconstruction

The understanding of the shape of an object is critical for any pose estimation technique. This is evidenced by the ability of some methods to accurately predict occluded keypoints, even though they are not directly optimized to learn the shape of the target object [98, 128, 52]. By directly incorporating the 3D topological information of the target object, the learning of its pose can be effectively disentangled from its shape and reduces its task complexity. Moreover, by reconstructing a dense estimation from a holistic view of the object, we construct a vote-based orientation extractor, which is robust to noisy regions. In this first chapter, we introduce a pipeline for guiding our optimization through a GraphCNN capable of reconstructing a pose-conditioned 3D mesh. Combined with other location cues, i.e. centroid projection and mask segmentation, extracted from the image using a lightweight backbone, we can recover the full pose estimation. The results of this approach beat prior state-of-the-art methods, specially with respect to rotation accuracy. We later show that we can use the reconstructed mesh quality to devise a pose confidence score, which is unavailable for other pose estimation methods.

1.3.2 Multi-Frustum Network

Li et al. [74] first noticed that extracting translation from dense representations like Normalized Object Coordinate Space (NOCS) [138] resulted in a poor performance compared to directly estimating it. This leads to *ad-hoc* solutions where rotation is disentangled from translation and both are separately retrieved. However, while these solutions suffice, they do not solve the root of the problem. The appearance of an object changes by different magnitudes when comparing the effects of rotation and translation transformations on its pose. Due to the projective function, small rotations can largely

impact the appearance of an object while a small translation away from the camera, i.e. zoom, might have very little. In this chapter we propose Multi-Frustum Network, to address the root of the problem and by estimating the rotation and translation though dense correspondence in 3D space, instead of 2D projection space, i.e. image. This method achieves superior pose accuracy and processes images in half the time required by earlier approaches. The novel representation can be converted into a full 6D pose using a Kabsch step, circumventing the need for a computationally expensive Perspective-n-Point (PnP) step.

1.3.3 Scaling to multi-object pose estimation

The methods discussed in the previous subsections are supported by deep neural networks that were trained for a single object. The main BOP datasets collectively contain 125 objects. Training a model from scratch for each object, for each dataset, consumes tremendous resources which not only has a big economic impact but an environmental one as well. While most methods can be easily scaled to multi-object, if one is prepared to sacrifice accuracy, other methods cannot be extended due to their design. The method described in the first chapter of this thesis is one of such. You cannot scale to multi-object if the network topology graph is, by design, specific for each object. Other examples cannot easily implement the extension. For example, methods that require heavy computational resources, such as refinement [67, 73] or domain adaptation through rendering [118] cannot iterate through ideas and ablation studies on a multi-object paradigm simply if the author does not have access to multiple server-grade machines. Moreover, some methods are realized by using complex intermediate representations [120] which might make scaling to multiple objects under a single lightweight model unfeasible. In this work, we propose a simple yet effective way model parameterization that, while introducing little overhead, can alleviate multi-object accuracy drops. In order to match one-object models, networks need to be scaled to parameters that might be unfeasible under restricted hardware availability. For this reason, we show how to recycle previously trained one-object models by distilling from multiple teacher networks in order to avoid training a large network.

1.3.4 CRT-6D: Cascaded Refinement Transformers

A pose estimation application might choose to refine the predicted erroneous pose. The most commonly used methods rely on a costly *render-compare* iterative process [73, 153, 67, 135], making them unsuitable for real-time applications. *Ad-hoc* refinement methods require large models, designed and trained only for refinement and leaving the initial pose estimation as an exercise for other methods [67, 73, 153]. In this chapter, we propose a novel refinement based method, called CRT-6D, that combines pose estimation and pose refinement in a single network. It achieves accurate estimations by removing redundant computations around areas where the object is not present, while oversampling the image regions where it is. We achieve this by using a simple yet effective new intermediate offset representation. CRT-6D is the fastest state-of-the-art 6D object pose estimation method running $2\times$ faster than the closest real-time state while supporting up to 30 objects on a single model.

1.3.5 PoseMatcher: Cascaded Refinement Transformers

When provided with a new object, in order to create a pose estimation model we either train a new single object model from scratch or extend a multi-object model and fine-tune on the new object. In order to truly scale, one should not have to train a new model or fine-tune an existing one every time a novel object is introduced. For this reason, we design a one-shot 6D object pose estimation model that can accurately estimate the pose of a previously unseen object given a limited amount of data and without retraining, which we call PoseMatcher, based on keypoint matching transformer methods [121, 140]. We design the training pipeline in order to cheaply construct an approximation of the full object point cloud during each training step. Additionally, we create a new attention layer that has been specifically design to accommodate point cloud to image matching without additional parameterization. Finally, we create a novel 3D-refinement step, which accepts pixel-wise 2D-refinement offsets as input and fuses them to generate a unified 3D transformation. In the one-shot paradigm, we achieve state-of-the-art on multiple datasets by a significant margin and even outperform instance-level methods such as PVNet [92].

1.4 Publications

This thesis has produced the following list of publications:

- [17] **P. Castro**, A. Armagan, and T.-K. Kim. Accurate 6d object pose estimation by pose conditioned mesh reconstruction. In ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020.

- [15] **P. Castro** and T.-K. Kim. Crt-6d: Fast 6d object pose estimation with cascaded refinement transformers. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023 Code and model weights made publicly available on Github.

- [117] J. Sock*, **P. Castro***, A. Armagan, G. Garcia-Hernando, and T.-K. Kim. Tackling two challenges of 6d object pose estimation: Lack of real annotated rgb images and scalability to number of objects. arXiv preprint arXiv:2003.12344, 2020.

- [16] **P. Castro** and T.-K. Kim. Posematcher: One-shot 6d object pose estimation by deep feature matching. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, 2023. Code and model weights made publicly available on Github.

Chapter 2

Background Theory

This chapter gives a through overview of the work on the 6D object pose estimation problem preceding this thesis. The chapter is split three sections. The first delves into classical approaches that were used before the introduction of deep learning in computer vision tasks. The second section presents a comprehensive review of exiting deep learning based 6D pose estimation methods. It is further split into subsections according to their pose estimation output design. In the last section, we conclude with an overview of the exiting evaluation datasets and the methodology and metrics used to assess the accuracy of an 6D object pose estimation method.

2.1 Classical Methods

Classical 6D object pose relied on template-matching [45], local descriptors [109] and 3D object alignment with Chamfer edge matching [77] or Hausdorff distance [56]. Although very fast and somewhat robust to occlusion, these methods often struggle on poorly textured objects and low resolution images. Some recent non-learning based methods were shown effective in the 6D object pose challenge [49]. Until recent versions of the Benchmark for Object Pose (BOP) challenge [49], Point Pair Features methods [30, 46, 133] held the top spot on its leaderboard. It relies on densely sampling keypoints and retrieving its surface normals from an existing 3D object model and matching these to a depth map at test time. At training time, one needs to simply store these keypoints which are sampled from the mesh. Matching is done through an iterative voting loop which then outputs a set

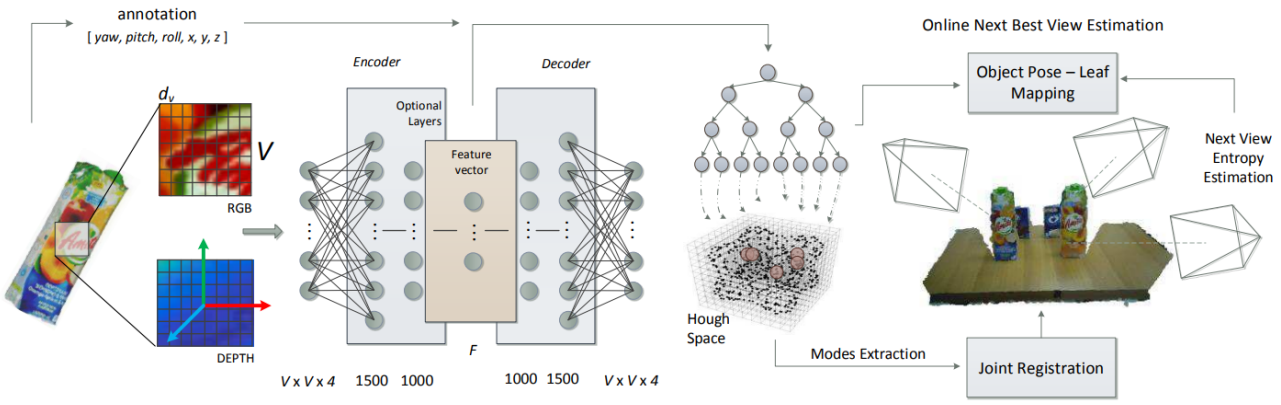


Figure 2.1: **Hough Voting by Doumanoglou et al. [28].** RGB-D patches are extracted and fed into a Sparse Autoencoder. The bottlenecked features produced by the autoencoder are forwarded to a Hough Forest, which then generates the final hypotheses through Hough voting space. Joint registration is used to refine the hypotheses which can then be used to we estimate the next-best-view.

of feasible poses. An Iterative Closest Point (ICP) [7] refinement step is then needed to recover finer poses, which are then re-evaluated based on their inlier count. This method relies on capturing information with a depth sensor at inference time which is not commonly available. Various voting-based methods have been proposed to tackle the problem of 6D pose. Tejani et al. [127] and Doumanoglou et al. [28] extend the template matching by Hinterstoisser et al. [45], in order to incorporate scale-invariant patch descriptors and integrate it into a regression hough forest. The latter extends the patch descriptors by using an auto-encoder to remove noise.

For RGB only methods, the over-reliance on template matching results in techniques susceptible to illumination changes, a naturally common occurrence in computer vision tasks. Moreover, patch based descriptor matching trigger false positives when an object has ambiguous, symmetric or repetitive features.

2.2 Deep learning based methods

Over the last years, deep learning based techniques have been dominating the 6D research field. Detection is a crucial part of recovering the 6D pose of an object and the field of 2D detection has been the focus of remarkable works [78, 102, 103]. In this section we will go over the current state-of-the-art methods using deep learning approaches. We divide these approaches into 5 separate categories which are explained in this order: 1.) keypoint detection; 2.) dense 3D keypoint detection; 3.) direct

pose estimation; 4.) template matching; 5.) pose refinement.

2.2.1 2D Projection of 3D Keypoints

Camera pose estimation is a problem closely related to object pose estimation. In camera pose estimation, the world is assumed to be static and the camera moves, wheras in object pose estimation, the camera is static and the object is dynamically positioned. Object pose estimation can be seen as the inverse of camera pose estimation. A common solution to estimating the camera pose in 3D space is by solving the Perspective-n-Point (PnP) problem, which involves determining the position and orientation of a camera given a set of 3D keypoints and their corresponding 2D projections. From these control points, pose can be efficiently extracted using openly available implementations [68]. By inverting the recovered transformation, we can transform the camera pose into object pose. In order to be solved, the PnP problem requires a sufficient number of non-collinear 3D points and the intrinsic parameters of the camera available.

Firstly, one must chose which keypoints are ideal to detect. Rad and Lepetit [98] propose detecting the 2D pixel coordinate projection of the corners of the 3D bounding box of the target object. In their implementation, they chose to estimate these positions as heatmaps, a 2D probability distribution map with a peak at the center of the 2D location of each keypoint. Tekin et al. [128] followed by implementing it within a single network, which was able to perform detection and pose estimation in under 50 Frames Per Second (FPS) on a TitanX GPU, additionally adding a confidence estimation to guide training. It achieved this throughput by being built on top of the YOLO [102] detector single-stage framework.

Occlusion being a serious challenge for pose estimators as it introduces foreground distractions which may hide critical features led Oberweger [86] to successfully alleviate occlusion by independently locating 2D keypoints over multiple small patches of the image and then accumulating the results into a single robust prediction, although the results are only supported by synthetic experiments. Hu et al. [52] proposed a similar idea but built on top of YOLO [102].

However, the projected 3D bounding box keypoints usually lie outside the silhouette of the objects which potentially reduces the effectiveness of the extraction of local information. The detection

of individual and of a limited number of keypoints presents a drawback, whereby image patches containing crucial information may not be utilized due to the absence of the selected keypoints in those regions or simply under external or self occlusion. For that reason, Peng et al. [92] proposed regressing a pixel-wise unit vector 2D map to identify and vote for keypoint locations, overcoming the challenges of occlusion and truncation. This representation as an additional advantage as it allows for estimating uncertainties associated with keypoint locations, which can be leveraged by a novel uncertainty based PnP solver for improved accuracy.

The process of keypoint localization is not without the limitations discussed in this section, such as external or self occlusion, truncation or its sparseness, but it also hindered by the need for identification and selection of optimal keypoints. In addition to the need to determine which keypoints should be selected, the number of keypoints to be chosen is also a crucial factor in achieving the desired level of accuracy in keypoint detection while also having an impact on its complexity. While these handcrafted keypoints and surface representation have been proven to work, [125] are motivated to learn the optimal surface keypoints for pose estimation without any keypoint annotation supervision. The set of learnt optimal 3D keypoints are geometrically and semantically consistent from any viewpoint. They learn to predict the same set of 3D keypoints from 2 different views, recover the relative pose between views by aligning the 2 sets of 3D keypoints [113] and optimize their model by minimize the relative pose error.

2.2.2 Dense 3D Keypoint Detection

NOCS [138] introduced a more comprehensive approach to keypoint selection, in contrast to using only a few pre-selected keypoints. In this method, every pixel within the silhouette of the object is employed to estimate its surface coordinate (in normalized space) projected at that specific pixel. Essentially, this means that every point on the surface is treated as a keypoint, which can then be utilized for creating the 2D-3D correspondence set required for solving the PnP problem. By leveraging all the points on the surface, this method ensures a more robust and detailed representation of the object, allowing the model to better understand the spatial relationships between different keypoints. In turn, this results in improved pose estimation accuracy and a more comprehensive understanding of the 6D pose of the object within the scene. Furthermore, the increased number of keypoints allows the

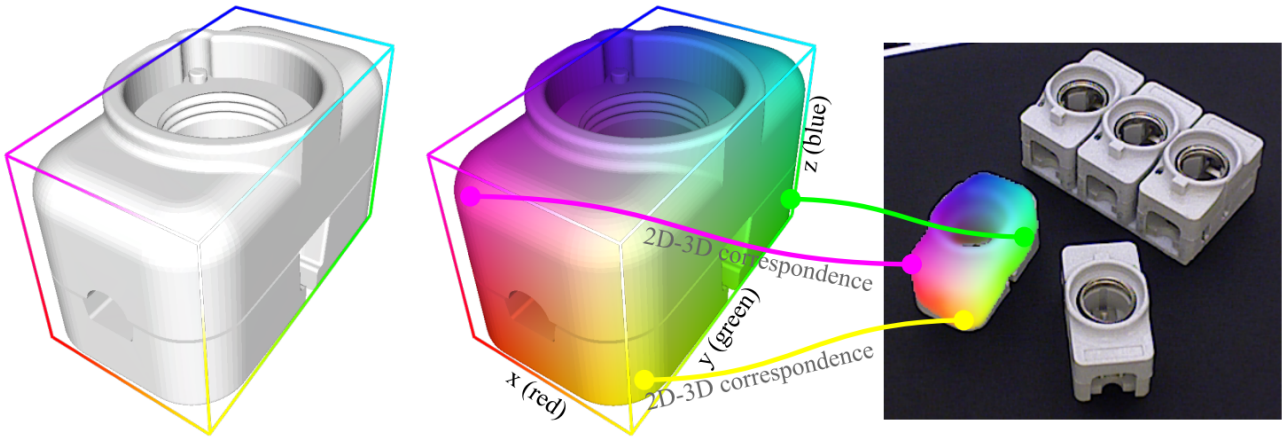


Figure 2.2: **NOCS representation by [90].** This dense representation extends 3D keypoint detection to every visible surface area of an object in an image. With the 2D-3D correspondence map, the 6D pose can be extracted by solving the PnP problem, usually assisted by RANSAC based outlier removal.

model to account for a wider variety of object geometries and surface features, enhancing its overall performance and adaptability. While originally proposed to tackle category level object pose estimation problems, it became widely used on inference level as well. Park et al. [90] addresses the challenge of occlusion in dense 3D keypoint estimation by introducing an adversarial condition in order to hallucinate the occluded regions of the object.

Zakharov et al. [153] proposed using a different correspondence mapping by estimating the texture UV coordinates of an object instead of its 3D coordinates. This representation ensures that every keypoint estimation, regardless of its accuracy, is located on the surface of the object. By focusing on the UV coordinates, the method provides a more consistent and constrained estimation space, which leads to improved pose estimation results and increased robustness against errors. The authors also found that reformulating the color regression problem as a discrete color class classification problem proved to be advantageous for faster convergence and superior quality of 2D-3D matches. By treating the problem as a classification task, the model is able to learn more effectively and efficiently, resulting in improved alignment of the 2D and 3D coordinates and ultimately, better pose estimation. As 3D coordinates are still required to apply the PnP step, a straightforward lookup table is employed to retrieve the 3D locations during inference, which has virtually no computational costs.

Extending 3D keypoint position discretization, some have proposed discretizing the full object model, dividing into meaningful and separable surface fragments. Hodan et al. [51] proposed using the fur-

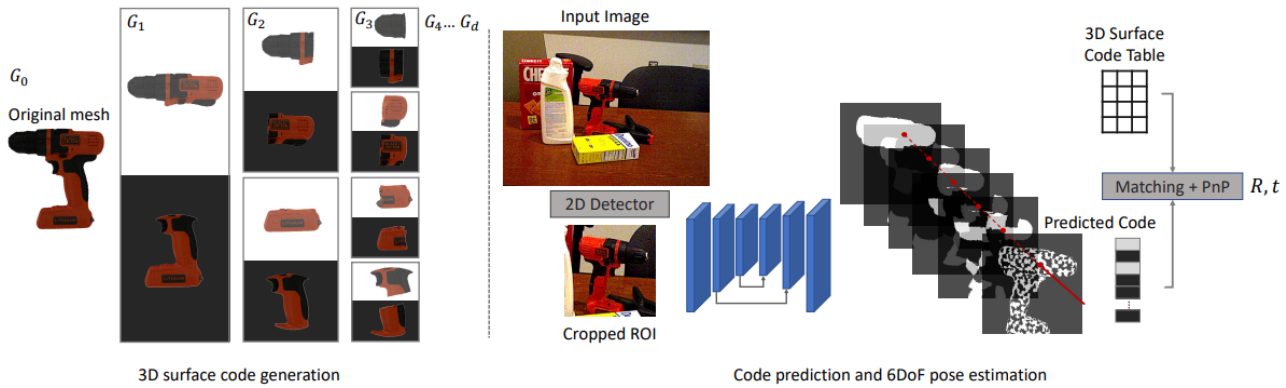


Figure 2.3: **Dense binary encoding in ZebraPose [120].** Su et al. [120] generates a hierarchical encoding that is efficiently densely estimated and for each encoding a 3D keypoint correspondence can be retrieved from a lookup table.

thest point sampling method to generate the fragment centers and assigns each object keypoint to the closest fragment center. The model predicts for each pixel the probability of the projected surface keypoint belonging to each fragment. The surface fragments are large and do not provide enough precision for correct pose therefore for each fragment an offset position, centered at the fragment, is extracted. This new method allows Hodan to train EPOS Hodan et al. [51] without symmetry labels, as different fragment at positions with similar appearance (symmetric positions) will output a probability distribution with the same peak at the different positions. SurfEmb [41] generates a dense 2D-3D correspondence with continuous representation distributions over the surface of objects without symmetry annotations. The method uses contrastive loss to optimize correspondence which are represented in object-specific latent spaces with a small fully connected representation model. More recently, ZebraPose [120] proposed a hierarchical binary encoding for regions of the object model. They propose learning this representation with a coarse to fine strategy, where bits representing larger regions should be learned before finer regions. The correspondence between encoding and 3D region can be recovered at test time with a simple precomputed lookup table.

2.2.3 Direct 6D Pose Estimation

The most straight forward way of recovering the pose of an object is by estimating its pose parameters directly. In order to design such a method, one must find a representation for the decoupled rotation and translation extrinsic parameters. This choice will impact both the runtime of the model, its robustness to external factors and pose precision.

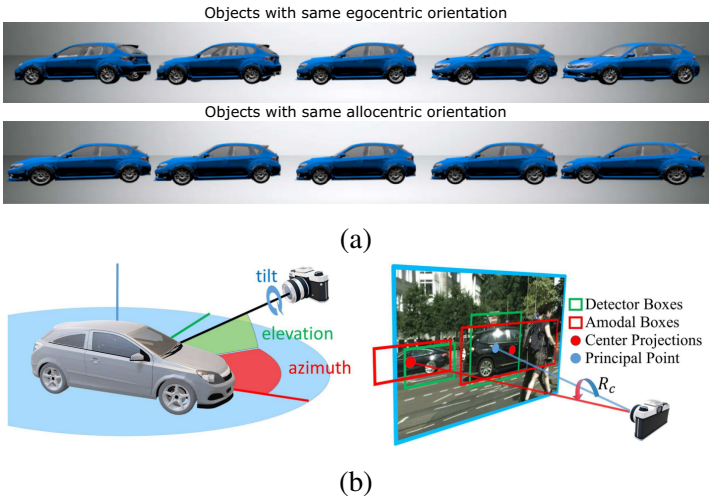


Figure 2.4: **Allocentric vs. egocentric orientation differences by Kundu et al. [66].** (a) illustrates the difference between cars in an image that share the same egocentric orientation with respect to the camera but exhibit significant changes in appearance, versus cars that share the same allocentric orientation and exhibit no appearance changes. (b) refers to the representation used and transformation applied to recover the egocentric pose.)

One problem arising from decoupling the components of the pose is the egocentric deformation that occurs by moving the object away from camera principal ray. Object orientation can be represented as egocentric (orientation with respect to the reference frame of the camera) or allocentric (orientation with respect to the object itself). Objects that share a similar allocentric orientation will have a similar appearance irrespective of their position in the image [66]. However, objects with a similar egocentric representation will exhibit significant changes in appearance based on their position. Since the orientation is predicted based on an cropped section of the full image, which is generated from a box centered on the object, it is preferable to adopt an object-centric (allocentric) representation for optimization purposes. For that reason, in two-stage approaches involving detection/cropping and pose estimation, it is essential to estimate the allocentric orientation of the target objects and then recover the egocentric pose by utilizing the bounding box information.

To address the task of recovering orientation, the rotation space can be discretized into 3D orientation bins, as commonly done in estimating yaw for object car detection [84, 114]. This approach transforms the task of recovering the orientation of an object into a classification problem. The larger the number of bins used, the more complex the task becomes while decreasing the number of bins results in low precision. Works from Kundu et al. [66] and Manhardt et al. [84] apply a bin-conditioned regression task. On the latter, rotation from center of the bin is refined in order to alleviate discretization

imprecisions, while on the former expectation is used to recover a precise orientation. SSD-6D [63], which build on top of the Single Shot Detector (SSD) [78], discretizes the possible viewpoints, i.e. spherical coordinates space by disentangling the allocentric orientation and independently estimates the in-plane rotation applied to the scene. Regression based direct pose estimation faces problems when dealing with particular unseen poses and occlusion due to its holistic view approach [4, 148]. Instead of discretizing the rotation space, the orientation estimation can be designed as a regression problem. Now the problem of choosing the best 3D representation arises. Euler angles can be used [4] to represent the target rotation however they are susceptible to three-dimensional gimbal lock, which occurs when two of the three rotational axes become aligned. This alignment reduces the degrees of freedom of the rotation, as two axis become *locked* and effectively leaving only one axis of rotation available. Lie algebra can be used directly as proposed by Do et al. [26], however it becomes a complex task as the rotation matrices need to be orthonormalized. For camera pose estimation, Posenet [64] proposed learning quaternions to predict rotation on camera pose estimation tasks, because enforcing the quaternion to be unit vector is simpler than rotation matrix orthonormalization. The same 4D representation was used for pose estimation by PoseCNN [148] and by [83], as well as for refinement with DeepIM[73]. However, this representation is not without its fault, as it still requires unitary constraint. Recently was found that that using two orthogonal 3D vectors and retrieving the third one by cross product produced the best results [157]. This strategy has been widely used to great effect since then [137, 25, 15].

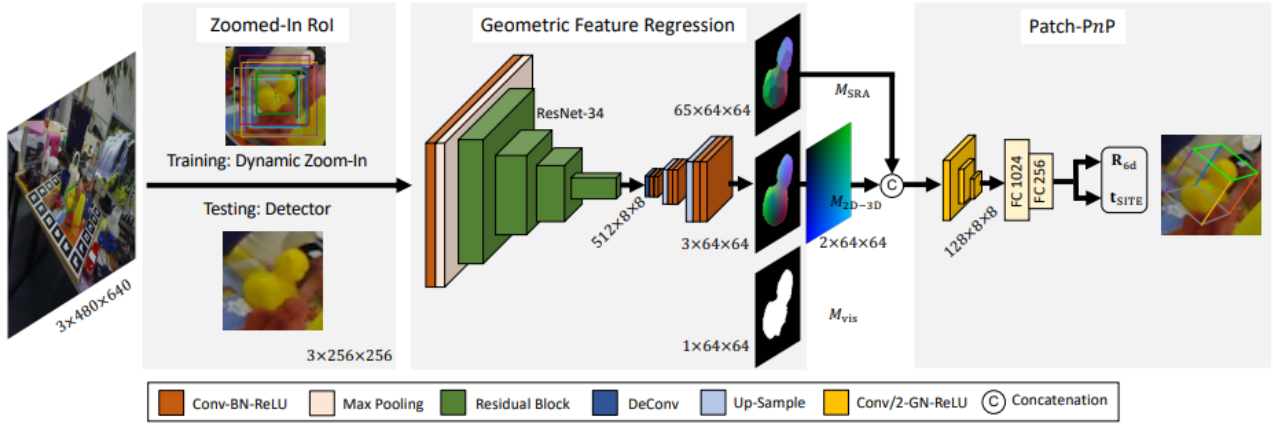


Figure 2.5: **Direct pose estimation through intermediate representations by Wang et al. [137].** The pipeline of GDR-Net generates the most widely used dense intermediate representations and uses them to generate a final pose, which means that the pipeline can be trained end-to-end, along with intermediate representation supervisions.

The common way to directly represent the translation transformation is by locating the 2D projection of the object centroid and independently find the distance of the object to the camera. SSD-6D [63] and the pose Augmented AutoEncoder (AAE) [124] find both by comparing the predicted bounding box information, used for detection and subsequent cropping, with bounding box information pre-computed for each viewpoint. Since the silhouettes size is inversely proportional to the distance to the camera for a given viewpoint, the size of the bounding box is used to recover depth. This algorithm, although it achieves good results for most examples, fails under occlusion, if bounding box regression has errors or if the viewpoint is wrongly predicted. Instead of relying on detection, refining the centroid position is possible by regression its 2D location within the cropped region of interest. Li et al. [74] noticed that by decoupling regression and translation, it could use dense estimations, based on Normalized Object Coordinates (NOCS) [138] for rotation purposes, and direct centroid and depth regression for translation. Pairing it with a bounding box relative depth estimation, this approach is heavily used [137, 25, 67, 74, 12] and remains a fundamental component of state-of-the-art pose estimators. Recently, Cai et al. [12] proposed replacing the depth regression with a discrete classification, but this approach has yet to see wide adoption. Wang *et. al.* [137] proposed the Geometry-Guided Direct Regression Network (GDR-Net) which combines the intermediate representations, NOCS and fragment classification, and from those directly regresses the 6D pose. It proposed Patch-PnP, a module that takes as input a 2D mapping of the fragment distribution probability and pixel-wise 3D coordinate regression which outputs the rotation in the 2-axis representation and translation using the dynamic depth translation used by Coordinates-Based Disentangled Pose Network (CDPN) [74]. This method has the advantage that it can be optimized in an end-to-end manner, similar to PoseCNN [148], directly using the common pose accuracy metrics. Di et al. [25] extended by adding another intermediate representation where it would predict object inner plane collisions with pixel rays. This work improved upon GDR-Net with virtually no additional overhead.

2.2.4 Template Matching

Sundermeyer et al. [124] creates a database of pose conditioned encoded latent vectors which serves as global descriptors. Each descriptor is generated by an auto-encoder network, where the bottleneck vector is said descriptor. The autoencoder is trained with renderings of the 3D object models,

heavily augmented by randomizing light positions, material properties, random background images and color and noise augmentations, hence the name "Augmented Autoencoder". At test time, the image is encoded into the latent space and a pre-computed pose database is searched for the closest match using nearest neighbours. Later this method was improved Sundermeyer et al. [123], where all objects shares a single encoder, resulting in an increased capacity for multi-object within a single model. SC6D [12] introduce an orientation estimation through contrastive loss [87], without being a holistically template based such as the methods proposed by AAE [124]. This method allows for the implementation of a fast direct pose estimation method with similar advantages as EPOS [51], where similar appearances share neighbouring vector representations and it can capture symmetric distributions without the need to carefully supervise each object with its symmetric properties. Shugurov et al. [115] and Liu et al. [80] rely on deep template matching for pose estimation in the one-shot paradigm. The former, uses encoded synthetic templates of the object at different poses and matches it to the query image. The matched template is then combined with the query for further information extraction which is then used for pose refinement. Gen6D [80] uses template matching in the same way as Shugurov et al. [115], but refines it using a 3D cost volumes generated from the projected features of a set of neighbouring pose templates. Wen et al. [145] tackles the problem in the category level domain and uses template matching to extract the category and shape of an unseen object and again to match the extracted pose feature to a set of pre-computed pose representations.

2.2.5 Pose Refinement

In addition to estimating the pose, applications might choose to refine the predicted erroneous pose to achieve higher accuracy. Traditionally, if depth information is available, the Iterative Closest Point (ICP) algorithm [7] is used [41, 46, 63]. The primary goal of ICP is to identify an aligning transformation between two sets of points by iteratively performing two key steps: establishing putative correspondences between the closest points and estimating a transformation that aligns them accordingly. In the context of refining object poses, the source point cloud is typically defined by the vertices of the 3D object model in the initially estimated pose or calculated from a rendered depth image of the model while the target is the point cloud generated by the available depth information. By refining the pose through this iterative process, the algorithm is able to minimize the discrepancy between the

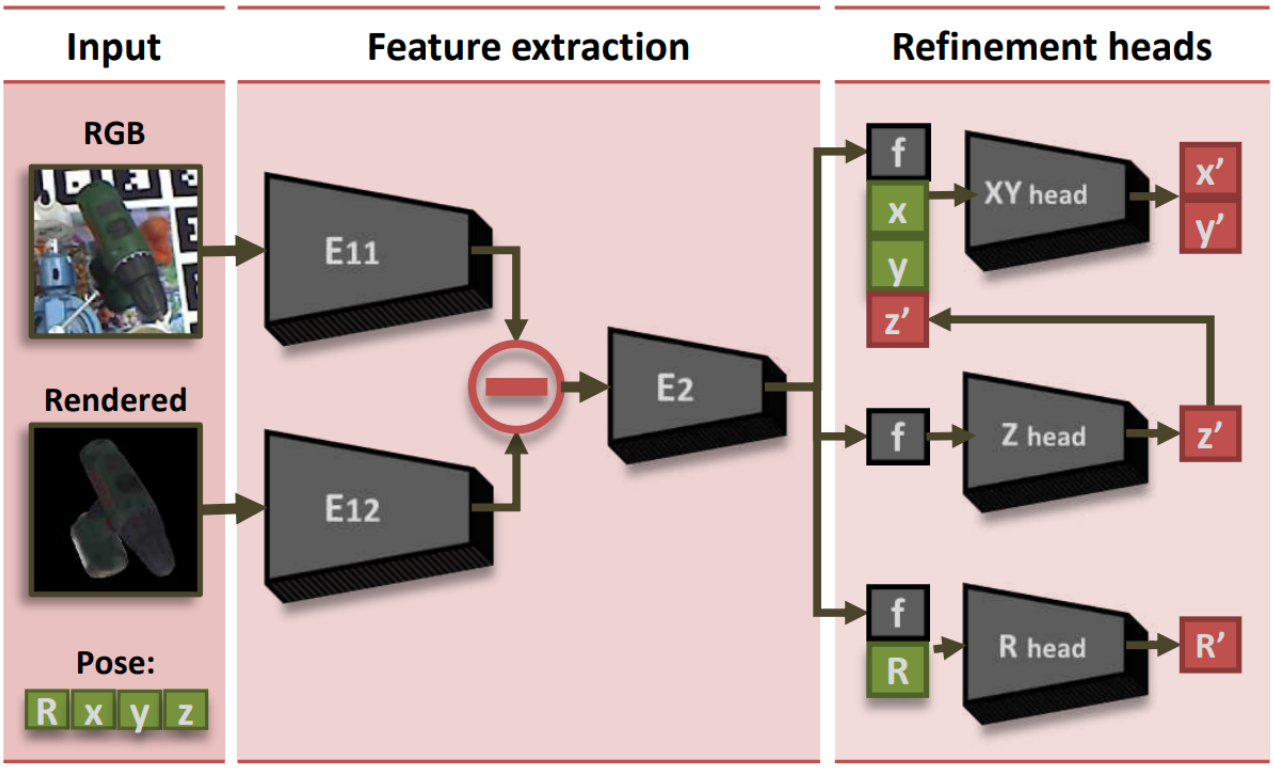


Figure 2.6: **Refinement diagram by Zakharov et al. [153].** Taking as input the query image and a rendering of an approximate pose, the offset rotation and translation are extracted. This process can be repeated multiple times until satisfied.

source and target point clouds, resulting in a more accurate pose estimation. Despite its widespread use, ICP has some limitations, such as its sensitivity to the quality of the initial pose estimation and the potential to converge to a local minimum due to its iterative nature.

