Deep Models for Rigid Objects Real-Time Pose Estimation

by

Jianyu Zhao

A thesis submitted in partial fulfilment for the requirements for the degree of Doctor of Philosophy at the University of Central Lancashire

January 2024

RESEARCH STUDENT DECLARATION FORM

Type of Award **Doctor of Philosophy**

School **School of Engineering and Computing**

1. Concurrent registration for two or more academic awards

I declare that while registered as a candidate for the research degree, I have not been a registered candidate or enrolled student for another award of the University or other academic or professional institution.

2. Material submitted for another award

I declare that no material contained in the thesis has been used in any other submission for an academic award and is solely my own work.

3. Collaboration

Where a candidate's research programme is part of a collaborative project, the thesis must indicate in addition clearly the candidate's individual contribution and the extent of the collaboration. Please state below:

This work described in the thesis is entirely the candidate's work.

4. Use of a Proof-reader

No proof-reading service was used in the compilation of this thesis.

Signature of Candidate Jianyu Zhao

Print name: **Jianyu Zhao**

Acknowledgements

First and foremost, I would like to express my deepest gratitude to Professor Bogdan Matuszewski, my Director of Studies, for his invaluable guidance, support, and encouragement throughout my PhD journey. His expertise, insight, and feedback have been instrumental in shaping my research and in helping me overcome numerous challenges.

I am also profoundly thankful to my second supervisor, Dr Wei Quan, for his insightful comments and thoughtful suggestions, which significantly contributed to the refinement and enhancement of my work.

My heartfelt appreciation extends to my colleagues and seniors, Dr Edward Sanderson, Dr Liyang Wang, Dr Yunbo Guo, Dr Malcolm Peacock, Mr Kerr Fitzgerald, Mr Bowen Han, Mr Jingdong Li, and Mr Ziyu Ye. I am grateful for the stimulating intellectual discussions and the collective support provided during difficult times.

Finally, I dedicate this thesis to my dear wife Lu Dong - the most beautiful, kind, lovely, and perfect lady in the world, my greatest source of motivation. Thank you for standing by me, for your unwavering support and love throughout this journey, even during the most challenging and stressful times. Your encouragement, companionship, and constant belief in me are always a constant source that keeps me moving forward.

Abstract

Accurate and robust six degrees of freedom (6-DoF) pose estimation of rigid objects is one of the fundamental tasks in computer vision, with wide-ranging applications that span industrial automation, augmented reality, and medical intervention. However, most existing methods typically rely on knowledge of objects' 3D models and depth measurements, and often require time-consuming iterative refinement to improve accuracy, which can be seen as limiting factors for broader applications.

This PhD thesis is primarily motivated by the desire to overcome these limitations. It presents a comprehensive study of the 6-DoF pose estimation problem. Drawing inspiration from the latest deep learning pose estimation methods, a novel 6-DoF pose estimation framework named Auto-Pose is proposed, which incorporates latent space representations of deep neural networks with supervised learning algorithms. The proposed framework consists of three novel autoencoder-based methods: DALSR-Pose, CVML-Pose, and CVAM-Pose. These proposed methods are specifically designed to address the limitations of the existing methods enabling the estimation of rigid objects' 6-DoF poses from a single colour image in real time, without access to any explicit 3D models of the objects or depth data or performing a post-refinement.

The fundamental idea is to implicitly learn intermediate representations of objects in the latent space from only colour images, and the 6-DoF poses are estimated from the latent representations using multiple regression-based algorithms such as multilayer perception (MLP), k-nearest neighbours (KNN), and random forest (RF).

The proposed methods can operate in real time and are applicable in complex scenarios, including textured/texture-less objects represented in low-resolution images with heavy occlusion and clutter.

Extensive experiments and evaluation results across multiple publicly available benchmark datasets demonstrate the superiority of the proposed framework in pose estimation accuracy over existing methods that similarly use latent space representations, with accuracy improved by 30%. It also achieves comparable results to other state-of-the-art methods that use 3D models.

The thesis makes significant contributions to the field of 6-DoF pose estimation facilitating development of model-free estimation algorithms. The novelty of the work rests in the proposed autoencoder-based methods that achieve competitive performance compared to the state-of-the-art using only data from a monoscopic camera, without the need for the object's 3D model, depth measurement, or further iterative refinement often essential for the existing methods.

Contents

List of Figures

List of Tables

List of Abbreviations

2D	Two-dimensional
3D	Three-dimensional
6D	Six-dimensional
6-DoF	Six Degrees of Freedom
AAE	Augmented Autoencoder
ADD	Average Distance of Model Points with distinguishable views
ADD(I)	Average Distance of Model Points
ADI	Average Distance of Model Points with indistinguishable views
APC	Amazon Picking Challenge
AR	Augmented Reality
Auto-Pose	Autoencoder-based Pose Estimation
BOP	Benchmark for 6D Object Pose Estimation
BOP18	BOP Challenge 2018
BOP19	BOP Challenge 2019
BOP20	BOP Challenge 2020
BOP22	BOP Challenge 2022
BRIEF	Binary Robust Independent Elementary Features
CNNs	Convolutional Neural Networks
CVAE	Conditional Variational Autoencoder

CVAM-Pose	Conditional Variational Autoencoder for Multi-Object 6-DoF Pose Estimation
CVML-Pose	Convolutional Variational Autoencoder-Based Multi-Level Network for Object 6-DoF Pose Estimation
CVPR	Computer Vision and Pattern Recognition Conference
DAE	Denoising Autoencoder
DALSR-Pose	Denoising Autoencoder Using Latent Space Regression for Object 6-DoF Pose Estimation
DRN	Dilated Residual Network
DT	Decision Tree
ECCV	European Conference on Computer Vision
ELBO	Evidence Lower Bound
ELU	Exponential Linear Unit
FPS	Farthest Point Sampling
fps	Frames per Second
GAN	Generative Adversarial Network
GELU	Gaussian Error Linear Unit
gt	Ground Truth
HMD	Head-mounted Display
ICCV	International Conference on Computer Vision
ICP	Iterative Closest Point
IoU	Intersection over Union
KL	Kullback–Leibler
KNN	K-Nearest Neighbours
LUT	Lookup Table
MAE	Mean Absolute Error
MLP	Multilayer Perceptron
MNIST	Modified National Institute of Standards and Technology

MSER	Maximally Stable Extremal Regions
MSPD	Maximum Symmetry-Aware Projection Distance
MSSD	Maximum Symmetry-Aware Surface Distance
Multi-Path	Multi-path Learning for Object Pose Estimation Across Domains
NNS	Nearest Neighbour Search
NCA	Neighbourhood Component Analysis
PBR	Physically-based Rendering
PCA	Principal Component Analysis
PnP	Perspective-n-Point
RANSAC	Random Sample Consensus
RE	Rotational Error
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
ROI	Region of Interest
SIFT	Scale-Invariant Feature Transform
SiLU	Sigmoid Linear Unit
SITE	Scale-Invariant Translation Estimation
SO(3)	3D Rotation Group
SSD	Single-shot Detector
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
TE	Translational Error
t-SNE	t-Distributed Stochastic Neighbor Embedding
UURIP	UCLan Undergraduate Research Internship Programme
VAE	Variational Autoencoder
VR	Virtual Reality
VSD	Visible Surface Discrepancy

List of Symbols

$\mathbf{P}$	pose
$\mathbf{R}$	rotation
$\mathbf{T}$	translation
T_x	translational offset on the x axis
T_y	translational offset on the y axis
T_z	projective distance on the z axis
$\mathbb{R}$	real number
M	3D model vertices
z	latent space variable
n	dimensionality of the latent space
$\|\cdot\|_F$	Frobenius norm
$\times$	cross product
N	normalisation
E_ϕ	encoder network
μ	mean
σ^2	variance
$\mathcal{N}(0, I)$	Normal distribution with zero mean and identity matrix
D_θ	decoder network
$\mathcal{D}$	dataset
$q(z\|x)$	variational posterior distribution
$p(z\|x)$	true posterior distribution
$p(x\|z)$	likelihood of observing x given z

$p(z)$	prior distribution
D_{KL}	Kullback-Leibler divergence
$\mathbb{E}$	mathematical expectation
$diag()$	diagonal
D	distance maps
V	visibility mask
p	visible pixels
S_M	global symmetry transformations
proj()	2D perspective projection
$\mathbf{H}$	rigid transformation in ICP
$\mathbf{q}$	point in the target point cloud
$\mathbf{p}$	point in the source point cloud
C	correspondence set of $\mathbf{p}$ and $\mathbf{q}$
$\mathbf{n_p}$	surface normal of each $\mathbf{p}$

Chapter 1

Introduction

In human vision, the ability to perceive and characterise objects appears to be inherent. Children learn about the world by observing, touching, and grasping the objects around them, even without understanding what these objects are. As they mature, their reliance shifts to knowledge accumulated from extensive experience; for instance, they can estimate an object's pose from just a glance. This observation raises a fundamental question: Can machines be endowed with a similar visual perception ability, enabling them to recognise an object's pose from a single image?

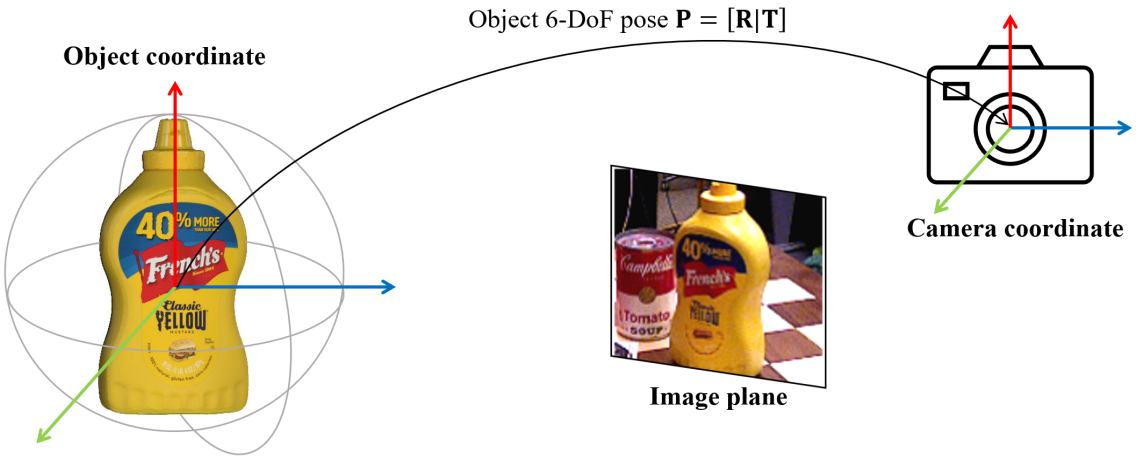


Figure 1.1: **Definition of a rigid object's 6-DoF pose.** The 6-DoF pose, denoted as **P**, of a rigid object is characterised by a rigid transformation with six degrees of freedom, from the object's coordinate system to the camera's coordinate system. This transformation comprises a 3D rotation **R**, and a 3D translation **T**. The 3D model and image are taken from the YCB-Video dataset [5, 6], and the image of the model is generated using the Pyrender software [9].

Scene understanding and automatic manipulation of objects present in such scenes are of fundamental importance to autonomous systems. A substantial amount of work in computer vision has been dedicated to developing reliable vision perception in machines. This includes recognition [10, 11, 12, 13], detection [14, 15, 16, 17, 18], segmentation [19, 20, 21, 22], and pose estimation [23, 24, 25, 26]. This thesis specifically focuses on the fast and accurate estimation of the 6-DoF pose of rigid objects, which includes both 3D rotation and 3D translation (as depicted in Figure 1.1). This capability is vital for a range of real-world applications, such as explorative navigation, augmented reality, and automated medical intervention. In industrial applications, precise pose estimation enables machines to grab and manipulate objects effectively. A notable example is the Amazon Picking Challenge (APC) [27, 28], which aims to encourage the development of intelligent machines capable of picking items from warehouse shelves, to automate packaging processes. The proposed work can be seen as a stepping stone towards the construction of systems where machines can be endowed with something resembling intelligence by replicating these perceptual capabilities of humans, which will enable the construction of fully autonomous systems operating safely and efficiently. This is symbolised through an AI-generated image, shown in Figure 1.2, paraphrasing the famous Michelangelo's *The Creation of Adam*, with the touching hands signifying God imparting knowledge to Adam.