In order to overcome these limitations, learned based approaches have been proposed, the most common of which rely on a resource-intensive *render-compare* iterative process[73, 153, 67, 135]. These methods rely renderings to represent the pose of the object. The process involves multiple rounds of rendering the object with the predicted pose and comparing it with the query image to iteratively refine the pose estimation. While these methods can achieve improved accuracy, the computational demands associated with the repeated rendering and comparison steps make them unsuitable for real-time applications, ultimately rendering them unsuitable for real-time applications. These learning-based methods aim to leverage the power of deep learning and other advanced techniques to overcome the inherent challenges faced by traditional algorithms like ICP. More recently, new approaches that rely on different pose representations have been specifically designed to strike a balance between runtime efficiency and initialization quality. RePose [58] is a fast method, but its accuracy largely

depends on the quality of the initial pose estimation. In other words, it requires a good initialization to deliver accurate results. A similar approach has also been proposed by Xu et al. [149], where the optimization process of RePose [58] is upgraded with a Recurrent Neural Network module. On the other hand, SurfEmb [41] is a method that excels in precision and robustness, even in the presence of occlusions and symmetries. However, its major drawback is the slow inference time, which is significant even when used with server-grade equipment. In Chapter 6, a more efficient representation for pose refinement is proposed allowing for real-time pose refinement.

2.2.6 Conclusion

Deep learning has become an invaluable tool for object pose estimation. However, as we have seen in the previous sections, we have learned how it is merely a tool, rather than a solution. Moreover, we have explored how deep learning can be applied to the object pose estimation task as the modeling tool of many intermediate tasks and representations. Nevertheless, each approach comes with its own advantages and limitations, inherently present due to the intrinsic properties of the chosen solution. For example, keypoint prediction, while surpassed in accuracy by denser estimation methods, can be much lighter [98, 137] and therefore suitable for real-world applications. Direct pose estimators are fast variants of pose estimators [137, 74, 25], however they fail to adapt to new objects and must be retrained if a new object needs to be supported. Refinement methods can be used on top of any method to achieve a finer pose estimation, however they add considerable overhead as they are designed to be an add-on [153, 73]. Addressing these practical challenges is essential for deploying object pose estimation in real-world applications. This thesis tackles these challenges and proposes solutions that enhance the feasibility and efficiency of object pose estimation for real-world applications.

2.3 Datasets and Evaluation Metrics

In order to evaluate, and train in the case of learning based approaches, we need high quality datasets, where the 6D pose of target objects is annotated. Recently, the Benchmark for Object Pose (BOP) [49], a workshop which has run at every International Conference on Computer Vision and European Conference on Computer Vision since 2018 to the time of writing this thesis, was created and it aggregated

the most commonly used instance-level datasets into one large benchmark. The core of this benchmark is composed of 7 datasets. They vary considerably in characteristics. Some datasets provide real annotated training images, while others do not. Linemod Occlusion [45] (a challenging subset of the Linemod dataset), YCB-V[13], HomeBrewed6D [61], IC-Bin [28] and TUD-L [49] are all datasets containing household common items. MVTec ITODD [31] and T-Less [48], on the other hand, provide industry-relevant objects that exhibit symmetries, no significant texture or discriminative color. MVTec ITODD, T-Less and IC-Bin provide images of scene with heavy cluttering, with most objects under heavy occlusion. MVTec ITODD is the only entirely synthetic and grayscale dataset. The combined datasets form a comprehensive evaluation benchmark that requires methods to overcome all the classical challenges of 6D object pose estimation. In its entirety, the benchmark requires accurately estimating the pose of 125 different objects. While Sock et al. [118], Wang et al. [136] and Chen et al. [20] proposed methods that could overcome the lack of real training data, they still relied on synthetic data for training supervision and initialization. Generating photo and physically realistic datasets can become very expensive. [50] rendered 2.5M images within a day with a server-grade setup of 400 nodes of 16-core Intel Xeon 2.3GHz processors with 112GB RAM, which is not cheaply available. Methods trained with different datasets are subject to different data biases, which results unfair comparisons when evaluating pose estimation results. For this reason, as part of the BOP [49], the organizers defined that the only accepted training data, real or synthetic, was chosen by them. The PBR [23] split of every dataset supported by BOP became the standard synthetic dataset. This set of synthetic images is rendered using Blender within a photorealistic scene generated from a randomized environment. To enhance realism, each environment introduces variations in background, distractors, 3D scene geometry, and illumination. A physics engine is utilized to ensure the objects are positioned in a physically plausible manner. The final renderings are captured from uniformly sampled viewpoints, ensuring comprehensive coverage of all possible object orientations. In order to remove detection noise from the evaluation of the pose accuracy, the BOP challenge provided standardized bounding box detection from a fine-tuned RetinaNet [75] generated by Labbé et al. [67]. Two sets of bounding boxes are provided, one for detectors trained with synthetic and real data, and another for detectors trained only with synthetic. The latter guarantees synthetic only methods do not leak real data through the detector stage. The use of these synthetic datasets splits is now standard across the

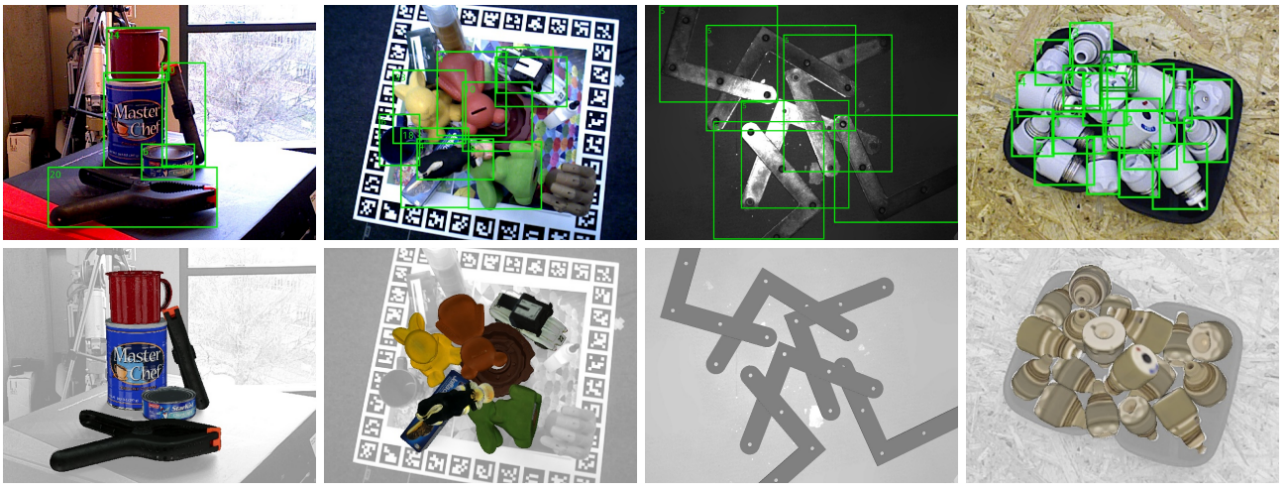


Figure 2.7: **Illustration of several datasets contained by the BOP challenge Hodaň et al. [49].** The first row refers to 2D detections while the second one illustrates pose estimation. From left to right, the examples belong to the following datasets: YCB-V[13], HomeBrewed6D[61], MVTec ITODD [31] and T-Less [48].

field and methods using a differently sourced synthetic datasets have to go through considerably more scrutiny to be accepted by the community.

In order to evaluate and compare object pose estimators, standard metrics have been proposed. The simplest metrics to use are the angular distance between the predicted and ground truth orientation and the translation difference measured in Euclidean space. The ADD-(S) is the most commonly used metric [148]. The ADD-(S) metric measures the mean Euclidean distance between the correspondent vertices of the estimated pose transformed model and the groundtruth pose transformed model. We consider a correct pose if the ADD score is below 10% of the diameter of the target object . However, ADD heavily penalizes symmetric objects whose appearance is rotational invariant along the symmetry axis. For this reason, we measure ADD-(S) on objects which display symmetry properties. Formally, these metrics are defined as:

$$ADD = \frac{1}{|V|}\sum_{i=0}^{|V|} \|(RV_i + T) - (\hat{R}V_i + \hat{T})\|_2 \tag{2.1}$$

$$ADD - (S) = \frac{1}{|V|}\sum_{i=0}^{|V|} \min_{V_k \in V} \|(RV_i + T) - (\hat{R}V_k + \hat{T})\|_2 \tag{2.2}$$

where V is the set of object vertices, i the vertex indices, R and T are the estimated rotation and translation transformation with their groundtruth counterpart being denoted with $\hat{}$. Compared to the

former metric, 2D reprojection is lenient when compared to zoom-in errors due to its projective nature, which makes it more suitable for evaluating methods aimed at AR applications. Visible Surface Discrepancy (VSD) was proposed by Hodaň et al. [48] to measure the pixel-wise difference between the reprojected depth maps using the estimated pose and the ground truth pose. Since translation errors command a much larger weight in the outcome of the metric, it was designed to be employed in measuring errors in industrial and robotic manipulation environments, where the extraction of the 3D location of the object is essential. ADD-(S), on the other hand, strikes a balance between 3D location and orientation errors. In order to considered an estimated pose to be correct, the result of metric function must be below a pre-defined threshold. The recall accuracy is percentage of predicted poses with a metric below the threshold. In literature, and in this thesis, we refer to this as simply ADD-(S) accuracy. In most works, the threshold used for angular distance is 5° while translation thresholds can vary from $5mm$ to $200mm$. For ADD-(S) it is common to use 10% of the objects diameter as a threshold and average 2D reprojection error below $5px$ is usually considered correct. PoseCNN [148] and the BOP challenge[49] have been pushing for a standardized metric, regardless of object size. Instead of defining a single threshold, accuracies are collected using a range of tolerance errors. The average accuracy is then measured by computing the area under the curve (AUC) over all thresholds. Currently, the BOP challenge [49] ranks methods according to the average AUC of the three metrics, ADD-(S), 2D reprojection and VSD, which are all computed using a pre-determined range of errors, agnostic to the object size.

Chapter 3

Pose Estimation by Mesh Reconstruction

In instance-level 6D object pose scenarios, full object 3D information is usually provided prior to pose inference [86, 124]. Recent trends show that learning to detect object keypoints based on their appearance can achieve state of the art results [128, 98]. However, even these techniques indirectly learn the shape of the object, shown by their ability to accurately predict occluded keypoints. Therefore, understanding the shape of an object is crucial for any pose estimation technique. In this chapter, we present a novel deep learning architecture for recovering the 6D pose of an object from a single 2D RGB image by utilizing 3D reconstruction. We directly incorporate the 3D topological information of the target object, effectively disentangling the learning of the object pose from its shape. This approach simplifies the task of pose estimation by reducing the complexity of the learning problem, resulting in a more accurate and efficient model. When it comes to instance-level 6D object pose estimation, it is common practice to provide full 3D information of the object prior to inferring its pose [86, 124]. While recent advancements have shown that detecting object keypoints based on their appearance can lead to state-of-the-art results [128, 98], these techniques indirectly learn the object shape as evidenced by their capability to predict occluded keypoints with a high degree of accuracy. Nonetheless, having a comprehensive understanding of the object shape remains critical for any pose estimation technique to achieve optimal results. In shape estimation fields, several methods have leveraged the graph-like structure of 3D meshes. In the face reconstruction domain, a mesh-based autoencoder was proposed by [99], with operations applied to the face represented as a mesh surface. A similar approach was presented by [35] in the hand pose domain, reconstructing a full 3D hand

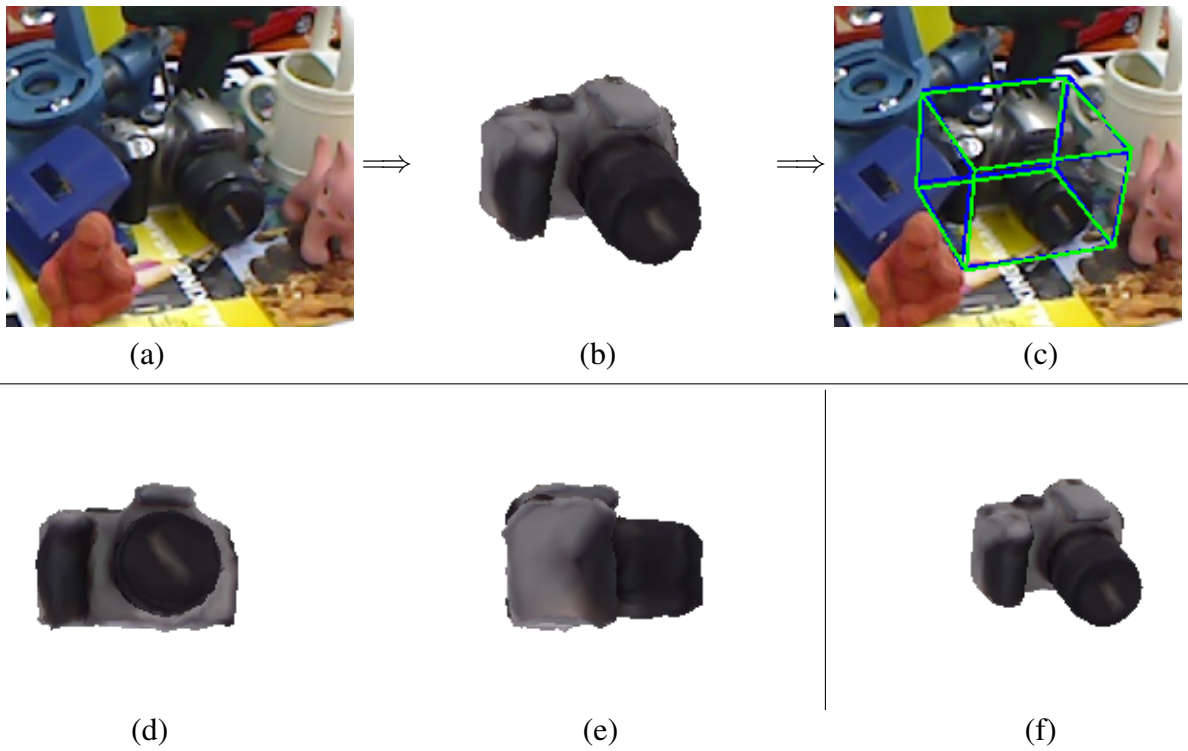


Figure 3.1: This method recovers the 6D pose of an object from a single image by means of an intermediate object reconstruction. From an RGB image of a detected object (a), a pose conditioned mesh (b) is reconstructed from which 6D pose can be extracted (c). (d) and (e) present alternative viewpoints of reconstruction (b) while (f) displays the groundtruth mesh.

mesh using spectral convolutions [22]. Given that the object mesh can also be represented as a graph, and inspired by the success of these approaches, a Graph Convolutional Neural Network (GraphCNN) capable of estimating a pose-conditioned 3D mesh is proposed.

By explicitly exploiting the 3D object dense mesh for pose estimation, this approach differs from prior art methods that either learn sparse 3D keypoints or estimate dense per-pixel poses. More specifically, this is achieved by utilizing the 3D object model information prior to prediction, which is different from previous methods that fit the object 3D keypoints to correspondent 2D predictions in a *post-processing* step [128, 98, 138]. The problem at hand involves rigid objects with 3D shapes that are known *a priori*, meaning that the object structure and spatial properties have been pre-determined and are available for use in the pose estimation process. This availability of full 3D shape supervision, as opposed to 2D mask supervision used in previous works [3, 66, 35], allows for the learning and inference of pose-dependent shapes, making this approach more computationally affordable and effective.

3.1 Related Work

The recent work by Wang et al. [138] introduces an new object surface coordinate representation and proposes a dense per-pixel estimation of surface points in 2D space. This change in paradigm is analogous to the one seen in the fields of human body [1] and hand [134] pose estimation. In concurrent pose and reconstruction domains, there have been introduced techniques leveraging the graph-like structure inherent in 3D meshes in order to extract dense representations. For instance, within the realm of face reconstruction, a mesh-based autoencoder was introduced by Ranjan et al. [99], which applied operations to a facial mesh surface encoding. A comparable strategy was employed in the hand pose domain by Ge et al. [35], who reconstructed a complete 3D hand mesh utilizing spectral convolutions [22], in order to recover its hand pose. Recognizing that the object mesh can similarly be characterized as a graph, and drawing inspiration from the success of these methods, a GraphCNN is proposed, designed to estimate a pose-conditioned 3D mesh. This approach capitalizes on the network capacity to harness known topological information effectively.

3.2 Methodology

The goal of our method is to estimate the full 6D pose of a rigid object given a monocular RGB input image $\mathcal{I}$. In order to successfully estimate the pose, a deep pipeline is proposed which is design to incorporate a per-object architecture, which fully exploits the prior information of the object. Our proposed method is inspired by successful approaches from prior works [35, 139], and it combines the object known topology with the encoded pose information extracted from $\mathcal{I}$ during the extraction stage. By integrating the object known topology with the image encoded pose information, our method generates a pose conditioned reconstructed mesh that closely approximates its true shape. The estimated mesh information is then used to recover the allocentric orientation of the target object, allowing us to determine the object spatial orientation relative to a fixed point in space. Finally, to obtain the complete 6D pose of the object, we adopt various approaches from the literature [66, 63] to recover the egocentric orientation and lift it to 6D space.

A pretrained FasterRCNN [103], the improved version of the ground breaking 2D object detector

Region-based Convolutional Neural Network (R-CNN) [36], is fine tuned on the existing pose estimation datasets in order to locate an object in 2D space. After the 2D object detector has successfully detected the object of interest in the input image, it is utilized to crop a region-of-interest (*ROI*) $\mathcal{I}_{ROI}$ containing the object. The *ROI* is then used in subsequent stages of our pipeline, where it is processed at a high resolution to extract fine details of the object appearance.

We were inspired by previous research on the discovery of latent keypoints alignment for object pose estimation [125], which has been shown to be highly effective in achieving accurate 6D pose estimation, and by the success of dense mesh estimators, such as those used in [1, 35], which have demonstrated impressive performance in reconstructing detailed 3D object meshes from 2D input images. To estimate the 3D keypoints position of the object, we employ a similar approach, where we reconstruct a 3D object mesh, denoted as $\hat{\mathcal{M}}$. In Section 3.2.2, we provide detailed discussion on how we learn to estimate the mesh vertices using a GraphCNN [22]. By leveraging the power of GraphCNNs, we are able to effectively encode the object topology and utilize the available prior information to achieve accurate and robust estimation of the 3D object mesh.

Three critical cues, $\hat{\mathcal{H}}_m$, $\hat{\mathcal{H}}_c$, and $\hat{\mathcal{M}}$, the estimated mask, centroid and recontructed mesh, respectively, are essential for accurately recovering the allocentric orientation of $\hat{\mathcal{M}}$. The allocentric orientation of $\hat{\mathcal{M}}$ is then lifted into a 6D egocentric pose relative to the camera. Utilizing a similar technique to the one used by Suwajanakorn et al. [125], we align the keypoints of $\hat{\mathcal{M}}$ with those of a canonical mesh, denoted as $\mathcal{M}$, using a differentiable alignment method, as detailed in Section 3.2.3. This enables us to recover the orientation of $\hat{\mathcal{M}}$ in a robust and accurate manner, while leveraging the available prior information of the object topology.

The proposed method, illustrated in Figure 3.2, is composed of three main stages: **1)** a 2D fine localization module, which leverages a fine-tuned off-the-shelf detector to produce a segmentation mask and the 2D projection of the object centroid, **2)** a GraphCNN [22] which is optimized to reconstruct a pose conditioned dense 3D mesh of the target object, and lastly **3)** an estimation of the object allocentric orientation by computing the optimal rotation necessary to align the produced mesh to the object canonical mesh. To achieve a seamless and integrated learning process, the three main components of the proposed system are jointly trained end-to-end. The system is designed to combine the inputs and outputs, including the initally estimated padded 2D bounding box, centroid projection,

mask segmentation, and allocentric pose, along with the camera intrinsic parameters. By using these parameters, the method is able to accurately lift the 6D egocentric pose of the object in relation to the camera, providing a comprehensive understanding of the object spatial orientation and location.

In the rest of this section, the 3 main stages of our 6D object pose estimation pipeline are discussed in greater detail.

3.2.1 Fine Object Localization

As an important piece of our pipeline we want to make fine localization of the object appearance. Localization cues exploited in this section caries essential information for pose estimation and integral to lift 6D pose which will be discussed later in Section 3.2.3.

Prior to the first stage of our pipeline we use an off-the-shelf detector to retrieve the 2D bounding box location of the object of interest in the image. For this purpose, we use a Faster-RCNN [103] based object detector, which offers a reasonable inference speed with a very high detection IOU accuracy rate. Object is detected from $\mathcal{I}$ and cropped according to the estimated bounding box to get $\mathcal{I}_{ROI}$. Afterwards, image padding is applied to the cropped region in an effort to make X robust to small inaccuracies introduced by the detector. During training, the image is randomly padded while at test time, we fix the padded region to avoid introducing randomness to our final estimation. Padded $\mathcal{I}_{ROI}$ is then resized to a higher resolution of 128×128. The reason for such a high resolution will become clear on the following section. This stage is trained independently from the rest.

As the task of 6D pose estimation requires high precision, both $\hat{\mathcal{H}}_c$ and $\hat{\mathcal{H}}_m$ are estimated at a high resolution, which makes our method robust to inaccuracies in the 2D object detector. Given a $\mathcal{I}_{ROI}$ recovered by the fine-tuned 2D object detector, we employ a U-Net[108] like encoder-decoder architecture, with two distinct decoders for each output. Both decoders share a ResNet-18[43] encoder, which encodes the input image features, and the decoders upsample the encoded features up to a resolution of 128x128. Importantly, the decoders do not share parameters between each other, which allows them to learn distinct features and effectively capture the required details for each output. The estimated binary segmentation mask $\hat{\mathcal{H}}_m$ enables accurate separation of the object from the background while the projected 2D $\hat{\mathcal{H}}_c$, provides information about the location of the object within the

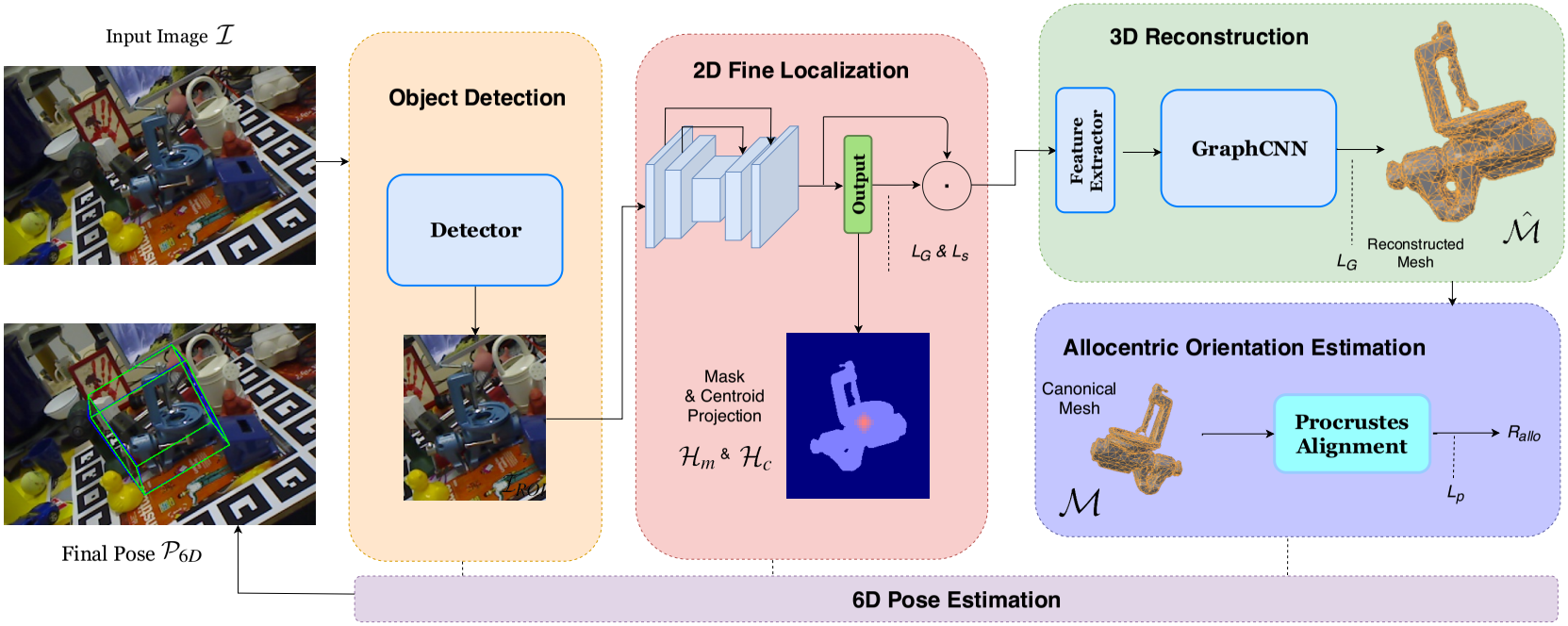


Figure 3.2: **Overview of the pipeline the object topology is exploited for 6D pose estimation.** Given an RGB input image $\mathcal{I}$, an object region of interest (ROI), denoted as $\mathcal{I}_{ROI}$, is obtained by cropping with a fine-tuned 2D object detector (depicted in orange). This cropped region is employed to precisely localize the object by estimating a binary object mask, $\hat{\mathcal{H}}_m$, and a heatmap, $\hat{\mathcal{H}}_c$, for the object centroid (depicted in pink), which are subsequently used to reconstruct a pose-conditioned 3D mesh, $\hat{\mathcal{M}}$, of the object (depicted in green). An allocentric orientation, $\mathcal{R}_a$, of $\hat{\mathcal{M}}$ with respect to a corresponding canonical object mesh, $\mathcal{M}$, can be estimated in a differentiable manner (indicated in purple) by determining the relative rotation of $\hat{\mathcal{M}}$ to the corresponding canonical object mesh, $\mathcal{M}$, through Procrustes alignment. The cropped image locations, $\hat{\mathcal{H}}_m$, $\hat{\mathcal{H}}_c$, and $\mathcal{R}_a$ are later utilized to estimate the complete 6D pose of the object. The symbol $\odot$ signifies the channel-wise concatenation operation.

ROI.

To estimate the 2D projection of the object centroid, we utilize a heatmap $\hat{\mathcal{H}}_c$ defined as a 2D Gaussian distribution centered on the expected 2D projection of the centroid, with $\sigma = 5px$. This effectively models the probability distribution of the object centroid around its expected 2D projection in the input image. By using a Gaussian distribution, we are able to preserve translation equivariance w.r.t. the image, which is crucial for accurate 6D pose estimation.

Inspired by Monodepth [85], the $\hat{\mathcal{H}}_m$ and $\hat{\mathcal{H}}_c$ are optimized by minimizing a multi-loss function L_s which is constructed from multiple scales and stages of the decoders. Extra resources are spent on optimizing the mask and centroid tasks because these are the main cues for translation estimation and small pixel-level inaccuracies can lead to large real world errors. In practice, during training only three scales are used for supervision. At test time, only outputs from the scale with the highest resolution are used for pose estimation. During inference, $\hat{\mathcal{H}}_m^{|S|}$ and $\hat{\mathcal{H}}_c^{|S|}$ are utilized as estimations for $\hat{\mathcal{H}}_m$ and $\hat{\mathcal{H}}_c$, respectively. This is done using the output from the highest resolution scale among a set of scales S, with $\mid S \mid$ corresponding to the index for the highest scale among a set of scales S. The loss function L_s can be formalized by:

$$L_s = \sum_{i=0}^{|S|} \frac{\lambda_m L_m^{(i)} + \lambda_c L_c^{(i)}}{2^i} \tag{3.1}$$

where i is the index of outputs at different scales S and λ_m and λ_c are the weight ratios for the mask and the centroid losses.

L_m and L_c, the mask and 2D projection losses at each scale S_i, are defined as a binary cross entropy and mean squared error functions, respectively:

$$L_m^{(i)} = \frac{1}{|\mathcal{H}_m^{(i)}|} \sum_p -(\mathcal{H}_m^{(i)}(p) \cdot log(\hat{\mathcal{H}}_m^{(i)}(p)) + (1 - \mathcal{H}_m^{(i)}(p)) \cdot log(1 - \hat{\mathcal{H}}_m^{(i)}(p))) \tag{3.2}$$

$$L_c^{(i)} = \sum_p \|\hat{\mathcal{H}}_c^{(i)}(p) - \mathcal{H}_c^{(i)}(p)\|_2^2 \tag{3.3}$$

where $\mathcal{H}_m^{(i)}$ and $\mathcal{H}_c^{(i)}$, the groundtruth mask and 2D centroid heatmap with their corresponding predic-

tions $\hat{\mathcal{H}}_m^{(i)}$ and $\hat{\mathcal{H}}_c^{(i)}$ are used for each pixel location p within a length of $|\hat{\mathcal{H}}_m^{(i)}|$.

3.2.2 3D Mesh Reconstruction

The next stage of the pipeline involves the reconstruction of a pose conditioned 3D mesh of the target object. In order to achieve an accurate reconstruction, a GraphCNN [22] which has been successfully applied to structured modelling [139, 35], is employed to output the 3D coordinate position of each vertex of the mesh. Assuming that the topology of the object surface remains fixed for each rigid object, its surface is represented as a pose conditioned 3D mesh $\mathcal{M}$. $\mathcal{M}$ can be represented as an undirected graph $G = (V, E, W)$, where V is a set of $|V(G)| = v$ vertices, E a set of $|E(G)| = e$ undirected edges and $W = \{0, 1\}^{v \times v}$ is an adjacency matrix, where $\{w_{ij} = 1 | (i, j) \in E, w_{ij} \in W\}$. The canonical mesh, which is the mesh without any transformation applied, can be represented as $G_{can} = (V_{can}, E, W)$, while the fully reconstructed mesh can be represented by $\hat{G} = (\hat{V}, E, W)$. By utilizing a canonical mesh, we are able to effectively encode the information of the object topology within the model itself.

To integrate the outputs from the previous stages of the pipeline, we encode them into a latent vector and resize them into a mesh signal $f \in \mathbb{R}^{|V| \times D}$, where D represents the data dimensionality in each vertex. The mesh signal is then forward propagated through a hierarchical GraphCNN architecture [22, 35, 99]. The hierarchical structure of the GraphCNN is composed of intercalated graph convolution operations and mesh feature upscaling, which allows us to operate on a finer mesh version.

The task of pose conditioned 3D mesh reconstruction is fully supervised by a loss function $L_G = L_v + L_e + L_l$, which combines multiple 3D loss constraints in the mesh generation process, similar to previous works such as [35, 139]. The vertex loss L_v constrains the 3D location of each vertex in the reconstructed mesh, ensuring that they are accurate and align with the ground truth object surface. The edge loss L_e penalizes incorrect edge lengths, which helps to maintain the structural integrity of the reconstructed mesh. Finally, the Laplacian loss L_l enforces mesh surface smoothness, ensuring that the reconstructed mesh is free of artifacts. Formally, the losses are defined by:

$$L_v = \sum_{i=1}^{v} \|\hat{V}_i - V_i\|_2^2 \tag{3.4}$$

$$L_e = \sum_{i}^{v} \sum_{V_k \in \mathcal{N}(V_i)} \left(\|\hat{V}_i - \hat{V}_k\|_2^2 - \|V_i - V_k\|_2^2 \right)^2 \tag{3.5}$$

$$L_l = \sum_{i=1}^{v} \left\| (\hat{V}_i - V_i) - \frac{\sum_{V_k \in \mathcal{N}(V_i)} (\hat{V}_k - V_k)}{|\mathcal{N}(V_i)|} \right\|_2^2 \tag{3.6}$$

where $\mathcal{N}(V_i)$ indicates is the set of neighbouring vertices defined as $\{V_k \in \mathcal{N}(V_i) | W_{i,k} = 1\}$. For the experiments, loss weighting parameters are set to $\lambda_v = 1.0$, $\lambda_e = 1.0$ and $\lambda_l = 1.0$.

3.2.3 Allocentric Pose Estimation & 6D Egocentric Pose Lifting

To recover the allocentric orientation of the object, we aim to compute the optimal alignment rotation between the two sets of vertices from the set of vertices of the reconstructed mesh $\hat{G}$ and the set of vertices of the canonical mesh pointcloud G_{can}. This task is known as the Generalized Orthogonal Procrustes Problem [113], and it is formally defined as $\hat{V}(\hat{G}) = R_a V_{can}(G_{can})$, where R_a represents the relative rotation between the two sets of vertices. To compute the optimal solution for R_a, we use the following equation:

$$R_a = \mathcal{V} diag(1, 1, \ldots, sign(det(\mathcal{U}\mathcal{V}^T))\mathcal{U}^T \tag{3.7}$$

where $\mathcal{U}, \mathcal{E}, \mathcal{V}^T = SVD\left(\hat{V} V_{can}^T\right)$.

As SVD can be implemented in a differentiable manner, it used to incorporate the relative pose error it into the optimization process. To further improve the accuracy of the allocentric orientation estimation, we add this additional constraint to the optimization process, similar to [125]. This constraint takes the form of a pose loss L_p, which is defined as:

$$L_p = arcsin\left(\frac{\|\hat{R}_a - R_a\|_2}{2\sqrt{2}}\right) \tag{3.8}$$

which penalizes the difference between the ground truth and estimated allocentric orientations. All the individual loss terms from the previous stages and this stage, namely L_s, L_g, and L_a, are combined into a final loss L which jointly learn parameters of the differentiable three stages of the pipeline by optimizing L:

$$L = \lambda_s L_s + \lambda_g L_g + \lambda_p L_p \tag{3.9}$$

λ_s, λ_g, and λ_p are hyperparameters that control the relative importance of each term.

In order to overcome symmetric challenges, for each object displaying symmetric properties, an object-specific axis of symmetry is defined, as different labeled poses might have identical appearances. When computing the vertex loss L_v and the allocentric orientation loss L_a,additional ground truth annotations are generated by rotating the vertex positions V and the rotation matrix R_a around the pre-defined axis of rotation. The error is measured for L_v and L_a on all available annotations and backpropagated through the minimum error.

At last, the egocentric 6D object pose with respect to the camera is determined by utilizing the information acquired from previous stages.The process is initiated by positioning the estimated 2D centroid projection relative to the bounding box location of the *ROI*. In accordance with the approach delineated in [66], the computation of $\mathcal{P}_{6D} = \left[\begin{smallmatrix} R & T \\ 0^T & 1 \end{smallmatrix}\right]$ is carried out, where $R = R_c R_a$ and $T = R_c[0, 0, \hat{d}]^T$. $R_c \in SO(3)$ represents the rotation required to align the camera principal axis $a = [0, 0, 1]^T$ with $r = K_c^{-1}\hat{c}$, in which r is the ray that passes through the repositioned 2D centroid projection and is determined using the inverse camera intrinsics matrix K^{-1} and the estimated 2D centroid projection $\hat{c}$. Consequently, R_c can be calculated as $R_c = I + [v]\times + \frac{[v]\times^2}{1+a\cdot r}$, where $v = a \times r$ and $[v]_\times$ is the skew-symmetric cross-product matrix of v, while I denotes the identity matrix. To compute the translation vector T, the distance from the object to the virtual *ROI* camera is required. Given the allocentric pose and the 3D model of the target object, the mesh is rendered at an arbitrary distance d with the estimated allocentric pose. By utilizing the rendering and the estimated masks' bounding box diagonal lengths, l and $\hat{l}$ respectively, the inference of $\hat{d} = \frac{l \cdot d}{\hat{l}}$ can be achieved [63].

3.3 Experimental Evaluation

We evaluate our method and compare it against previous RGB based state of the art approaches[128, 98, 63, 148, 86] that do not rely on iterative refinement stages or use of depth information for accurate 6D pose estimation.



Figure 3.3: **Qualitative results obtained utilizing the reconstruction intermediate representation.** For every example: (a) estimated mask and centroid projection, (b) reconstructed pose-conditioned mesh, and (c) groundtruth (in green) and our estimated (in blue) pose visualized on the input image. The last column features failure examples, which typically occur due to poor $\hat{\mathcal{H}}_m$ and $\hat{\mathcal{H}}_c$ estimations or reconstruction errors caused by occlusion, extreme illumination conditions, or ambiguous viewpoints. It is important to note that the mask and centroid projection are blended on the same image, and test images are zoomed in for improved visualization.

Method	ADD-(S) Accuracy			2D Projection Accuracy		
Objects	BB8[98]	Tekin[128]	**OURS**	BB8[98]	Tekin[128]	**OURS**
Ape	27.90	21.62	**35.11**	95.30	92.10	**98.10**
Benchvise	62.00	81.80	**85.01**	80.00	**95.06**	88.20
Camera	40.10	36.57	**45.46**	80.9	**93.24**	80.80
Can	48.10	**68.80**	68.31	84.10	**97.44**	94.10
Cat	**45.20**	41.82	41.98	97.10	**97.41**	97.10
Driller	58.60	63.51	**72.73**	74.10	79.41	**83.20**
Duck	32.80	27.23	**35.57**	81.20	94.65	**96.70**
Eggbox*	40.00	69.58	**75.80**	87.90	**90.33**	81.10
Glue*	27.00	**80.02**	53.40	89.0	**96.53**	67.60
Holepuncher	42.40	42.63	**44.46**	90.50	92.86	**97.30**
Iron	67.00	74.97	**81.25**	78.90	**82.94**	82.50
Lamp	39.90	71.11	**75.71**	74.40	76.87	**82.60**
Phone	35.20	47.74	**55.35**	77.60	**86.07**	83.70
Average	43.55	55.95	**59.32**	83.9	**90.37**	87.15

Table 3.1: **ADD-(S) results of our experiment on the LINEMOD dataset.** Our method outperforms all prior methods w.r.t. ADD-(S) accuracy falling short on the 2D projection metric. Symmetric objects are denoted with a *.

3.3.1 Experimental Setup

Datasets Setup. To train the model on the LINEMOD and OCCLUSION datasets, synthetic data is generated by combining existing real images of the LINEMOD [45] dataset and digitally rendered versions of the object. These digital renderings are subjected to a series of random transformations, which introduces a degree of variation to the synthetic data. To sample the training and evaluation sets, 15% of each object image set are selected for training and the rest for evaluation, with a constraint that the relative pose between the selected images is larger than $\frac{\pi}{12}$ [98]. The objects are cropped according to their silhouettes and placed on random backgrounds using the "Cut and Paste" technique [33]. To further augment the training dataset, rotation transformations are applied to the cropped instances in pixel domain, and their groundtruth poses are updated accordingly. This strategy results in the generation of 15000 training instances per object. The YCB-Video dataset provides its own training set [148]. Online augmentation is performed on all datasets by randomly adjusting the hue, saturation, and exposure of the image. Additionally, Gaussian noise is added to the images for further variability.

Implementation Details. The loss weighting parameters are set to $\lambda_s = 1.0$, $\lambda_g = 1.0$ and $\lambda_l = 5.0$. The networks' parameters are updated for 20 epochs using the Adam [65] optimizer, with its initial learning rate set to 10^{-3} and batch-size of 8. Given an input image 480×640, our method runs at

Method	ADD-(S) Accuracy				2D Projection Accuracy			
Objects	PoseCNN [148]	Tekin [128]	Oberweger‡ [86]	**OURS**	PoseCNN [148]	Tekin [128]	Oberweger‡ [86]	**OURS**
Ape	9.60	-	16.50	14.90	34.6	7.01	64.70	42.50
Benchvise†	-	-	-	84.01	-	-	-	87.73
Can	45.20	-	42.05	40.11	15.10	11.20	53.00	30.72
Cat	0.93	-	2.82	1.23	10.40	3.62	47.90	22.32
Driller	41.40	-	47.10	53.31	7.40	1.40	35.10	**44.89**
Duck	19.60	-	11.00	7.98	31.80	5.07	36.10	**16.80**
Eggbox*	22.00	-	24.70	29.44	1.90	-	10.30	15.57
Glue*	38.50	-	39.50	25.98	13.80	6.53	**44.90**	4.28
Holepuncher	22.10	-	21.90	29.40	23.10	8.26	52.90	**57.5**
Average w/o †	24.9	-	25.80	25.29	17.20	6.16	43.10	29.32

Table 3.2: **ADD-(S) comparison of our approach on the OCCLUSION dataset with state-of-the-art.** Our method outperforms prior state of the art, specially when it comes to 2D projection accuracy. Symmetric objects are denoted with a *. The method by Oberweger et al. [86], denoted with a ‡, is evaluated on a combined dataset of OCCLUSION and synthetic data, which is not public.

17 FPS on a GTX1080 Ti GPU. The most computationally expensive operation is the detection stage, which takes around 28ms to compute. This step can be further improved by using a lighter detector. The mask segmentation and centroid estimation stages follow closely, taking only 8ms to process. The GraphCNN forward pass stage takes 14ms, while the lifting to 6D takes only 9ms. The total processing time for the entire pipeline is 59ms, showcasing the real-time applicability of our method. The method was implemented using PyTorch [91]. All the experiments were done on a computer equipped with a GTX1080Ti.

Evaluation Metrics. To study and compare the reconstruction-based approach with prior methods, the three main benchmark datasets for 6D object pose estimation, LINEMOD [45], OCCLUSION [10], and YCB-Video [148], will be evaluated using the ADD-(S), 2D projection, and AUC over ADD-(S) metrics. A correct pose is considered as an ADD-(S) score below 10% of the diameter of the target object. The AUC metric over ADD-(S), which was first used by Xiang *et. al.* [148], will be used and is defined as the area under the accuracy-threshold curve when using this metric. The threshold is varied up to a maximum of $10cm$ to build the curve. It is worth noting that this metric is commonly not used on the LINEMOD and OCCLUSION datasets, and thus, it will only be presented on the YCB-Video dataset.

3.3.2 Evaluation Results

LINEMOD. Higher ADD-(S) accuracies are reported for all but three objects, achieving an overall average accuracy approximately 4% higher than Tekin [128], as shown in Table 3.1. However, the results for the 2D Projection accuracy evaluation are not as favorable. This reduced performance is due to the imprecision in the 2D localization tasks. While our pose estimation is done in 3D space, YOLO6D [128] perform keypoint localization in 2D space. For this reason, we outperform in metrics that reward 3D precision. Additional qualitative results can be found in Figure 3.3.

OCCLUSION. The results on the OCCLUSION dataset are displayed in Table 3.2, with the results from PoseCNN and Tekin reported in [86]. The presented outcomes demonstrate competitive scores compared to prior works. This work outperforms PoseCNN and Tekin by 12.2% and 23.16% respectively when compared on the 2D projection accuracy metric. Nonetheless, the occlusion-aware approach of [86] surpasses the proposed method. The object topology-based pose estimation approach outperforms the more recent work of Wang et al. [138], which requires RGB-D input, by 5.61% when considering the *Benchwise* object, achieving an overall 35.81% 2D Projection accuracy.

YCB-Video. Furthermore, the results of the proposed method on the more recent YCB-Video dataset are presented. The performance significantly exceeds both PoseCNN [148] and Oberweger [86] across all evaluated metrics. As the method is not designed solely for occluded scenarios, it can better generalize across various scenes by leveraging its 3D reasoning with precise localization, leading to more accurate pose estimation. A detailed analysis of the performance on this dataset is available in Table 3.3, and qualitative examples are shown in Figure 3.3.

Methods	PoseCNN [148]	Oberweger[‡] [86]	**OURS**
ADD-(S)	21.30	33.60	**47.09**
AUC	61.00	61.40	**67.52**
2D Projection	3.72	23.10	**55.22**

Table 3.3: **Comparison of YCB-Video dataset results with state-of-the-art approaches.** Our method outperforms prior state of the art by a significant margin in all metrics. While the method by Oberweger et al. [86], denoted with a *, is evaluated along with synthetic test data, we still outperform it on YCB-V.

Information	w/o Priors		Ours w/ GT Priors		
Objects	**Ours**	SSD6D [63]	w/ $\mathcal{H}_c$	w/ $\mathcal{H}_m$	w/ $\mathcal{H}_c$&$\mathcal{H}_m$
Ape	35.11	18.04	35.52	51.13	53.16
Benchwise	85.01	67.38	85.83	86.49	87.07
Camera	45.46	27.81	48.38	70.52	73.61
Can	68.31	64.30	69.57	80.18	81.35
Cat	41.98	33.33	42.49	58.44	59.46
Driller	72.73	62.04	72.90	75.42	76.60
Duck	35.57	16.67	36.13	55.98	57.26
Eggbox*	75.8	75.20	91.90	97.50	97.60
Glue*	53.4	52.50	56.00	56.10	56.60
Holepuncher	44.46	35.57	45.43	62.57	63.62
Iron	81.25	62.59	81.77	79.08	79.51
Lamp	75.71	70.09	76.12	83.70	84.60
Phone	55.35	43.36	56.72	69.75	70.64
Average	**59.32**	48.37	61.44	**71.30**	**72.39**

Figure 3.4: ADD-(S) comparison of our approach by using groundtruth priors for the centroid and the mask estimation intermediate steps on LINEMOD.

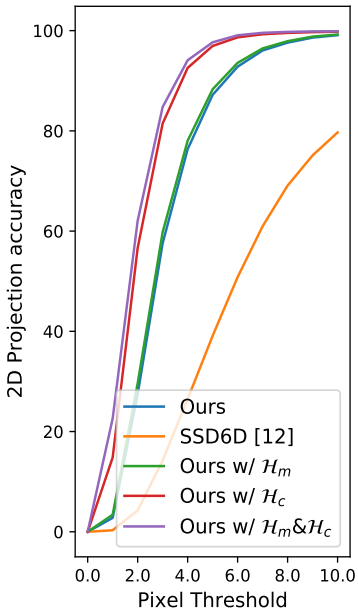


Figure 3.5: 2D Projection accuracy using different priors on LINEMOD.

3.3.3 Aditional Discussions

In this section, the individual impact of the 2D location and reconstruction stages of the pipeline on the final 6D pose estimation is examined and discussed. The experiments presented in this section were conducted using the LINEMOD dataset.

2D Localization As discussed earlier, the employment of a detector and the estimations of both $\hat{\mathcal{H}}_m$ and $\hat{\mathcal{H}}_c$ serve as *ad hoc* components within the comprehensive 6D pose estimator. In Table 3.4, the ADD-(S) accuracy of the model is presented when ground truth information is accessible for $\hat{\mathcal{H}}_m$ and $\hat{\mathcal{H}}_c$, while Figure 3.5 displays each respective 2D Projection accuracy. In [63], the authors leverage their orientation and bounding box estimations to recover the centroid position, and it is demonstrated that the proposed method of estimating the 2D centroid projection surpasses the one employed by [63] for all objects. The more precise localization of the centroid projection is attributed to the high-resolution $\hat{\mathcal{H}}_c$ estimation. When utilizing non-available ground truth information for secondary tasks, the proposed method achieves exceptionally high accuracy for the majority of objects. Additionally, the influence of each 2D location task on the respective metrics can be observed. The 2D Projection accuracy is significantly impacted by the quality of $\hat{\mathcal{H}}_c$, as it dictates where the object will be projected within the 2D plane of the image. On the other hand, ADD-(S) is correlated with the quality of the

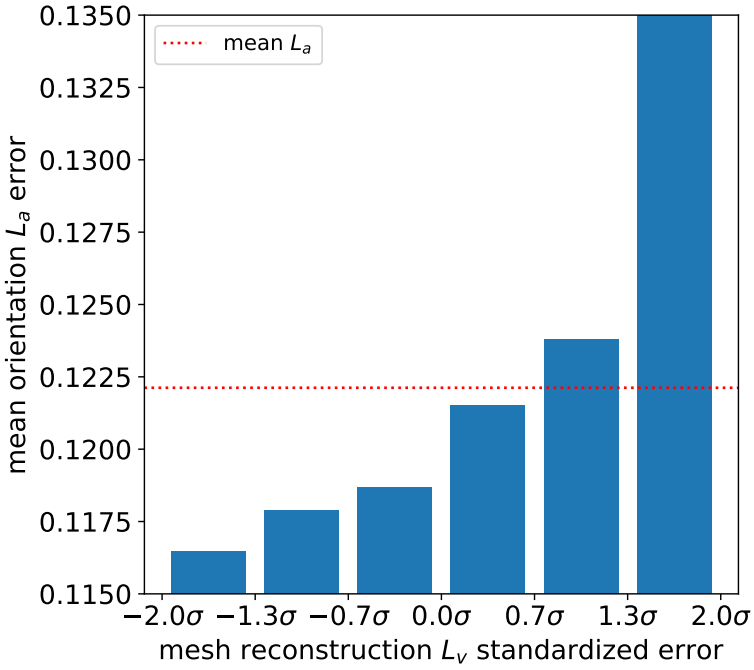


Figure 3.6: Histogram of mean pose losses L_a over standardized mesh reconstruction error L_v thresholds. This measurement reveals a directly proportional relationship between the mesh reconstruction error, which can be evaluated at inference time, and the predicted pose error, which is unknown at that stage.

$\hat{\mathcal{H}}_m$, since it determines the distance between the object and the camera, thereby heavily affecting $\mathcal{P}_{6D}$ in 3D space.

Mesh Reconstruction. In addition to evaluating the quality of the 2D detection tasks, a more detailed assessment of the mesh reconstruction errors is also conducted. To accurately recover the allocentric pose, it is essential that the reconstructed mesh closely resembles its canonical counterpart; otherwise, the alignment strategy will be incapable of retrieving the correct orientation. As a result, it is important to determine how the quality of the reconstructed mesh influences orientation recovery. To quantify this relationship, the mean orientation error L_a is measured for meshes with a reconstruction error L_v that falls within specific thresholds. Since L_v varies among objects, it is standardized for each object prior to examining the correlation. The histogram depicted in Figure 3.6 assesses the impact of reconstruction quality on pose prediction. When the reconstruction error is within the top or lower ~ 16 percentile, a correspondingly high or low pose error is observed. This finding suggests that the quality of the recovered orientation is indeed correlated with the quality of the reconstructed mesh. By taking this observation into account, the model can generate an intrinsic confidence score for the recovered pose simply by measuring the reconstruction error. Since the reconstruction error can

be calculated without requiring ground truth data, the orientation estimation can be independently verified during inference.

3.4 Conclusion

In this chapter, a novel method for instance-level 6D object pose estimation has been introduced. Unlike previous approaches, the proposed method leverages the topological information of the target object to its full potential. Through high-resolution extractions of both the object mask and the 2D centroid projection, the 3D position of the object can be recovered. Its pose recovery is conducted in 3D space through the reconstruction of a pose conditioned 3D mesh of the target, from which the orientation is recovered by comparing it with the canonical version of the object 3D mesh. The performance of the method has been validated on three challenging benchmark datasets: LINEMOD, OCCLUSION and YCB-Video. The method outperformed non-occlusion aware approaches while being comparable to methods optimized for occlusion. Additionally, it was shown that the reconstruction error, used as a training signal, can also serve as a reliable indicator of pose estimation accuracy at inference time, enabling the model to produce a pose confidence score.

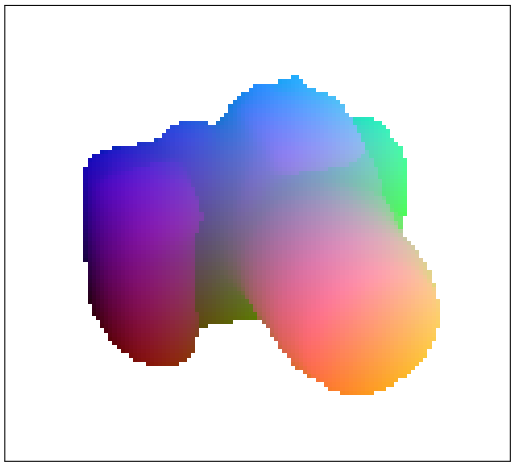
Chapter 4

Multi-Frustum Network

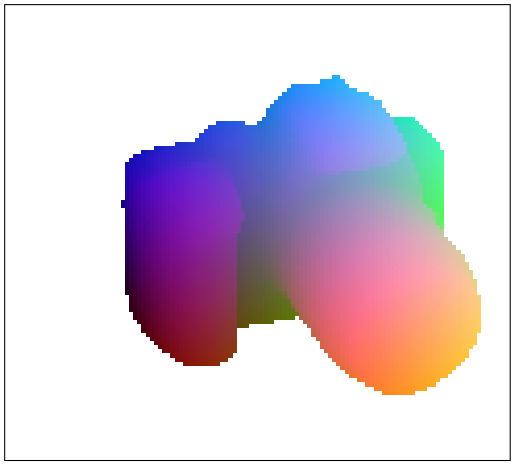
In the last chapter, the proposed method fails to properly recover the distance to the camera. Table 3.4 indicates that a significant increase in accuracy is observed when ground truth cues for translation are available. While the 2D location within the cropped image was specifically found through the estimation of its centroid 2D projection and its orientation through the mesh reconstruction, the distance to the camera was found through the estimated amodal mask diagonal size. An amodal mask refers to the segmentation region of an object including occluded regions. In practice, this is hard to achieve, as for accurate amodal segmentation the model must implicitly learn to recreate the occluded region, which itself is viewpoint dependent. In other words, it requires the model to learn the object's pose, which is recovered by another module. Moreover, even if the amodal segmentation is acceptable, the estimation of the object mask is prone to precision errors around the boundary of the object. Moreover, accurate translation relies on an external orientation estimator model. Dependencies on multiple independent tasks, where inaccuracies in each intermediate step can propagate and compound, can ultimately degrade the precision of the overall translation estimation. This chapter proposes a novel method that is specifically aimed at recovering the full pose in a fast and accurate manner without compromising the translation estimation precision and without relying on off-the-shelf external detectors.

RGB only methods rely on clues from an image in order to extract pose information, which means these methods are highly dependent on the appearance of the object. Yet, object appearance changes in different magnitudes with respect to the rotation or translation of the object. Due to the projective

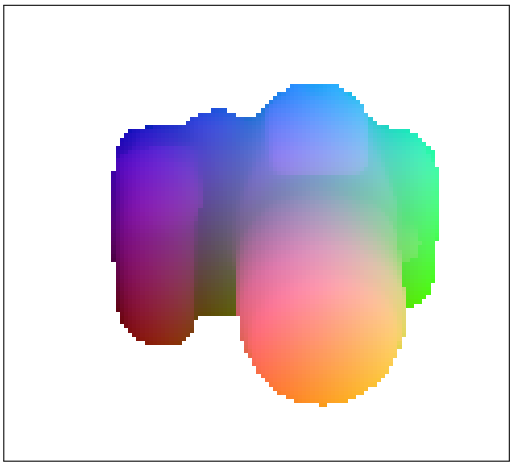
function, small rotations can largely impact the appearance of an object while a small translation away from the camera, i.e. zoom, might have very little. Most state-of-the-art methods rely on an intermediate 2D representation, whether that is detecting dense object coordinates maps [138, 74, 153,



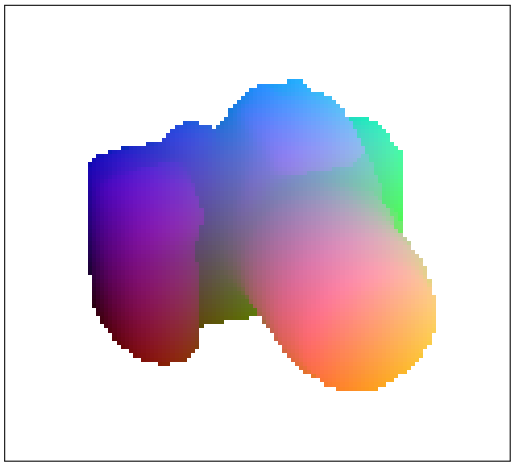
(a) Ground Truth Pose



(b) Shift Perturbation
Translation error : 17mm
ADD error : 17mm
MSE NOCS error : 220



(c) Rotation Perturbation
Translation error : 0mm
ADD error : 17mm
MSE NOCS error : 2.5



(d) Zoom Perturbation
Translation error : 17mm
ADD error : 17mm
MSE NOCS error : 0.5

Figure 4.1: **MFNet motivation.** The motivation for this work is illustrated using a 2D NOCS representation in a crop, similar to the approaches in [138, 137, 25, 58]. A ground truth pose (a) is taken and perturbed in terms of translation parallel to the camera plane (b), distance to the camera (c), and rotation (d). All perturbations result in a final ADD-(S) error of 17mm (the standard threshold error for the object). When applying the common mean squared error(MSE) loss for optimizing NOCS-based methods, it can be observed that the magnitude of the loss varies for all perturbations. The priority becomes evident: the network first learns 2D detection, followed by object rotation, and only then the distance to the camera.

137, 11] or keypoints [128, 92] and this rotation/translation contrast persists: due to the collapse of the depth dimension, the 2D representation is more heavily impacted by changes in rotation of the object than image zoom (See Fig. 4.1). For that reason, during optimization, the trade-off between rotation and translation errors is heavily skewed to the former, resulting in poor translation estimations. The resulting underperformance of translation accuracy has been noticed by the object pose estimation community however no true solution has been proposed. An *ad-hoc* technique was proposed by CDPN [74] and other works have followed the same approach [137, 67]. However, this method does not tackle the problem directly and instead expertly balances the trade-offs of direct pose estimation and 2D-3D correspondence methods.

MFNet addresses this issue with an innovative design. By generating an intermediate representation directly in 3D, the inductive bias arising from previous 2D-3D correspondence methods is eliminated, resulting in improved performance, particularly in terms of translation errors. Our method achieves superior pose accuracy and processes images in half the time required by earlier approaches. The novel representation can be converted into a full 6D pose using a Kabsch step to solve the procrustes alignment, circumventing the need for a computationally expensive Perspective-n-Point (PnP) step. Kabsch can be efficiently solved through SVD while PnP requires solving a non-linear optimization problem which usually requires an iterative solution. Moreover, the PnP problem is not inherently differentiable, and imposing differentiability upon it significantly increases computational complexity and slows down the overall optimization process [18]. With the proposed method, not only is it possible to outperform previous techniques on standard metrics, but extensive comparison studies also reveal the high degree of translation precision that can be achieved.

4.1 Related Work

Detector-free 6D object pose estimators have been proposed in the past. EPOS [51] and DPOD [153] estimate the 3D object keypoint correspondences for every available pixel in the image. PVNet [92] similarly extracts pixel 2D offsets on each pixel. As discussed in Chapter 1, this leads to increased complexity and computational requirements. Instead, researches have chosen to use detectors in an end-to-end manner. Kehl et al. [63] first proposed extending the single-stage SSD detector [78] in

order to incorporate viewpoint classification. However, while fast it cannot properly recover the 3D translation transformation. YOLO [102] is one of the fastest single-stage detectors in the 2D detection field. Naturally, researchers have found ways to extend it to object pose estimation. YOLO6D [128] extends the classification and bounding box prediction per cell to additionally generate the 2D projection location of keypoints. Hu et al. [52] also extends YOLO but instead of generating the keypoints projections, it generates an 2D offset direction akin to Peng et al. [92]. Two-stage detectors have also been used, such as NOCS [138], that build on top of Faster RCNN [103]. However, the RPN followed by bounding box refinement is an expensive procedure and in this chapter we aim to replace it. Zhou et al. [156] proposed a very fast 2D detection based on heatmap estimation by model the object detection output as a single position, the center point of its bounding box. The center can then be used to recover the classification of the object and its bounding box. We use this method as the backbone of the work on this chapter due its fast throughput and implement it as a two-stage method.

4.2 Methodology

In this section, each stage of the deep learning based method MFNet, that takes an RGB monocular image as input and outputs a 6D egocentric object pose output, is explained in great detail. All stages apart from the Kabsch alignment are differentiable and therefore optimized in an end-to-end manner.

4.2.1 Detection Stage

MFNet follows a two stage framework for detection and pose estimation. This framework allows for an easy extension to multi-instance pose estimation. For detection, the single point strategy, as introduced in [156], is employed. We chose this approach over RPN methods [42, 103] due to its rapid runtime and minimal overhead. Moreover, it is found that detection precision is not critical for accurate pose estimation, and a coarse detection suffices for the proposed method.

An input image $\mathcal{I} \in \mathbb{R}^{W \times H \times 3}$ at full resolution is processed through a feature pyramid network. From the highest level feature map, the model begins by outputting a centroid heatmap $\hat{L} \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times C}$, where R represents the stride used and C denotes the number of classes. Instead of using a Gaussian

kernel as described in [156], it is determined that a single flag at the centroid position is sufficient to generate the ground-truth L. This step is optimized with the multi-class version of focal loss [75].

The discretization error introduced by R is rectified by outputting an offset map $\hat{O} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$. To recover bounding box size, a scale map $\hat{S} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ is output. Both of these maps are optimized with respect to their respective ground truths, following Zhou et al. [156], with L2 loss serving as their loss functions. The full detection loss $\mathcal{L}_{det}$ can be expressed as follows:

$$\mathcal{L}_{det} = \mathcal{L}_L + \mathcal{L}_O + \delta\mathcal{L}_S \tag{4.1}$$

where $\mathcal{L}_L$, $\mathcal{L}_O$ and $\delta\mathcal{L}_S$ refer to the centroid localization, offset and bounding box scale losses, respectively, and $\delta = 0.1$ for all experiments as suggested by Zhou et al. [156].

For each object, a tight image crop $\mathcal{I}_c$ is taken around the object using ROI Align [103]. For each crop, the objective is to find the pose of a target object with respect to the camera, with its loss function being formally defined by:

$$F : \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 3} \rightarrow [SO(3), \mathbb{R}^3] \tag{4.2}$$

$$\mathcal{P}_c = \phi(\mathcal{I}_c, \theta), \tag{4.3}$$

where θ are the learnable parameters of ϕ and $\mathcal{P}_c$ is the full 6D pose.

4.2.2 Perspective Warping

Due to the projective function applied by the camera, the appearance of an object is affected not only by its orientation but also by its position. Since MFNet relies on a cropped region, the problem $\mathcal{I}^C$ becomes a one-to-many function, where the same appearance might correspond to different egocentric orientations as pointed out by 3DRCNN [66]. A common strategy to solve this problem is to apply an allocentric to egocentric transformation to the recovered pose. Several methods utilizing direct pose estimation have implemented this step with success [137, 25, 84, 67].

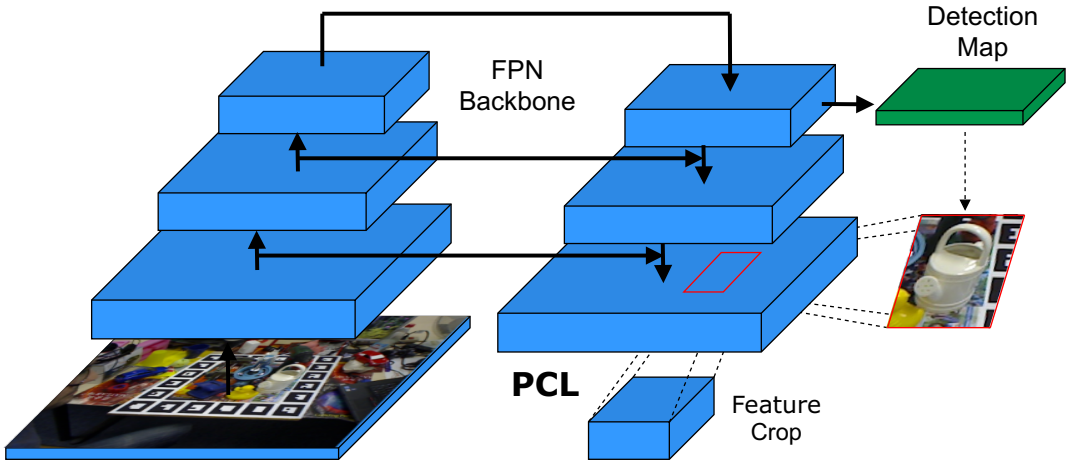


Figure 4.2: **Detection Pipeline.** Diagram of the first stage of **MFNet**, where the object is found and cropped using PCL [151].

It is crucial for the subsequent step to construct a common pose search space for every crop, ensuring that $\mathcal{I}^C$ is independent of the crop position in the image. As a result, the output of $\mathcal{I}^C$ relies solely on the object appearance in the crop, which simplifies the complexity of the task. To achieve this, a Perspective Cropping Layer [151] is applied during the cropping process. In essence, this operation defines a new virtual camera in which the optical center of the camera aligns with the center of the crop. This guarantees that $\mathcal{I}^C$ will be computed in allocentric space. By utilizing the inverse of the PCL, the space can be converted back to egocentric space. For further details, readers should referr to PCL [151]. In practice, this operation closely resembles those described in previous 3D detection estimation works [66, 84].

4.2.3 Single-Frustum Generation

A novel intermediate pose representation, the Multi-Frustum grid is introduced in this section. In contrast to prior methods, a 3D-3D correspondence paradigm is adopted to prevent the degeneration of optimization due to the projective function. The approach is inspired by the default 2D-3D object representation NOCS, along with their corresponding pixel coordinates, which are widely used in numerous 6D pose estimation methods [138, 90, 153, 74]. This representation is extended into 3D world coordinates. Instead of employing image pixels as the 2D correspondence grid, as done in previous methods, the correspondence grid is constructed directly in 3D world coordinates.

Since an image is generated by projecting information from the viewing frustum into an image plane,

we perform the inverse transformation and propose using the frustum, generated through the projective function, as our NOCS correspondence output space. The 3D detection field has had great success using frustum projections for operating on point clouds [97, 142]. A regular rectangular grid could have been chosen, such as in [106]. However, in practice, this approach did not yield good results.

A viewing frustum coordinate grid $\mathcal{V}_c \in \mathbb{R}^{V \times U \times D \times 3}$ can be constructed given a virtual camera K_c with f_c focal length and $c = 0.5$ center point:

$$\mathcal{V}_c(u, v, z) = K_c^{-1}[u, v, Z + \gamma z]^T \tag{4.4}$$

where u and v define pixel positions in $\mathcal{I}_c$ and γ is the size of the distance sampling step.

Ideally, one would choose Z such that the center of the target object would coincide with the center of $\mathcal{V}$. However, this is not feasible as this would require knowledge of the translation of the target object at this stage.

SSD-6D [63] proposed an ingenious way to chose Z. Given their estimated orientation, Kehl et al. exploit a simple linear relationship between pre-computed masks, the object diameter d their estimations to recover distance. [66] uses a very similar approach. At inference time, methods that require orientation recovery cannot be used, as we will employ $\mathcal{V}$ to recover both orientation and translation. Instead of assuming a known orientation during inference, we construct $\mathcal{V}$ to be robust to object orientation and translation. In other words, irrespective of $\mathcal{P}$, the viewing frustum will always contain the object.