The primary objective of this thesis is to develop methods for estimating the 6-DoF pose of a rigid object from a single colour image. The proposed pose estimation algorithms aim to be efficient and capable of real-time operation, while not requiring access to an explicit 3D model of the target object. To clarify, an object model typically includes several elements that define its structure and appearance in 3D space, such as vertices, textures, and surface normals. The term "real-time" throughout the thesis refers to the theoretical suitability of a method to complete computations within a specific time period. This means that a method is considered real-time if it is possible to predict the number of computations required to process a given class of input data (e.g. images of a given size). In this sense, iterative methods,

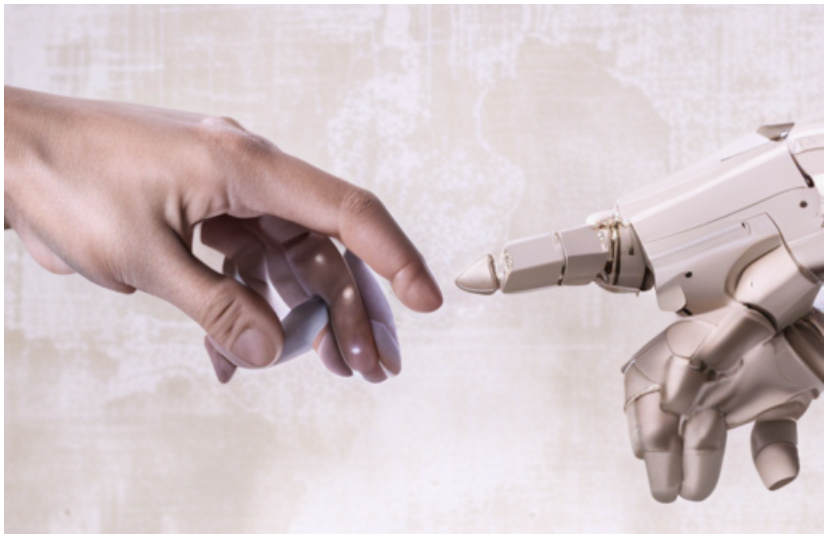


Figure 1.2: **The "Creation of Artificial Intelligence".** The main idea of the thesis is to enable machines to estimate a rigid object's 6-DoF pose from only images/videos. This figure is generated based on the stable diffusion model [29] using the DreamStudio© platform, with the prompt "mimic the creation of Adam with a robot arm on the right", accessed 19 May 2023.

such as iterative closest point (ICP) used by some pose estimation methods, are not considered real-time because it is not possible to determine the number of iterations required for the algorithm to converge to an acceptable solution. Using this definition, meeting specific time limits, e.g. 50 ms per image (for 20 frames per second video stream), can theoretically be achieved by adjusting computational resources (such as the graphics card) rather than the method itself. This presents a multitude of challenges from various perspectives, as detailed in Section 1.1. These challenges include, but are not limited to, problems arising from occluded objects, cluttered scenes, varying object appearances, and diverse lighting conditions.

1.1 Challenges

Estimating the 6-DoF pose of a rigid object from images presents numerous challenges, which can be categorised from the perspectives of the object, sensor, and environment.

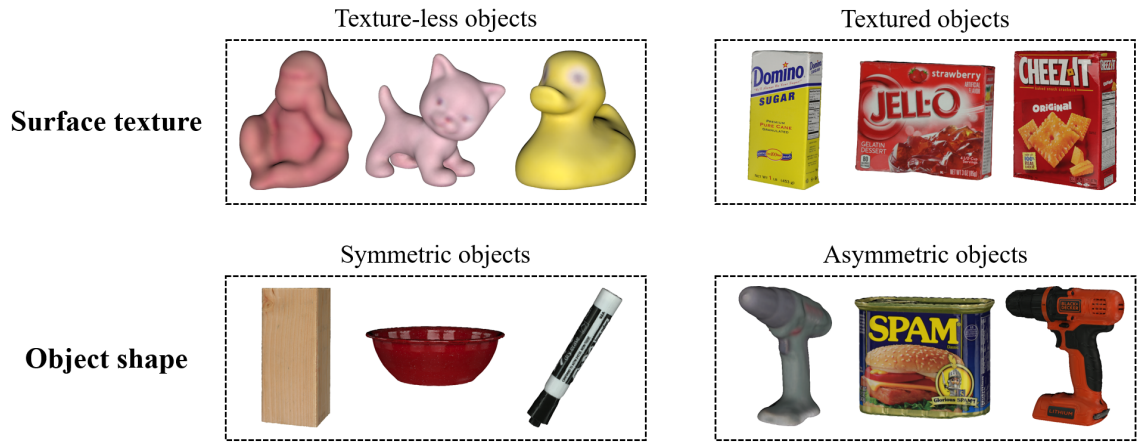


Figure 1.3: **Examples of object-orientated challenges.** The example cases highlight two significant challenges in pose estimation: the lack of distinctive features in texture-less objects which inhibits feature extraction and matching, and the inherent pose ambiguity associated with symmetric objects. The 3D models are taken from the Linemod [1, 2] and YCB-Video [5, 6] datasets, and the images of the models are generated using the Pyrender software [9].

Figure 1.4: **Examples of clutter and motion blur.** The example scene images depict scenarios with cluttered backgrounds and motion blur that add visual complexity. Simultaneously, the objects of interest demonstrate challenging cases such as occlusion, which complicates the pose estimation task. The images are taken from the Linemod dataset [1, 2].

- Object-orientated challenges, as illustrated in Figure 1.3, are inherently tied to the properties of the object itself, such as its shape and surface texture. For example, objects with rich textures facilitate the extraction of discriminative feature descriptors [30, 31, 32], while texture-less objects are problematic in

feature detection and matching. In these scenarios, pose estimation methods specifically designed for texture-less objects must rely on alternative representations, such as surface normals [1, 33]. Additionally, an object's shape can influence pose estimation algorithms, especially for symmetric objects, where multiple poses may appear visually similar, resulting in a many-to-one mapping problem between the input data and potential poses.

- Sensor-orientated challenges relate to the quality of images captured by sensors. The quality of these images is influenced by factors such as sensor noise and image resolution. For instance, low-resolution images may not guarantee reliable feature extraction. Sensors can also suffer from noise, manifesting as random variations in brightness or colour, and can be influenced by the material properties of the objects being imaged, as discussed in [34]. Furthermore, images may exhibit motion blur, as shown in Figure 1.4, a phenomenon that occurs when the sensor moves during image acquisition, adding complexity to pose prediction.

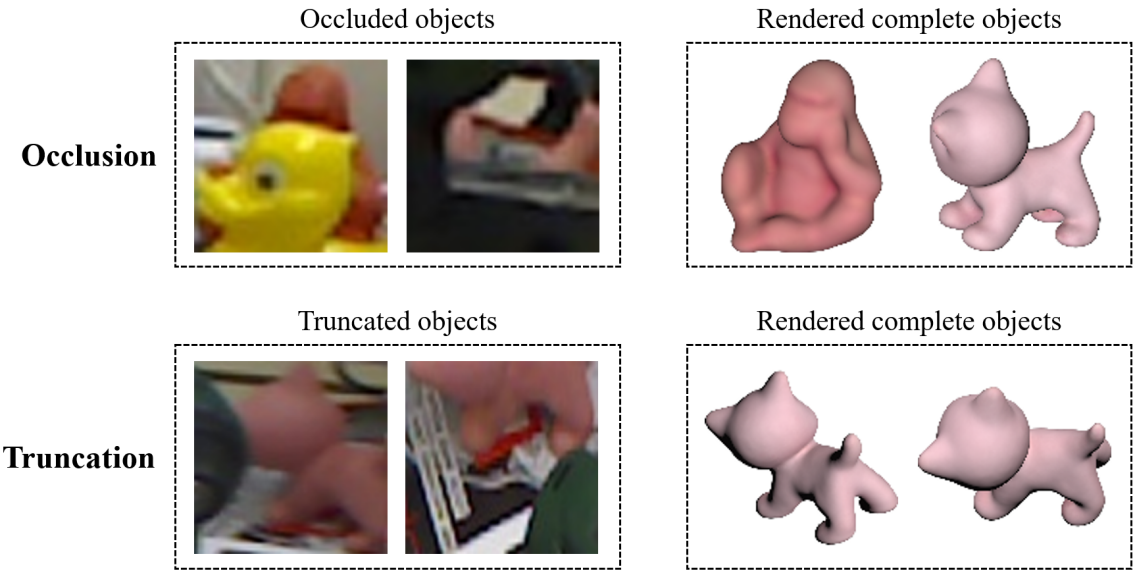


Figure 1.5: **Examples of object occlusion and truncation.** The rendered images show a complete view of the occluded and truncated objects, where both cases increase the difficulty of recognising an object's pose. The occluded and truncated images and 3D models are taken from the Linemod-Occluded dataset [3, 4], and the rendered images are generated using the Pyrender software [9].

- Environment-orientated challenges, as depicted in Figure 1.4 and 1.5, are primarily related to the surroundings of the objects. These challenges include occlusion, truncation [35], and cluttered background. Occlusion occurs when parts of the object of interest are obscured, typically by other objects. Truncation happens when parts of the object are cut off by the edges of the image frame, leaving the object partially outside the visual field. Both occlusion and truncation significantly reduce the available visual information, complicating accurate object identification and pose estimation. Different from them, clutter refers to images with backgrounds containing a large number of objects or features that are irreverent to the object of interest. These irrelevant features increase the complexity of identifying which features belong to the target object, leading to inaccurate prediction in object detection, segmentation, and consequently, pose estimation.

1.2 Problem Formulation

In this thesis, the problem formulation for estimating a rigid object's 6-DoF pose is conceptualised in Figure 1.6. The primary task involves developing fast and accurate methods to determine the 6-DoF pose of an object from a single colour image, given input data such as training images, ground truth (gt) bounding boxes, and gt 6-DoF poses of the object. In particular, the research reported here imposes several constraints on the proposed methods (detailed in Chapter 4), which highlight their novelties and advantages. For example, both the training and test data are restricted to colour images only[1], without any supplementary information from depth sensors. The 3D models of objects are utilised solely for pose evaluation and image synthesis, indicating that the proposed pose estimation methods do not rely on 3D models for inference. These methods are designed to function even in the absence of 3D models. The object's 6-DoF pose is fully defined by a matrix $\mathbf{P} = [\mathbf{R}|\mathbf{T}]$, which consists of

[1]The methods should also work with grayscale images because they are capable of handling texture-less objects.

3D rotation $\mathbf{R}$ and 3D translation $\mathbf{T} = (T_x, T_y, T_z)^T \in \mathbb{R}^3$, as depicted in Figure 1.1. The 3D translation $\mathbf{T}$ is a process of moving an object from one place to another in 3D space, typically represented by three numerical values. The 3D rotation $\mathbf{R}$, often more complex due to its diverse representations such as rotation matrix, axis-angle, or unit quaternion, will be detailed in Section 3.3.

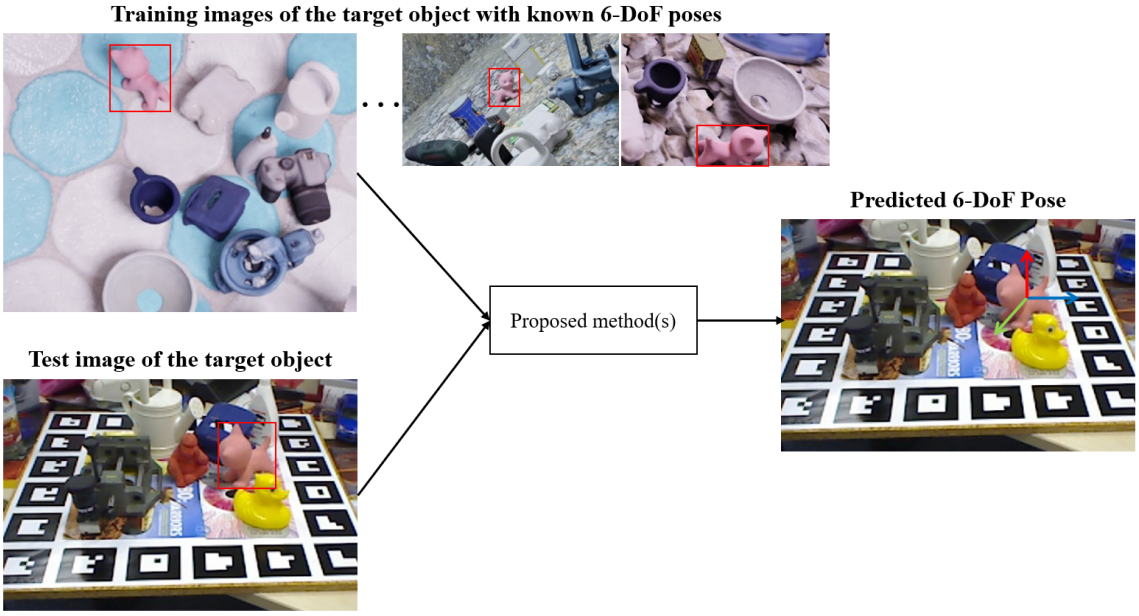


Figure 1.6: **Formulation of rigid object 6-DoF pose estimation problem.** The proposed method should be trained with only colour images of the target object with annotated 6-DoF poses. During inference, the method takes colour images as well and predicts 6-DoF poses of the detected object. Notably, neither object 3D models, depth information, nor post-refinement should be used in testing. Images are taken from the Linemod PBR dataset [1, 2, 7, 8].

The formulated pose estimation problem is multifaceted and challenging. For instance, the data used for training and inference inherently face several challenges as detailed in Section 1.1, including object occlusion, truncation, lack of texture, cluttered background, and motion blur. The constraints on the proposed methods also force the methods to abstract information solely from colour images, rather than relying on other sources like object 3D models or depth sensors, thereby adding significant complexity to the task. Furthermore, to meet the requirements for real-time processing, the adoption of time-consuming iterative post-refinement techniques is not necessary, which demands that the proposed methods are both accurate and fast.

Another technique adopted in the proposed pose estimation methods is 2D object detection, which is also important as it provides several fundamental components for evaluation (see Chapter 5). To clarify, object detection is not proposed in this thesis, and should be distinguished from the pose estimation problem. Object detection mainly aims to localise objects in the 2D image space, whereas pose estimation operates in the 3D camera coordinate space, which is more complicated, as explained in Figure 1.1. Consequently, pose estimation requires more information than object detection, such as camera intrinsic parameters.

1.3 Importance of 6-DoF Pose Estimation

Pose estimation is a cornerstone across a broad spectrum of real-world applications, including robotics [36, 37, 38, 39, 40, 41], autonomous driving [42, 43, 44, 45], and augmented reality [46, 47, 48, 49, 50]. This technique requires considering both rotational and translational movements of an object in 3D space. For example, precise pose perception is crucial for machines to interact with and manipulate their surroundings effectively, achieving operational goals and ensuring safety, particularly in complex environments. In augmented reality (AR) and virtual reality (VR), pose estimation plays a key role in creating immersive experiences, facilitating seamless interaction between virtual objects and real/simulated environments. In the context of scene understanding, reliable pose estimation algorithms are also fundamental to safety protocols in autonomous driving, enabling self-driving vehicles to precisely determine the poses of obstacles and other entities.

Delving into specific applications like industrial automation, fast and accurate pose estimation algorithms enable machines to skilfully handle tasks ranging from basic object manipulation [27, 28, 51], to complex activities such as performing surgical procedures [52, 53]. These applications often require a combination of sensors and algorithms. For instance, in automatic navigation [54, 55], various image sensors, including colour and depth cameras, collaborate with algorithms designed to

predict the object's 6-DoF pose in real time, thereby facilitating navigation and interaction with the environment. In AR applications, the pose of a user's camera or head-mounted display (HMD) needs to be constantly estimated and updated, for accurately rendering virtual objects based on the spatial information about the real world.

In medical applications, real-time pose estimation is vital for precision and safety in diagnostics and interventions [3, 56]. A notable example is robot-assisted surgery, such as machines used in the da Vinci® Surgical System [57], enabling fine manipulation and control of surgical instruments within the patient, and minimising the risk of accidental damage to surrounding structures. Here, the real-time capability is key, as surgeons can operate through a console that translates their hand movements into precise movements of instruments inside the patient in real time. In this scenario, the pose estimation algorithm helps to accurately map the surgeon's movements with the surgical instruments, ensuring that the instruments operate optimally to perform the required tasks such as tissue removal, dissection, and stitching. Moreover, real-time pose estimation reduces the risk of possible surgeon tremors or unintended movements, providing a stable and controlled surgical environment.

The importance of 6-DoF pose estimation cannot be overemphasised in these diverse applications. Real-time, seamless interpretation of pose plays a crucial role in bridging 2D and 3D spaces, which is the key to ensuring accurate and fast navigation, interaction, and integration in different environments. The pose estimation methods proposed in this thesis (see Chapter 4) not only address the key challenges often encountered in conventional approaches, such as the dependence on 3D models and post-refinement, and lack of robustness in complex environments, but also introduce a novel fusion of deep learning-based latent space representation with traditional machine learning-based regression algorithms. A real-time video demonstration[2] of the proposed CVML-Pose method (see Section 4.3) also exhibits the robustness in handling a variety of complex scenarios, which signifies a step forward in the field.

[2]video available: https://ieeexplore.ieee.org/document/10040668.

1.4 Novelty

In this thesis, a unified pose estimation framework named Auto-Pose is proposed. It encompasses three novel convolutional autoencoder-based methods, designed to address the problem of estimating the 6-DoF pose of rigid objects from a single colour image, without the need for accessing object's 3D model, depth information, or performing a post-refinement. The core concept shared across these methods is the integration of the autoencoder's latent space representation with regression-based algorithms. This involves training autoencoder models to learn intermediate representations of objects in the latent space, and subsequently using various regression-based algorithms to interpolate the learnt representations to continuous pose representations, including multilayer perception (MLP) [58], k-nearest neighbours (KNN) [59], and random forest (RF) [60, 61].

Each method within the framework is distinguished by the type of autoencoder model it employs, including denoising autoencoder (DAE) [62], variational autoencoder (VAE) [63], and conditional variational autoencoder (CVAE) [64], resulting in various latent space representations. The novel contributions of each method are outlined as follows:

DALSR-Pose: Denoising Autoencoder Using Latent Space Regression for Object 6-DoF Pose Estimation

DALSR-Pose differs from existing 6-DoF pose estimation methods, such as those described in [65, 66], which typically discretise pose into a finite set of instances using a lookup table (LUT) based on the DAE's latent space. Instead, DALSR-Pose introduces a novel continuous pose regression approach, combining the DAE's latent space representation and regression-based algorithms to interpolate the continuous pose of an object. The main idea is to implicitly learn an intermediate representation of the object. Subsequently, multiple regression algorithms, including MLP and KNN, are trained to separately regress continuous pose representations based on

the learnt representation. Finally, the complete object 6-DoF pose can be computed based on the pinhole camera model.

This work has been evaluated on the Linemod [1, 2] and the BOP version [7] of the Linemod-Occluded [3, 4] benchmark datasets. It outperforms methods based on latent representation [65, 66], e.g. 20.8% and 14.8% better on Linemod-Occluded using the VSD metric [7] as reported in Table 5.3, and achieves comparable results to state-of-the-art methods that use 3D models [67, 68], e.g. within 0.1% difference compared to [68] on Linemod-Occluded using the MSSD metric [7]. Although the proposed DALSR-Pose method does not yet match the accuracy of the leading methods based on 2D-3D model correspondence [35, 69, 70, 71, 72, 73, 74] and model-point refinement [75], it marks an advancement in facilitating continuous latent space regression for object 6-DoF pose estimation, contributing to the development of the proposed CVML-Pose approach.

CVML-Pose: Convolutional Variational Autoencoder-Based Multi-Level Network for Object 6-DoF Pose Estimation

CVML-Pose builds upon the DALSR-Pose method, introducing a novel convolutional variational autoencoder-based multilevel network for object 6-DoF pose estimation. The main originality and novelty is the proposal of a fast and accurate pose estimation algorithm that implicitly learns the pose of an object from only colour images encoded in the regularised latent space of a VAE. Similar to DALSR-Pose, after encoding in the latent space, multiple supervised learning methods, such as MLP and KNN, are trained to regress the continuous pose representation from the frozen latent space. Through an analysis of the latent variables using clustering algorithms, the method is possible to infer other characteristics beyond pose, such as object class and shape topology.

The robustness and effectiveness of the CVML-Pose method have been comprehensively validated through extensive ablation tests and experiments conducted on several benchmark datasets, including Linemod, the BOP version of Linemod-

Occluded, and the BOP version of YCB-Video [5, 6]. For example, it has been shown to significantly outperform state-of-the-art methods based on latent space representation [65, 66] (46.0% and 36.8% better) and 2D-3D model correspondence [67, 68] (43.6% and 16.4% better) on Linemod-Occluded using the MSPD metric [7], as reported in Table 5.3. Although it does perform as well as the most state-of-the-art methods that do not use latent space, e.g. showing 26.0% and 27.0% worse performance than [71] and [76] on the YCB-Video dataset using the VSD metric, it can achieve results somewhat comparable to the leading methods reliant on 3D models [35, 69, 70, 73]. For example, it performs within a 4.0% difference compared to [35] on Linemod-Occluded using the MSPD metric.

CVAM-Pose: Conditional Variational Autoencoder for Multi-Object 6-DoF Pose Estimation

CVAM-Pose is a multi-object[3] pose estimation method, developed upon the CVML-Pose method. It draws inspiration from the CVAE model, which leverages label conditions to generate specific data samples. The key concept is to use a CVAE that incorporates the one-hot encoding technique within the autoencoder architecture, which facilitates learning constrained representations of the multi-object 6-DoF poses in the latent space. In particular, the one-hot encoded conditional labels are embedded into every convolutional layer or block of the model, which learns more high-level feature representations. These learnt multi-object representations are concatenated with the conditional labels, and subsequently interpolated to the continuous poses of multiple objects using regression-based algorithms including MLP and RF.

Empirical evaluation of the CVAM-Pose method, conducted on the BOP version of the Linemod-Occluded dataset, indicates its superior performance over the state-of-the-art latent representation methods in both single-object and multi-object prediction scenarios [65, 66] (45.5% and 36.3% better using the MSPD metric as reported in Table 5.3). Notably, the CVAM-Pose method improves the scalability and compu-

[3] "multi-object" means more than one object.

tational efficiency in multi-object predictions within a single-encoder-single-decoder architecture, and achieves almost the same level of pose accuracy compared to the proposed CVML-Pose method.

1.5 Thesis Organisation

The structure of this thesis is organised to provide a comprehensive exploration of rigid object 6-DoF pose estimation, presented as follows:

This chapter, Chapter 1, introduces the problem of 6-DoF pose estimation of rigid objects, highlighting its challenges, and defining the specific problem formulated in this thesis. It also describes the wide-ranging applications and the novel contributions of the proposed methods.

Chapter 2 reviews both classical machine learning-based methods and state-of-the-art deep learning-based 6-DoF pose estimation methods. The focus primarily is on deep learning-based methods, which are categorised according to their approaches to solving the problem. The chapter also engages in a discussion regarding the use of object 3D models and concludes with a summary of the advantages and disadvantages of the reviewed methods.

Chapter 3 details the tools and techniques employed in the thesis, following a structured sequence of data, method, and evaluation. It begins with an introduction to the pose estimation datasets used on the project and the relevant data processing methods. Subsequent sections delve into the pose representations and autoencoder models used in the proposed methods, along with a comprehensive explanation of the evaluation metrics.

Chapter 4 addresses the problem of estimating a rigid object's 6-DoF pose from a single colour image, without access to the object's 3D model, depth information, and iterative post-refinement. It describes three novel pose estimation methods based on the latent space of different autoencoder models: DALSR-Pose, CVML-Pose, and

CVAM-Pose. These methods involve learning implicit representations of objects in the autoencoder's latent space, followed by regression of the learnt representations to the continuous pose representation. The chapter highlights the importance of continuous regression in the DALSR-Pose method, as well as emphasises the regularised latent space in the CVML-Pose method for comprehensive object characterisation, including the object's pose, category, and shape topology. It also discusses the proposed label embedding technique in the CVAM-Pose method, demonstrating its effectiveness in encoding constraint representations for multiple objects within a single latent space, without losing accuracy compared to the single-object methods. Comprehensive ablation studies are presented to determine favourable parameters of the proposed methods, with concluding remarks provided for each method.

Chapter 5 presents the evaluation pipeline, results, and discussions for the three proposed methods. It starts with an overview of the evaluation procedure, followed by a comparative analysis of the pose accuracy of the proposed methods against state-of-the-art methods. This comparison includes methods employing latent space representation, as well as those that utilise 3D models. The effectiveness of the proposed methods is assessed across multiple benchmark datasets, using a variety of evaluation metrics. The chapter also analyses and discusses the strengths and weaknesses of each category based on the categorisation of these state-of-the-art methods introduced in Chapter 2. Furthermore, the chapter explores additional tests designed to examine the impact of object detectors, data quality, and occlusion on pose estimation accuracy.

Chapter 6 investigates the impact of refinement techniques on 6-DoF pose estimation. The chapter begins by introducing typical refinement techniques, including an online learning-based refinement using object 3D models, and an iterative post-refinement using both 3D models and depth measurements. Following this, the chapter presents detailed experiments and results using the proposed CVML-Pose method as the baseline model. A comparative analysis of the pose accuracy and the inference time is included, as well as conclusions at the end of the chapter.

The final chapter, Chapter 7, concludes the thesis, summarises the novel contributions, lists the limitations of the proposed methods, and presents possible further work.

Chapter 2

Literature Review

2.1 Introduction

Estimating an object's 6-DoF pose is a fundamental problem in computer vision. The pose refers to the rigid transformation with six degrees of freedom that defines the rotation and translation of an object in 3D space with respect to the camera coordinate system. Initially, traditional methods mainly focused on extracting and matching discriminative feature descriptors, from which the pose was recovered through established 2D-3D correspondences. The advent of depth sensors, such as the Microsoft Kinect [77] and Intel RealSense [78], shifted the focus towards template matching, a process of comparing different scene representations. The introduction of convolutional neural networks (CNNs) [58, 79] marked a significant evolution in 6-DoF pose estimation. Deep learning-based approaches, leveraging the capabilities of CNNs, have achieved remarkable results. Consequently, the focus is now turned towards building appropriate loss functions, better rotation representations, and finding good intermediate data representations.

This chapter reviews existing 6-DoF pose estimation methods, dividing them into classical machine learning-based methods (Section 2.2) and deep learning-based methods (Section 2.3). The main focus is on deep learning-based methods, further

categorised into direct methods (Section 2.3.1), indirect methods (Section 2.3.2), and latent representation methods (Section 2.3.3). Additionally, this chapter includes a discussion on the use of object 3D model (Section 2.4) and concludes with a summary of these methods (Section 2.5). For more comprehensive reviews on 6-DoF pose estimation, please refer to [23, 24, 25, 80, 81, 82, 83].

2.2 Classical Machine Learning-based Methods

Before deep learning became mainstream, numerous traditional machine learning techniques were employed for object 6-DoF pose estimation. Early methods typically relied on edge features obtained from edge detectors, such as the Canny detector [84]. One notable example is the work by Lowe [85], which proposes a scale-independent segmentation algorithm for detecting and matching edge segments, referred to as edgelets. The pose of the object can be recovered by matching between sets of edgelets with specific viewpoints of the object's 3D model. Extending this concept, Lamdan and Wolfson [86] introduce a multi-object pose estimation using geometric hashing. Further, Lowe et al. [87] advance this method to work with objects having arbitrarily curved surfaces, and varying internal parameters representing articulation, variable dimensions, or surface deformations. Another significant contribution is from Damen et al. [88], combining tractable edgelet constellations with a lookup table (LUT) represented by a transformation-invariant descriptor. Based on [88], Hodan et al. [89] propose an improved work by incorporating a better edge detector [90], faster Hough-based tracing, and more accurate constellation representations.

These edge-based methods [85, 86, 87, 88, 89, 91] typically rely on the performance of edge detection, which can work well with both textured and texture-less objects (see example objects in Figure 1.3). However, they are sensitive to the cluttered background (see examples in Figure 1.4), which can significantly impact the results of pose estimation.