To construct the 3D correspondence grid, two assumptions are made. First, the objects possess known and fixed geometry, enabling the exploitation of the object shape and size. Second, $\mathcal{I}_c$ is acquired as a tight crop. A naïve approach might involve defining the distance to the camera using the object diameter, such that $Z = f_v \cdot d$ would represent the center of the frustum. However, this method is only applicable if the object is spherical, where the bounding box would maintain a consistent size regardless of the actual orientation. Since most objects are irregular, accounting for this is essential. Robustness to crop tightness is also considered, given that detectors are not infallible, and this aspect will be discussed in the subsequent section.

Given the two prior assumptions, every target is presumed to be an irregular object, and a range of

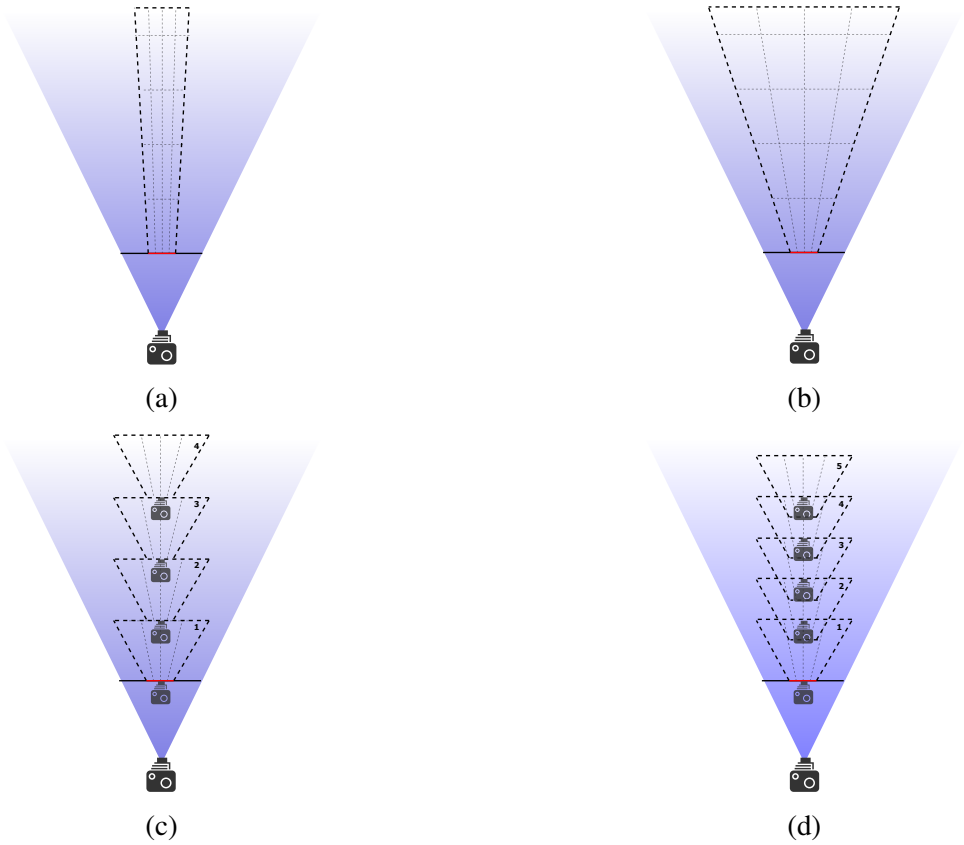


Figure 4.3: **Choosing frustum focal lengths.** This figure illustrates the need to carefully consider the focal lengths MFNet should use to generate. We illustrate in 2D for ease of understanding. a shows that high focal length does not cover the full search space, while b creates a sparse grid. Instead, we chose c and d, a structure where multiple high resolution short frustums are generated at different positions, therefore maintaining a large search scope with high resolution.

distances at which the object is situated can be calculated. For this purpose, the object minimum and maximum diameter, d_{min} and d_{max}, are used. By employing the inverse projection function and assuming a tightly cropped image, the minimum and maximum depth for this distance can be recovered as follows:

$$Z_{min} = f_v \cdot d_{min} \tag{4.5}$$

$$Z_{max} = f_v \cdot d_{max}, \tag{4.6}$$

Finally, to generate the frustum coordinate grid, we can apply Eq. 4.4 with $Z \in [Z_{min}, Z_{max}]$, with D resolution.

4.2.4 Multi-Frustum Generation

In theory, a single frustum should suffice to capture the space surrounding the target object; however, in practice, this is not always the case. The previously described method encompasses two major constraints. When defining the frustum coordinate grid, the crop camera intrinsics K_c, generated through the PCL, can be utilized. However, using this virtual intrinsic matrix would result in each $\mathcal{V}_c$ being dependent on the crop, which must be avoided. Consequently, intrinsic parameters K_v common to every crop must be selected to render the search space independent of the 2D position and distance to the target. As a result, Eq. 4.4 is updated. While unit dimensions for the crop are maintained ($cx = cy = 0.5$), the choice of focal length presents an interesting dilemma, with a trade-off existing between detection error robustness and $\mathcal{V}_c$ resolution.

Low f_v. As previously mentioned, $\mathcal{V}_c$ must exhibit robustness to detection errors. Naturally, selecting a low value for the focal length would result in complete frustum viewing coverage. This choice would either render $\mathcal{V}_c$ very low-resolution (particularly at greater distances from the camera) or, by increasing its resolution, would make $\mathcal{V}_c$ extremely large.

High f_v. Simultaneously, by increasing the focal length, the viewing frustum would be narrow, but due to detection errors, the target object might not be encompassed within its volume.

These issues are illustrated in Fig. 4.3a and 4.3b. To address this problem, a multi-frustum coordinate system is proposed. N short frustums are concatenated along the optical center. Each frustum shares the same intrinsic parameters K_v but is defined at different spatial positions. Let $\mathcal{V}_c^i$ be defined as one of several frustums that compose this system. Each frustum is uniformly positioned between the known distance ranges such that:

$$\mathcal{V}_c^i(\tfrac{H}{2}, \tfrac{W}{2}, \tfrac{D}{2}) = K_v^{-1}[cx, cy, Z_{min} + \tfrac{Z_{max} - Z_{min}}{N-1}] \tag{4.7}$$

where $\mathcal{V}_c^i(\frac{H}{2}, \frac{W}{2}, \frac{D}{2})$ is the center of frustum i and N is the total number of frustums. The multi-frustum system now solves both of the prior issues.

With this approach, a small focal length can be defined without compromising faraway positions while covering significant lateral space to ensure robustness against detection errors. However, this

strategy introduces sparsity at the base of each frustum, which may impact performance instead of only affecting further distances. To counteract this, Eq. 4.7 is modified, enabling the frustums to overlap and creating regions with redundant coordinate positions. This results in both dense and sparse regions of two different frustums. A visualization of each advantage offered by this system can be found in Fig. 4.3c, while the extension to overlap is illustrated in Fig. 4.3d.

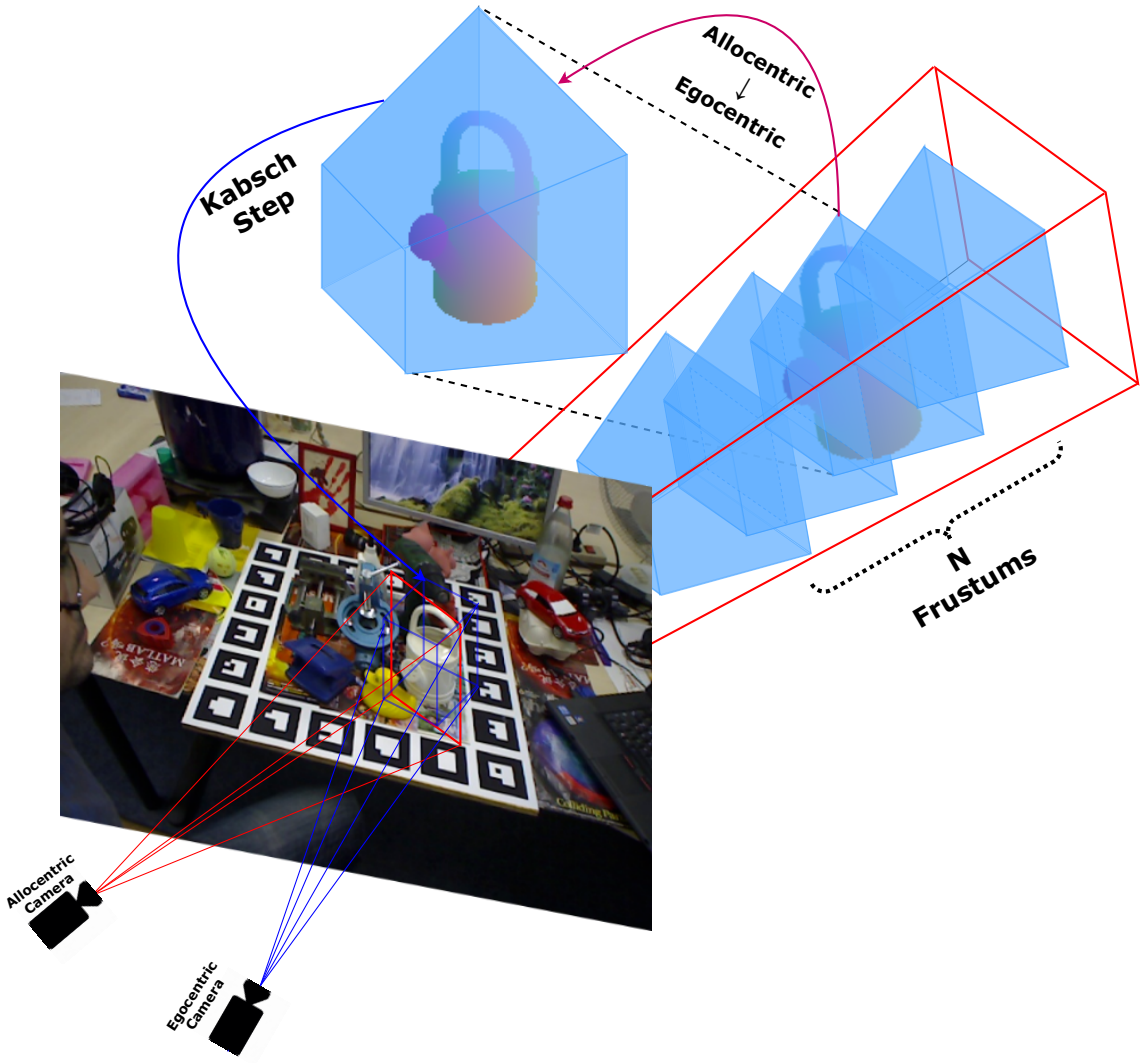


Figure 4.4: **Multi-Frustum inference illustration.** Our method starts by generating, from an allocentric virtual camera, a 3D NOCS representation on a Multi-Frustum grid (in red). We then project the Multi-Frustum grid back to egocentric space where the Kabsch algorithm can be applied to retrieve full 6D pose (in blue).

4.2.5 3D-3D NOCS correspondence

In parallel, MFNet generates the NOCS grid $\hat{\mathcal{G}}$. For every cell in $\hat{\mathcal{G}}$, MFNet outputs the 3D position of the respective cell in object coordinates. $\hat{\mathcal{G}}$ is the output of the Frustum Module, which is fed $\mathcal{I}_c$, the crop taken of the target object. The objective function to learn $\hat{\mathcal{G}}$ can be described as:

$$\mathcal{L}_{NOCS} = \|\mathcal{V} - (R'\hat{\mathcal{G}} + t')\|_2, \tag{4.8}$$

with R' and t' being the groundtruth rotation and translation in allocentric space. By optimizing in 3D space, the projective inductive bias present in previous NOCS based methods is removed. Additionally, the Frustum Module also outputs a confidence occupancy mask, optimized by $\mathcal{L}_m$, in the form of binary focal loss [75].

Objects with symmetry are common and introduce pose ambiguity. If not properly addressed, will yield poor training results. To this end, the previous loss $\mathcal{L}_{NOCS}$ is augmented as suggested in prior methods Wang et al. [138] in order to support symmetries. Given a set of n symmetries $\mathcal{S} = \{\mathcal{S}^0, ..., \mathcal{S}^{n-1}\}$, the objective function for $\mathcal{F}$ becomes:

$$\mathcal{L}_{NOCS} = min_{i=0,...,n-1}\|\mathcal{V} - (\mathcal{S}^i R'\hat{\mathcal{G}} + t')\|_2 \tag{4.9}$$

At inference time, $\hat{\mathcal{G}}'$ is projected to egocentric space by inverting the operation described in Section 4.2.2. The Kabsch algorithm is then used to align the projected $\hat{\mathcal{G}}$ and $\mathcal{V}$, which returns the estimated 6D pose $\hat{\mathcal{P}} = [\hat{R}, \hat{t}]$. Translation $\hat{t}$ is found by taking the distance between the centroids of the grids, while rotation is recovered with SVD:

$$\hat{R} = V diag(1, 1, \ldots, sign(det(\mathcal{U}\mathcal{V}^T))\mathcal{U}^T, \tag{4.10}$$

$$\hat{t} = \frac{\sum \hat{\mathcal{G}}}{|\hat{\mathcal{G}}|} - \frac{\sum \mathcal{V}}{|\mathcal{V}|}, \tag{4.11}$$

where $\mathcal{U}, \mathcal{E}, V^T = SVD\left(\hat{\mathcal{G}}\mathcal{V}^T\right)$. SVD is extremely fast and GPU implementations are readily available. A diagram of the Multi-Frustum pipeline at inference time can be seen in Fig. 4.4.

4.3 Experimental Evaluation

We conducted experiments on two benchmark dataset Linemod (LM) [45] and YCB-V [13] where we present strong evidence of the potential of MFNet. LINEMOD serves as the primary benchmark dataset for object pose evaluation due to its diverse range of object shapes, against which all methods are typically compared. However, due to its relatively small size and limited scene complexity we perform an additional evaluation against the YCB-V dataset, which features significantly more challenging scenes with higher levels of occlusion and image noise. Moreover, YCB-V includes objects with uniform textures and continuous symmetries, both of which are critical challenges for object pose estimation methods. We demonstrate that our method can achieve an impressive translation accuracy, as a result of our new 3D-3D correspondences.

4.3.1 Evaluation Setup

Datasets Setup. For Linemod [45], we create a synthetic dataset with 10000 images for each object, with each image having between 3 and 10 instances of the objects. Pose is sampled from dataset statistics provided by BOP [49]. Each synthetic object is generated with a set of random illumination and material properties. Our online augmentations include standard color transformations as well blur and noise. We also perform spatial transformations including scaling, rotation and homography transformations, always accounting for the respective pose transformation. MFNet also uses augmented detection which is achieved with perturbations to the bounding boxes used prior to PCL. For YCB-V, we use training data made available through BOP: the original real training images, simple synthetic images and PBR [23] synthetic data. We apply the same online augmentations for this dataset.

Implementation details. The detection stage employs a feature pyramid structure with a ResNet18 [43] backbone, which is found to have sufficient capacity for the tasks at hand. The detection map is produced and unraveled during inference as described in [156]. Each batch normalization layer is replaced with GroupNorm [147]. After detection and the subsequent forward pass through the PCL layer, a feature map is obtained, which is then fed into the Multi-Frustum module. This module consists of six convolutional layers, with two deformable upscaling [21] operations and GroupNorm following every layer except the last.

The model is trained end-to-end using Radam [76] with Lookahead [155] and a batch size of eight images. A learning rate of 5×10^{-4} follows a flat-cosine scheduler with decay commencing after 65% of the training iterations. The overall objective function is:

$$\mathcal{L} = \mathcal{L}_{NOCS} + \mathcal{L}_m + \mathcal{L}_{det} \tag{4.12}$$

During experiments, it was found that γ equaling the error threshold for each object produces the best results and with the grid size defined with $U = V = D = 32$. The detection feature map is produced at $R = 32$ stride. The output of the Multi-Frustum module, $\mathcal{G}$, is defined in normalized object coordinates as described in [138]. Prior to $\mathcal{L}_{NOCS}$ these are revert it back to canonical coordinates. This method was implemented using PyTorch [91]. All the experiments for MFNet were done on a computer equipped with a GTX1080Ti.

Evaluation metrics. The *ADD-(S)* metric is commonly used to measure 3D errors and is the standard metric for evaluating methods on Linemod. It is computed by taking the model vertices $\mathbb{V} = \{\mathbb{V}_0, \mathbb{V}_1, \ldots, \mathbb{V}_{n-1}\}$, with n being the number of vertices of the object 3D CAD model, and measuring the euclidean distance between the points of the object at the estimated pose $\hat{P}_{6D}$ and $\mathbb{V}$ and at ground truth pose P_{6D}.

α defines the percentage of object diameter to use as the threshold distance to consider a pose correct. Later on the manuscript, will we share an in-depth study on translation distances and use it as an evaluation metric. On YCB-V, traditionally performance is measure using *AUC-AD*, the area under the curve of *ADD-(S)*, with thresholds up to 10cm.

4.3.2 Evaluation Results.

Detection on the Linemod dataset. Detection of an object must be accurate as it is a key part of the object pose estimation task. However, the former is a much simpler task than the latter. With a small network, MFNet has 99.69 % detection accuracy (at 75% IOU) on the full Linemod test set. Moreover, when using ground truth bounding boxes, our method produces an *0.1-ADD-(S)* accuracy of 95.42%, a negligible difference to when using MFNet detection outputs. In part, this is due to the

robustness MFNet has to bounding box tightness and unknown orientation, described in section 4.2.3. We show that, with MFNet, there is no need for complex and slow off-the-shelf detectors, since their addition to the pipeline minimally improves results.

Comparison with previous state-of-the-art on the Linemod dataset We can observe in Tab. 4.1,

Table 4.1: **Comparing ADD-(S) results on Linemod.** Our method outperforms previous RGB only methods. Both CDPN [74] and GDR-Net [137] rely on an external object detector while MFNet detection and estimates pose with the same model.

Method Type	Keypoint Regression		2D-3D Dense Correspondence				**3D-3D Correspondence**	
Method Name	YOLO6D [128]	PVNet [92]	Pix2Pose [90]	DPOD [153]	CDPN [74]	GDR-Net [137]	**MFNet**	Relative Perf.
Ape	21.62	43.62	58.10	53.28	64.38	76.29	**87.46**	+14.6%
Benchvise	81.80	**99.90**	91.00	95.34	97.77	97.96	98.35	-1.6%
Camera	36.57	86.86	60.90	90.36	91.67	**95.29**	93.84	-1.5%
Can	68.80	95.47	84.40	94.10	95.87	98.03	**98.41**	+0.4%
Cat	41.82	79.34	65.00	60.38	83.83	93.21	**93.72**	+0.5%
Driller	63.51	96.43	76.30	97.72	96.23	**97.72**	96.89	+0.9%
Duck	27.23	52.58	43.80	66.01	66.76	80.82	**85.73**	+6.1%
Eggbox*	69.58	99.15	96.80	97.72	99.72	99.53	**99.92**	+0.4%
Glue*	80.02	95.66	79.80	93.83	**99.61**	98.94	99.26	-0.4%
Holepuncher	42.63	81.92	74.80	65.83	85.82	91.15	**92.64**	+1.6%
Iron	74.97	98.88	83.40	99.80	97.85	**98.06**	97.92	-0.1%
Lamp	71.11	99.33	82.00	88.11	97.89	**99.14**	98.78	-0.4%
Phone	47.74	92.41	45.00	74.24	90.75	92.35	**94.45**	+2.3%
Average	55.95	86.27	72.40	82.98	89.86	93.69	**95.19**	+1.6%

Table 4.2: **ADD-(S) lower threshold comparison.** Our translation precision is far superior than GDR-Net[137] however our accuracy for low threshold *AD* is lower.

Threshold	*0.02-ADD-(S)*			*0.05-ADD-(S)*		
Method	CDPN [74]	GDR-Net [137]	**MFNet**	CDPN [74]	GDR-Net [137]	**MFNet**
Ape	10.95	14.86	**16.42**	38.57	46.48	**59.71**
BVise	36.66	**43.06**	42.59	83.71	87.39	**92.34**
Camera	25.98	**28.82**	23.06	70.20	71.86	**73.52**
Can	27.66	**35.43**	32.11	73.13	83.76	**87.63**
Cat	23.05	**25.85**	21.12	58.48	68.86	**70.14**
Driller	29.53	**42.91**	20.54	73.44	**84.14**	79.80
Duck	11.92	**19.06**	9.41	43.19	50.89	**56.06**
Eggbox	56.53	**61.97**	54.59	94.84	95.31	**97.13**
Glue	43.15	**45.95**	44.51	86.97	90.83	**92.46**
HPunch	**28.07**	27.40	50.52	64.03	64.03	**68.63**
Iron	34.01	**43.21**	42.36	81.10	86.31	**88.37**
Lamp	41.84	46.45	**48.74**	83.11	91.55	**92.99**
Phone	22.47	**26.63**	19.31	65.25	69.88	**72.16**
Average	30.14	**35.51**	30.19	70.46	76.25	**79.30**
# of Wins	1	**10**	2	0	1	**12**

that we are close to fully saturating this dataset using our most currently used metric *0.1-AD*. Nonetheless, we outperform prior state-of-the-art, in accuracy and # of objects it outperforms. We analyzed in more detail MFNet as well as three landmark openly sourced 6D pose estimation methods, PVNet [92], DeepIM [73] and GDR-Net [137] and show that an accuracy to precision trade-off is present yet elusive if one does not use finer error thresholds. These 3 methods are specifically chosen for being the state-of-the-art contemporaries of MFNet, each with different intermediate representations. PVNet [92] uses sparse keypoint regression to find 2D to 3D correspondences, DeepIM [73] a refinement of pose through direct estimation and GDR-Net [137] a dense estimation of the object 3D surface in 2D image space. In contrast to GDR-Net, MFNet estimates the 3D object surface in 3D space, which is why it is referred to as a 3D-3D correspondence method in Table 4.1. As noted on our comparison with previous works, we outperform all three methods for *0.1-AD* and *0.05-AD*. However, observing Table 4.2, we can see that if we reduce the error threshold, MFNet suffers a steeper decline of accuracy. While MFNet still outperforms both methods on *0.05-AD*, both in average accuracy or # of objects it outperforms, we underperform using the finer *0.02-AD*, having poorer results than both compared methods. This observation serves as a reminder to the community that current metrics might not be enough to fully evaluate a 6D pose estimator. For this reason, we would like to suggest the presentation of finer metrics to future authors, as these are not saturated.

Translation accuracy on the Linemod dataset. While reading the previous section, the reader might induce that our method underperforms when it comes to translation. Translation accuracy is crucial for robotic applications, and ADD-(S) is considered the benchmark metric for evaluating a 6D

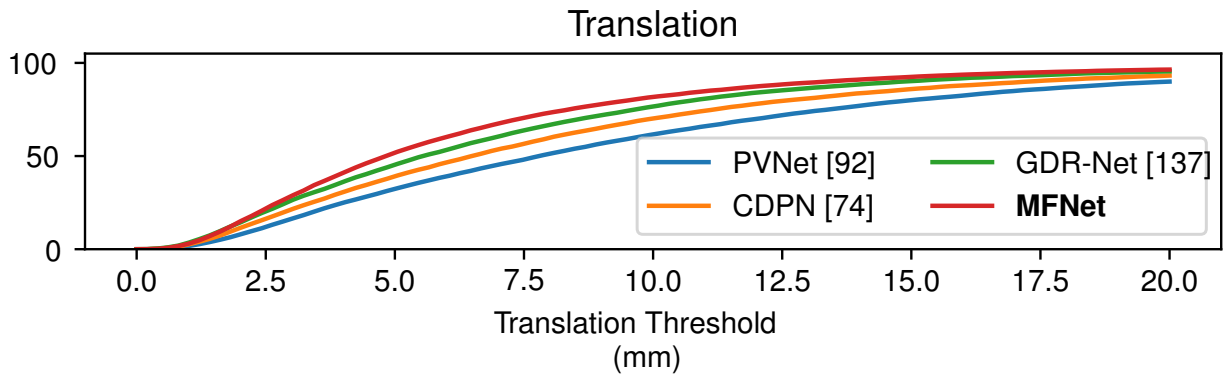


Figure 4.5: Translation accuracy curve on Linemod at multiple thresholds.

Table 4.3: **Translation error evaluation and comparison study.** Evaluate the impact of the design of **MFNet** in translation errors compared to previous state-of-the-art. AUC refers to the area under the curve of translation errors with a threshold up to 2cm.

Threshold	Object	Ape	BVise	Camera	Can	Cat	Driller	Duck	Eggbox	Glue	HPunch	Iron	Lamp	Phone	Average
0.5 cm	PVNet[92]	24.00	47.77	30.00	34.45	32.73	36.17	29.86	24.69	22.30	35.87	26.97	39.73	35.06	32.28
	CDPN[74]	40.38	42.10	44.41	38.39	40.92	30.72	41.97	45.73	30.12	50.14	32.69	38.00	32.58	39.09
	GDR-Net[137]	46.00	47.24	47.35	49.61	49.80	43.81	48.08	**54.65**	33.98	48.05	41.68	43.95	36.07	45.40
	MFNet	**60.19**	**58.81**	**48.38**	**61.20**	**53.94**	**44.10**	**54.15**	50.52	**38.61**	**53.03**	**54.86**	**57.13**	**41.11**	**52.00**
1.0 cm	PVNet[92]	51.05	79.94	59.90	69.49	62.87	69.47	55.31	51.46	42.86	65.56	58.02	73.22	61.96	61.62
	CDPN[74]	69.14	75.27	77.45	72.93	71.46	62.64	72.21	77.93	58.59	77.26	64.76	70.06	62.42	70.16
	GDR-Net[137]	75.14	79.15	79.80	83.56	79.64	74.23	77.46	82.54	63.42	78.50	73.85	80.33	68.46	76.62
	MFNet	**87.14**	**89.37**	**80.77**	**89.63**	**81.85**	**75.84**	**82.70**	**83.96**	**68.93**	**83.27**	**81.94**	**86.06**	**73.29**	**81.90**
2.0 cm	PVNet[92]	83.81	97.67	92.75	96.06	87.72	94.15	84.04	86.20	76.45	90.58	92.34	96.55	91.26	89.97
	CDPN[74]	90.38	96.02	95.88	96.56	92.91	90.39	93.99	96.15	88.90	95.05	92.95	93.47	89.05	93.21
	GDR-Net[137]	95.14	96.70	**96.67**	97.93	96.61	**94.95**	94.84	96.34	**92.28**	96.57	94.79	97.50	90.93	95.48
	MFNet	**98.06**	**98.11**	96.42	**98.58**	**97.71**	94.53	**96.97**	**97.92**	91.15	**97.25**	**95.49**	**98.04**	**93.56**	**96.45**
AUC	PVNet[92]	46.48	67.56	54.02	60.15	54.70	59.60	49.80	47.57	41.38	57.39	53.16	62.10	55.65	54.58
	CDPN[74]	59.89	63.67	65.08	61.96	61.26	54.42	62.76	66.57	52.45	66.80	57.07	59.95	54.63	60.50
	GDR-Net[137]	65.45	66.56	66.98	69.43	67.71	**63.34**	65.70	**70.30**	57.07	67.65	63.34	66.40	58.66	65.28
	MFNet	**73.69**	**73.32**	**67.50**	**74.64**	**69.65**	63.22	**70.10**	69.86	**58.69**	**70.09**	**68.59**	**71.76**	**62.16**	**68.71**

Table 4.4: **YCB-V study.** . We report results of MFNet on YCB-V and compare it with recent non-refinement methods.

Metric	PoseCNN[148]	PVNet[92]	GDR-Net[137]	**MFNet**
AUC-AD	61.3	73.41	**80.19**	78.42
T. Acc. - 2cm	-	-	55.03	**57.04**

object pose method suitability for robotics. However, the proposed method significantly outperforms previous approaches in translation accuracy, owing its accuracy to the elimination of the projective function optimization degeneration. MFNet surpasses prior techniques at every threshold and is capable of achieving an impressive 52% accuracy using a threshold of $5mm$. Qualitative examples are provided in Fig. 4.6, demonstrating the precise estimations.

The performance of MFNet in terms of translation accuracy is meticulously analyzed and compared to contemporary methods. A comprehensive study can be found in Tab. 4.3. In Fig. 4.5, the translation curves for both MFNet and earlier methods are displayed. It is evident that MFNet outperforms all previous approaches across all levels of translation precision.

Result on YCB-V dataset In Tab. 4.4, we share the results of our method on the YCB-V dataset. This dataset contains several instances of objects highly occluded, which is our model is not suited for. Nonetheless, we outperform PVNet [92] and produce competitive results when compared to GDR-Net [137] on *AUC-AD*. The difference stems from the fact that MFNet has a lower recall than GDR-Net [137] due to hard occlusions. Nonetheless, our translation estimation is still extremely accurate:

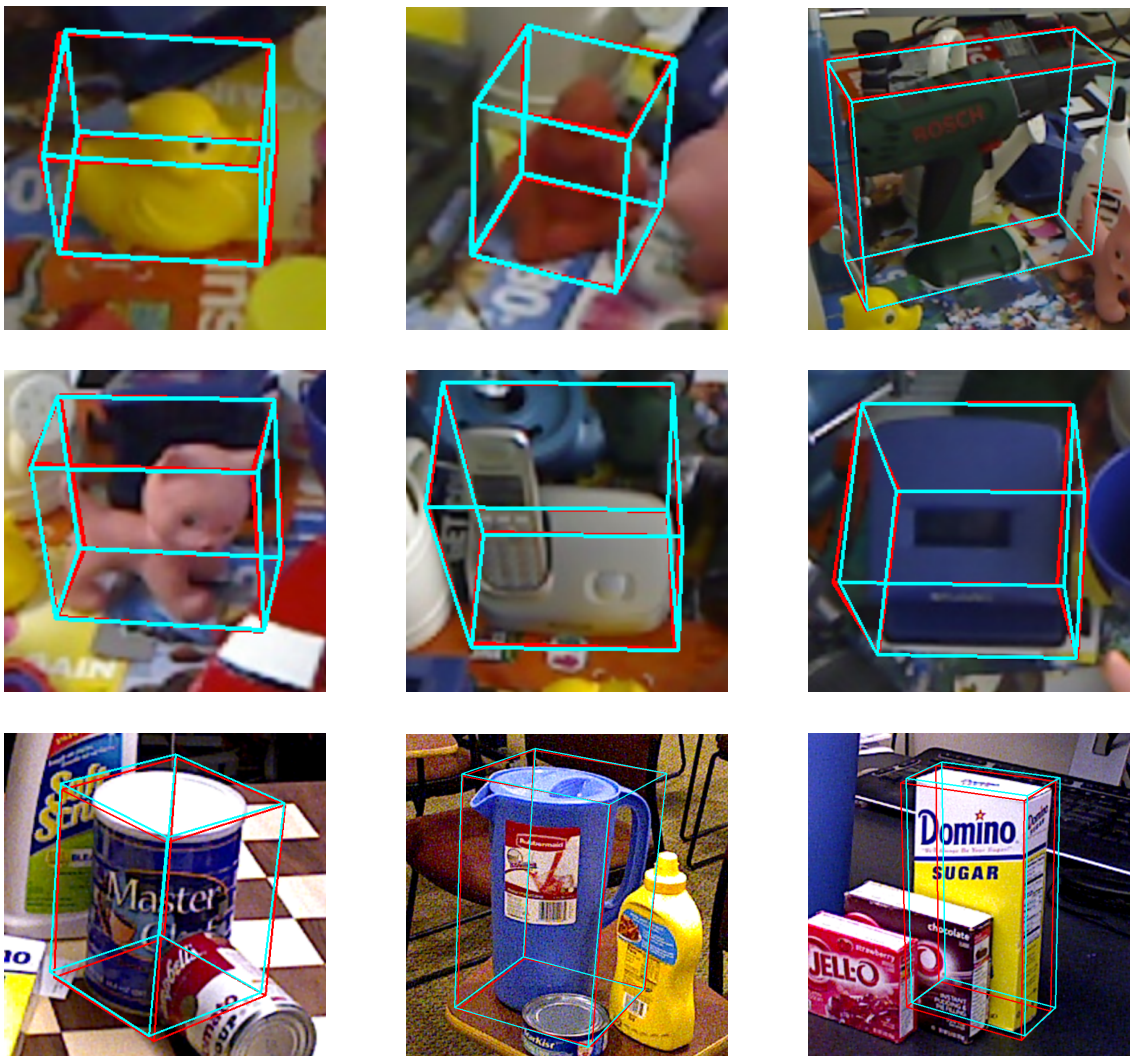


Figure 4.6: **Qualitative Results.** We show some qualitative results of MFNet on the Linemod dataset and YCB-V dataset. Predicted poses are shown by their light blue bounding boxes while groundtruth are color-coded with red.

while GDR-Net [137] achieves a 27% and 5% translation accuracy on 2cm for object *"Bowl"* and *"WoodBlock"*, MFNet achieves 60% and 37%, respectively.

Runtime discussion Recent methods are capable of running at real-time, which one could roughly define as 25 FPS. In other words, the processing of one image by the method should not take more than 40ms. However, higher frequency processing enables the use of more frames in the same amount of time, resulting in increasing redundancy that can help decrease errors. It can also enable applications needing lower latency or with worse hardware. At the same time, the method must still be accurate. MFNet can satisfy both these requirements: for a single object, detection and pose estimation takes 13ms, including copping, building frustum correspondence grid and computing the Kabsch algorithm,

surpassing all previous methods by a large margin, without compromising accuracy, as can be seen in Tab. 4.5.