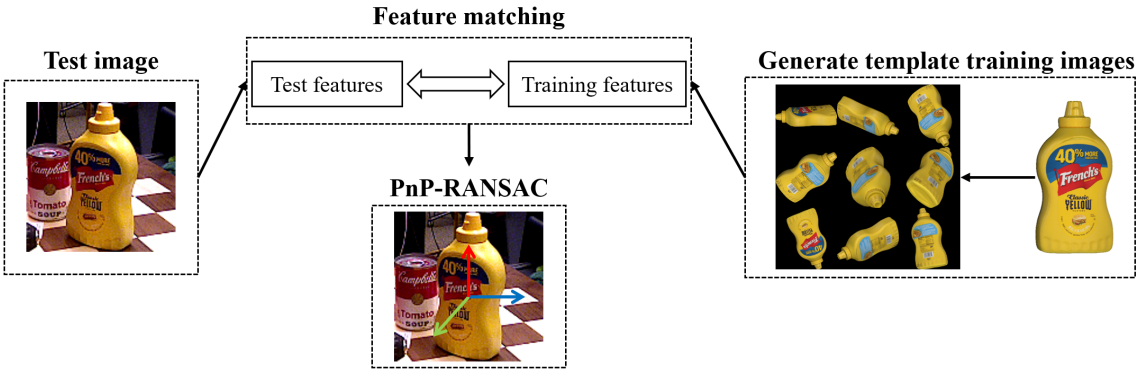


Figure 2.1: **Example procedure of estimating an object's 6-DoF pose by extracting and matching discriminative features.** A set of 2D local feature points of the test instance are extracted and matched against those extracted from the training set, which results in 2D-3D correspondences, and the final pose can be retrieved by RANSAC [92] and PnP [93]. The 3D model and images are taken from the YCB-Video dataset [5, 6]. The template images are generated using Pyrender [9].

With the success of local feature descriptors such as Scale-Invariant Feature Transform (SIFT) [30], Speeded Up Robust Features (SURF) [31], Maximally Stable Extremal Regions (MSER) [94] and Binary Robust Independent Elementary Features (BRIEF) [32], many pose estimation algorithms began focusing on extracting such discriminative features from textured objects. A typical procedure, illustrated in Figure 2.1, involves extracting 2D feature points from training images. These features are then matched against those in test images, with the matched features mapped to predefined 2D-3D correspondences obtained from the object's 3D models. The 6-DoF pose is subsequently computed using a Perspective-n-Point (PnP) algorithm [93]. For instance, Lourakis and Zabulis [95] propose a feature-based pose estimation method where the SIFT descriptors detected in the observed image are matched with those contained in the 3D model, and the object's 6-DoF pose is estimated using the PnP algorithm with Random Sample Consensus (RANSAC) [92]. Another method, as proposed in [96], builds a global model description using the point pair feature, with the final pose estimated through an efficient voting loop if the reference point lies on the surface of the object. In addition to SIFT-like local feature descriptors, the authors in [97, 98] employ various machine learning algorithms, such as nearest neighbour search (NNS) [99] or randomised trees [100], to classify prominent feature points based on the object's 3D model.

Despite their high precision, approaches using explicit feature descriptors [95, 97, 98, 101, 102] require accurate 3D models to establish 2D-3D correspondences, and are therefore not model-free. Moreover, these feature-based methods face difficulties with such objects due to the lack of discriminative features for extraction and matching. This is an important limitation as texture-less objects are almost everywhere in practical applications. These methods are also prone to issues related to occlusion, which leads to a reduction in available features, thereby compromising reliable pose estimation.

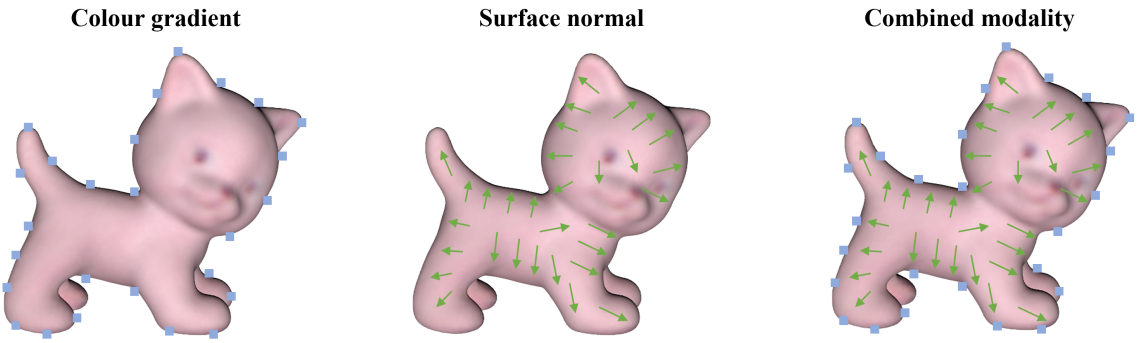


Figure 2.2: **Example texture-less object with colour gradient information and surface normal features.** Left: location of the silhouette colour gradient vectors. Middle: location of the surface normals obtained from the 3D model of the object. Right: the combined features to estimate and refine the 6-DoF pose of the object. The 3D model is taken from the Linemod dataset [1, 2]

To work with texture-less objects, template-based methods [1, 33, 103, 104, 105, 106] utilise colour gradient information from silhouettes and interior surface normal features (Figure 2.2). For instance, Hinterstoisser et al. [1] generate template images with ground truth (gt) object poses and retrieve the most similar template based on the silhouette. The final pose is then refined using surface normal orientations obtained from 3D models. Similarly, Li et al. [107] propose an efficient method capable of handling texture-less objects based on stably observed point pair features, requiring both depth measurement and knowledge of the object's 3D model. In another approach, Tsai and Tsai [108] develop a method that utilises both texture and shape information.

Although template-based methods are useful for handling texture-less objects, they

often suffer from clutters and occlusions [109, 110]. Moreover, they heavily rely on the access of object 3D models for template generation, or depth information for determining surface normal orientations, which may not always be feasible in real-life applications.

2.3 Deep Learning-based Methods

The development of computer vision algorithms has significantly influenced the field of object 6-DoF pose estimation. The classical methods mentioned earlier (Section 2.2) mainly rely on different kinds of image representations, such as edge-based features, local/global feature descriptors, and image templates. With the rapid development of deep learning techniques, numerous state-of-the-art pose estimation methods [5, 35, 65, 67, 69, 70, 71, 72, 73, 74, 75, 76, 111] based on CNNs have been proposed. These deep learning-based methods differ from classical approaches in that they typically rely on features or parameters learnt from CNNs and use them for different purposes.

This section categorises deep learning-based methods into three distinct groups according to how neural networks are used: direct methods (Section 2.3.1), indirect methods (Section 2.3.2), and latent representation methods (Section 2.3.3). The direct methods train CNNs to regress 3D rotation and translation from images directly. The indirect methods focus on learning 2D-3D correspondence mapping using CNNs, with the 6-DoF pose subsequently estimated using different variants of the PnP algorithms. The latent representation methods aim to learn implicit latent space representations of objects using specific network architectures, typically autoencoders. The pose of a test instance is then retrieved by finding the nearest neighbours, computing observation likelihoods, or directly regressing from the learnt representations.

2.3.1 Direct Methods

The development of deep learning-based direct methods (referred to as direct methods), which regress 6-DoF poses directly from CNNs (illustrated in Figure 2.3), has significantly influenced object pose estimation. One of the most representative methods is PoseCNN [5], which trains an end-to-end network that decouples the pose estimation task into three subtasks: semantic labelling, 3D translation estimation, and 3D rotation regression. This method first predicts semantic labels for each pixel. It then estimates the 2D projective centre on the image and the 2D projective distance from the camera for each object. The 2D projection centre is localised through Hough voting [112], and the 3D translation is retrieved using the pinhole camera model with known camera intrinsic parameters. The 3D rotation for each object is parameterised to a unit quaternion representation and estimated from each region of interest (ROI), using a model point-based loss function (Eq. 2.1). This model-based training refers to training with 3D model points, where the loss function is calculated in 3D space. Therefore, it requires 3D models of the objects, with 3D points being part of the models.

$$\begin{aligned} \text{PoseLoss} &= \frac{1}{2m} \sum_{x \in M} ||R(\tilde{\mathbf{q}})x - R(\mathbf{q})x||_2 \\ \text{ShapeMatch-Loss} &= \frac{1}{2m} \sum_{x_1 \in M} \min_{x_2 \in M} ||R(\tilde{\mathbf{q}})x_1 - R(\mathbf{q})x_2||_2 \end{aligned} \tag{2.1}$$

where m is the number of model points, M depicts the 3D model vertices, $R(\tilde{\mathbf{q}})$ and $R(\mathbf{q})$ are the rotation matrices transformed from the estimated unit quaternion representation $\tilde{\mathbf{q}}$ and the gt unit quaternion representation $\mathbf{q}$. Details of the unit quaternion representation are described in Section 3.3.4.

Similarly to PoseCNN, Wu et al. [113] train a dilated residual network (DRN) [114] to generate segmentation masks with corresponding object class labels. This is followed by a pose interpreter network comprising a modified ResNet-18 network [115] and a multilayer perceptron (MLP) [58], trained to output the 3D rotation (parameterised

as a unit quaternion) and 3D translation. The pose interpreter network is also trained with a model point-based loss function to minimise the distance between the object's 3D model points transformed by the gt pose and the predicted pose. EfficientPose [116] modifies the EfficientDet detection pipeline [117] and introduces two subnetworks to predict the rotation and translation respectively, where the rotation is parameterised as an axis-angle representation, and the translation is estimated following the similar procedure to PoseCNN. DeepIM [111] implements an iterative refinement where the network predicts a relative pose transformation between the input image and the rendered image of its predicted pose. CosyPose [75] extends DeepIM with a recently proposed continuous rotation representation [118] (detailed in Section 3.3.5) and a more advanced network architecture, and won the BOP Challenge 2020 [7]. GDR-Net [71] trains a geometry-guided network to learn model correspondences, and employs the continuous rotation representation for pose regression, which won the BOP Challenge 2022 [26].

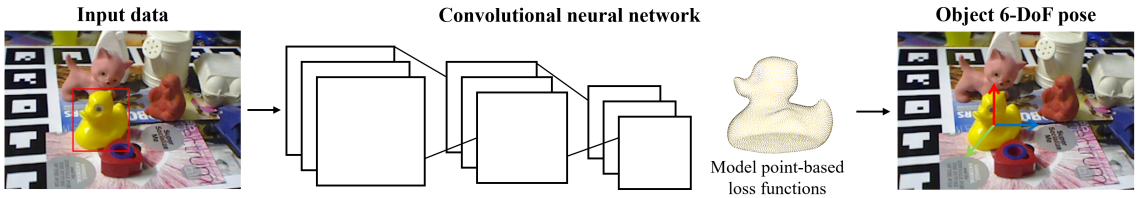


Figure 2.3: **Example procedure of estimating an object's 6-DoF pose by directly regressing 3D rotation and translation from CNNs.** The deep learning-based direct methods typically train an end-to-end network to perform semantic labelling or object detection, then estimate the pose from the detected region or segmented mask, with specifically designed loss functions using object 3D model points. The 3D model and images are taken from the Linemod dataset [1, 2].

Instead of regression, SSD6D [119] converts the pose estimation task to a classification task, modifying the popular single-shot detector (SSD) paradigm [120]. It divides the 3D space into a set of classifiable viewpoints by rendering all possible views and in-plane rotations, discretising the 3D rotation group (denoted as SO(3)) into a finite number of instances. This approach necessitates post-refinement techniques, as the initial prediction is only a rough approximation of the pose. A similar discretisation of the pose is also implemented in several latent representation meth-

ods [36, 65, 66, 121], which will be discussed later in Section 2.3.3.

Training a pose regression network is challenging, as illustrated in [111]. Using separate loss functions for rotation and translation, such as angular distance between rotations and L2 distance between translations, often suffers from difficulty in balancing the two losses [122]. Consequently, most direct methods prefer using model point-based loss functions like PoseLoss and ShapeMatch-Loss (see Eq. 2.1) proposed by PoseCNN [5], which calculate the distance between 3D model points at the gt rotation and the estimated rotation. These loss functions can be further refined by considering the translation vector T, as implemented in [116]. Here, the PoseLoss and ShapeMatch-Loss are extended to L_{asym} and L_{sym}, respectively.

$$\begin{aligned} L_{asym} &= \frac{1}{m} \sum_{x \in M} ||(R(\tilde{\mathbf{r}}, x) + \tilde{\mathbf{t}}) - (R(\mathbf{r}, x) + \mathbf{t})||_2 \\ L_{sym} &= \frac{1}{m} \sum_{x_1 \in M} \min_{x_2 \in M} ||(R(\tilde{\mathbf{r}}, x_1) + \tilde{\mathbf{t}}) - (R(\mathbf{r}, x_2) + \mathbf{t})||_2 \end{aligned} \tag{2.2}$$

where $R(\tilde{\mathbf{r}}, x)$ and $R(\mathbf{r}, x)$ are the model points transformed from the estimated axis-angle representation $\tilde{\mathbf{r}}$ and the gt axis-angle representation $\mathbf{r}$, using the Rodrigues' rotation formula [123, 124, 125]. Details of the axis-angle representation are described in Section 3.3.3.

Model point-based training is preferable for improving the accuracy of pose estimation, as shown in many state-of-the-art methods [5, 75, 111, 113, 116]. However, as outlined in Section 1.2, the proposed methods should achieve state-of-the-art performance without explicitly using 3D models or prior 3D information about the object. Thus, neither model point-based loss functions nor model-based refinement fulfils the requirements set for the methods developed in this thesis. It will be demonstrated later in Chapter 6 that the proposed CVML-Pose method does not require the model point-based training but can still achieve competitive pose estimation results. Furthermore, it is difficult to build an accurate 3D model for every possible object of interest in real-life applications, methods relying on object's 3D model would fail to work when the model is not available.