Table 4.5: **Runtime comparison.** Comparing with monocular RGB only methods (including detection) measured on a GTX 1080Ti, or equivalent, except GDR* [137] which is measured on a GTX 2080Ti.

Method	Runtime Speed (ms) ↓	Frames-per-second (FPS) ↑
PVNet [92]	40	25
CDPN [74]	30	33
DPOD [153]	25	40
DPOD+ [153]	30	33
GDR-Net* [137]	22	46
MFNet	**13**	**77**

4.4 Conclusion

This chapter introduces MFNet, specifically aimed at accelerating 6D object pose estimation by uniting the detector and pose estimator in a single model and removing the need for PnP step. The method addresses the challenges of accurately estimating the translation by extracting the intermediate representation directly in 3D space thereby mitigated the 2D inductive error bias typically present in previous 2D-3D correspondence methods. Experiments show that MFNet achieves state-of-the-art, with a considerable decrease in translation errors and with a $2\times$ speedup over previous work.

Chapter 5

Scaling to multi-objects

The methods proposed on the first chapters relied on heavy neural networks to achieve high object pose accuracy. However, for each object, an individual network was trained from scratch, which is also the case for previous state-of-the-art methods that were compared. State-of-the-art methods typically train a neural network per object, ending up with as many such networks as the number of objects. Its scalability is a major issue, and has been overlooked in favor of accuracy. Although for other vision problems such as image classification or object detection, a deep network is able to learn and perform well over numerous object categories, the same thing has not been observed in the literature of 6D object pose estimation, but rather the opposite [61]. Unlike in image classification or object detection, where 2D features alone often suffice, 6D pose estimation requires learning detailed spatial and rotational characteristics unique to each object. A model with limited parameterisation will struggle to capture object-specific details, which makes generalization to multiple objects significantly harder.

The most straightforward approach to support multi-objects without significantly increasing the capacity of a model is through careful injection of object specific parameterisation. In multi-task scenarios, the simplest form of additional parameterisation is the addition of task specific parameters at output level [42, 105, 52]. In the transfer learning domain, efforts have been made in the pursuit of representation learning across multiple tasks or domains [100, 9]. A universal parametric network was proposed by [101], where a subset of parameters in intermediate layers is task specific and easily interchangeable. However, their approach introduces a sizeable overhead, doubling the original

capacity of the network when tackling 10 different tasks. When task selection is available prior to inference, one can design their system to be preemptively conditioned thereby reducing the overhead. Conditional and Adaptive Instance Normalization [32, 54] were introduced in style transfer as conditioning strategies for generating images with similar styles by learning style specific normalization parameters, which can be arbitrary changed with a parameterisation lookup table. These parameters add considerably less overhead than prior attempts, both in terms of model capacity and necessary computation, and have been successfully used outside the style transfer domain [152, 119]. Other traditional conditioning strategies include one-hot embedding at input level [69].

In this chapter, we examine the issue, and confirm that 6D pose accuracy rapidly drops when increasing the number of objects for a single DNN. We then investigate methods to relieve the problem and present scalable multi-object architectural modifications.

5.1 Related Work

Adaptive Instance Normalization (AdaIN) has been used to a great extent in the style transfer domain [54, 104] and image generation [141, 60, 93]. It has been demonstrated that instance normalization effectively performs style normalization by manipulating feature statistics, which are identified as carriers of style information within an image. This revelation has been corroborated by the success of these methods and their wide applicability.

Knowledge Distillation (KD) has emerged as a prominent method for model compression, particularly when leveraging pre-existing, fully trained models [47]. This approach involves a smaller student network learning from the distilled knowledge provided by a larger teacher network. The knowledge transfer process typically relies on the softened label outputs or feature distribution of the teacher network. There have been successful implementations of KD methods that employ multiple teachers, each with their own specialization in a specific domain [47, 79]. While KD methods were initially developed for classification tasks [47, 6, 107, 96], their adaptation to regression problems presents a more challenging endeavor [111, 19]. As a result, we research how to effectively apply and extend KD techniques to the 6D pose estimation regression problem.

In addition to the methods previously discussed, there are other compression techniques available, such as quantization and pruning. Quantization is a process that reduces numerical precision by diminishing the bit representation of floating point operations. This results in high throughput and low memory usage, albeit at the expense of accuracy [37, 131, 55, 96]. On the other hand, pruning aims to eliminate redundant and unimportant network connections, thereby reducing both unnecessary computation and memory usage [82, 39, 40]. However, it is worth noting that pruning may introduce increased pipeline complexity and necessitate specialized equipment [144]. These approaches offer alternative strategies for managing the trade-offs between performance, memory usage, and accuracy in compression techniques. However, these experiments are out of the scope of this work since they can be applied without paying special attention to the task at hand.

5.2 Methodology

Given a deep neural network model, we consider adding more parameters to efficiently embed multiple objects to the model. For parameterisation, we propose primarily the use of Conditional Instance Normalization (CIN) [32] as a conditioning strategy. Conditional Instance Normalization (CIN) is a constrained alternative to AdaIN which conditions normalization parameters on specific identity rather than input style statistics. This feature leads to well-defined and consistent features for each object, allowing multiple objects to be embedded efficiently within the model, which can be efficiently applied with almost negligible overhead. The conditional information contains object identity, which is obtained by a state-of-the-art 2D object detector. We also apply Knowledge Distillation [47] and Hint Training [107] to help compress multiple single-object pose networks to a single network.

Conditioning a Multi-Object Pose Estimator. Conventional pose estimators either have a single network per object, or a network that has multiple outputs for multiple objects as shown. The goal is to condition the network to increase its capacity for multiple object pose estimation. Therefore, a new object conditioning strategy is built on top of CIN [32]. A feature map F is conditionally normalised by:

$$cin(F, c) = \gamma_c \left(\frac{F - \mu}{\sigma} \right) + \beta_c, \tag{5.1}$$

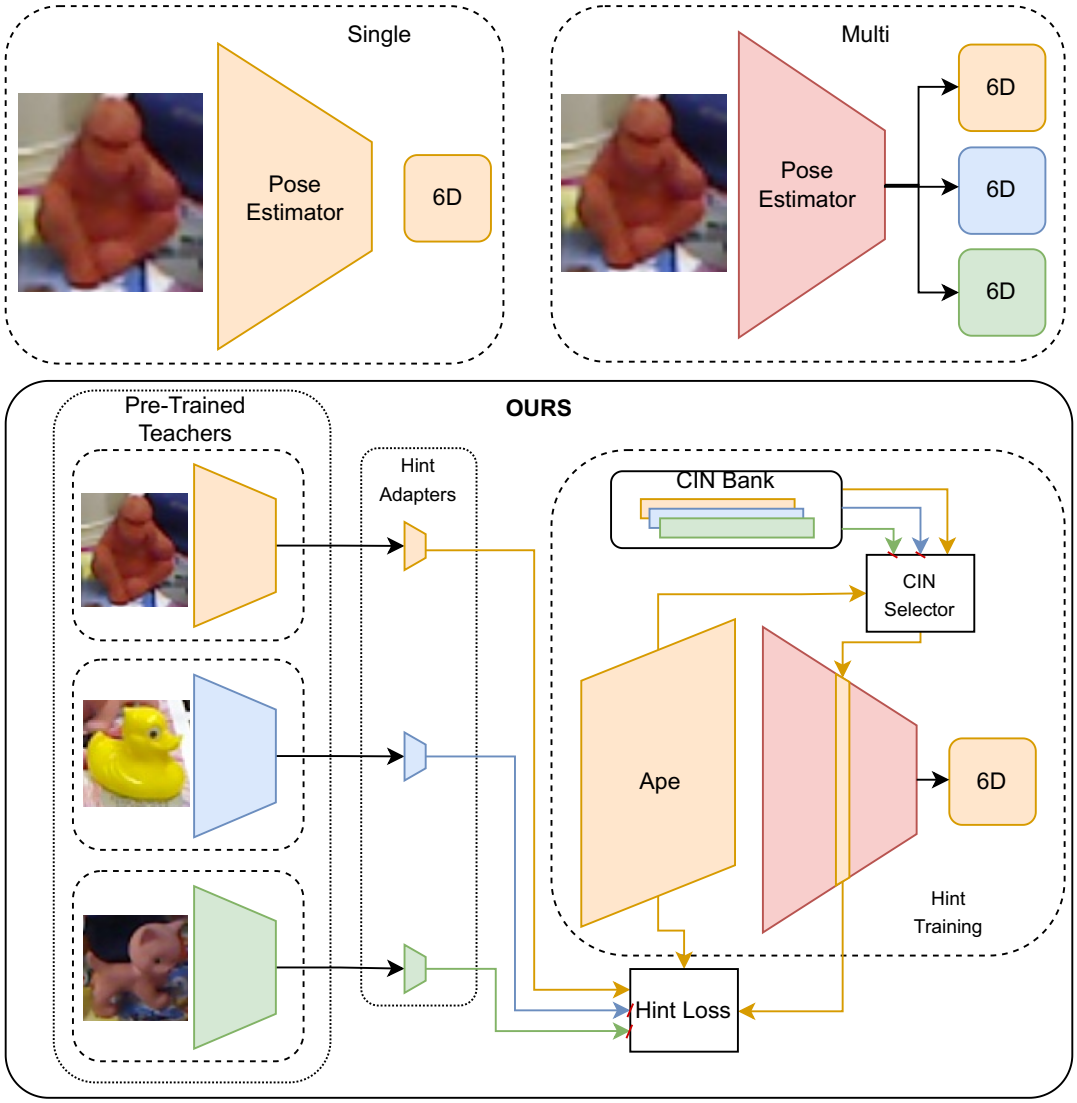


Figure 5.1: **Diagram of CIN conditioning.** This diagram shows our compression pipeline from multiple networks to a single network by Conditional Instance Normalization (CIN) [32] and Knowledge Distillation (KD) [47].

where μ and σ are the channel-wise mean and standard deviation of the feature map, and γ_c and β_c are the scalar and bias parameters of object c. Although originally formulated for style transfer purposes [32], we found that CIN can be successfully applied on multi-object 6D pose estimation tasks as shown in the experimental section. When CIN is applied, we observe that the multi-output models performed on par with the single-output models. Therefore, we further reduce the existing overhead by merging the keypoints of every object [9], which replaces the multi-output formulation with a single output.

Guided optimization with Knowledge Distillation and Hint Training. We examine Knowledge Distillation (KD) for compressing multiple single-object networks to a single multi-object network.

While KD is typically applied to classification tasks [47], applying it to regression problems is less straightforward.

Efforts in optimizing KD in this context have shown that a loss formulation needs to be carefully constructed [111, 19]. In our scenario, the upper-bounded KD loss [19] is defined as:

$$L_{KD}(h_s, h_t, h_{gt}) = \begin{cases} ||h_s - h_{gt}||^2, \text{ if } ||h_s - h_{gt}||^2 < ||h_t - h_{gt}||^2 \\ ||h_s - h_t||^2 + ||h_s - h_{gt}||^2, \text{ otherwise} \end{cases}, \tag{5.2}$$

where h_s and h_t are the student and teacher output and h_{gt} is the ground truth 3D bounding box label. The loss function ensures that the teacher output guides the student only when the error from the teacher is lower than that of the student. When the student produces a more accurate keypoint prediction, only the ground truth serves as supervision.

Another useful component of the distillation process is Hint Training (HT) [107]. A hint is a intermediate feature map from the teacher which serves as an objective for the student during training. In our approach, HT is used as an additional regularization mechanism which allows our multi-object student to better capture the teacher knowledge. For this purpose, we pick the output of the last convolutional layer of as our hint. In [107], the hint loss is defined by the Euclidean distance between feature maps.

An adaptation layer is added if both feature maps are not of the same size. In line with [19], we found the adaptation layer to be crucial to effectively train the network using hints even if the feature maps have identical sizes. Our adapted hint training loss is:

$$L_{HT}(F_t, F_s) = \|\Psi_t(F_t; \theta_t) - F_s\|^2, \tag{5.3}$$

where Ψ_t is a 1x1 convolutional layer with learnable parameters θ_t. F_t and F_s are the feature maps of teacher and student respectively. The transformation layer Ψ_t ensures the student features are guided by those of the teacher, ensuring effective knowledge transfer. At inference, this adaptation layer is discarded, as it is only used during for knowledge distillation process and once the process is complete, the adaptation and teachers become redundant.

5.3 Experimental Evaluation

In order to validate our method, we present experiments using two separate methods. The first is a vanilla BB8 by Rad and Lepetit [98], which has been proven to be a great baseline pose estimator. In addition to validating our results on a vanilla version, we also validate using the self-supervised BB8 version by Sock et al. [118]. We use the work by Sock et al. [118] as baseline since the default fully supervised BB8 version has its metrics somewhat saturated. The applicability of the method described in this chapter is demonstrated by overcoming the prohibitive cost of the per-object training described by Sock et al. [118], thus enabling a practical approach to multi-object scenarios that would otherwise be unfeasible. KD^* refers to the self-supervised knowledge distillation version, where the equations 5.2 is propagated if and only if, the training instance is synthetic.

Datasets Setup.

To evaluate the robustness of our method, two datasets with complementary characteristics were selected. The LINEMOD dataset [45] serves as a standard for evaluating pose estimation methods as mentioned in Chapter 2. In contrast, the T-LESS dataset [48] was chosen for its extensive set of diverse objects, allowing us to test scalability across multiple instances. For the LINEMOD dataset, following the standard practice, 15% of the images are used *without* labels for training and the rest for testing. The selection of training images follows the strategy in [11]. The T-LESS dataset provides real images for training. For synthetic data, we use the synthetic dataset described in Chapter 3, with 15000 instances of each object. To be robust against inaccurate detection, we randomly perturb center position and rescale the bounding box during training. When performing experiments on Sock et al. [118], the real images annotations are not used.

Implementation Details. We use the official BB8 codebase provided by the authors. For training, a batch size of 64 is used for each iteration with the ADAM optimiser[65]. The initial learning rate is $1e^{-5}$ for the first 15 epochs and reduced by a factor of 10 for the next 10 epochs. Loss gradients are not propagated through the teacher models. This method was implemented using PyTorch [91]. All the experiments for MFNet were done on a computer equipped with a GTX1080Ti.

Evaluation Metrics. All reported results are percentage of correctly predicted poses (i.e. object recall rate) by one of the following metrics. We measure ADD-(S) accuracy with a 10% threshold

for Linemod experiments and we measure VSD accuracy as the predictions with $err_{vsd} < 0.3$, with tolerance $\tau = 20mm$ and $\delta = 15mm$, while only considering objects with visibility higher than 10% as proposed by Hodaň et al. [48].

5.3.1 Evaluation Results

Linemod Experiments. We first evaluate our proposed methods for multi-objects with full supervision. Using a smaller baseline model, we perform an ablation study by measuring the performance of the variations of our methods on several numbers of LINEMOD objects. The Upper Bound (UB) is defined as the average accuracy of the single-object models while the lower bound (LB) refers to a multi-object model with the standard multi-output formulation. Fig. 5.2c shows the change in the accuracy of our proposed methods relative to the UB. We also compare CIN to one-hot conditioning [69] and show that CIN also outperforms as one-hot conditioning did not improve the results in our experiments, while yielding similar parameter reduction of 90.7%. With our implementation of method [101] however, the reduction was only 80%. When applying CIN conditioning, the models learned object specific features more easily, resulting in an improvement over LB for all different numbers of objects. We further boost the performance by applying KD and KD+HT, where we distill the knowledge from multiple teachers specialised in each object. We also present several studies on

Table 5.1: **Linemod Ablation study using Sock et al. [118].** Showing the impact of the combined solutions using ADD metric on LINEMOD dataset.

Object \ Method	Multi	Multi+KD*	OH[69]+KD*	CIN+KD*	Multi+KD*+HT	CIN+KD*+HT	Single
Ape	5.91	12.53	15.22	14.48	21.86	16.62	20.98
Benchvise	12.86	25.86	23.51	43.11	30.64	46.20	65.97
Cam	5.89	17.61	12.67	10.75	20.39	18.54	39.20
Can	12.11	31.67	33.09	31.95	30.33	40.02	54.50
Cat	20.70	30.70	28.19	22.50	27.31	43.22	64.12
Driller	21.69	27.49	29.65	37.66	34.08	49.08	68.05
Duck	2.56	13.87	13.04	16.59	18.95	12.60	26.75
Eggbox	34.33	48.77	34.33	43.33	52.23	42.22	69.15
Glue	27.85	35.00	39.96	37.05	37.47	53.90	71.59
Holepuncher	8.44	11.55	7.15	16.64	16.18	10.54	19.09
Iron	19.12	41.10	41.87	38.40	45.30	48.00	71.90
Lamp	22.76	34.81	31.47	35.56	35.19	43.98	67.68
Phone	10.04	18.06	19.82	21.11	22.30	29.49	47.83
Mean	15.71	26.85	25.38	28.39	30.17	34.95	52.83

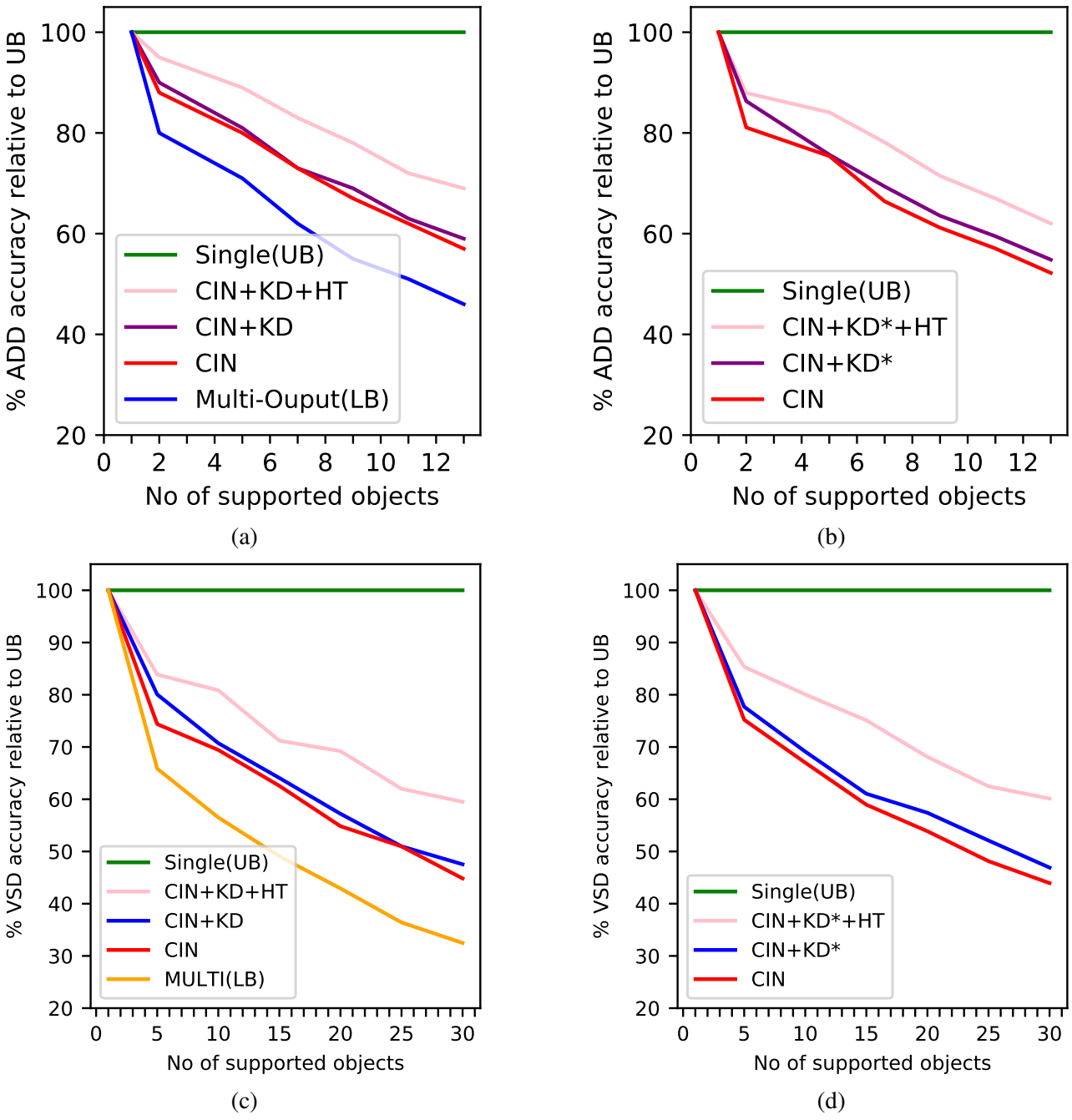


Figure 5.2: **BB8 Ablation Study.** (a,c) Relative performance comparing the single model (upper bound, UB) to a naive extension to multi object (lower bound, LB) and our multi-object proposed methods (b,d) Relative performance of BB8 combined with self-supervision, trained with self-supervised single-object as teachers and synthetic data. (a,b) refer to the 13 LINEMOD dataset, while (c,d) refer to T-LESS. Best viewed in colour.

BB8 using the work done by Sock et al. [118], which can be seen in Table 5.1 and Fig. 5.2. Due to domain gap issues, training a multi-object pose estimation with only synthetic data results in extremely poor results. We show through those experiments each contribution of our method.

T-LESS Experiments. We also validate our results on the more challenging T-LESS dataset [48], which has 30 different objects. In Table 5.2, we compare multiple compression rates. In comparison to AAE [124], our model achieves better accuracies at 90% compression rate (10 objects per model) and below.

Table 5.2: **Ablation study:** Different model capacities on the T-LESS dataset, reported with the VSD metric. The numbers in the bracket indicate the number of teacher models compressed into each student model.

Compression Rate	**CIN KD***	**CIN KD*+HT**
0% (1)	48.75	
97% (30)	20.84	22.95
93% (15)	25.94	27.08
90% (10)	34.82	37.02
80% (5)	36.36	39.12
AAE[124]	36.79	

5.4 Conclusion

This chapter introduced a multi-object model compression framework using Conditional Instance Normalization and multi-teacher distillation to develop an efficient pose estimator with minimal capacity increase. This framework was used to careful extend stable and widely adopted state-of-the-art approaches, thereby demonstrating that a 90% compression rate could be achieved while outperforming naive multi-object implementations. Experiments on large object-count datasets supported its applicability to both supervised and self-supervised learning.

Chapter 6

CRT-6D: Cascaded Refinement Transformers

Recent methods employing Convolutional Neural Networks (CNN) have outperformed classical approaches and now constitute the foundation of many state-of-the-art 6D object pose estimators [124, 17, 92, 118, 98, 137, 25, 67, 120, 90, 12]. The computational pipeline of these methods is broadly divided into three stages: 1.) Object detection in the image, typically performed using an off-the-shelf object detector [103, 102] fine-tuned on the target dataset; 2.) Feature extraction from the cropped image surrounding the 2D area containing the object, using well-established CNN architectures [43, 126]; 3.) Transformation of these features into an intermediate representation [92, 90, 137, 98, 51], which is then employed to extract pose (utilizing PnP [68] or other variations [120, 51]) or directly extracting pose [18, 137, 25, 67]. By presuming that the necessary information for pose extraction is provided by step 2.), the pose extraction step becomes crucial for rapid and accurate estimation. In prior research, a large number of intermediate pose representations have been proposed, such as NOCS [138] and variants of NOCS [51, 153, 90] or keypoint regression through heatmaps or votes [92, 52]. These representations are designed to cover the entire input crop area, calculating pixel values across a significant spatial dimension, regardless of whether the area of the image is occupied by the object or not. As a consequence, a substantial amount of redundant and computationally expensive calculations are performed, negatively impacting the efficiency of these methods. Furthermore, some of these approaches require the integration of an additional RANSAC PnP step, which further complicates the pose estimation process and increases computational time. These drawbacks highlight the need for more streamlined and efficient methods in pose estimation. Alternative methods

have been proposed that focus on directly learning the PnP operation [137, 18, 25, 118]. While these approaches have demonstrated improvements in speed and accuracy compared to their counterparts, they introduce complexity, making it harder to achieve high accuracies without additional parameterization, which in turn slows down the runtime of the method. Their increased complexity does not adequately address the issue of information-less regions within the computational pipeline, which continue to contribute to unnecessary calculations. In addition to the aforementioned challenges, applications may seek to refine the predicted pose in cases where the initial estimation is erroneous. Commonly used methods addressing the refinement challenge rely on a computationally expensive *render-compare* iterative process [73, 153, 67, 135], which renders them unsuitable for real-time applications. *Ad-hoc* refinement methods necessitate large models that are specifically designed and trained solely for refinement, leaving the task of initial pose estimation to other methods [67, 73, 153]. More recent approaches attempt to strike a balance between runtime efficiency and the quality of initialization. RePose, for instance, offers a faster solution but requires a high-quality initialization to perform effectively [58]. On the other hand, SurfEmb [41] provides high precision and robustness against occlusions and symmetries, but suffers from extremelly slow inference times.

In the previous chapter, we showed that given a limited amount of parameterization could help extend single-object models to multi-object models with limited degeneration. That lead to a postulation: we can represent an object with a set of keypoints. If this is true, then instead of refining the pose of an object using its rendered image at a given pose, we could instead use a limited number of keypoints, reducing the model complexity and increasing its throughput. In this chapter, we propose a novel efficient method named CRT-6D (Cascaded Refinement Transformers), which demonstrates substantial speed improvements over previous state of the art methods, as evidenced by the performance results in Figure 6.3. Our method eliminates redundant computations in areas where the object is absent while focusing on image regions where the object is present. This is achieved through the implementation of a simple yet effective intermediate offset representation called *Object Surface Keypoint Features* (OSKF), a set of keypoints that can be used to represent an object. Given an initial coarse pose, predetermined object surface keypoints are projected onto the image plane. OSKF are generated by sampling the extracted feature pyramid at each keypoint current 2D location. As the initial pose is not guaranteed to be accurate, deformable attention is employed to guide the sampling around

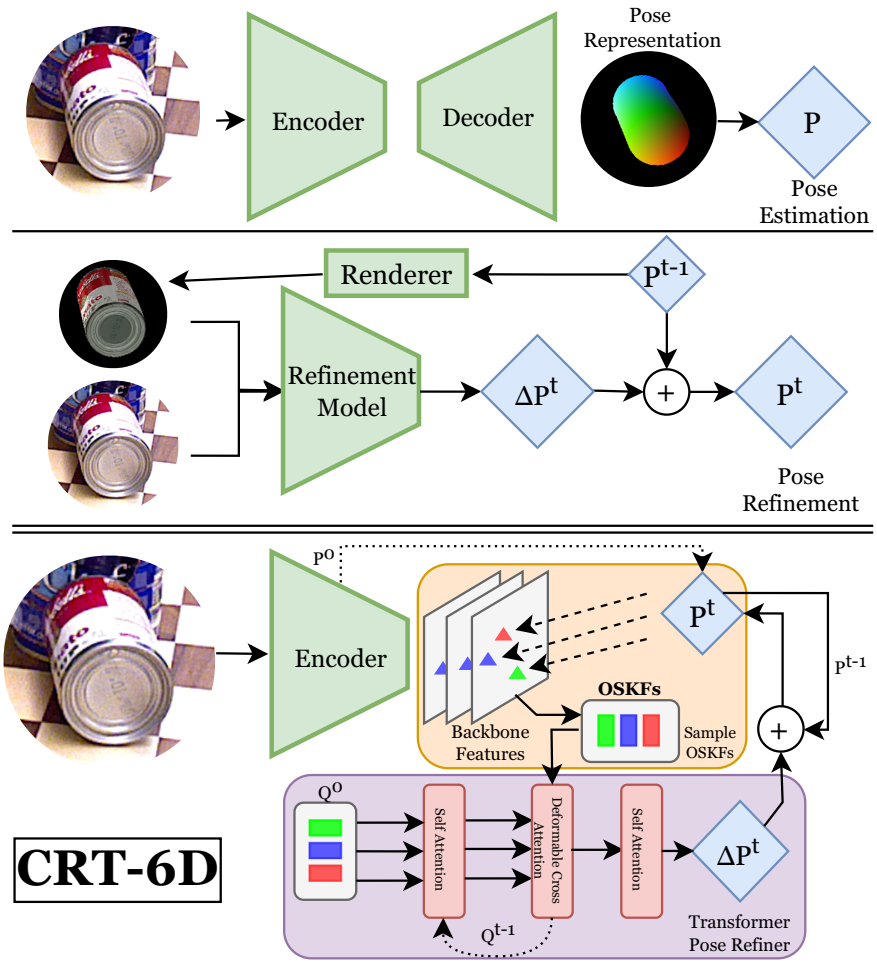


Figure 6.1: **Illustration of CRT-6D.** The feature pyramid is initially generated, and the last feature level is utilized to produce a pose estimation P^0, which serves as the initial reference for the refinement module. **(a)** With P^0, a set of surface keypoints $\mathcal{S}$ is projected into the image plane. **(b)** The deformable attention mechanism employs the location of the set of projected keypoints to sample the feature pyramid. **(c)** Self-attention is then performed over the sampled feature set, and an offset pose ΔP is estimated, such that $P^t = \hat{P}^{t-1} + \Delta P^t$, where t represents the refinement step.

the original 2D location, mitigating potential errors in the coarse pose. To refine the pose estimation, we propose *OSKF-PoseTransformers* (OSKF-PT), which is a transformer module incorporating deformable attention mechanisms [158]. Self-attention and cross-attention operations are conducted over the OSKF set, resulting in an improved pose. Since OSKF represent a computationally efficient solution, multiple OSKF-PT modules are combined in a novel Cascaded Pose Refinement (CPF) module. This module allows for iterative pose refinement in a cascaded manner and can be trained end-to-end. CRT-6D demonstrates a promising approach to 6D object pose estimation, overcoming limitations of existing methods while maintaining accuracy and computational efficiency.

6.1 Related Work

Given the rising effectiveness of transformers in computer vision tasks, there have been attempts to use attention modules to improve human [57, 72], hand [53], depth estimation [8], keypoint detection and matching [121, 81, 140] and object [2, 159, 115, 38, 59] pose estimation. Dosovitskiy et al. [27] proposed an image classifier entirely built with multi-headed attention layers [132], by leveraging long range dependencies. Concurrently, DETR [14] showed it could leverage a more efficient CNN for local feature extraction, which would then feed to a transformer module to extract 2D object detections. Deformable-DETR [158] improved on its predecessor by performing multiple feature sampling at different resolutions of the feature pyramid. For object pose estimation, Amini et al. [2] extends the DETR [14] framework in order to incorporate single-stage pose estimation with a transformer backbone. Zou et al. [159] introduces 6D-Vit, a category level approach using transformer attention modules to combine dense representations extracted from an image and depth map. Hampali et al. [38] propose using a transformer to model the interactions between the object and the hands. OSOP [115] leverages an attention module to extract meaningful information from a sparse set of synthetic templates. However, these improvements come at the cost of runtime, making them unsuitable for real-time applications. Our novel approach, not only improves results, but also decreases the runtime when compared to prior object pose estimation methods.

6.2 Methodology

In this section, the stages of **CRT-6D**, are described. Given an image $\mathcal{I}$, the objective of CRT-6Dis to predict $P_i = [R_i|t_i]$, the 6D pose of the objects targeted by the camera, where i denotes the i^{th} object in the set of N objects $\mathcal{O} = \mathcal{O}_i \mid i = 0, , N-1$ present in the image. The setup followed by other methods [137, 25, 66] is adopted, which involves disentangling pose from object detection, meaning that an off-the-shelf detector is used to crop out regions of the image where objects are present. These regions are then independently processed and supplied to CRT-6D for pose estimation.

6.2.1 Coarse Pose Estimation

A ResNet34 [43] is employed as the backbone for CRT-6D. Although alternative backbones could have been utilized, ResNet offer fairer comparisons with previous works, as they are commonly used by numerous 6D pose estimation approaches[137, 25, 120, 58, 74]. With this backbone, a multi-scale feature pyramid $\mathcal{F} = \mathcal{F}_l \mid l = 1, , L$ of sizes $[s/4, s/8, s/16, s/32]$ is constructed, where $L = 4$.

CRT-6D starts by estimating a coarse pose $P^0_{6D} = [R^0|t^0]$ from the flattened feature map of F_3 using a simple MLP: $FC_\theta(f_3) = R^0_{6D}, \tilde{t}^0$, where θ are learnable parameters, and f denotes the flattened 2D maps. In order to retrieve the rotation matrix $R = [R_1, R_2, R_3]$, the 6D rotation representation $R_{6D} = [r_1, r_2]$ introduced by Zhou *et. al.* [157] is employed. This representation has been successfully applied in previous methods [137, 25, 67] for effective pose estimation. It can be formally defined by:

$$\begin{cases} R_1 = r_1 \\ R_3 = R_1 \times r_2 \\ R_2 = R_3 \times R_2 \end{cases}, \tag{6.1}$$

where $r1$ and $r2$ are unit vectors.

As a consequence of the projective function applied by the camera sensor, the appearance of an object is influenced not only by its orientation but also by its position. Given that a cropped image and known camera intrinsics K are utilized, predicting P a becomes a one-to-many function, where identical appearances may correspond to various egocentric orientations, as highlighted by 3D-RCNN [66]. A prevalent strategy for addressing this issue, which is also adopted here, involves estimating the allocentric orientation and implementing a transformation to retrieve the egocentric orientation during the inference stage.

Although the objective is to obtain the global translation $t = [t_x, t_y, t_z]$, this information cannot be directly acquired by CRT-6D due to the cropping step. Wang *et. al.* [137] demonstrated that selecting an appropriate translation representation impacts the performance of the method. As a result, the use of $t' = [O_x, O_y, t_z]$ is employed, which enables the recovery of translation t through back-projection. CRT-6D adopts the scale-invariant representation $\tilde{t} = [\gamma_x, \gamma_y, \gamma_z]$ [137, 25]:

$$\begin{cases} \gamma_x = (O_x - c_x)/s_{bbox} \\ \gamma_y = (O_y - c_y)/s_{bbox} \\ \gamma_z = t_z/r_{bbox} \end{cases}, \tag{6.2}$$

where the scale of the crop bounding box is $s_{bbox} = max(w_{bbox}, h_{bbox})$ and ratio is $r = s_{bbox}/s$, with s referring to the original size of the image. The network outputs, denoted by γ, are modeled with respect to the crop region of interest, which is performed independently of the pose estimation step. This separation effectively enables the disentanglement of the pose estimation and object detection tasks, allowing us to focus specifically on the latter in this chapter.

6.2.2 Object Surface Keypoint Features - OSKF

As previously mentioned, refinement methods typically depend on a separate, robust pose estimator [67, 58, 73], which can slow down both the inference and training processes. In contrast, CRT-6D is specifically designed to estimate and refine the pose by reutilizing the multi-scale feature pyramid. This approach not only streamlines the process but also enhances the overall efficiency and effectiveness of the pose estimation and refinement, making it a more suitable solution for real-time applications.

The rendering step utilized by pose refiners [67, 73, 58] is replaced, and instead of using an expensive pose representation, a pose offset representation $\mathcal{Z}$ is instead generated. This representation consists of a compact set of backbone features obtained by sampling at 2D keypoint locations $\mathcal{P}^t = \pi(\mathcal{S}k^t, P6D^t, K_{cam}) \mid k = 1, ..., K$, where π represents the projection function, K_{cam} denotes the known camera intrinsic parameters, P^t_{6D} refers to the reference pose, and $\mathcal{S}$ is a set of pre-defined keypoints, with t indicating the iteration step. Since the input image I does not undergo any perturbations from the pose estimation step to the refinement step, or even between refinement iterations, these features are reused rather than being recalculated at every iteration, in contrast to prior methods [153, 73, 67, 58].

In summary, CRT-6D uses OSKF $\mathcal{Z} = \{\mathcal{Z}_k \mid k = 1, ..., K\}$, a lightweight offset pose representation generated by sampling the feature pyramid at all spatial scales:

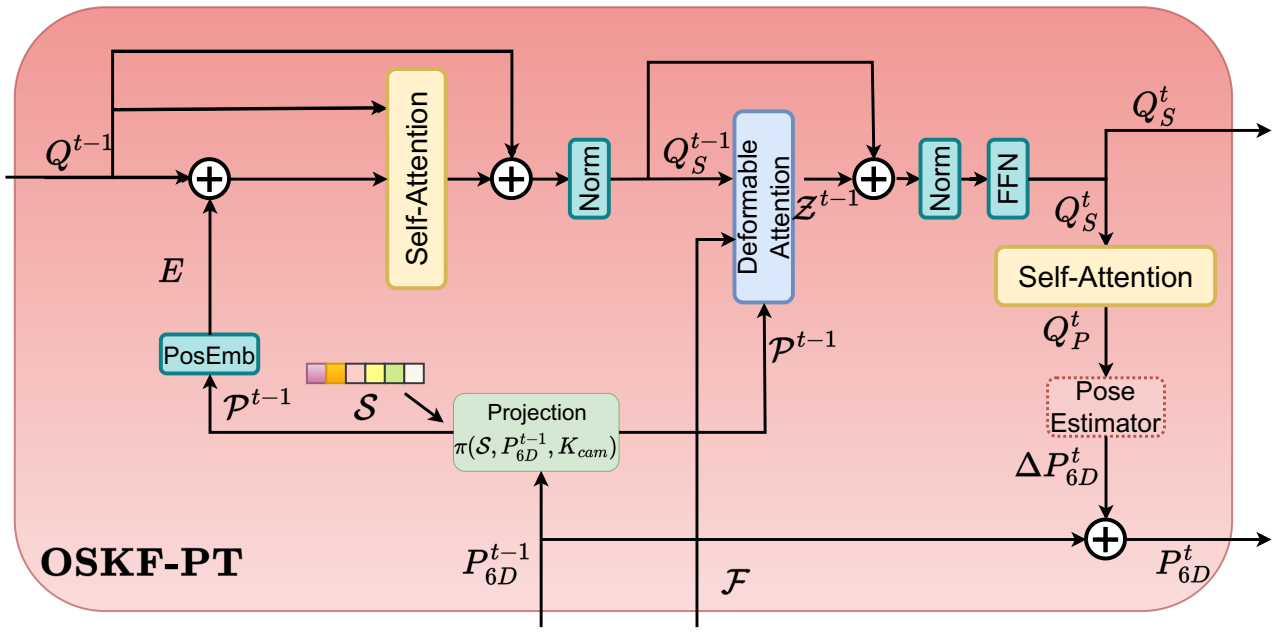


Figure 6.2: **Illustration of OSKF-PT.** OSKF-PT receives as an input the previous Q^{t-1}, P_{6D}^{t-1} and $\mathcal{F}$, the last one being the same on every iteration. It outputs a new refined pose P_{6D}^t and Q^t, an initial query to guide the next iteration.

$$\mathcal{Z} = \{\mathcal{F}_l(\mathcal{P}_k) \mid k = 1, ..., K,\ l = 1, ..., L\} \tag{6.3}$$

where l denotes the l^{th} element of the feature pyramid.

The set of object keypoints $\mathcal{S} = \{\mathcal{S}_k \mid k = 1, ..., K\}$ chosen for OSKF are generated using the farthest point sampling algorithm [92], where K is a hyperparameter of the number of used keypoints. Each of these features represent the local information around $\mathcal{P}$, with coarse information at higher feature levels and finer on lower. The following modules will learn to recover the pose offset embedded in $\mathcal{Z}$.

6.2.3 OSKF Pose Transformer (OSKF-PT)

The untopologized set of OSKF $\mathcal{Z}$ presents a use case for transformers which are able to process data without introducing implicit spatial bias [132] allowing the attention mechanisms to learn spatial and structural relationships. CRT-6D is designed to interpret $\mathcal{Z}$ and learn to output the offset pose ΔP such that:

$$P_{6D}^t = \hat{P}_{6D}^{t-1} + \Delta P_{6D}^t, \tag{6.4}$$

at the t refinement step.

However, the estimated pose and keypoints may not always be accurate and can be externally or self occluded in certain scenarios. As a result, the optimal information for a keypoint may not be located in close proximity to its 2D position in the image. Inspired by the Deformable-DETR method [158], the OFKF-PT module addresses this challenge by leveraging deformable attention mechanisms to learn to sample information around the reference keypoint position. A diagram of the OFKF-PT module is shown in Figure 6.2.

OFKF-PT are composed by a self-attention layer, a deformable self attention layer and another attention layer for pose estimation, in this order. We start by performing self-attention over a query matrix Q, where each row corresponds to a keypoint in the set $\mathcal{S}$. The multi-head attention operation takes $\hat{Q}$, $\hat{Q}$ and Q as query, key and value, where $\hat{Q} = Q + E$. $E_k = \textit{PosEmb}(\mathcal{P}_k)$ are the positional embeddings for keypoint k and *PosEmb* is the high frequency cosine embedding for position parameterization [132]. Following [158], we then apply residual addition and layer normalization. We denote the output of this operation as Q_S.

The multi-scale deformable attended OSKF are formally defined by:

$$\tilde{\mathcal{Z}}_{kl} = \sum_{j=0}^{J} A_{ljk} W \mathcal{F}_l(\mathcal{P}_k + \Delta\mathcal{P}_{ljk}) \tag{6.5}$$

where $\Delta\mathcal{P}_{ljs}$ refers to the predicted sampling offset for the j^{th} deformable position, spatial level l and object keypoint k. J refers to the number of sampling points used for deformation. $\Delta\mathcal{P}_l js$ and A_{ljs} are computed by a simple linear transformations of Q_S.

We apply a self-attention operation over $\tilde{\mathcal{Z}}$ in order to capture long distance information which we denote as Q_P. We chose to use a global pooling operation over Q_P instead of using a CLS-Token (which is shown to make no difference in practice [27]), followed by a small MLP to output $[\Delta R, \Delta\gamma_x, \Delta\gamma_y, \Delta\gamma_z]^t$. We update the reference pose P^t:

$$\begin{cases} R^t = \Delta R^t \; R^{t-1} \\ \gamma^t_{\{x,y\}} = \gamma^{t-1}_{\{x,y\}} + \Delta\gamma^t_{\{x,y\}} \\ \gamma^t_z = \gamma^{t-1}_z \cdot (1 + tanh(\Delta\gamma^t_z)) \end{cases} . \tag{6.6}$$

We found that using a *tanh* parameterization over $\Delta\gamma_z$ performed better than linear [67] or exponential [73].

Since each OFKF-PT operates independently and builds on the output of the preceding one without degrading its performance, stacking can continue indefinitely. However, diminishing returns are observed, as discussed later in this Chapter (see Fig. 6.3. Since we want to maintain a focus on runtime performance, the experiments are limited to three OFKF-PT modules. The output of the deformable attention operation is reused on the next iteration, as the input matrix becomes $Q^t = \tilde{\mathcal{Z}}^{t-1}$. The initial query Q^0 is a learnable embedding.

6.2.4 Objective Functions

The main goal of CRT-6D is to process an image and produce the 6D pose of the target object. To perform this operation, it goes through an iterative refinement process. At each iteration, we compute a valid 6D pose and reuse it on the next iteration. Due to the simplified objective of CRT-6D, which does not output an intermediate pose representation, our overall objective $\mathcal{L}$ is guided by the pose estimation error and can be defined as:

$$\mathcal{L} = \lambda\mathcal{L}^0 + (1-\lambda)\sum_{i=1}^{N}\mathcal{L}^i \tag{6.7}$$

with i indicates the refinement iteration, $N = 3$ is fixed for all experiments and $\mathcal{L}^0$ the loss for the coarse estimation. The pose loss $\mathcal{L}$ is disentangled and separated into rotation and position loss:

$$\mathcal{L} = \alpha\mathcal{L}_R + \mathcal{L}_{pos}. \tag{6.8}$$

Recalling the pose parameterization described in Sec. 6.2.1, the loss functions are defined as:

$$\begin{cases} \mathcal{L}_R = \underset{x \in \mathcal{P}}{avg} ||Rx - \hat{R}x||_1 \\ \mathcal{L}_{pos} = ||\gamma_x - \hat{\gamma_x}, \gamma_y - \hat{\gamma_y}, \gamma_z - \hat{\gamma_z}||_1 \end{cases}, \tag{6.9}$$

where $\hat{\cdot}$ refers to the ground truth data. When the target object is symmetric, a variation of $\mathcal{L}_R$ is used [138], while $\mathcal{L}_{pos}$ is invariant to symmetries. We employ a sparse loss function, in this case L1, due to the nature of the training process, which often encounters highly challenging and unconventional poses. These difficult scenarios arise from the various image augmentations applied to the training data. By using a sparse loss, we prevent the optimization process from excessively penalizing these outlier cases, thereby allowing the model to focus on learning generalizable features rather than being degraded by rare training examples. Note that while our refinement module outputs an offset pose ΔP, the transformation in Eq. 6.4 is differentiable which means CRT-6D can be directly optimized through the set of predicted poses $P_{6D} = \{P_{6D}^t | t = 0, ..., N\}$.

6.3 Experimental Evaluation

We conducted experiments on two benchmark dataset LM-O[10] and YCB-V[13] where we present strong evidence of the potential of CRT-6D. These datasets were selected not only to enable direct comparison between CRT-6D and previous state-of-the-art methods but also because they are consistently used in previous chapters and other state-of-the-art approaches. The commonly used Linemod dataset (LM) [45] has become extremely saturated within the instance-level pose estimation task with most recent methods achieving over 95% accuracy [67, 137, 25] and is now not even considered one of the main benchmark datasets for the BOP challenge [49]. For this reason the experiments are aimed at the more challenging Linemod Occlusion (LM-O), a subset of 1214 LM images, where ~ 8 objects are annotated on every image. YCB-V, a larger dataset with 21 target objects, is also used as it presents objects with very challenging symmetries. We also show through ablation studies our key contributions, including the improvements stemming from the use of iterative refinement and the high accuracy it achieves with an impressively low inference time. Results for all BOP datasets [49] are available on the challenge website and on Table 6.3.

6.3.1 Experimental Setup

Datasets Setup. For LM-O, in accordance with prior art [137, 25, 120], the available LM real images, $\sim$ 1200 images are available per object, for training. The YCB-V contains over 100k real images readily available for training. However, the dataset is generated through video resulting in similar frames where the objects are seldom fully visible. On top of real images, we also make use of synthetic data. For a fair comparison, we employ the readily available PBR splits [23], available for both LM-O and YCB-V, a dataset of photo and physically realistic synthetic images containing the target models with challenging poses and under heavy occlusion. We also perform common on-the-fly image augmentations such as color jittering, blur and noise as well as more complex operations such as in-plane image rotations and background removal. For experiments under the BOP [49] settings, LM-O methods are trained only with PBR synthetic data. We also implement Dynamic Zoom-In (DZI) [74] in order to be robust to detection errors. During training we apply uniform perturbations to the center and scale of the bounding box. Padding detection predictions at test time is a useful technique that ensures robustness to detection inaccuracies [74, 137, 25], which can stem from weaker detectors or complex occlusions. However there is a trade-off there is a trade-off between accommodating such errors and the loss of information caused by downsampling the image through zoom-out operations stemming from padding and resizing. We found that increasing the detection bounding boxes by 20%, to ensure the object is fully visible, yielded the best results.

Implementation details. CRT-6D is implemented using PyTorch [91]. The number of heads and reference points used for deformable attentions in CRT-6D is determined by the recommendations made in Deformable-DETR [158], with 8 heads and 4 reference points being used. The entire model, including the cascaded refinement step, is trained end-to-end. All ablation experiments are optimized with the same number of training iterations, and the training of LM-O experiments is done for 250k iterations with a batch size of 32, where PBR images make up 50% of the batch, or 100% under BOP standards [49]. YCB-V is trained for 350k iterations, with the same PBR ratio and batch size. The optimizer used for training CRT-6D is Ranger [146]. We begin training with a learning rate of 10^{-4} and use a cosine annealing schedule that starts at 85% of the total training time. While similar methods have used a lower learning rate for backbone weights in their experiments [14, 158, 71], it was found that this was not optimal for CRT-6D. $\lambda = 0$ for the first 20% of iterations because

P^0 initially has very poor pose estimations, which makes it difficult for CRT-6D to learn. For the remainder of training, $\lambda = \frac{N-1}{N}$, where N is the number of pose refiners used. For loss equilibrium, $\alpha = 3$ on all experiments.

In order to maintain fairness when comparing with other methods, on the LM-O dataset, publicly available 2D detections from a Faster-RCNN [103] are adopted, the same ones utilized by in [137, 25, 120]. Similarly, for the YCB-V dataset, publicly available detections from FCOS[129], trained by Li et al. [74], are employed. In the case of BOP results, the standardized detections provided by the challenge.

Evaluation Metrics. The same evaluation practices as previous methods are followed [137, 25, 153, 148, 92]. For the LM-O dataset, results are presented using the ADD(-S) metric [98, 49]. According to the ADD(-S) metric, a pose is deemed accurate if the distance between the transformed model points and the ground truth points is less than $d \cdot 10\%$, where d represents the object diameter. For symmetric objects, the distance from each transformed point to the closest ground truth is determined, and the same threshold is applied [49, 45].

When conducting experiments on the YCB-V dataset, the Area Under the Curve (AUC) of the ADD metric is also considered, with a maximum threshold distance of 10cm [148].

Under BOP standards, the average recalls utilized by the challenge, including AR_{vsd}, AR_{mssd}, and AR_{mspd}, are presented along with their mean. For more information about these metrics, readers are referred to the BOP challenge [49]. Inference time measurements for both the proposed method and prior techniques are obtained using publicly available code on a GTX1080ti. For simplicity, detection time is disregarded as all methods require an initial detection and crop and are evaluated using the same detections.

6.3.2 Evaluation Results

CRT-6D is directly compared to various methods categorized under different types of object pose estimation. The first category refers to simple pose estimation approaches where an image is taken as input and the target pose of the object directly estimated. For this comparison, we selected commonly used state-of-the-art methods, including PVNet [92] and GDR [137], the latter of which offers both

Table 6.1: **Comparison study on LM-O.** The results for the LM-O dataset under the ADD(-S) metric are presented and compared to other state of the art methods. In terms of accuracy, **CRT-6D** is outperformed only by ZebraPose [120], a method with an inference time of $\sim$ 191ms for a single object, while **CRT-6D** estimates the pose for all objects in a single LM-O image ($\sim$ 8 objects) in 36ms. Best results are bolded while second best are underlined. *N.S.O.* refers to the number of objects supported by a model and * denotes symmetric objects.

Type	Pose Estimation					Refinement		Hybrid
N.S.O.	1	8	1	8	1	1	8	8
Method	PVNet [92]	GDR [137]	GDR [137]	SO-Pose [25]	ZebraPose [120]	RePose [58]	DeepIM [73]	**CRT-6D**
Ape	15.8	44.9	46.8	48.4	<u>55.2</u>	31.1	**59.2**	53.4
Can	63.3	79.7	90.8	85.8	**94.9**	80.0	63.5	<u>92.0</u>
Cat	16.7	30.6	40.5	32.7	**56.6**	25.6	26.2	<u>42.0</u>
Driller	65.7	67.8	82.6	77.4	**94.7**	73.1	55.6	<u>81.4</u>
Duck	25.2	40.0	46.9	48.9	**60.9**	43.0	<u>52.4</u>	44.9
Eggbox*	50.2	49.8	54.2	52.4	**64.7**	51.7	<u>63.0</u>	62.7
Glue*	49.6	73.7	75.8	78.3	**84.5**	54.3	71.7	<u>80.2</u>
Holepuncher	36.1	62.7	60.1	75.3	**83.2**	53.6	52.5	<u>74.3</u>
Average	40.8	56.1	62.2	62.3	**74.3**	51.6	55.5	<u>66.3</u>

single and multi-object versions, as well as its enhanced variant, SO-Pose [25]. Additionally, we compare our method to slower but more accurate pose estimators, such as ZebraPose [120], CosyPose [67] and SurfEmb [41]. Since CRT-6D adopts a hybrid approach, integrating both pose estimation and pose refinement, we also include comparisons with existing refinement methods such as RePose [58] and DeepIM [73].

LM-O results. We present the results under the ADD(-S) metric for LM-O on Tab. 6.1. We present competitive results, with the second best overall accuracy behind ZebraPose [120]. ZebraPose is significantly more expensive to run than CRT-6D and this contrast is examined in further detail later in this experimental section. Compared with real-time methods, CRT-6D achieves a 6.4% improvement over SO-Pose [25], the former state of the art. In accordance with the BOP challenge guidelines, CRT-6D attains state-of-the-art performance, achieving an Average Recall of 67.2%. This outperforms all other RGB-based methods, irrespective of their inference times. The exceptional result can be attributed to the method robustness to occlusion, which is facilitated by the utilization of deformable attention. This allows the method to focus on regions far from the reference position when keypoints are occluded, as demonstrated in Fig. 7.4.

YCB-V results. On the YCB-V dataset, the proposed method delivers highly competitive results, achieving 72.1% and a state-of-the-art 87.5 ADD-(S) and AUC of ADD(-S), respectively. In comparison to the nearest real-time methods, the proposed method outperforms GDR-Net[137] and SO-

Table 6.2: **LM-O and YCB-V datasets under the BOP standards [49].** We present the results for all metrics used on the challenge with *Mean AR* referring to average *AR* over both datasets. Best results are **bolded** while second best are underlined.

Method	LM-O				YCB-V				**Mean AR**
	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	**AR**	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	**AR**	
EPOS [51]	0.389	0.501	0.750	0.547	0.626	0.677	0.783	0.695	0.621
GDR-Net [137]	-	-	-	-	0.584	0.674	0.726	0.661	-
SO-Pose [25]	0.442	0.581	0.817	0.613	0.652	0.731	0.763	0.715	0.664
SurfEmb [41]	-	-	-	0.656	-	-	-	0.718	0.687
CosyPose [67]	0.480	0.606	0.812	0.633	**0.772**	**0.842**	**0.850**	**0.821**	**0.727**
SC6D [12]	-	-	-	-	0.695	0.796	0.804	0.765	-
CRT-6D	**0.504**	**0.640**	**0.837**	**0.660**	0.707	0.776	0.774	0.752	0.706

Table 6.3: **BOP challenge results.** At the time of writing, CRT-6D is the fastest method on the BOP challenge website achieving state-of-the-art results, with its average runtime being than 60ms for all objects in an image.

AR_{Core}	$AR_{LM\text{-}O}$	$AR_{T\text{-}LESS}$	$AR_{TUD\text{-}L}$	$AR_{IC\text{-}BIN}$	AR_{ITODD}	AR_{HB}	$AR_{YCB\text{-}V}$	Time (s)
0.599	0.660	0.644	0.789	0.537	0.208	0.603	0.752	0.059

Pose[25] by 46% and 27% on the ADD(-S) metric, as well as 9% and 4% on AUC of ADD(-S). When contrasted with slower methods, the proposed method is surpassed on the ADD(-S) metric only by ZebraPose [120], which operates at an order of magnitude slower than the proposed approach. Additionally, YCB-V results are presented under the BOP standard on Table 6.2, where the proposed method is outperformed solely by a slower technique CosyPose [67], which is also an order of magnitude slower than CRT-6D. The difference in performance between the datasets stems from both the inherent challenges of the datasets and the metrics used. LM-O is a more challenging subset of Linemod, where objects are often heavily occluded, contrasting with the standard Linemod dataset in which objects are seldom occluded. Moreover, the metrics used are sensitive to object sizes and YCB-V includes some of the largest objects, which further influences the results.

BOP results. The full detailed results of CRT-6D for the BOP challenge are available on its website. The AR accuracy for all main datasets can be seen in Table 6.3.

Runtime comparison to state of the art. DeepIM [73] represents a significant milestone in the field of learned pose refinement. Nevertheless, it exhibits a considerable drawback in terms of speed, with an inference time of 41ms per object per iteration (with a recommendation of two iterations), excluding the time required for model initialization, which is also a large contributing factor to its

accuracy. In contrast, CosyPose [67] introduces a unified model that leverages two fully trained CNN, where the first CNN provides an initial coarse estimation while the second refines it. On the hardware used to measure runtime, the substantial size of the employed models results in an inference time of approximately 100ms for a single object. In an effort to enhance the speed of refinement, Repose [58] presents a method that achieves an impressive 18ms inference time with 5 refinement iterations. However, Repose requires an accurate pose initialization (utilizing PVNet [92], which itself takes over 25ms) and solely supports a single object per model.

One the other hand, CRT-6D takes on average 26ms and 34ms for YCB-V and LM-O images (each with $\sim$ 4 and $\sim$ 6 object instances) for all objects on a single model, making CRT-6D an order of magnitude faster compared to other refinement techniques. Contemporary state-of-the-art real-time methods, such as GDR-Net [137] and SO-Pose [25], require 2 and 3 times longer, respectively. Moreover, CRT-6D not only is faster than the previously mention methods but it more accurate, on both LM-O and YCB-V. A more detailed comparison to other methods w.r.t. inference time can be seen in Fig. 6.3.

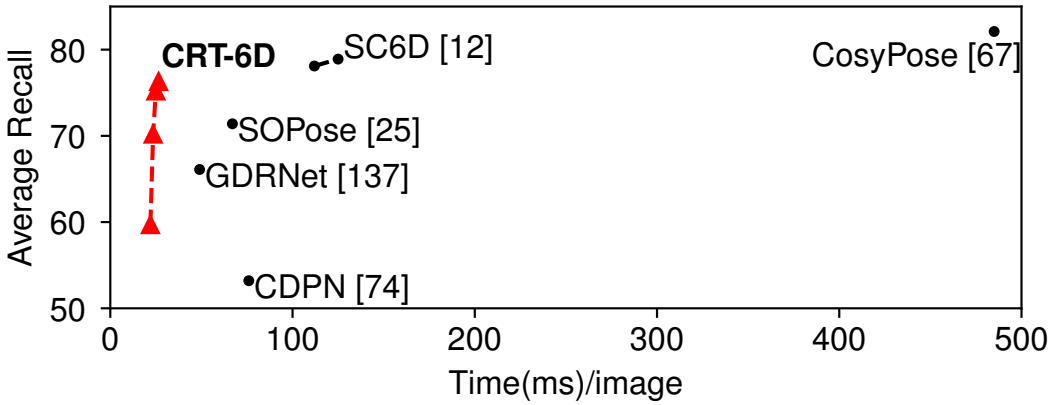


Figure 6.3: **Comparing the inference time between CRT-6D and other state-of-the-art methods.** CRT-6D is the fastest method, even after three refinement steps, compared to previous approaches while maintaining highly competitive results relative to methods that are 10 times slower. The average time required to estimate the pose for all objects in an image (approximately 4.75 objects per image on YCB-V) is measured. It should be noted that ZebraPose [120] and SurfEmb [41] have been excluded from the comparison due to their significantly longer runtimes, as the pose estimation takes over 250ms and 2000ms per crop, respectively.

Table 6.4: **Effects of the Cascaded Pose Refinement**. We present the improvement with each refinement step. We can observe the diminishing returns where the first refinement improvement is larger than all other iterations combined.

Dataset	LM-O		YCB-V		
Metric	ADD-(S)	AR	ADD-(S)	AUC	AR
0	40.5	59.8	50.6	81.0	59.7
1	61.4	69.0	63.2	84.9	70.2
2	64.9	71.1	70.8	87.1	75.2
3	66.3	71.5	72.1	87.5	76.3

6.3.3 Ablation studies

Ablation Study on the Number of OSKF: With the experiments results shown on Fig. 6.4 it can be seen that the performance of CRT-6D is not heavily reliant on an extensive number of reference keypoints. While using 8 keypoints falls short of the desired accuracy level, the distinction between 16 and 128 keypoints is relatively insignificant. Consequently, for all experiments, $K = 64$ keypoints are chosen as this configuration produces optimal results for both the LM-O and YCB-V datasets.

Cascaded Pose Refinement. CRT-6D iteratively refines the target object initial pose in a cascaded manner. In Tab. 6.4, we present the influence of multiple iterations on accuracy. It can also be seen that marginal improvements diminish as multiple refinements are applied. Notably, the reduction in error achieved during the first refinement iteration surpasses the combined impact of subsequent iterations.

Effect of coarse pose accuracy. In Table 6.5 we show the impact of the initial coarse pose on

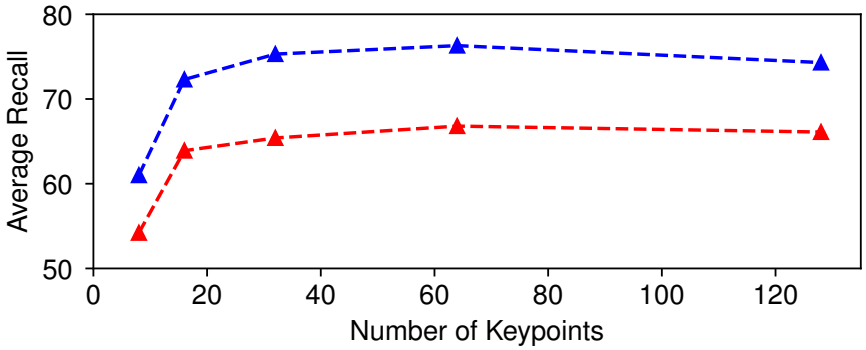


Figure 6.4: **Experiments over number of OSKF used.** While there is a noticeable decline in performance when using only 8 keypoints, the differences become minor as the number of keypoints increases. The optimal configuration employs 64 keypoints for both datasets, striking the right balance between performance and complexity. CRT-6D is not reliant on a large number of keypoints.

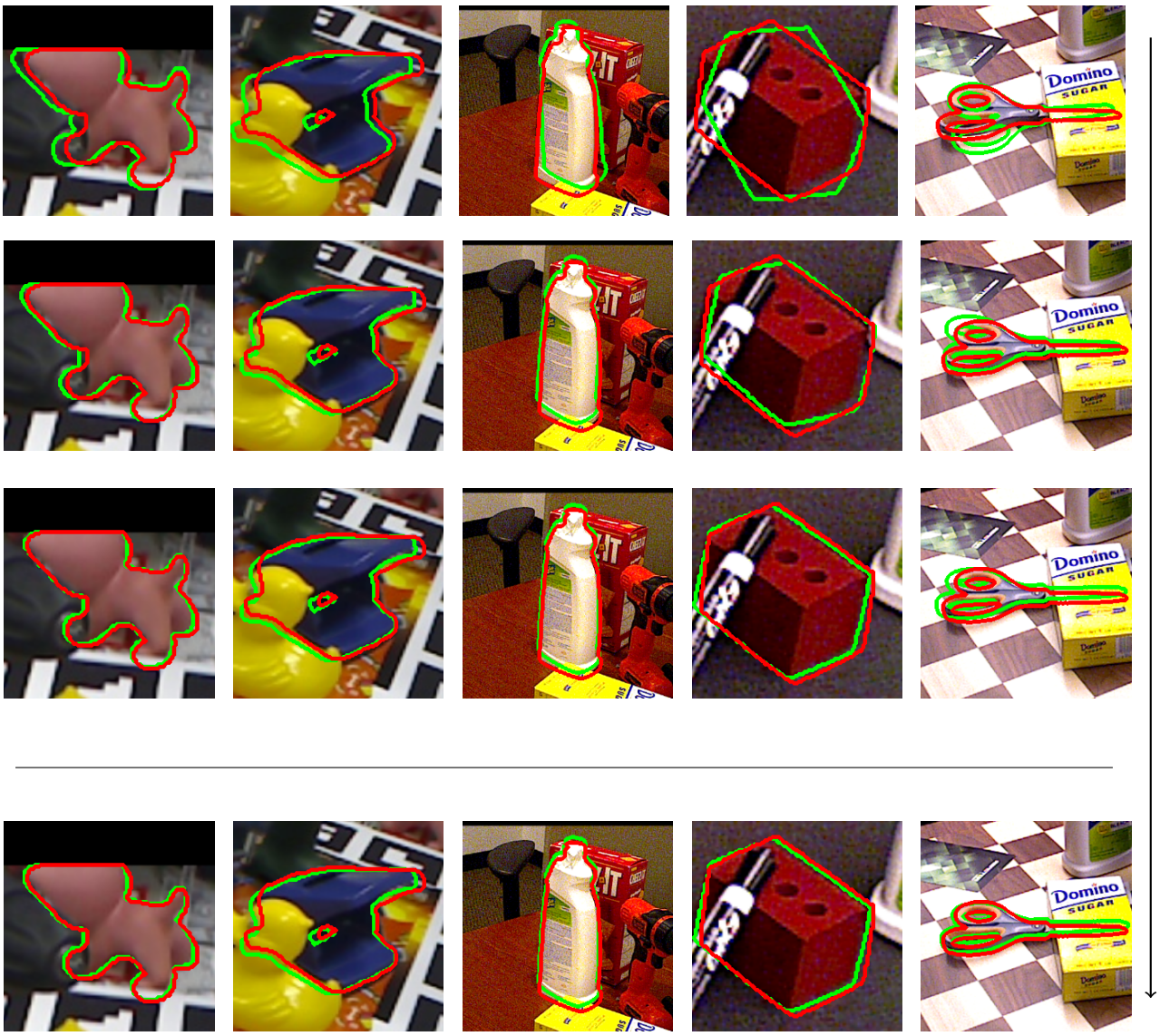


Figure 6.5: **Qualitative results on refinement steps.** The progress in pose accuracy over iterative refinements is demonstrated, with the initial coarse pose in the first row and the final results in the last row. Despite the initial pose being of low quality according to evaluation standards, it is found to be sufficient as a reference for CRT-6D.

the OSKF-TP module. We show OSKF-TP can serve as an independent refiner module, such as DeepIM [73] or Repose [58], as it can receive pose estimations from other approaches and refine them to a state of the art level. Using 3 OSKF-TP modules, we improve GDR [137] and SO-Pose [25] results by 22% and 10% respectively. This experiment is done for the sake of completion: CRT-6D must generate the feature pyramid $\mathcal{F}$ to compute OSKF, which is the most expensive operation in the pipeline, making the use of these initial poses redundant and unnecessary as we achieve better results with our coarse initialization.

Table 6.5: **Initial Pose ablation study.** The experimental results are presented, demonstrating the effects of applying the proposed refinement on top of prior art. The initial poses are shown in the top three rows, while the refined poses can be found in the bottom three rows. An * indicates the utilization of the refinement technique from CRT-6D on the estimated pose of the respective method.