2.3.2 Indirect Methods

The deep learning-based indirect 6-DoF pose estimation methods (referred to as indirect methods) typically train CNNs to regress predefined 2D-3D representations of the target object (illustrated in Figure 2.4). These representations include pixel-wise dense correspondence or a number of predetermined sparse keypoints. The pose estimation problem is then formulated as a PnP problem, which estimates the pose given a set of 3D keypoints of the object, their corresponding 2D projections in the image, and the camera intrinsic parameters.

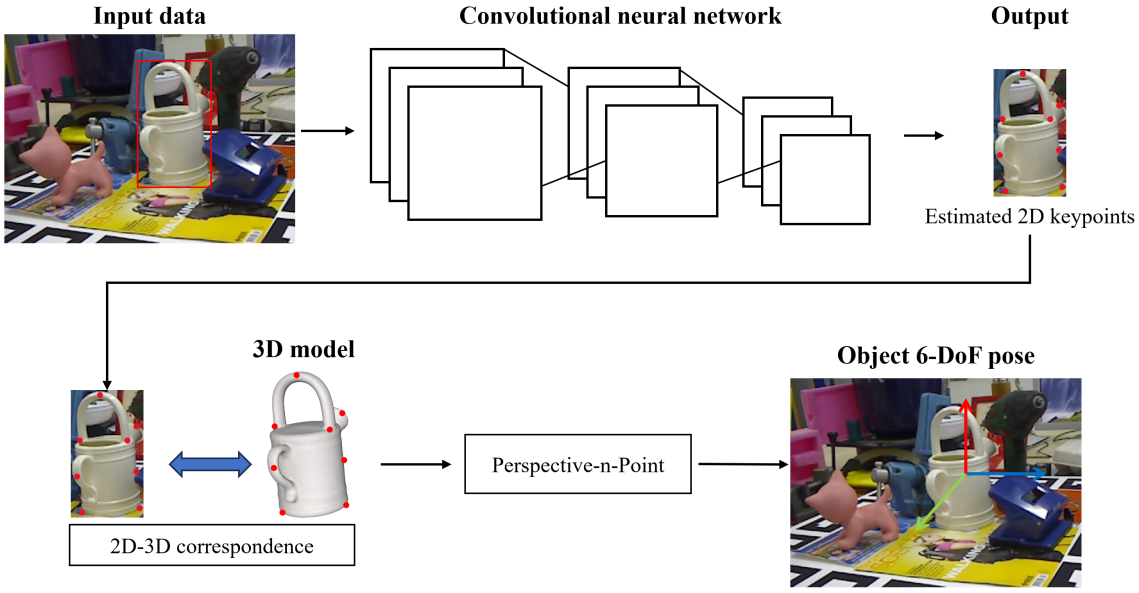


Figure 2.4: **Example procedure of estimating an object's 6-DoF pose by regressing 2D-3D correspondence.** The deep learning-based indirect methods typically train CNNs to predict 2D projections of 3D keypoints or pixel-wise dense keypoints, the 6-DoF poses are subsequently recovered using the PnP algorithm based on the established 2D-3D correspondence. The 3D model and images are taken from the Linemod dataset [1, 2].

For instance, BB8 [126] trains a segmentation network to localise objects of interest in 2D images, followed by a modified VGG network [127] predicting the 2D projections of the corners of 3D bounding boxes, and the final pose is estimated using an efficient PnP algorithm [93]. YOLO-6D [128] modifies the YOLO detection model [129] to predict the 2D locations of an object's 3D bounding box vertices. The pose is then recovered using the same PnP algorithm as BB8. YOLOv5-6D [74]

extends YOLO-6D with a more advanced YOLOv5 detector [130] and expands it to X-ray imaging tasks. However, methods using the eight corners of an object's 3D bounding box as keypoints often suffer from occlusion due to failure in keypoint detection. Inspired by human pose estimation [131], some methods instead train CNNs to predict heatmaps of the keypoints to address occlusion. For example, Oberweger et al. [132] predict heatmaps for the 2D projections of the corners of the object's 3D bounding box from multiple image patches independently. The heatmaps are then aggregated, and the PnP algorithm is established to recover the 6-DoF pose. Pavlakos et al. [133] localise a set of class-specific keypoints from the output heatmap responses, where these keypoints are manually defined on the 3D model surface. The final 6-DoF pose is computed based on the 2D-3D correspondences using the PnP algorithm. The PVNet method [35] also suggests that establishing a set of voting-based 2D-3D keypoints on the 3D model surface is more effective than selecting the eight corners of the 3D bounding box. The method selects 8 keypoints on the object surface using the farthest point sampling (FPS) algorithm. A segmentation network modified from the ResNet-18 [115] is trained to output the semantic labels and the pixel-wise unit vectors pointing to the keypoints. These keypoints are then localised by voting hypothesis, and the final pose of the object is estimated using a RANSAC-based PnP algorithm which prunes the matched pairs. Although not using CAD models directly, recent methods such as OnePose [47] use Structure from Motion (SfM) to build sparse models of objects and match 2D keypoints in the inference image with 3D keypoints in the sparse models.

Except for manually selecting a number of sparse keypoints from 3D models, 3D object coordinates can also be utilised for establishing 2D-3D correspondence. Early work by Brachmann et al. [3] predicts the 3D coordinates using traditional machine learning algorithms such as random forest from RGB-D images. With a more advanced auto-context Hough forests [134], a RANSAC-based pose sampling, and marginalisation of the 3D coordinate distribution over depth, the extended version [50] can also predict the 6-DoF pose from a single RGB image. Subsequent

approaches mainly focus on predicting the 3D object coordinates from RGB images using CNNs. For example, EPOS [69] predicts the correspondence between densely sampled pixels and the object's 3D fragments, before the pixels are linked to the predicted 3D locations and a variant of the PnP-RANSAC algorithm [93, 135] is used to estimate the final pose. Similar to EPOS, SurfEmb [73] links image pixels to object surfaces through a shared embedding space, allowing for dense and continuous correspondence distributions. Pixel2pose [68] adopts an image-conditional Generative Adversarial Network (GAN) [136] to estimate the 3D coordinates, and then the pixel-wise predictions are used to form 2D-3D correspondence and finally compute the pose via a RANSAC-based PnP algorithm. DPOD [67] regresses the object mask and 2D-3D correspondence UV map through an encoder-decoder network, computing the object's pose based on PnP and RANSAC. RNNPose [72] trains a recurrent neural network (RNN) [137] to iteratively refine the object pose based on the estimated dense correspondences between the input data and the reference data (data generated from 3D models). Unlike them, CDPN [70] disentangles 3D rotation and translation, which proposes a coordinate-based method to estimate only rotation using PnP from the predicted 2D-3D correspondence. The translation is then estimated directly from the local image patches using a Scale-Invariant Translation Estimation (SITE) strategy. Instead of directly predicting the translation vectors, the method predicts the scale-invariant offset and recovers the final translation based on the pinhole camera model, which improves the pose estimation accuracy on the Linemod dataset by 14.8% using the Average Distance of Model Points (ADD(I)) metric as reported in [70].

Despite indirect methods generally performing well with the help of prior 2D-3D correspondence information from the object's 3D model, they require additional computation for setting up either 2D-3D keypoints or pixel-wise dense correspondence. In summary, they need an accurate 3D model to acquire such information.

2.3.3 Latent Representation Methods

Another group of pose estimation methods have been proposed based on the latent space of autoencoders, categorised as latent representation methods (Figure 2.5). A notable example is the method proposed by Sundermeyer et al. [65], known as Augmented Autoencoder (AAE), trains a denoising autoencoder (DAE) [62] with strong data augmentation. It learns implicit pose representations from rendered 3D model views, and builds a codebook from the latent space representation, where the training process utilises the pixel-wise L2 loss (Eq. 2.3).

$$l_2 = \sum_i ||x_i - \hat{x}_i||_2 \tag{2.3}$$

where x_i represents the gt data, $\hat{x}_i$ represents the reconstructed data.

At the inference stage, the pose is estimated using a lookup table (LUT) from the codebook built with all trained representations $z_i \in \mathbb{R}^{128}$, $i = 1...m$, m represents the number of training images. The cosine similarity (Eq. 2.4) is calculated between z_i and the representation produced by the test image $z_{test} \in \mathbb{R}^{128}$. Subsequently, the rotation with the highest similarity is assigned to the test instance, determined using the k-nearest neighbours (KNN) algorithm [59].

$$cos_i = \frac{z_i \cdot z_{test}}{||z_i|| \cdot ||z_{test}||} \tag{2.4}$$

where z_i represents the trained representation, z_{test} is the representation produced by the test image.

The extended Multi-Path method [66] jointly estimates the implicit latent space representations of multiple objects using the same DAE network but in a single-encoder-multi-decoder architecture. This demonstrates that the learnt representations can be shared through different objects. Similar to AAE, PoseRBPF [121] maps the autoencoder's latent space to a codebook but samples object poses through a Rao-Blackwellised particle filter [138]. The 6-DoF pose is then estimated by computing

observation likelihoods for each particle. The extended work [36] takes RGB-D images as input together with a physical robot arm for grasping which boosts the performance. Although not using the autoencoder architecture, similar ideas exist in earlier work from [139, 140], which train CNNs to learn discriminative descriptors of target objects and map the descriptors to 6-DoF poses using the NNS algorithm.

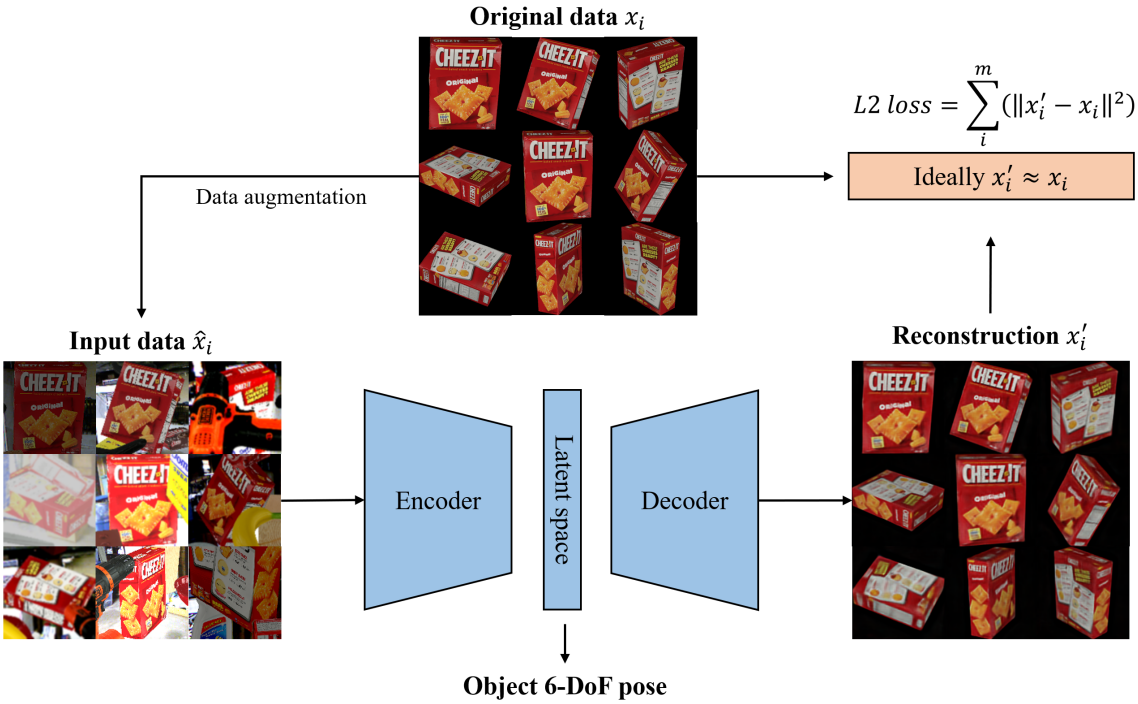


Figure 2.5: **Example procedure of estimating an object's 6-DoF pose using autoencoder's latent space representation.** The deep learning-based latent representation methods typically train an autoencoder network that encodes images of objects with strong data augmentation in the latent space, and reconstructs objects with a clean background from the latent representation. The final 6-DoF pose is interpolated from the latent representation. The images are taken from the YCB-Video dataset [5, 6].

Latent representation methods are capable of handling occluded objects due to their robust data augmentation strategies and inherent autoencoder architecture. However, methods using the LUT assign the most likely pose to the instance, leading to notable errors due to the discretisation of SO(3). Additionally, they primarily focus on the 6-DoF pose, ignoring other attributes of the object which can be crucial for specific tasks. In contrast, the proposed methods (see Chapter 4) aim to construct an informative latent space that maps not only to object 6-DoF poses but also to other characteristics, e.g. object category and shape topology. In terms of object

characterisation, most existing 6-DoF pose estimation approaches are restricted to object 6-DoF pose only, and have to be retrained or redesigned to incorporate other attributes of objects. In this case, the latent space of the proposed CVML-Pose method (see Section 4.3) has the potential to be extended to multiple tasks without retraining the main network, reducing computational costs. The object characterisation tasks, including object class and genus, will be explained in Section 4.3.3.

Both the AAE and Multi-Path methods are compared with the proposed methods across multiple benchmark datasets in Section 5.3. Although the proposed methods are not the first to use latent representation for 6-DoF pose estimation, it will be demonstrated later that the proposed methods outperform AAE, Multi-Path, and AAE-ICP (AAE with 3D model and depth refinement) by a certain margin when tested on the more challenging BOP version [7] of the Linemod-Occluded [3, 4] and the YCB-Video [5] datasets. The proposed CVAM method (see Section 4.4) also shows superiority over the Multi-Path method by enabling a more efficient single-encoder-single-decoder architecture with the conditional variational autoencoder (CVAE) network [64] for multi-object pose estimation. It is also noted that the PoseRBPF method and its extended version cannot be directly compared with the proposed methods because the authors neither report results on the Linemod [1, 2] dataset evaluated in this thesis, nor participate in the BOP Challenge [7, 26, 141].