Initial Pose	ADD(-S)			2°2cm	5°5cm	Total
	0.02d	0.05d	0.1d			
GDR [137]	3.42	26.5	56.1	3.14	35.1	24.9
SO-Pose [25]	**4.55**	**31.4**	**62.3**	**3.39**	**39.0**	**28.1**
CRT-6D-Coarse	1.53	16.3	40.5	1.99	29.45	18.0
GDR* [137]	6.08	34.4	62.5	5.67	43.5	30.43
SO-POSE* [25]	6.22	34.8	63.4	5.38	44.4	30.84
CRT-6D	**6.23**	**36.2**	**66.3**	**5.48**	**45.4**	**31.9**

6.3.4 Qualitative Results

Visualization of Deformable Attention. In Fig. 6.6 we visualize the attention sampling points generated by the deformable attention operation. We only show sampling positions (white circles) for high attention weights $A > 0.25$ (see Eq. 6.5). When the keypoints is visible, the deformations occur near the keypoint projection location, whereas for occluded keypoints attention is scattered around the image. The transformer can guide its sampling to regions with clues to recover subtle pose differences.

Refinement Qualitative Examples. On Fig. 6.5 we present qualitative results showing the impact of each refinement step. We can see that for most objects the original pose is significantly inaccurate. Nonetheless, CRT-6D can recover an accurate pose after 3 iterations of the refinement module. The last column shows the results on the *scissors* object, the hardest object in YCB-V, CRT-6D was not able to recover the correct pose due to a poor initial pose.

6.4 Conclusion

This chapter presents a novel object pose refinement technique that, in contrast to existing methods, is built for real-world applications due to its efficient and rapid inference capabilities. CRT-6D matches or surpasses state-of-the-art performance on challenging datasets while having over $2\times$ the throughput. This efficiency is attained through a novel pose guided sampling strategy, which minimizes unnecessary computations by attending to regions where the object is known to be present, based on coarse initial pose estimates, which are subsequently refined for precise object pose estimations.



Figure 6.6: **Deformable attention sampling locations.** The red circle indicates the reference point while the white circles indicate deformed sample positions. On the last column, we show attention sampling for the same object and keypoint, with the first row having the keypoint visible whereas on the bottom two the keypoint is self or externally occluded. When the keypoint is occluded, attention learns to sample clues elsewhere on the object silhouette.

Chapter 7

PoseMatcher

By one-shot pose estimation, we refer to estimating the pose of an novel object based on information seen only at test-time, without the object,or its category, being present in the training dataset. Older methods have some very hard constraints such as needing masking and depth maps at test-time [46], a full colored 3D model of the target object [73], need to be fine-tuned on images of the same dataset for difficulties of overcoming domain gap [80] or rely on an extremely expensive test-time optimization through renderers [150, 70, 89]. More recently, feature matching methods have been shown to achieve impressive results. Particularly, OnePose [122] and OnePose++ [44] make use of existing state of the art pre-trained descriptor extractors and build a pose estimation pipeline on top. However, by relying on a fixed pre-trained model to extract descriptive keypoints from templates which fails to capture the optimal keypoint object descriptions. Ideally, the template extraction model should be the same as the query model and should be jointly optimized.

We rethink the problem and introduce a three-view pipeline that allows us to jointly train the template and query extractors. Just by redesigning the training problem, we can improve OnePose++[44] without changing anything else about its pipeline. Morover, we also introduce a new efficient image to object attention layer which we call IO-Layer, which reduces both parameters when compared to the modules used by OnePose++ [44] but also separates the two modalities, allowing the model to optimize weights specifically for either image or objects keypoints. We also found that working with object at all stages of the feature matching is redundant and reduces pose estimation accuracy and we introduce. On top of these changes, we further employ a 3D-refinement technique based on zoom

refinement used on instance-level pose estimation [73, 15, 67], inspired by the work developed in Chapter 6.

7.1 Related Work

Methods such as Pitteri *et. al.*[95] and CorNet[94] rely on training with a subset of similar objects of the same dataset and/or scene, which includes biases towards illuminations, noise, background and shape. More recent attempts have shown researchers are keen on tackling one-shot pose estimation. OSOP [115] introduces a global matching method, by maximizing local feature matching, where a given query is matched to the closest viewpoint from a database of pre-processed synthetic viewpoints, which requires the object model, an assumption we do not make. Gen6D [80] works in a similar fashion by matching a query to a small amount of real annotated images and then refining the pose. However, it is susceptible to poor pose initialization and poor 2D bounding box detections. As we have seen in the previous chapter, transformers can be applied to the 6D object pose estimation field with success. Although traditionally thought to be compute heavy and unsuitable for real-time tasks [8, 159, 27], the last chapter showed that with careful design we can exploit the transformers powerful capture of long-range relationships with minimal overhead. Image matching is a field that shares keypoint detection with the topic of this thesis [5, 121, 81, 140]. A key requirement of these methods is their generalization to unseen scenes. Some approaches to 6D object pose estimation methods, built on top of image matchers, have already been proposed. Sun et al. [122] propose using SuperPoint [24] detection and descriptors to extract and store object texture information and then match them to a set of keypoints generated from a query image. The matching process is improved using a graph based attention network, similar to what SuperGlue [112] proposed for image to image matching, where the graph network is encouraged to push matching descriptors closer together while pushing non-matching descriptors apart. The closest work to our own is OnePose++ [44], an improved version of OnePose [122], where the feature extraction is replaced by the detector free feature matcher LoFTR [121]. However, OnePose++ [44] adapts a pre-trained LoFTR to one-shot pose estimation without taking consideration the different modalities of image and point cloud, self-occlusion redundancy and the possibility of using 6D pose based refinement.

7.2 Method

The goal of the PoseMatcher is to establish matches between the two sets of keypoints. We establish correspondence between the 2D keypoints features extracted from the query image $\mathcal{I}^Q$ and the object point cloud $\{\mathcal{P}_O\}$. From those matches we can apply PnP and recover full 6D pose ζ. A diagram of PoseMatcher can be seen in Fig. 7.1.

7.2.1 Template Based Training

We adopt a new training methodology that allows us to design a one-shot object-image detector free feature matching model from scratch, specifically aimed at pose estimation task. We therefore remove the need for pre-trained descriptor extractors used by OnePose and OnePose++ [122, 44].

However, a single viewpoint will result in a partial reconstruction of the object. At inference time,

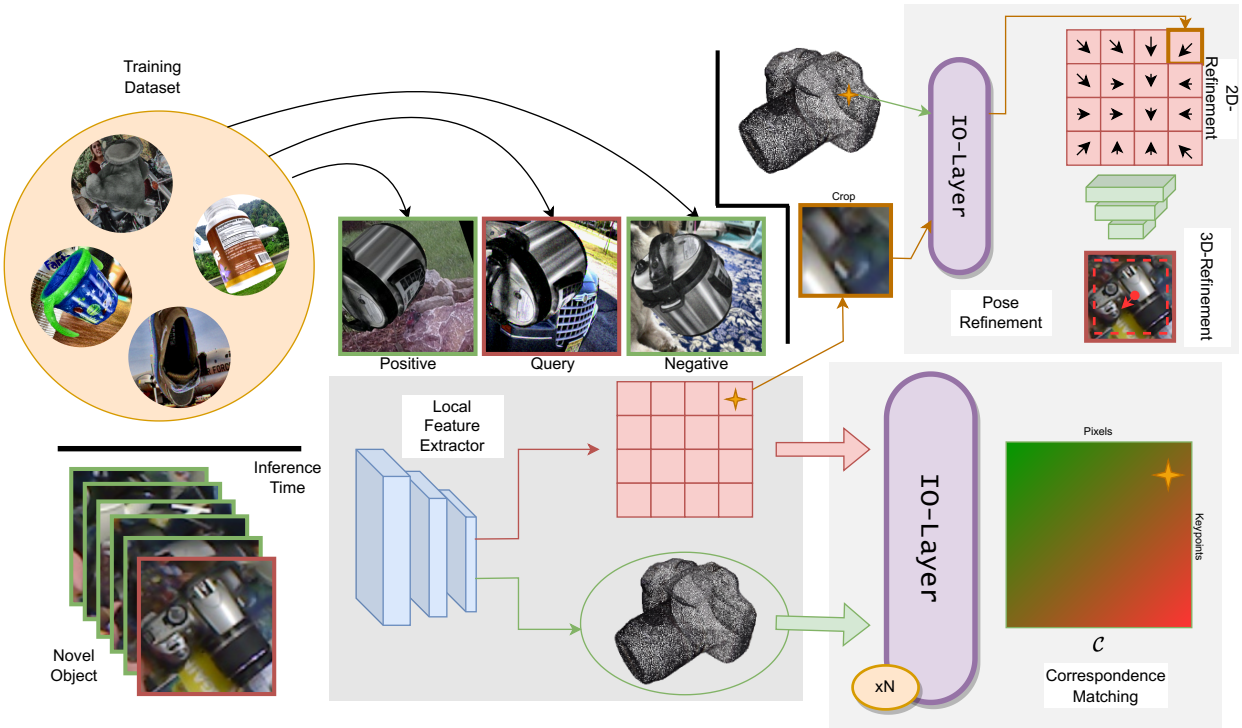


Figure 7.1: **Diagram of PoseMatcher.** At training time we sample from the synthetic training dataset and from two opposing views templates we construct a partial point cloud. At test time, a novel unseen object is used instead. We extract local features from both the query and the templates. Using our novel IO-Layer, we compute dense correspondences between the image and the object. We then take the matches (yellow star) and further refine them using 2D and 3D techniques.

both the visible and occluded regions of the object will be represented in the template point cloud. Therefore, a single template is not enough to emulate the conditions at test-time.

To address these limitations, we sample an additional template image we refer to as the negative template $\mathcal{I}^-$. This template shares low co-visibility area with the anchor image. The keypoints extracted from $\mathcal{I}^-$ serve to generate nearly complete reconstruction of the target object, with the visible sections being sampled from $\mathcal{I}^+$ while the self-occluded ones from $\mathcal{I}^-$. This 2-template paradigm simulates the full object point cloud available at the inference time.

In order to optimize through the matching task we apply the differentiable the dual-softmax operator as proposed by LoFTR [121] and subsequently used by both OnePose and OnePose++ [122, 44].

In order to output matches, we start by using a local feature extractor, such as a ResNet [43] to extract coarse and fine level feature maps, $\hat{\mathcal{F}}_{\mathcal{I}^Q} \in \mathbb{R}^{H \times W \times \hat{C}}$ and $\tilde{\mathcal{F}}_{\mathcal{I}^Q} \in \mathbb{R}^{H \times W \times \tilde{C}}$ from the query image. We repeat the same process for both $\mathcal{I}^+$ and $\mathcal{I}^-$ and sample M points from both templates. We make sure that the template positions lie within the objects mask such that:

$$\begin{aligned} p^+ &= \{p_k^+ |\ k \in \mathcal{M}^+\}, \\ p^- &= \{p_k^- |\ k \in \mathcal{M}^-\}, \end{aligned} \tag{7.1}$$

where $\mathcal{M}$ refers to the segmentation mask. We backproject, using the a depth map only available at training time, and generate a template point ${\mathcal{P}_\mathcal{T}}^{3D} \in \mathbb{R}^{2M \times 3}$ with its corresponding extracted features $\hat{\mathcal{F}}_\mathcal{T} \in \mathbb{R}^{2M \times \hat{C}}$ and $\tilde{\mathcal{F}}_\mathcal{T} \in \mathbb{R}^{2M \times \tilde{C}}$, where $\mathcal{T}$ refers to the combined positive and negative templates.

At this point, we perform dense matching between the coarse level query features $\hat{\mathcal{F}}_{\mathcal{I}^Q}$ and the extracted point cloud $\hat{\mathcal{F}}_\mathcal{T}$. We now aim to globally match each pixel of the query image that contains the object to its respective point cloud keypoint. We make use of positional embeddings to encode positional information into each feature point. For 2D keypoints we use the sinusoidal fixed version used by DETR [14] while for 3D positional encoding we use a simple 3-layer MLP as used by SurfEmb [41] and OnePose++ [44].

We flatten both feature maps and apply self and cross attention layers following other feature matching methodologies [112, 121, 122, 44, 140] to generate more easily separable features on each set. Our objective is to construct a matrix $P^{HW \times 2M}$ that reflects the level of confidence of the correspondence

between $\mathcal{P}^{2D}$ and $\mathcal{P}^{3D}$. A score matrix $\mathcal{S}$ is computed by measuring the cosine similarity between the two sets of transformed features in a contrastive manner. We apply dual-softmax [130] over $\mathcal{S}$ to calculate the correspondence confidence matrix P:

$$P = softmax(\mathcal{S}(i,\cdot))_c \cdot softmax(\mathcal{S}(\cdot,c))_i, \tag{7.2}$$

where and $\mathcal{S} = \frac{1}{\tau}\langle \hat{\mathcal{F}}^{'}_{\mathcal{I}^Q} , \hat{\mathcal{F}}^{'}_{\mathcal{T}} \rangle$, τ being a temperature hyperparameter, with i and c being the indices of the flattened image pixels and point cloud, respectively. The image to object correspondences $\mathcal{C}$ are established by choosing only those that exceed a certain level of confidence, represented by the threshold value θ, and meet the mutual nearest neighbor (MNN) constraint to remove false matches:

$$\mathcal{C} = \{(i,c) | \forall (i,c) \in MNN(\mathcal{P}^{2D}_i, \mathcal{P}^{3D}), P_{i,c} \geq \theta\}. \tag{7.3}$$

The coarse loss $\mathcal{L}_c$ used to optimize $\mathcal{C}$ uses focal loss [75] as suggested by LoFTR [121].

7.2.2 IO-Layer: An object-image attention layer

Image to image feature matching has seen its effectiveness increase with the introduction of self and cross attention mechanisms [140, 112, 121, 81]. These layers are particularly efficient for this task seeing as both inputs are from the same modality and share the same spatial structure. For this reason, self attention layers can share weights and cross attention requires only to switch the encoding and decoding image, as you can see in Fig 7.2. OnePose++ [44] has suggested using the same approach for image to object matching however in this scenario we are working with two different modalities, a 2D image and a 3D point cloud.

We postulate that mixing modalities has an adverse effect on learning distinctive features. The positional encoding for each input are structurally different which means that attention layers will have difficulty separating the features of each modality. OnePose++ [44] shares a sign of this problem when they observe that adding 3D positional encoding results in a small ($< 1\%$) improvement. A simple ad-hoc solution would be to have modality specific weights, however that would double the amount of parameters and subsequently the amount of training data and compute. In standard self at-

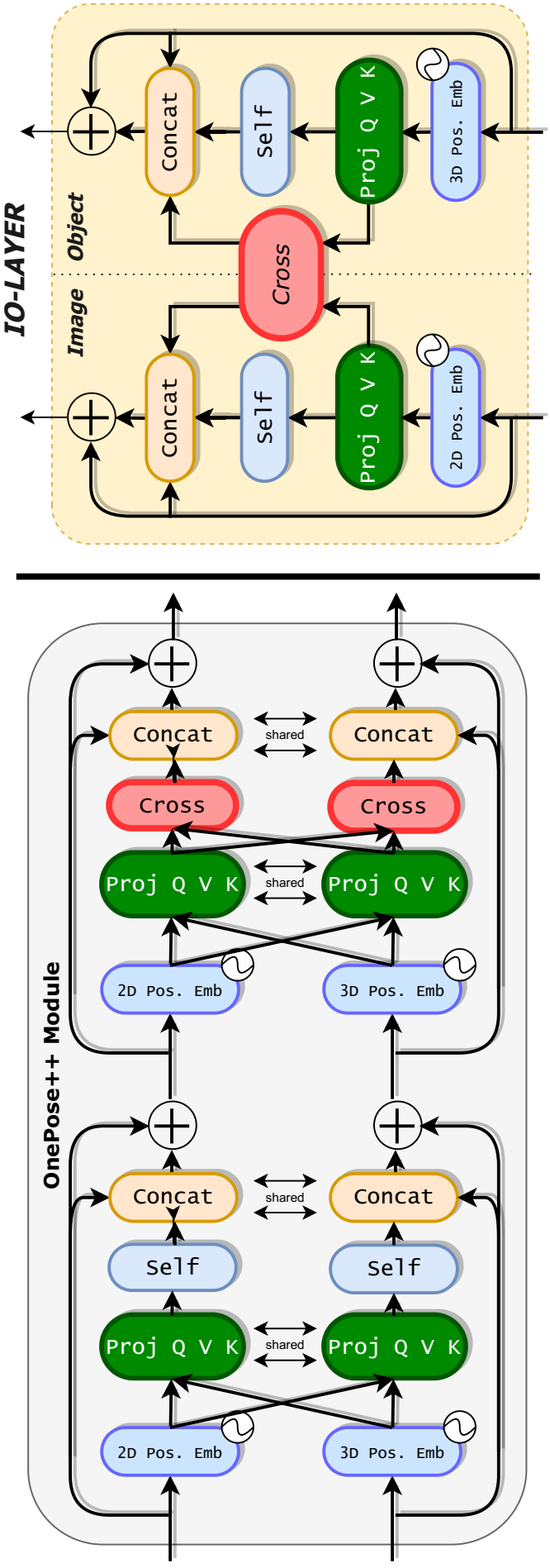


Figure 7.2: **IO-Layer architecture.** We improve on the attention modules used by LoFTR and OnePose++ [121, 122]. We have modality specific projections for image features embedded with 2D positional encoding and for object template features with its 3D positional encoding. We found that separating the two modalities improves the final pose accuracy.

tention layers an input sequence is linearly projected into a query, key and value: Q_i,K_i,V_i,stemming from the same input i. For cross attention, one would linearly project Q_i, K_j, V_j where i and j are different inputs. Usually, in attention mechanisms, the encoded message is parsed to the decoder and is not returned. However, since it is important for the image features to be distinct from the object features, and vice versa, the message must be passed bi-directionally and therefore requires careful redesign.

We propose a simple layer, specially designed for **I**mage to **O**bject matching we call **IO-Layer**. (Q_i,K_i,V_i) and (Q_o,K_o,V_o), the inputs to the attention mechanism for inputs and object respectively, are computed only once and used for both self and cross attention. Therefore cross attention between the image and the object is computed as $softmax(Q_i K_o^T/\sqrt{d})K_o$ where as for object to image is computed as $softmax(Q_o K_i^T/\sqrt{d})K_i$, requiring a single linear projection of the sequence which they both share. We modify self and cross attention accordingly to support Linear Attention [62]. Much like the layers used in LoFTR and OnePose++, these can be stacked together. We show exemplify IO-Layer on Fig.7.2.

7.2.3 Object Pruning

OnePose++ proposes sampling and using a point cloud template with over 15k keypoints per object. Performing attention over such a higher number of keypoints is expensive even if using more efficient mechanisms such as Linear Attention [62]. As a peripheral advantage, by removing a set of keypoints from the template point cloud, we reduce the complexity of the following attention layers. The set of keypoints that are not in visible in the query image should be quickly identifiable in early matching. Removing these keypoints from the point cloud allows for better separability of the features of the remaining visible keypoint, which leads to a better match. During pruning, we do not impose a hard threshold on the confidence of the matches but rather but rather we select the keypoints with higher confidences of existing in the image. Therefore, we extend the matching operation in Eq. 7.3 and sum over every pixel. We propose a pruning step after each IO-Layer. The amount of pruning is subjected to ablation studies in the Section 7.3.2.

7.2.4 Pose Refinement

As described by LoFTR [121] and OnePose++[44], a fine level 2D-refinement post-processing step was added. For each match, we crop a window at the match location and perform simple matching w.r.t. the matched keypoints. We supervise it by minimizing the Euclidean distance between the center of the grid and the projected keypoints. More details available in the suplementary or in OnePose++[122].

However, its impact is limited as we show in the experimental section. We believe this is due to the use of PnP following the keypoint position refinement. Prior 6D object pose literature has shown that estimating translation through PnP leads to poor results [74, 137]. An alternative approach, first introduced by CDPN [74], is to directly estimate the translation as the 2D offset of the object centroid as well as bounding box relative depth estimation. Directly applying this step to PoseMatcher cannot be done seeing as the objects are not known.

In order to perform translation estimation, we adopt translation refinement, which will refer to as 3D-refinement, a strategy used by iterative methods [15, 67]. Instead of estimating the translations directly, we estimate the translation errors given an initial pose. Intuitively, we estimate the necessary 2D translation and *zoom* necessary to align the initial pose to the target pose.

In order to generate the initial pose ζ^0, we perform PnP over the coarse matching keypoints. In the 2D-refinement step, the initial position of the matched keypoints are the image grid default positions. However, if we are performing a refinement over an initial pose, this alignment might be broken due to erroneous matching, pose estimation or even sub-pixel positioning. Therefore we project the matched keypoints using the initial pose and use those 2D locations as the center for the fine-sampling grid crops F^i_{crop}. We perform self and cross attention over the grids and the corresponding 3D fine feature T^i_f using a single IO-Layer. We compute the 2D expectation and supervise its output using the same methodology as described for the 2D-refinement.

We propose a lightweight model, that collects a sequence of keypoint locations and its refinement step as computed by the *3D-refinement* and output ΔT and ΔZ, the 2D location and zoom offsets respectively. Moreover, in order to supervise the refinement step, LoFTR [121] proposes computing the total variance of the matching heatmap in order to penalize more confident but erroneous estimations. We

propose reusing this variance to introduce a confidence measure to each keypoint refinement output and append to each input its corresponding refinement confidence.

We model the *zoom* as a classification problem and discrete the possible zooms in K classes [12]. We output the estimated *zoom* δ_z as the expectation over all classes and 2d translation δ_{2d} necessary to align ζ^0 with ζ^{GT} as following:

$$\begin{cases} \zeta_z^{GT} = \zeta_z^{GT} \cdot \delta_z, \\ p(\zeta^{GT}) = p(\zeta^{GT}) \ \cdot \ \delta_{2d}, \end{cases} \tag{7.4}$$

where $p(P) = \mathcal{K}\zeta[0,0,0]^T$ is the 2D projection location of the object centroid adjusted for the crop bounding box and K are the camera intrinsic parameters.

We supervise the *zoom* and 2D location offset directly with a sparse loss:

$$\begin{cases} \mathcal{L}_z = ||\delta_z - \varepsilon_z||_1 \\ \mathcal{L}_{2d} = ||\delta_{2d} - \varepsilon_{2d}||_1 \end{cases} \tag{7.5}$$

where ε is noise introduced at training time.

The full loss with the new additions becomes:

$$\mathcal{L} = \mathcal{L}_c + \mathcal{L}_f + \alpha \, (\mathcal{L}_z + \mathcal{L}_{2d}), \tag{7.6}$$

where α is a hyperparameter controlling the refinement losses relative weighting.

7.3 Experiments

In this section, a comprehensive evaluation of PoseMatcher is presented to assess the effectiveness of the method in one-shot 6D object pose estimation. The evaluation results highlight the ability of the method to generalize to unseen objects while performing comparably to supervised state-of-the-art methods. Performance comparisons were conducted against state of the art methods using widely used Linemod and YCB-V, both of which were also used in the methods introduced in Chapters 3, 4 and 6.

7.3.1 Experimental Setup

Datasets Setup. In order to train PoseMatcher from scratch we must use a sufficiently large dataset to encompass a large number of shapes and textures. We use the Google Scanned Objects[29] dataset with 1023 household objects. Additionally, we also use the renderings provided by [80] for fair comparison where each object is rendered at 250 different viewpoints. For each training instance we find a close viewpoint, $\mathcal{T}^+$ is sampled within $[5^\circ - 25^\circ]$ of rotation difference to ensure high co-visibility while being sufficiently disparate. $\mathcal{T}^-$ is sampled randomly from a set of the 5 farthest viewpoints from the query, which ensures enough data variety. We apply the standard color and noise augmentations [41, 137, 15] as well as bounding box *zoom-ins* as proposed by CDPN [74]. We also perform object level augmentations by changing the object canonical reference frame. Although Google Scanned Objects only has 1000 objects, by providing strong augmentations we are able to avoid overfiting.

Our results on Linemod and the YCB-V datasets use the standard 2D bounding boxes provided by the BOP Challenge [49]. To note these are trained using synthetic versions of the target datasets but this way we have a fair comparison to other methods that rely on the same defections.

Implementation Details. For fair comparison with OnePose++, we use 3 IO-Layers. We set the contrastive temperature term $\tau = 0.1$. For 3D-refinement we set $K = 20$ and $\alpha = 100$. During training we sample $M = 2048$ from each templates for a total of 4096 and do not prune at any stage. For refinement, we use the top P_i matches with confidence $\theta = 0.2$ up to. If not enough matches during training, we pad the pointcloud with groundtruth matches. At test time, we sample

Table 7.1: **Comparison study on Linemod.** We present the results for ADD(-S) metric and compare them to state of the art. While Linemod is close to saturated for fully supervised methods, one-shot pose estimation is still challenging. PoseMatcher achieves the best results for all objects for the one-shot category, surpassing self-supervised methods and close to fully supervised methods. Best results for one-shot are bolded. * denotes symmetric objects.

Type	Fully Supervised			Self-Supervised		One-Shot			
Method	PVNet [92]	GDR [137]	SO-Pose [25]	Self6D [136]	Sock *et. al.* [118]	Gen6D [80]	OnePose [122]	OnePose++ [44]	**PoseMatcher**
Ape	43.6	85.9	-	38.9	37.6	-	11.8	31.2	**59.2**
Benchvise	99.9	99.8	-	75.2	78.6	62.1	92.6	97.3	**98.1**
Camera	86.9	96.5	-	36.9	65.6	45.6	88.1	88.0	**93.4**
Can	95.5	99.3	-	65.6	65.6	-	77.2	89.8	**96.0**
Cat	79.3	93.0	-	57.9	52.5	40.9	47.9	70.4	**88.0**
Driller	96.4	100.	-	67.0	48.8	48.8	74.5	92.5	**98.4**
Duck	52.6	65.3	-	19.6	35.1	16.2	34.2	42.3	**54.1**
Eggbox*	99.2	99.9	-	99.0	89.2	-	71.3	**99.7**	97.8
Glue*	95.7	98.1	-	94.1	64.5	-	37.5	48.0	**91.5**
Holepuncher	81.9	73.4	-	16.2	41.5	-	54.9	69.7	**73.4**
Iron	98.9	86.9	-	77.9	80.9	-	89.2	97.4	**97.9**
Lamp	99.3	99.6	-	98.2	70.7	-	87.6	97.8	**98.1**
Phone	92.4	86.3	-	50.1	60.5	-	60.6	76.0	**92.1**
Average	86.3	91.0	96.0	58.9	60.6	-	63.6	76.9	**87.5**

16k keypoints from the available training images. We prune 50% of the point cloud after each IO-Layer, to 8k and 4k after the first two layers, respectively. For all datasets and experiments, we use AdamW [65] with cosine learning rate decay and we linearly warmup the learning rate for 5 epochs. We train PoseMatcher for 50 epochs with a batch size of 8 and an intial learning rate of 0.0001. At each epoch, we sample 10 different viewpoint queries for each of the 1k objects. The method was implemented using PyTorch [91]. All the experiments for PoseMatcher were done on a computer equipped with a GTX1080Ti.

Evaluation Metrics. To measure the performance of PoseMatcher, we employ the same metrics used by prior methods. The ADD(-S) metric considers a pose correct if the average distance to the groundtruth falls below a threshold, usually 10% of the objects diameter, with a slighly modified version for symmetric objects [148].

7.3.2 Evaluation Results

RGB vs Grayscale images Interest point descriptor models are mostly used over grayscale images as it has been shown to improve generalization. OnePose and OnePose++ [122, 44] are applied to grayscale images because they make use of pre-trained descriptor models trained solely on grayscale, SuperPoint [24] and LoFTR respectively [121]. PoseMatcher does not rely on prior methods thus is not limited to grayscale. To provide a comprehensive analysis and ensure thoroughness we train our

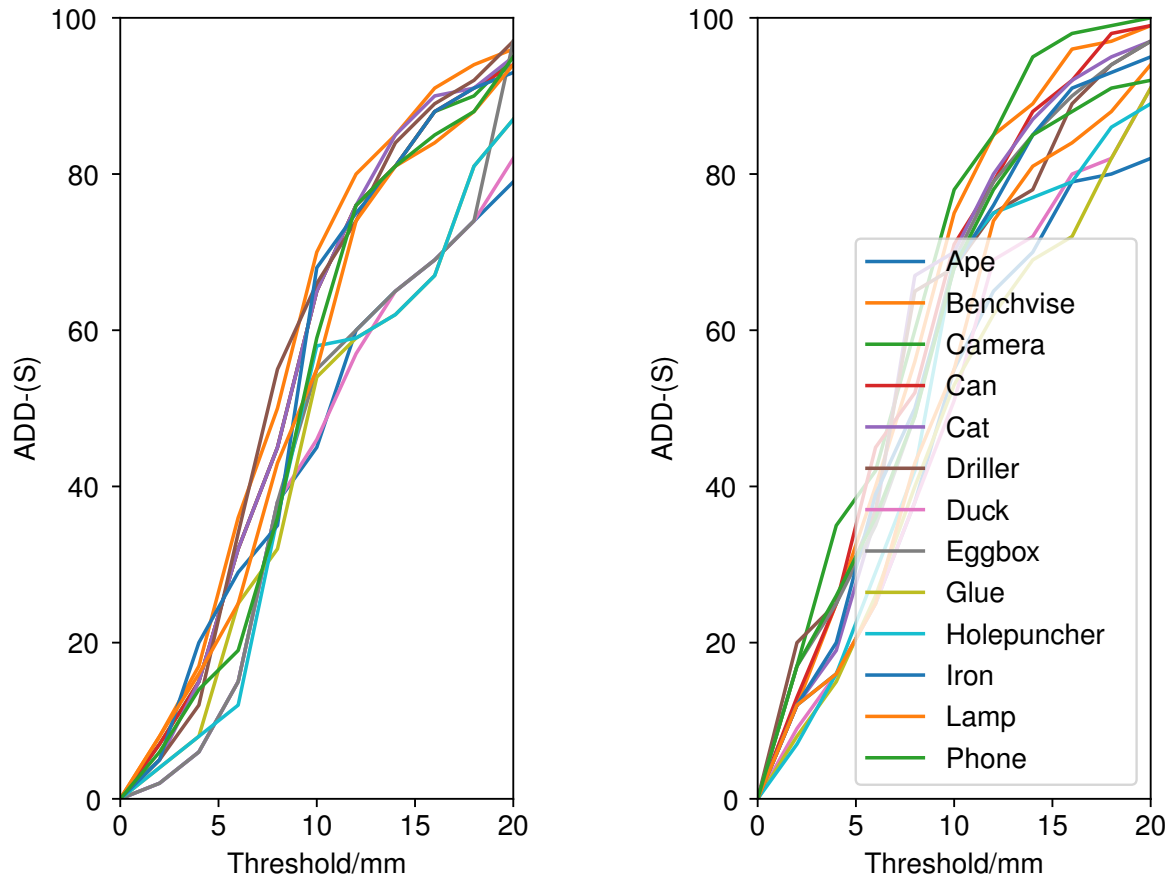
Table 7.2: **Ablation studies for each component.** We present the results on Linemod and YCB-V measured by ADD-(S). The first line are the results gathered from OnePose++ [122] The model used for the second and third rows refer is identical to the OnePose++ [44] model trained with our three-view based pipeline.

Three-View	IO-Layer	Pruning	2D-Ref	3D-Ref	Grayscale	Linemod	YCB-V
			✓			76.9	-
✓						81.1	22.1
✓			✓			81.4	22.5
✓	✓					83.4	24.1
✓	✓		✓			83.4	24.2
✓	✓	✓	✓			83.9	25.8
✓	✓	✓		✓		**86.1**	**30.1**
✓	✓	✓		✓	✓	84.1	27.4

method on both RGB and grayscale input. Intuitively, RGB inputs should provide more information and allow PoseMatcher to better separate textured regions of an object as well as from the background. Surprisingly, we found that using RGB data improves our method. Grayscale only reaches an ADD-(S) accuracy of 84.1% on Linemod while RGB inputs reach an accuracy 86.1%. All our experiments and ablations are with RGB inputs.

We have a large advantage over DeepIM [73] which needs a strong pose initialization from external pose estimators, which for its specific case the authors chose PoseCNN [148], whereas PoseMatcher does not require any type of pose initialization or the 3D mesh model. We outperform every existing one-shot method by a significant margin, specially on more complicated objects such as the Ape or the Cat. It is interesting to observe that we also outperform PVNet[92], an instance-level pose estimator.

Ablation Studies Our new training pipeline elevates the results from OnePose++ [44]. By learning meaningful descriptors specifically trained for pose estimation instead of pretrained ones, we are able increase the quality of the feature matching thereby improving the pose accuracy by 5%. This confirms that carefully designing a training pipeline can yield better results. The IO-Layer, our novel attention based layer, yields an improvement of 2% on both datasets. While it does not increase runtime performance when compared to OnePose++ [122], we can see an improvement stemming from the use of specialized modality weights. Replacing 2D based refinement with pose specific 3D-Refinement improved our results significantly. We found improvements in both Linemod and the YCB-V datasets. The improvement cannot be solely observed by measuring the ADD-(S) since the average threshold for Linemod is $15mm$. We provide further information for Linemod on Fig. 7.3



(a) 2D-refinement

(b) 3D-refinement

Figure 7.3: **Difference between refinement operations.** We can see that for all objects in Linemod, the 3D-refinement has a big impact on lower thresholds.

where the error curves for each object in Linemod can be seen. The 3D-refinement has a much larger impact on lower thresholds accuracies than higher ones. An addition like 3D-refinement is invaluable for applications that require high precision.

We found 2D-refinement to result in no significant improvements. This step does not correct significant matching errors and PnP seems to overcome small imperfections stemming from resolution errors. PnP-based estimation is also subject to biasing for translation as observed by prior works [137, 74]. Our results on 3D-refinement show a large increase in the translation accuracy when we use 3D-refinement over 2D-refinement, which is also reflected on ADD-(S).

Object pruning ablations. As discussed before, pruning an object provides two advantages. Firstly, it eliminates self-occluded regions of the object which removes possible noise during feature matching. Secondly, it reduces the complexity of the attention mechanism by decreasing the amount of

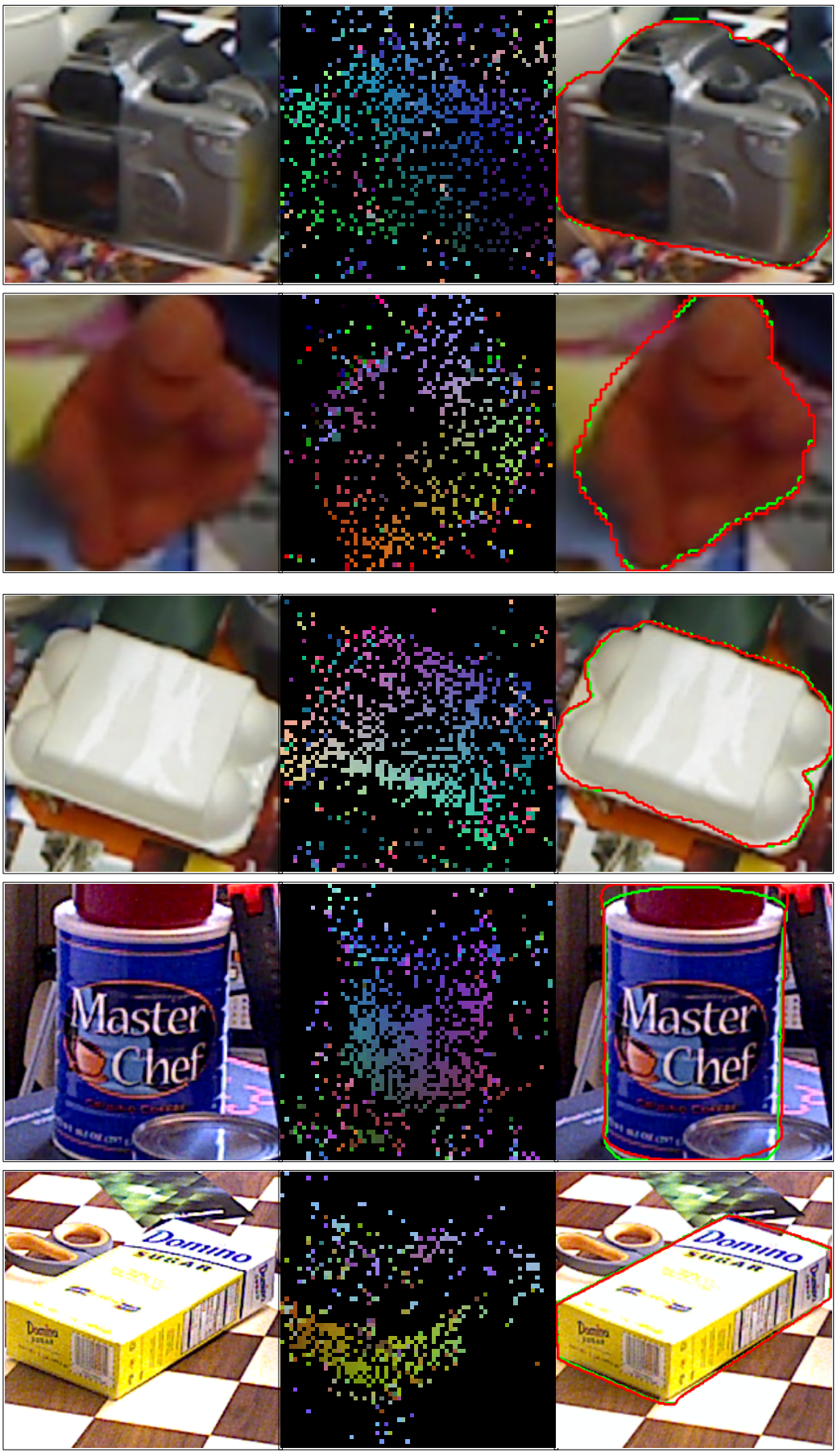


Figure 7.4: **Visualization of matching keypoints.** In this figure, we show PoseMatcher output $\mathcal{C}$, where we assigned a normalized coordinate of the highest confidence keypoint to the corresponding pixel. The last two rows are examples from the YCB-V dataset.

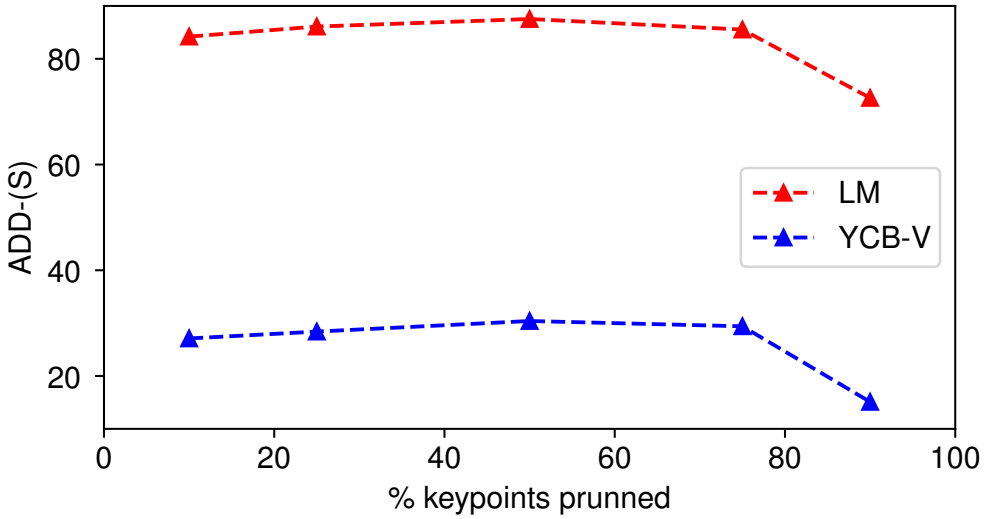


Figure 7.5: **Pruning ablations.** On this ablation study, we show that PoseMatcher does not degenerate when aggressively pruned. Since keypoints have very distinct descriptors, PoseMatcher can early on remove self or externally occluded keypoints or keypoints that fall on a textureless area.

keypoints needed to be *attented*, which leads to faster runtime. However, pruning a large amount of keypoints leads to poorer results. We perform inference time ablation studies to help us determine the optimal pruning percentage and plot the results of percentages ranging from 10% to 90% after each IO-Layer in Fig 7.5 and measure the ADD-(S) accuracy on Linemod. If a low amount of keypoints are removed, the performance is not highly affected. However, we see a small drop at 75% with a large drop when crop 90% of the existing pointcloud. The latter might be due to the number of initial keypoints as the number of keypoints are just 150 points. Although increasing the number of pruned keypoints leads to higher throughput, the peak accuracy results is at 50%.

Number of templates used. For all our experiments on Linemod, we have been using all the available training images in order to sample template keypoints. Due to our training pipeline, PoseMatcher is robust to incomplete and noisy templates. We perform an ablation study over the number of templates needed. Since each object in Linemod contains different number of training images, we present the percentage of training images used. To sample which images to use, we sample from the available viewpoints using furthest point sampling [92] w.r.t. orientations in order to cover the widest range of viewpoints as possible. In other words, on the Linemod dataset, CRT-6D almost achieves the same level of accuracy as OnePose++ [44] using only 75% of the available training images.

Table 7.3: **Number of templates.** We present the results on Linemod for % of training images used. Each object has around 180 images therefore 10% is only 18 templates to use.

%	Ape	BW	Camera	Can	Cat	Driller	Duck	EB*	Glue*	HP	Iron	Lamp	Phone	Avg
10	2.1	15.1	8.8	0.0	1.9	6.4	0	25	2.5	0	12.4	2.1	12.4	6.8
25	5.9	24.1	15.2	12.4	13.1	12.4	1.2	45.1	8.3	0	24.4	12.6	29.1	15.7
50	12.4	59.4	35.6	19.7	19.3	34.7	9.4	78.9	41.9	5.7	34.1	21.8	57.3	33.1
75	39.9	78.9	81.2	76.6	57.3	65.1	27.1	92.1	72.1	46.8	87.1	79.	88.5	68.6
100	59.2	98.1	93.4	96	88.0	98.4	54.1	97.8	91.5	73.4	97.9	98.1	92.1	87.5

7.4 Conclusion

In this chapter, the topic of unseen pose estimation was tackled from a practical perspective overcoming shortcomings stemming not only from the inherent challenges of the task, but also from the shortcomings of previous works. To this end, a novel image based model free one-shot pose estimator was proposed and given the name of PoseMatcher as it is built on deep feature matching techniques. Pre-trained descriptors were avoided by using an efficient training pipeline relying on a triplet of views, one serving as the query image and the rest for sparsely recreating the object, allowing for high data throughput during training. It is shown that with this single addition, we could improve the state-of-the-art OnePose++ [44], validated on the BOP benchmark datasets. Moreover, we showed improvements when applying the novel IO-Layer, a feature matching transformer based layer specially designed for image to object keypoint matching, as well as a customized 3D refinement module inspired by the CRT-6D pose refinement process, described in the previous chapter. Furthermore, we investigated the impact of utilizing different amounts of real-world data and explored the acceleration of inference through point cloud dynamic pruning. We found that PoseMatcher exhibits the potential for practical deployment in real-world applications.

Chapter 8

Conclusion and Future Work

This thesis has presented research contributions aimed at accelerating and extending 6D object pose estimators to multi-object for real-world adaptability. This chapter provides a comprehensive summary of these contributions and examines their impact on the field of 6D object pose estimation. A comprehensive summary of each contribution is provided, highlighting their distinct impacts and analyzed their limitations. Afterwards we discuss prospective future directions for extending the current state of 6D object pose estimation.

8.1 Contributions Overview

In Chapter 3, we introduced a new method for instance-level 6D object pose, which takes full advantage of the target object's topological information. This method extracts both the object's mask and 2D centroid projection with high resolution to precisely detect the object in 2D. We perform estimation in 3D space by reconstructing a 3D pose conditioned mesh of the target from which we recover its orientation by comparing it with the canonical version of the object's 3D mesh. Our method outperforms state-of-the-art on the standard 6D object pose dataset LINEMOD such as BB8 [98] and Tekin [128]. Additionally, we present our method's results on more challenging datasets, OCCLUSION and YCB-Video, outperforming previous non-occlusion aware approaches SSD-6D [63] and PoseCNN [148] while being comparable to methods optimized for occlusion. We also show how we can use the reconstruction error as an indicator of pose estimation precision which allows the model

to self validate at test time, which is crucial for real life applications. This method is currently limited by its translation accuracy, or more specifically, its depth estimation. We show that if ground truth mask segmentation and centroid projections are available, our method can be extensively improved, specially when the former is provided. By relying on a segmentation map, truncation and external occlusion leads to pose estimation failure. Future work would include a more robust occlusion-aware 2D localization method as well as reducing the impact of occlusion on the mesh reconstruction process.

Chapter 4 introduces Multi-Frustums Network, or MFNet, an innovative 6D object pose estimation approach designed to address translation errors commonly encountered in preceding 6D methods. By incorporating a novel intermediate representation, known as the Multi-Frustum grid, for 3D-3D correspondence, the model is enabled to optimize for translation with greater ease. When integrated into an efficient two-stage pose estimator pipeline, MFNet not only surpasses the previous state-of-the-art, GDR-Net [137] and CDPN [74], in accuracy but also achieves a runtime that is twice as fast as earlier methods. This optimization comes at the cost of scalability seeing as MFNet is limited by its single-object network framework, making it less adaptable for multi-object pose estimation tasks. MFNet is limited by the fact that it cannot scale well to multi-objects due to the large size of its output grid. *Ad-hoc* approaches would overparameterize the model and it would become unfeasible to train with the available hardware. Future works could include designing a object pose estimation specific conditional parametrization in order to incorporate detection level classification knowledge into the intermediate representation estimation. On another note, since MFNet uses NOCS [138] based representations, one could extend MFNet to category and possible achieve the fastest category level pose estimator.

In Chapter 5, we aim to provide solutions to the multi-object model extension problem. We presented multi-object model compression with Conditional Instance Normalization (CIN) parameterization and multi-teacher distillation techniques to achieve high performance multi-object pose estimator with minimal increase when compared to one-object model capacity. Rather than distilling information from a larger teacher model, we use multiple object specific teachers in order to recycle existing singe-object pose estimation methods, which decreases computational costs. By using CIN, we can leverage the knowledge of the object class prior to pose estimation. We condition the network using conditional instance normalization parameterization to soften the complexity of keypoint detection

by dynamically altering the *style* of the image [54, 104]. Our evaluations show the effectiveness of the proposed framework on multiple datasets and the improvement over *ad-hoc* multi-object pose estimators such as the work by Li et al. [69]. Not only did we present evaluations on multiple datasets, but we also showed that our multi-object extension works on both fully supervised and self-supervised methods. Compared to prior art, we can achieve 90% compression and still outperform previous methods. As future work, we could look to extend Conditional Instance Normalization to category or unseen instances of objects, seeing as the parameterization has shown it can alleviate inter-class appearance task complexities.

In Chapter 6, we have proposed a novel approach to 6D object pose estimation, named CRT-6D, based on iterative pose refinements. The input to our refinement modules is a set of feature vectors OSKF, sampled from feature pyramid at the location of known keypoints, 2D projected using a coarse pose. OSKF are the representation of the pose offset representation and are fed into an OSKF-PoseTransformer to extract the refined pose. We achieve state of the art on multiple datasets, when compared to existing methods such as GDR-Net [137], CDPN [74] and SO-Pose [25], while being over $2\times$ faster. For future directions, we can look to extend the use of keypoints to represent an object. We showed can learn them through a fully supervised method, however it leaves the door open to finding a method where OSKF could be accurately extracted at test-time from a set of templates, making this method able to extract the pose of novel unseen objects.

In Chapter 7, we proposed PoseMatcher, a novel model free one-shot pose estimator based on deep feature matching. In order to avoid using pre-trained descriptor models, we introduce a new training pipeline that allows us to train from scratch, based on a query/positive/negative template triplet. With this simple addition to the training pipeline, we show that we can improve upon OnePose++ [44] without any additional modification. We build on top of OnePose++ by designing a new attention layer IO-Layer, designed specifically for image to object matching. Additionally, we propose improvements to the pipeline including object pruning and 3D based refinement. PoseMatcher achieves state of the art results when compared to similar one-shot methods such as OnePose [122], OnePose++ [44] and Gen6D [80], and also surpasses some fully supervised methods, for instance PVNet [92]. Unfortunately we found that PoseMatcher has limited domain adaptation capability. If there is a large domain gap between the template domain (ex. synthetic renderings) and queries PoseMatcher is un-

able to correctly match keypoints, even coarsely. We specifically find this on YCB-V where each scene contains different levels of sensor noise and illumination. We also found symmetric objects particularly difficult, even though object pruning did show improvements. Looking forward, incorporating a domain adaptation pipeline or module could serve as a potential future direction. This enhancement may lead to PoseMatcher becoming solely dependent on synthetic data, even during feature template extraction at inference time.

8.2 Future Directions

While the methods described in this thesis were aimed at solving on going challenges in pose estimation, much still remains to be studied. One area which is very overlooked but an integral part of real world industrial application is 6D object tracking. Some current RGB based approaches [110, 143] rely on a *render-compare*, very similar to DeepIM [73], which as discussed in this thesis, is a very expensive pipeline. The proposed refinement module of CRT-6D and the novel pose representation of projected OSKF, we opened the doors to real-time 6D object tracking through iterative refinements. Direct pose estimation methods, while fast, leave perspective deformation issues as an afterthought. Our work described in Chapter 4 and Park and Cho [88] aim to reduce this deformation by way of reprojection. It remains an open question whether or not a direct pose estimation method could be created capable of explicitly addressing this issue, rather than regarding it as a shape noise inherent to the problem at hand. One-shot pose estimation breakthroughs are leading to new questions. PoseMatcher, proposed in this thesis, has surpassed instance-level methods that were until very recently state-of-the-art. Although it is still outperformed by current instance-level estimators, one must question if the gap can be closed. If all the information from the training set can be efficiently and completely extracted into multiple local and/or single global descriptor, it could be argued that a one-shot method could absorb the same information as an instance-level method. In this case, we would expect the gap to close and instance-level pose estimation to be considered a solved problem. However, research into how to efficiently extract information from templates is still in its early stages, so we expect instance-level pose estimators to reign over the BOP [49] leaderboard for the foreseeable future. Another interesting research path in the one-shot paradigm is the acceleration of the current iteration of methods. Current

one-shot methods rely on a matching step [122, 44, 80], including PoseMatcher, that although it adds a novel 3D-refinement step, it still performs coarse level keypoint matching, which slows down the method considerably. Moreover, due to their design, we cannot optimize the models directly through pose. In order to see full adoption of the one-shot paradigm models, either an acceleration or an end-to-end optimization of these methods is necessary. A GDR-Net [137] style breakthrough where one could retain part of the previous design and append a learnable match-to-pose module would provide such upgrade.

8.3 Epilogue

As the current world witnesses an exponential increase in the reliance on automation, it becomes of great importance to understand its implications and trajectory. The industrial revolution served as a catalyst for advancements in machinery and robotics, yet it is during the current age of information that the true potential of robotic automation has begun to manifest. Within the digital realm, automation has reached a state of dominance, rendering virtually all software processes automated. At the time of writing this thesis and to the best of the author's knowledge, humans have not developed the necessary technology to digitize themselves, therefore intelligent and automated software systems still require a physical medium to interact with us. The integration of software with real-world sensory systems presents a complex undertaking. While physical interaction is accompanied by its own set of challenges, it predominantly depends on perceptual information from which it derives conclusions. In the event of visual perceptual failure, the efficacy of robotic interaction is compromised. This dissertation presents solutions to the perceptual obstacles that hinder the deployment of pose estimators in real-world applications. It is the hope of the author that the methods proposed in this dissertation will serve as a foundation for future research endeavors, as the author too, has benefited from the intellectual contributions of those who came before him.

Bibliography

[1] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. DensePose: Dense human pose estimation in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[2] Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. Yolopose: Transformer-based multi-object 6d pose estimation using keypoint regression. In *International Conference on Intelligent Autonomous Systems (IAS)*, 2022.

[3] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. Pushing the envelope for rgb-based dense 3d hand pose estimation via neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] Vassileios Balntas, Andreas Doumanoglou, Caner Sahin, Juil Sock, Rigas Kouskouridas, and Tae-Kyun Kim. Pose guided RGBD feature learning for 3D object pose estimation. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[5] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key. net: Keypoint detection by handcrafted and learned cnn filters. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[6] Vasileios Belagiannis, Azade Farshad, and Fabio Galasso. Adversarial network compression. In *European Conference on Computer Vision (ECCV)*, 2018.

[7] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1992.

[8] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[9] Hakan Bilen and Andrea Vedaldi. Universal representations: The missing link between faces, text, planktons, and cat breeds. *arXiv preprint arXiv:1701.07275*, 2017.

[10] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[11] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[12] Dingding Cai, Janne Heikkilä, and Esa Rahtu. Sc6d: Symmetry-agnostic and correspondence-free 6d object pose estimation. In *International Conference on 3D Vision (3DV)*, 2022.

[13] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. In *IEEE International Conference on Advanced Robotics (ICAR)*, 2015.

[14] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, 2020.

[15] Pedro Castro and Tae-Kyun Kim. Crt-6d: Fast 6d object pose estimation with cascaded refinement transformers. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.

[16] Pedro Castro and Tae-Kyun Kim. Posematcher: One-shot 6d object pose estimation by deep feature matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2023.

[17] Pedro Castro, Anil Armagan, and Tae-Kyun Kim. Accurate 6d object pose estimation by pose conditioned mesh reconstruction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[18] Bo Chen, Alvaro Parra, Jiewei Cao, Nan Li, and Tat-Jun Chin. End-to-end learnable geometric vision by backpropagating pnp optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[19] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[20] Hanzhi Chen, Fabian Manhardt, Nassir Navab, and Benjamin Busam. Texpose: Neural texture learning for self-supervised 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[21] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[22] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[23] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc, 2019.

[24] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018.

[25] Yan Di, Fabian Manhardt, Gu Wang, Xiangyang Ji, Nassir Navab, and Federico Tombari. So-pose: Exploiting self-occlusion for direct 6d pose estimation. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2021.

[26] T Do, Trung Pham, Ming Cai, and Ian Reid. Real-time monocular object instance 6d pose estimation. In *British Machine Vision Conference (BMVC)*, 2019.

[27] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*, 2021.

[28] Andreas Doumanoglou, Rigas Kouskouridas, Sotiris Malassiotis, and Tae-Kyun Kim. Recovering 6d object pose and predicting next-best-view in the crowd. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[29] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

[30] Bertram Drost, Markus Ulrich, Nassir Navab, and Slobodan Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

[31] Bertram Drost, Markus Ulrich, Paul Bergmann, Philipp Hartinger, and Carsten Steger. Introducing mvtec itodd-a dataset for 3d object recognition in industry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) workshops*, 2017.

[32] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *International Conference on Learning Representations (ICLR)*, 2016.

[33] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, Paste and Learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[34] Yang Fu and Xiaolong Wang. Category-level 6d object pose estimation in the wild: A semi-supervised learning approach and a new dataset. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[35] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3D hand shape and pose estimation from a single RGB image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[36] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2015.

[37] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *International Conference on Machine Learning (ICML)*, 2015.

[38] Shreyas Hampali, Sayan Deb Sarkar, Mahdi Rad, and Vincent Lepetit. Keypoint transformer: Solving joint identification in challenging hands and object interactions for accurate 3d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[39] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[40] Song Han, Jeff Pool, Sharan Narang, Huizi Mao, Enhao Gong, Shijian Tang, Erich Elsen, Peter Vajda, Manohar Paluri, John Tran, Bryan Catanzaro, and William J. Dally. Dsd: dense-sparse-dense training for deep neural networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[41] Rasmus Laurvig Haugaard and Anders Glent Buch. Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[42] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[43] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image

recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[44] Xingyi He, Jiaming Sun, Yuang Wang, Di Huang, Hujun Bao, and Xiaowei Zhou. Onepose++: Keypoint-free one-shot object pose estimation without CAD models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[45] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2011.

[46] Stefan Hinterstoisser, Vincent Lepetit, Naresh Rajkumar, and Kurt Konolige. Going further with point pair features. In *European Conference on Computer Vision (ECCV)*, 2016.

[47] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *Deep Learning and Representation Learning Workshop, NIPS*, 2015.

[48] Tomáš Hodaň, Pavel Haluza, Štěpán Obdržálek, Jiří Matas, Manolis Lourakis, and Xenophon Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[49] Tomáš Hodaň, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiří Matas, and Carsten Rother. Bop: Benchmark for 6d object pose estimation. *European Conference on Computer Vision (ECCV)*, 2018.

[50] Tomáš Hodaň, Vibhav Vineet, Ran Gal, Emanuel Shalev, Jon Hanzelka, Treb Connell, Pedro Urbina, Sudipta N Sinha, and Brian Guenter. Photorealistic image synthesis for object instance detection. In *2019 IEEE international conference on image processing (ICIP)*, 2019.

[51] Tomas Hodan, Daniel Barath, and Jiri Matas. Epos: Estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[52] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[53] Lin Huang, Jianchao Tan, Ji Liu, and Junsong Yuan. Hand-transformer: Non-autoregressive structured modeling for 3d hand pose estimation. In *European Conference on Computer Vision (ECCV)*, 2020.

[54] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[55] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2016.

[56] Daniel P Huttenlocher, William J Rucklidge, and Gregory A Klanderman. Comparing images using the hausdorff distance under translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1992.

[57] A S M Iftekhar, Hao Chen, Kaustav Kundu, Xinyu Li, Joseph Tighe, and Davide Modolo. What to look at and where: Semantic and spatial refined transformer for detecting human-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[58] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M. Kitani. Repose: Fast 6d object pose refinement via deep texture rendering. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2021.

[59] Thomas Georg Jantos, Mohamed Amin Hamdad, Wolfgang Granig, Stephan Weiss, and Jan Steinbrener. Poet: Pose estimation transformer for single-view, multi-object 6d pose estimation. In *Conference on Robot Learning*, 2023.

[60] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative

adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[61] Roman Kaskman, Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[62] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning (ICML)*, 2020.

[63] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[64] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2015.

[65] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[66] Abhijit Kundu, Yin Li, and James M Rehg. 3D-RCNN: Instance-level 3D object reconstruction via render-and-compare. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[67] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision (ECCV)*, 2020.

[68] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International Journal of Computer Vision (IJCV)*, 2009.

[69] Chi Li, Jin Bai, and Gregory D Hager. A unified framework for multi-view multi-class object pose estimation. In *European Conference on Computer Vision (ECCV)*, 2018.

[70] Fu Li, Hao Yu, Ivan Shugurov, Benjamin Busam, Shaowu Yang, and Slobodan Ilic. Nerf-pose: A first-reconstruct-then-regress approach for weakly-supervised 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[71] Hui Li, Zidong Guo, Seon-Min Rhee, Seungju Han, and Jae-Joon Han. Towards accurate facial landmark detection via cascaded transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[72] Wenhao Li, Hong Liu, Hao Tang, Pichao Wang, and Luc Van Gool. Mhformer: Multi-hypothesis transformer for 3d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[73] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[74] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[75] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[76] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2019.

[77] Ming-Yu Liu, Oncel Tuzel, Ashok Veeraraghavan, and Rama Chellappa. Fast directional chamfer matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[78] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang

Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, 2016.

[79] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Association for Computational Linguistics (ACL)*, 2019.

[80] Yuan Liu, Yilin Wen, Sida Peng, Cheng Lin, Xiaoxiao Long, Taku Komura, and Wenping Wang. Gen6d: Generalizable model-free 6-dof object pose estimation from rgb images. In *European Conference on Computer Vision (ECCV)*, 2022.

[81] Xiaoyong Lu, Yaping Yan, Bin Kang, and Songlin Du. Paraformer: Parallel attention transformer for efficient feature matching. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2023.

[82] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[83] Fabian Manhardt, Diego Martin Arroyo, Christian Rupprecht, Benjamin Busam, Tolga Birdal, Nassir Navab, and Federico Tombari. Explaining the ambiguity of object detection and 6d pose from visual data. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*.

[84] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[85] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[86] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In *European Conference on Computer Vision (ECCV)*, 2018.

[87] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[88] Jaewoo Park and Nam Ik Cho. Dprost: 6-dof object pose estimation using space carving and dynamic projective spatial transformer. *arXiv preprint arXiv:2112.08775*, 2021.

[89] Keunhong Park, Arsalan Mousavian, Yu Xiang, and Dieter Fox. Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[90] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[91] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2019.

[92] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[93] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[94] Giorgia Pitteri, Slobodan Ilic, and Vincent Lepetit. Cornet: generic 3d corners for 6d pose estimation of new objects without retraining. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV) Workshops*, 2019.

[95] Giorgia Pitteri, Aurélie Bugeau, Slobodan Ilic, and Vincent Lepetit. 3d object detection and

pose estimation of unseen objects in color images with local surface embeddings. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2020.

[96] Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization. In *International Conference on Learning Representations (ICLR)*, 2018.

[97] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[98] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2017.

[99] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, 2018.

[100] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[101] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[102] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[103] 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 (NeurIPS)*, 2015.

[104] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[105] Iasonas Kokkinos Riza Alp Guler, Natalia Neverova. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[106] T Roddick, A Kendall, and R Cipolla. Orthographic feature transform for monocular 3d object detection. In *British Machine Vision Conference (BMVC)*, 2019.

[107] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *International Conference on Learning Representations (ICLR)*, 2015.

[108] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015.

[109] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International Journal of Computer Vision (IJCV)*, 2006.

[110] Denys Rozumnyi, Jiří Matas, Marc Pollefeys, Vittorio Ferrari, and Martin R Oswald. Tracking by 3d model estimation of unknown objects in videos. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2023.

[111] Muhamad Risqi U Saputra, Pedro PB de Gusmao, Yasin Almalioglu, Andrew Markham, and Niki Trigoni. Distilling knowledge from a deep pose regressor network. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[112] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[113] Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 1966.

[114] Xuepeng Shi, Qi Ye, Xiaozhi Chen, Chuangrong Chen, Zhixiang Chen, and Tae-Kyun Kim. Geometry-based distance decomposition for monocular 3d object detection. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2021.

[115] Ivan Shugurov, Fu Li, Benjamin Busam, and Slobodan Ilic. Osop: A multi-stage one shot object pose estimation framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[116] Juil Sock, Kwang In Kim, Caner Sahin, and Tae-Kyun Kim. Multi-task deep networks for depth-based 6d object pose and joint registration in crowd scenarios. In *British Machine Vision Conference (BMVC)*, 2018.

[117] Juil Sock, Pedro Castro, Anil Armagan, Guillermo Garcia-Hernando, and Tae-Kyun Kim. Tackling two challenges of 6d object pose estimation: Lack of real annotated rgb images and scalability to *arXiv preprint arXiv:2003.12344*, 2020.

[118] Juil Sock, Guillermo Garcia-Hernando, Anil Armagan, and Tae-Kyun Kim. Introducing pose consistency and warp-alignment for self-supervised 6d object pose estimation in color images. In *2020 International Conference on 3D Vision (3DV)*, 2020.

[119] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. Adaptis: Adaptive instance selection network. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[120] Yongzhi Su, Mahdi Saleh, Torben Fetzer, Jason Rambach, Nassir Navab, Benjamin Busam, Didier Stricker, and Federico Tombari. Zebrapose: Coarse to fine surface encoding for 6dof object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[121] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[122] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. OnePose: One-shot object pose estimation without CAD models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[123] Martin Sundermeyer, Maximilian Durner, En Yen Puang, Zoltan-Csaba Marton, Narunas Vaskevicius, Kai O Arras, and Rudolph Triebel. Multi-path learning for object pose estimation across domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[124] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3D orientation learning for 6D object detection from RGB images. In *European Conference on Computer Vision (ECCV)*, 2018.

[125] Supasorn Suwajanakorn, Noah Snavely, Jonathan J Tompson, and Mohammad Norouzi. Discovery of latent 3D keypoints via end-to-end geometric reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[126] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[127] Alykhan Tejani, Danhang Tang, Rigas Kouskouridas, and Tae-Kyun Kim. Latent-class hough forests for 3d object detection and pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[128] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[129] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[130] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[131] Vincent Vanhoucke, Andrew Senior, and Mark Z. Mao. Improving the speed of neural networks on cpus. In *Deep Learning and Unsupervised Feature Learning Workshop, NIPS*, 2011.

[132] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[133] Joel Vidal, Chyi-Yeu Lin, and Robert Martí. 6d pose estimation using an improved method based on point pair features. In *4th International Conference on Control, Automation and Robotics (ICCAR)*, 2018.

[134] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Dense 3D regression for hand pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[135] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. 2019.

[136] Gu Wang, Fabian Manhardt, Jianzhun Shao, Xiangyang Ji, Nassir Navab, and Federico Tombari. Self6D: Self-supervised monocular 6d object pose estimation. In *European Conference on Computer Vision (ECCV)*, 2020.

[137] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[138] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J. Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[139] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2Mesh: Generating 3D mesh models from single RGB images. In *European Conference on Computer Vision (ECCV)*, 2018.

[140] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. Matchformer: Interleaving attention in transformers for feature matching. In *Asian Conference on Computer Vision (ACCV)*, 2022.

[141] Xintao Wang, Yu Li, Honglun Zhang, and Ying Shan. Towards real-world blind face restoration with generative facial prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[142] Zhixin Wang and Kui Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019.

[143] Bowen Wen, Chaitanya Mitash, Baozhang Ren, and Kostas E Bekris. se (3)-tracknet: Data-driven 6d pose tracking by calibrating image residuals in synthetic domains. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[144] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[145] Yilin Wen, Xiangyu Li, Hao Pan, Lei Yang, Zheng Wang, Taku Komura, and Wenping Wang. Disp6d: Disentangled implicit shape and pose learning for scalable 6d pose estimation. In *European Conference on Computer Vision (ECCV)*, 2022.

[146] Less Wright. Ranger-Deep-Learning-Optimizer, 2019. URL `https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer`.

[147] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, 2018.

[148] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. *Robotics: Science and Systems (RSS)*, 2018.

[149] Yan Xu, Kwan-Yee Lin, Guofeng Zhang, Xiaogang Wang, and Hongsheng Li. Rnnpose: Recurrent 6-dof object pose refinement with robust correspondence field estimation and pose optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[150] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[151] Frank Yu, Mathieu Salzmann, Pascal Fua, and Helge Rhodin. Pcls: Geometry-aware neural reconstruction of 3d pose with perspective crop layers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[152] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[153] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE/CVF international Conference on Computer Vision (ICCV)*, 2019.

[154] Kaifeng Zhang, Yang Fu, Shubhankar Borse, Hong Cai, Fatih Porikli, and Xiaolong Wang. Self-supervised geometric correspondence for category-level 6d object pose estimation in the wild. In *International Conference on Learning Representations (ICLR)*, 2023.

[155] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[156] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[157] Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. On the continuity of rotation representations in neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[158] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2020.

[159] Lu Zou, Zhangjin Huang, Naijie Gu, and Guoping Wang. 6d-vit: Category-level 6d object pose estimation via transformer-based instance representation learning. *IEEE Transactions on Image Processing*, 2022.