2.4 Use of Object 3D Model

Considering the practical scenario in the current state-of-the-art, object 3D models are unavoidably used in image synthesis and pose evaluation. To address the lack of real data, several methods [5, 35, 65, 67, 69, 119] generate substantial amounts of synthetic training data, by rendering textured object 3D models from various viewpoints and placing them onto different real background images. To bridge the domain gap between synthetic training data and real test data, the BOP Challenge 2020 [7] (an annual challenge in the field of object 6-DoF pose estimation) addi-

tionally provides a large volume of photorealistic synthetic data using the physically based rendering (PBR) technique [142, 143], which dramatically improve the performance of many approaches (as will be described in Section 3.2.4). In the proposed pose estimation methods (Chapter 4), although the photorealistic training data are generated from object 3D models, the proposed methods themselves do not access any information from 3D models. Moreover, training with photorealistic data allows for a fair and straightforward comparison with the state-of-the-art methods participating in the BOP Challenge, as most participants train their methods using these data.

Regarding pose evaluation (to be explained in Section 3.5), most commonly used pose evaluation metrics such as Average Distance of Model Points (ADD and ADI) [1] and Visible Surface Discrepancy (VSD) [7] are model-dependent, which utilise object 3D model points to calculate model distance or fitness of model surface alignment. To facilitate comparison with the state-of-the-art methods [35, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 119], the proposed methods have to use these model-dependent metrics. However, when 3D models are not available, model-independent metrics such as rotational error (RE) and translational error (TE), could be acceptable alternatives.

Therefore, although the use of 3D models is inevitable in image synthesis and pose evaluation for the aforementioned reasons, the proposed methods can still work when the relevant 3D models are unavailable. In addition, there are always alternative ways to facilitate a completely model-free pose estimation method, such as using model-independent pose evaluation metrics and collecting real annotated data.

2.5 Summary

Most state-of-the-art methods for 6-DoF pose estimation typically require an object's 3D model or establish 2D-3D correspondence based on the model. Approaches that rely on 2D-3D correspondence estimate either pixel-wise dense correspondence [67,

68, 69, 70], or a number of sparse keypoints [35, 126, 128, 133]. These methods subsequently estimate the pose through various PnP algorithms [93, 144, 145]. Other approaches either construct functions utilising 3D model points [5, 113, 116], or iteratively match the image rendered from a 3D model at its estimated pose with the observed input image [75, 111]. However, the need for 3D models can be seen as one of the limiting factors for broader real-life applications. It is impractical to construct a 3D model for every object of interest or build the prior 2D-3D correspondence in real time. Different from most existing methods, this thesis introduces three novel methods (Chapter 4) based on the autoencoder's latent space representation, which does not require object 3D models during inference. These methods are trained and tested with colour images only, achieving comparable results to the state-of-the-art without any use of depth measurements and time-consuming post-refinement such as iterative closest point (ICP) [146, 147].

Methods using latent space representation are proposed in [36, 65, 66, 121], categorised as latent representation methods in Section 2.3.3. These methods typically use an LUT based on the learnt latent space representation, which assigns the most likely pose to the test instance. Although these methods also do not require 3D models during inference, the discretisation of the pose representation can lead to notable errors. In contrast, the proposed methods regress the learnt latent space representation to continuous pose representation instead of using LUT, which is reported to outperform existing latent representation methods by a certain margin in Chapter 5.

What's more, in terms of object characterisation, most methods are restricted to object 6-DoF pose only, in which case they are not able to generalise to other attributes of the object. The networks/algorithms have to be retrained or redesigned if other attributes of objects are required, where such attributes, such as object class and shape topology, are also important for specific tasks such as object grasping. Taking into account, for example, a mechanical arm tasked with grasping a mug, it would be beneficial to differentiate between mugs with and without a handle. In the

proposed CVML-Pose method (see Section 4.3), the learnt latent space representation is also shown to have the potential to be extended to multiple tasks without retraining the main network, thereby reducing computational costs.

Chapter 3

Tools and Methods

3.1 Introduction

This chapter covers the tools and techniques utilised in the proposed 6-DoF pose estimation methods, structured around the themes of data, method, and evaluation. Section 3.2 introduces the adopted datasets and outlines the associated data processing procedures. Section 3.3 delves into pose representations that have been utilised. Furthermore, Section 3.4 illustrates the adopted autoencoder models and latent space representation. Lastly, Section 3.5 explains the metrics used for the evaluation of pose estimation. For details of the proposed methods, please refer to Chapter 4.

3.2 Datasets

This section presents several publicly available datasets used in the thesis, including the Linemod dataset (Section 3.2.1), the Linemod-Occluded dataset (Section 3.2.2), the YCB-Video dataset (Section 3.2.3), and the physically based rendering (PBR) dataset (Section 3.2.4). In addition to these, the MNIST dataset, although not directly related to pose estimation but used for addressing a toy problem, i.e. 2D rotation estimation using a variational autoencoder (detailed in Section 4.3.1), is

also mentioned in Section 3.2.5. An overview of the data processing procedures applied to these datasets can be found in Section 3.2.6.

3.2.1 Linemod

The Linemod dataset [2] (licence: CC BY 4.0[1]) was introduced for objects detection and pose estimation by Hinterstoisser et al. [1]. This dataset contains 15 texture-less household objects (as shown in Figure 3.1) characterised by distinct shapes and colours. Each object in the dataset is represented by a series of images, each featuring a single annotated instance of the object set against a background with considerable clutter, but exhibiting little occlusion.

Figure 3.1: **Object 3D models provided in the Linemod dataset [1, 2].** The fifteen objects, in order, are ape, benchvise, bowl, cam, can, cat, cup, driller, duck, eggbox, glue, holepuncher, iron, lamp, and phone. Objects with the $*$ symbol refer to those also contained in the Linemod-Occluded dataset [3, 4]. The 3D rendering is implemented using the Pyrender software [9].

In the original version of the Linemod dataset, the authors provide 15 coloured 3D models for training, e.g. render synthetic training images, and over 1000 real images for each object from different viewpoints, complete with corresponding ground truth (gt) 6-DoF poses for test. Specifically, the test images are captured from uniformly distributed views, ranging from 0° to 360° around the object, 0° to 90° tilt rotation, 65 cm to 115 cm scaling, and ±45° in-plane rotation. However, in [3], the authors omit 2 objects due to incorrect 3D models (as illustrated in Figure 3.2), resulting in

[1]Licence link: https://creativecommons.org/licenses/by/4.0/.

a total of 13 objects. Furthermore, to reduce the dependency on synthetic training data, the original Linemod real images are split into training and test sets. The training images are sampled with at least 15° angular distance to roughly cover the upper hemisphere, while the remaining real images are allocated to the test set. This data split has been widely adopted by numerous state-of-the-art methods [3, 35, 50, 65, 67, 70, 119, 126, 128]. Although the exact composition of the training set varies, e.g. the AAE method [65] uses only rendered images for training, the test data remain consistent across the methods to enable a fair comparison.

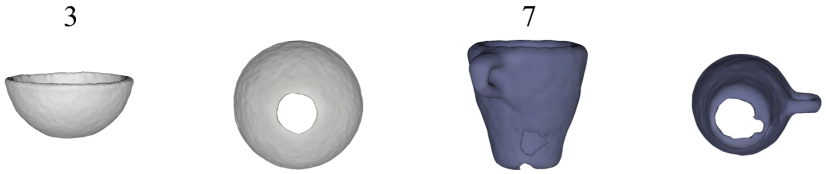


Figure 3.2: **Incorrect 3D models in the Linemod dataset [1, 2].** The 3rd and 7th objects (bowl and cup) are dismissed in most approaches due to their wrong 3D models. The 3D rendering is implemented using the Pyrender software [9].

3.2.2 Linemod-Occluded

Similar to the Linemod dataset, the Linemod-Occluded dataset [4] (licence: CC BY-SA 4.0[2]) was built alongside a pose estimation algorithm proposed by Brachmann et al. [3]. It provides additional gt annotations for a total of 8 objects (marked with a star in Figure 3.1) in the benchvise test sequence from Linemod. Characterised by various levels of occlusion, the Linemod-Occluded dataset presents a range of challenging test scenarios, which has been extensively used for evaluation in many state-of-the-art approaches [5, 35, 68, 69, 70], and it has also been chosen as one of the main datasets in the BOP Challenge [7, 26, 141].

In the thesis, both Linemod and Linemod-Occluded datasets are chosen for evaluation. Linemod is a widely used benchmark dataset for many state-of-the-art pose estimation approaches, while Linemod-Occluded introduces strong occlusions that increase the difficulty in pose estimation. To facilitate comparison with methods

[2]Licence link: https://creativecommons.org/licenses/by-sa/4.0/.

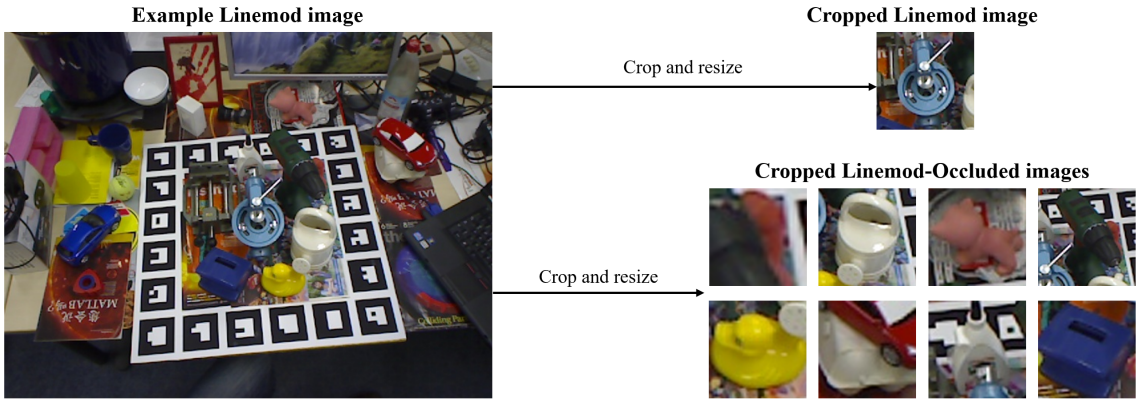


Figure 3.3: **Comparison between Linemod [1, 2] and Linemod-Occluded [3, 4].** The Linemod dataset contains 15 test sequences, and each of the sequences is associated with only one object of interest. The Linemod-Occluded dataset contains only one test sequence, but each of the sequences is associated with up to eight objects of interest with different levels of occlusion.

participating in the BOP Challenge, the BOP version [7] of the Linemod-Occluded dataset, a subset manually selected by the challenge organisers, is used for evaluation of the proposed methods. Besides, since the objects in Linemod-Occluded are a subset of the objects in Linemod, the same network (weights) trained for each Linemod object can also be applied to the corresponding objects in Linemod-Occluded, enhancing computational efficiency. Moreover, Linemod and Linemod-Occluded data are considered heterogeneous due to the distinct shape and appearance of the objects. The images of the objects, both with and without occlusion and clutter, exhibit very different characteristics, varied illumination and shadow patterns.

3.2.3 YCB-Video

Like Linemod and Linemod-Occluded, the YCB-Video dataset [6] (licence: MIT[3]) also came with a 6-DoF pose estimation approach proposed by Xiang et al. [5]. The dataset comprises 92 video sequences, with 80 designated for training and the remaining 12 for test, which encompasses a total of 133,827 images featuring 21 objects (see Figure 3.4) selected from the YCB dataset [148, 149, 150]. To facilitate deep learning-based approaches, the authors additionally generate 80K synthetic training images by rendering the objects' 3D models. The dataset features both

[3]Licence link: https://opensource.org/license/mit/.

textured and texture-less objects, often occluded with varying levels in cluttered scenes.

Figure 3.4: **Object 3D models provided in the YCB-Video dataset [5, 6].** The images of the 3D models are rendered using the Pyrender software [9].

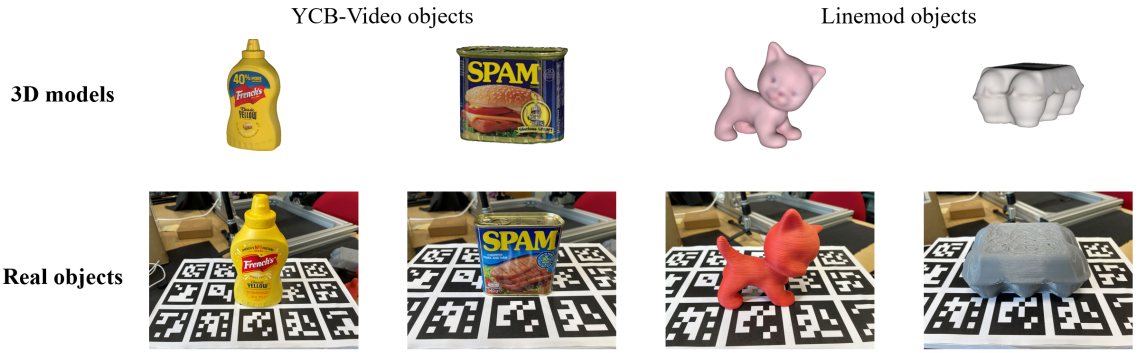


Figure 3.5: **Comparison between real objects in the YCB-Video [5, 6] and Linemod [1, 2] datasets.** Most objects in the YCB-Video dataset are easily accessed in life, e.g. the mustard and spam objects, though there may be subtle differences in appearance between 3D models and real objects. Most Linemod objects, e.g. the cat and eggbox, need to be 3D printed from the 3D models, and their appearance can be affected by the colour of printing materials and the capability of the printer. The images of the 3D models are rendered using the Pyrender software [9].

The test images in the YCB-Video dataset present a variety of challenging cases, where each of the test sequences is associated with 3 to 9 objects of interest with different levels of occlusion and clutter. This dataset has been used by many state-of-the-art approaches [5, 35, 69, 75], and chosen as one of the main datasets in the BOP Challenge [7, 26, 141]. Similar to the Linemod-Occluded dataset, the BOP version of the YCB-Video dataset is used for evaluation, to facilitate comparison with the state-of-the-art. The BOP version subset contains only a folder of 75 images that are manually selected for each of the 12 test scenes, to remove redundancies and to avoid images with inaccurate gt poses. Another reason for evaluating the YCB-Video

dataset is that most objects are common in life and easily accessible, facilitating real-time demonstration. The Linemod real objects are difficult to acquire, although they can be built using additive manufacturing techniques based on the provided 3D models, as well as further polishing and painting (see Figure 3.5).

3.2.4 The BOP Challenge Dataset

The Benchmark for 6D[4] Object Pose Estimation (BOP) Challenge [7, 26, 141] is a well-organised annual challenge that aims to continuously report the state-of-the-art in estimating rigid object's 6-DoF pose from RGB/RGB-D images. The challenge provides 12 publicly available datasets[5], including Linemod [1], Linemod-Occluded [3], T-LESS [151], MVTec ITODD [152], HomebrewedDB [153], NVIDIA Household Objects for Pose Estimation [154], YCB-Video [5], Rutgers APC [155], IC-BIN [156], IC-MI [157], TUD Light [141], and Toyota Light [141]. Each is provided in the unified BOP format[6] and including the object's 3D model with training and test images annotated with gt poses (example procedure of obtaining gt poses can be found in [151]). The objects' coordinate system is defined in the datasets as a right-handed system, with the coordinate origin coinciding with the centre of the 3D bounding box of the object 3D model. For objects not included in the dataset, the coordinate origin can similarly be set at the centre of the 3D bounding box, and axes can be defined arbitrarily, e.g. the axis along the largest dimension of the bounding box can be designated as the primary axis (x-axis). The training images are either captured by RGB-D/Gray-D sensors or obtained by rendering the object model, while the test images are captured in various scenes, each characterised by graded complexity, often involving significant clutter and occlusion.

In the BOP Challenge 2019 (BOP19), classical point pair methods [96, 158] still outperformed deep learning-based methods [65, 67, 68, 70], and Hodan et al. [7]

[4]6D here means 6 degrees of freedom not 6-dimensional.

[5]The datasets and the licence of each can be accessed through: https://bop.felk.cvut.cz/datasets/.

[6]Description of the format: https://github.com/thodan/bop_toolkit/blob/master/docs/bop_datasets_format.md.

pointed out the two main challenges for deep learning-based approaches:

- an insufficient number of real training images with annotations.
- a large domain gap between real test images and commonly used synthetic training images (objects rendered on random backgrounds).

Method name	PBR images	non-PBR images	real images	AR_{score}
CDPN [70]	✓			0.569
		✓		0.374
CDPNv2 [70]	✓			0.624
	✓		✓	0.624
EPOS [69]	✓			0.547
		✓		0.443
CosyPose [75]	✓			0.633
	✓	✓	✓	0.633
Pix2Pose [68]	✓			0.281
		✓		0.077

Table 3.1: **Approaches that benefit from the Linemod PBR images [1, 2, 7, 8].** Each approach has two entries with different training data and tests on the same BOP version of the Linemod-Occluded data [3, 4]. The performance score AR_{score} is calculated from the pose error metrics proposed in the BOP Challenge 2020 [7] (see Section 3.5). Non-PBR images refer to synthetic images with non-PBR techniques. For example, CDPN [70] gets $AR_{score} = 0.374$ with non-PBR images, while it achieves $AR_{score} = 0.569$ using only PBR images.

Method name	PBR images	non-PBR images	real images	AR_{score}
CDPN [70]	✓		✓	0.457
			✓	0.422
CDPNv2 [70]	✓			0.390
	✓		✓	0.532
EPOS [69]	✓			0.499
		✓	✓	0.696
CosyPose [75]	✓			0.574
	✓	✓	✓	0.821

Table 3.2: **Approaches that benefit from the PBR, non-PBR, and real images of the YCB-Video dataset [5, 6, 7, 8].** Each approach has two entries with different training data and tests on the same BOP version of the YCB-Video test data. As an example, CosyPose [75] gets $AR_{score} = 0.821$ with all three types of data, while it only achieves $AR_{score} = 0.574$ using PBR images.

In the BOP Challenge 2020 (BOP20) [7], participants were given access to additional 350,000 physically based rendering (PBR) images [142, 143] as part of their training data. This was intended to mitigate the substantial domain gap between

synthetic training images and real-world test images [159, 160, 161]. Subsequently, deep learning-based methods [68, 75] have caught up with the point pair methods from BOP20. Approaches that benefit from the Linemod PBR images and their corresponding results reported in the BOP20 are summarised in Table 3.1. For instance, CDPN [70], Pix2Pose [68], and EPOS [69] show obvious improvements of 19.5%, 20.4%, and 10.4% in AR_{score}, respectively, when using only PBR images, compared to when they use only non-PBR images (their own synthesised images with non-PBR techniques). CosyPose [75] and CDPNv2 [70] achieve competitive results with PBR images only, demonstrating the effectiveness of the PBR technique.

However, for the YCB-Video dataset, not only the PBR images but also other types of images can be used for training as well. As shown in Table 3.2, based on the reported results of the state-of-the-art methods [69, 70, 75] that participated in the BOP20, the real images are beneficial for training, providing as much or even better generalisation ability as the PBR images. For example, using only real images, CDPN achieves competitive results (3.5% lower in AR_{score}) compared to those obtained when using both PBR and real images. Its extended version, CDPNv2, also significantly improves the results (14.2% better in AR_{score}) using real images. Training with a combination of non-PBR synthetic and real images boosts the performance of EPOS (19.7% better in AR_{score}) and CosyPose (24.7% better in AR_{score}). Although there is no definitive evidence that non-PBR synthetic images are beneficial on their own, the two methods achieve much better results when using a combination of data types. This suggests that the proposed methods can be trained with different types of images when dealing with the YCB-Video dataset.

In the proposed methods (see Chapter 4), Linemod and Linemod-Occluded are chosen as the main datasets for benchmarking in evaluation and ablation tests. The corresponding Linemod PBR images are used for training instead of real images because, as demonstrated in Table 3.1, they provide a good enough generalisation for many state-of-the-art approaches to perform well on real test images. In addition, in the related task of object detection, the authors in [142] train a Faster

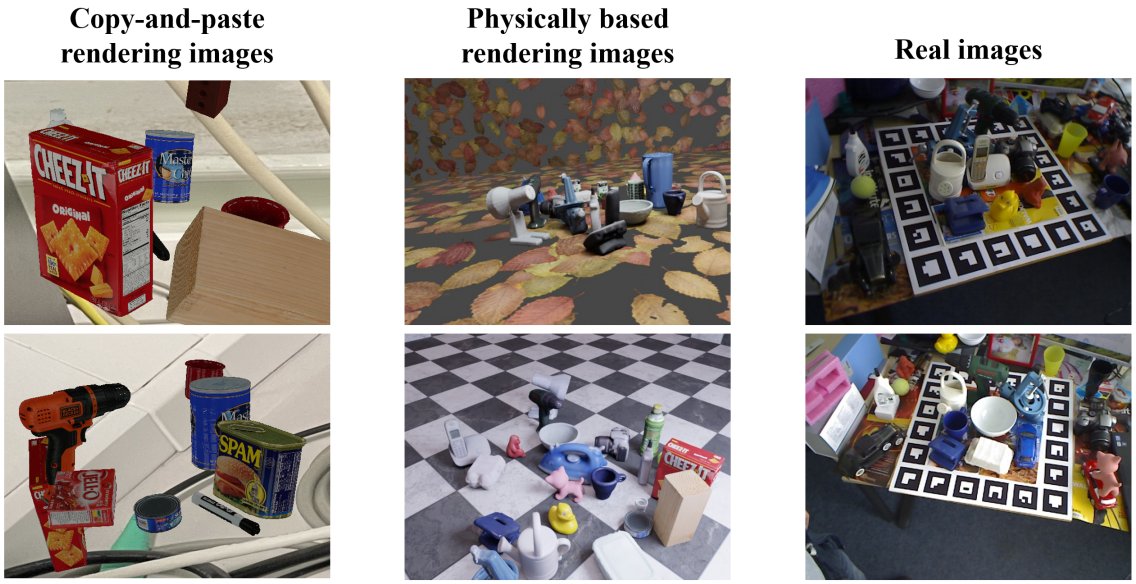


Figure 3.6: **Examples of the copy-and-paste, PBR, and real images.** The copy-and-paste images are generated using the YCB-Video dataset [5, 6], the PBR images are taken from the Linemod PBR dataset [1, 2, 7, 8], and the real images are taken from the Linemod dataset [1, 2].

R-CNN detector [162] using the Linemod PBR images. This approach demonstrates promising results, particularly when compared to the common practice of training with copy-and-paste images, which are generated by rendering 3D models onto random scene images. The YCB-Video dataset is selected as supplementary data, which can also be used to investigate whether the proposed methods are scalable across different datasets. The corresponding PBR images, non-PBR synthetic images, and real images of the YCB-Video dataset are all considered for training.

3.2.5 Digits dataset

MNIST (Modified National Institute of Standards and Technology) dataset [163, 164] is a widely recognised benchmark in image classification (licence: CC BY-SA 3.0[7]). It consists of 60,000 training images and 10,000 test images of handwritten digits, each accompanied by a label indicating the digit (ranging from 0 to 9). The MATLAB[8] digits dataset [165] includes 10,000 synthetic greyscale images of digits. Each image is 28-by-28 pixels and has two associated labels denoting the digit it

[7]Licence link: https://creativecommons.org/licenses/by-sa/3.0/.
[8]UCLan academic licence.

represents (0–9) and the angle of rotation ($[-45°, 45°]$) applied to the image.

Although the digits are not typically used in object 6-DoF pose estimation, because they naturally exist in 2D rather than 3D, the 2D rotation is a special case in SO(3). If the proposed method can successfully predict the 2D in-plane rotation, there is potential for extending it to 3D rotation estimation. This forms a toy problem in the proposed CVML-Pose method, exploring whether the latent space in a variational autoencoder is capable of encoding 2D rotations (will be introduced later Section 4.3.1).

3.2.6 Data Preprocessing for 6-DoF pose estimation

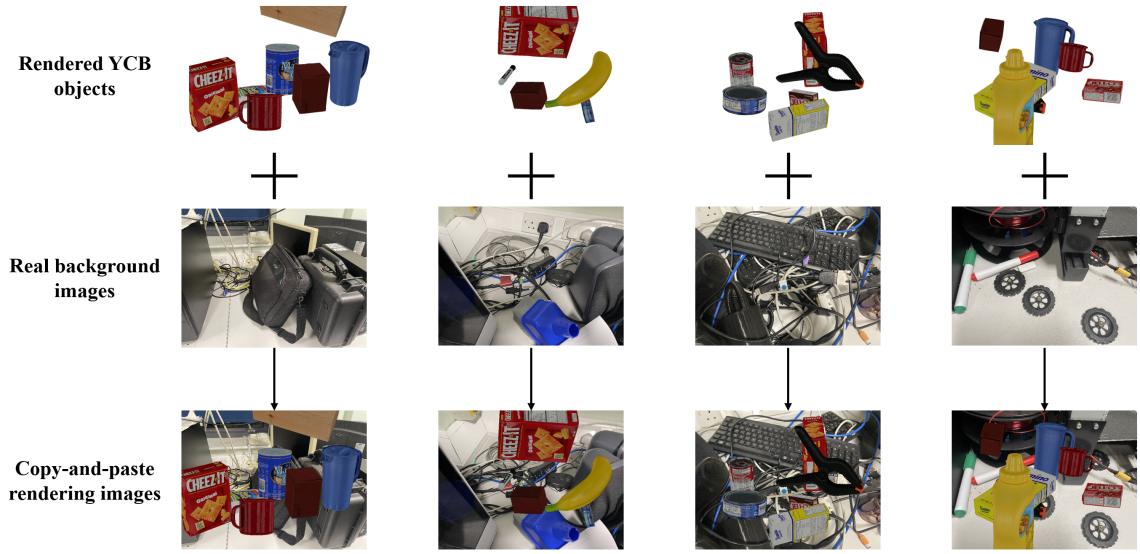


Figure 3.7: **Example procedure of replacing background with a real scene in the synthetic YCB-Video training data [5, 6].** The real scene images come from random captured photos by a cell phone, to replace the original uniform background.

As illustrated in previous sections, the Linemod test set, the BOP version of the Linemod-Occluded dataset, and the BOP version of the YCB-Video dataset, are selected for training and evaluation of the proposed methods (Chapter 4). For Linemod and Linemod-Occluded, the corresponding PBR images are employed for training as the PBR technique has been shown to enable better network performance than training with real and/or non-PBR synthetic images, as evidenced by the results reported in Table 3.1. For the YCB-Video dataset, a combination of PBR,

non-PBR synthetic, and real images is selected, as this combination can yield better performance compared to using only PBR images. In particular, the non-PBR synthetic images of the YCB-Video dataset only have a uniform background, which can be less generalisable to the real test data. Following the copy-and-paste pipeline illustrated in [142], the rendered images of objects are pasted on top of the real scene images from the NYU Depth Dataset V2 [166, 167], and the example procedure is shown in Figure 3.7.

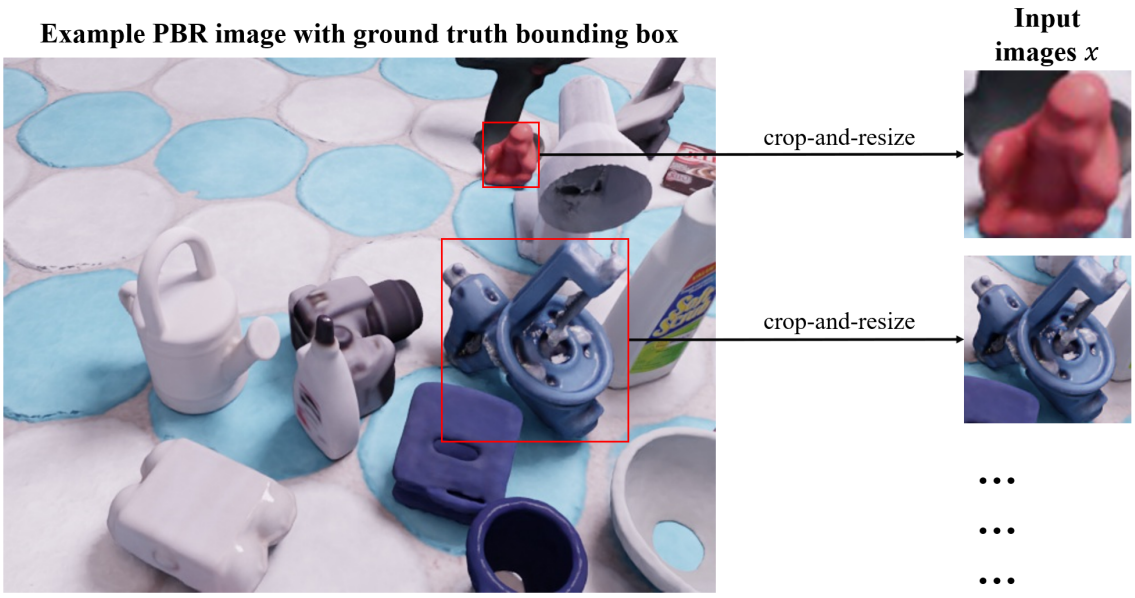


Figure 3.8: **Data preprocessing step one: crop-and-resize.** All the gt information including object 6-DoF pose, bounding box, and object visibility are provided by the Linemod PBR dataset [1, 2, 7, 8].

To prepare the training data, i.e. images to be fed into the autoencoder models (see Chapter 4), a crop-and-resize strategy is applied. For instance, as demonstrated with the Linemod PBR images shown in Figure 3.8, each image contains several Linemod objects. The crop-and-resize strategy first crops each target object into a square shape from the scene using the provided gt bounding box, with the square's size defined as the longer side of the bounding box. The cropped images of objects are then resized to $128 \times 128 \times 3$ using bicubic interpolation to match the autoencoder network's input size. Following the crop-and-resize strategy, approximately 50,000 images of objects can be achieved for each Linemod object.

Among these images, a fair amount is considered "bad data", due to nearly invisible

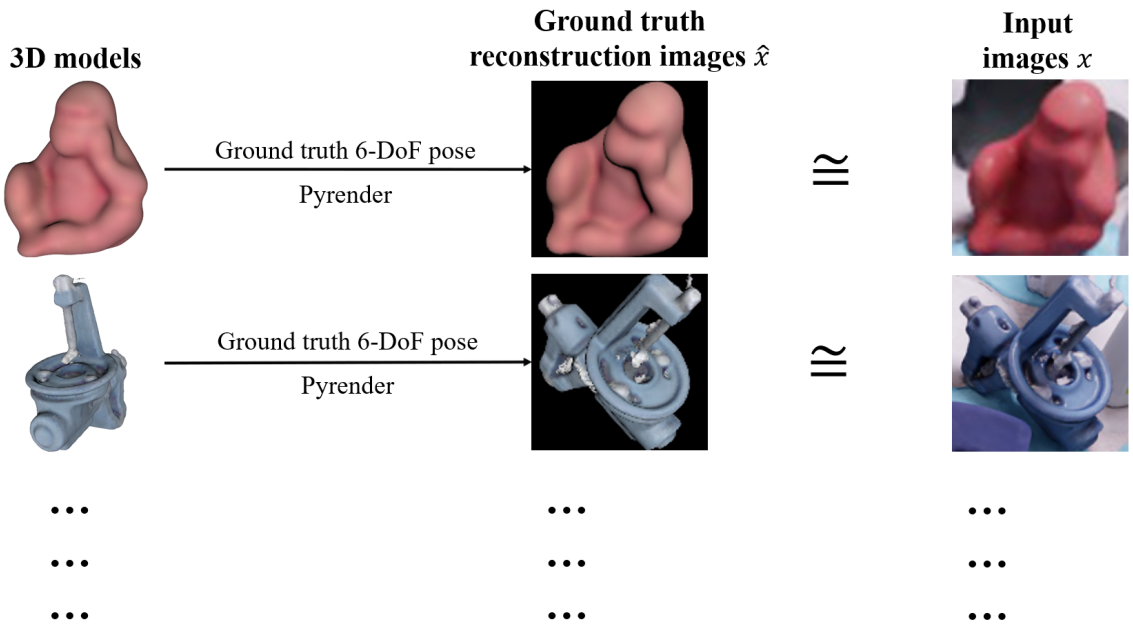


Figure 3.9: **Data preprocessing step two: data generation for the gt reconstruction images.** Each gt reconstruction image $\hat{x}_i$ is identical to the input image x_i in terms of the 6-DoF pose but slightly different in background and appearance, where $\hat{x}_i$ shows a complete object without any background or occlusions, x_i shows various backgrounds and may be occluded or truncated. The images are taken from the Linemod dataset [1, 2]. The gt reconstruction images are rendered using the Pyrender software [9].

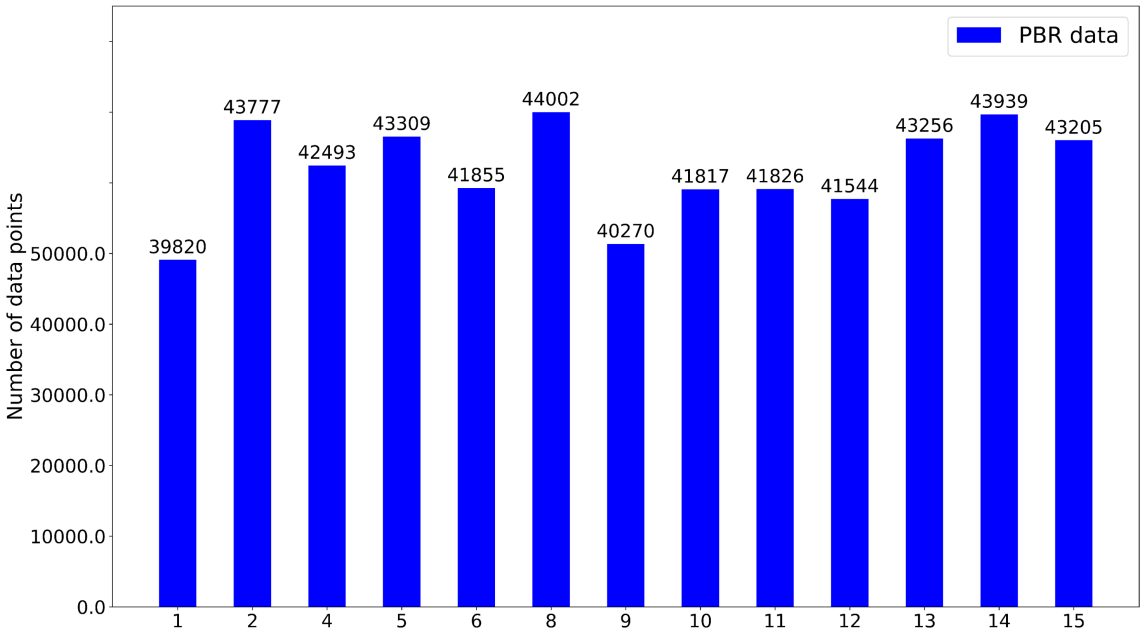
in the scene from occlusion or truncation (see examples in Figure 3.10 and A.1. Since the BOP Challenge only evaluates images of objects with visibility of at least 10% area in the scene, the same criterion is applied, i.e. images of objects that are heavily obscured (less than 10% area of the object is visible) are omitted, and eventually around 40,000 images per object are obtained as the final input x. To prevent overfitting, 90% of these images are randomly allocated to the training set, and the remaining 10% to the validation set for each object. An overview of the training data distribution for Linemod and YCB-Video objects is provided in Figure 3.11.

Another set of data, called the ground truth (gt) reconstruction data, is required for training the autoencoder model (will be explained later in Section 3.4). Unlike typical self-supervised learning where the input data can be treated as the gt reconstruction data, it is not feasible here due to the inherent noise in the input data, such as background, occlusion, and truncation. Therefore, to build the gt reconstruction dataset, an easy-to-use non-PBR rendering software, Pyrender [9], is used to syn-

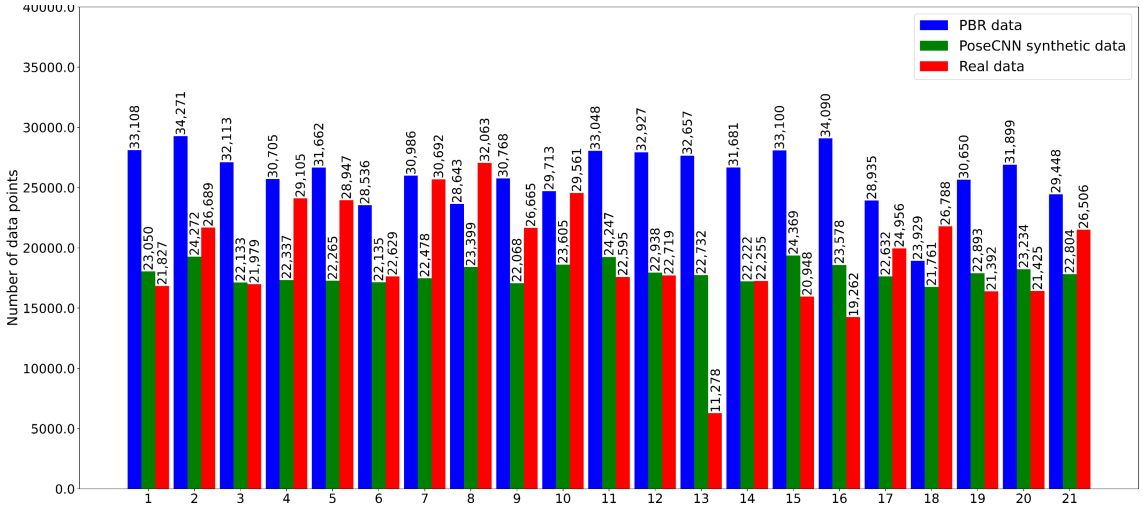
Figure 3.10: **Example bad case for the training data.** The object visibility information is provided by the Linemod PBR dataset, which is defined as the proportion of the number of visible pixels of the object. The images are taken from the Linemod PBR dataset [1, 2, 7, 8].

thesise images of objects with a clean background based on the gt 6-DoF poses and camera intrinsic parameters from the original datasets. Figure 3.9 illustrates the preparation of gt reconstruction images. Please notice that the gt reconstruction images $\hat{x}$ are different from the input images x of the object of interest. $\hat{x}$ shows a complete object without any background or occlusions, which could be possibly present in the original input image x.

To build the test set, the same crop-and-resize strategy is applied to the test data of these datasets. A slight difference is that the test set images can be cropped either from the gt or the detection bounding box, while the training images are all cropped from the gt bounding box. For both training and test sets, the bounding box provides not only the location of the object in 2D images, but also helps to estimate the 3D translation $\mathbf{T}$, which will be further explained in Chapter 4. Regarding object detectors, while they are not proposed in the thesis, different state-of-the-art pretrained detectors are critically assessed, to evaluate their impact on the accuracy of pose estimation. The complete evaluation procedure, as well as comparative analysis and evaluation of these detectors, are presented in Chapter 5.



(a) Linemod



(b) YCB-Video

Figure 3.11: **Distribution of the processed training data.** The training data for the Linemod objects only contain PBR images, while the data for the YCB-Video objects consist of PBR, own synthetic, and real images.

3.3 Pose Representations

An object's 6-DoF pose consists of two components: 3D rotation $\mathbf{R} \in \mathrm{SO}(3)$ and 3D translation $\mathbf{T} \in \mathbb{R}^3$. The 3D translation can be interpreted as 3D displacement in the camera coordinate system, which is typically denoted as $\mathbf{T} = (T_x, T_y, T_z)^T$. Here, T_z is the 2D projective distance from the camera to the object's projective centre, while T_x and T_y are the offsets along the x and y axis in 3D space, respectively. The 3D rotation, representing rotational motion in 3D space, is more complicated than 3D translation due to various rotation representations such as rotation matrix, axis-angle, and unit quaternion.

In many deep learning-based pose estimation methods (explained in Section 2.3), directly regressing a 3×3 rotation matrix is often not the first choice. The main reason is that the output from CNNs may not be a valid rotation i.e. not orthogonal, necessitating the Gram-Schmidt orthogonalisation process during network training. Alternatively, more compact rotation representations, such as axis-angle and unit quaternion, are typically preferred, as exemplified in [168]. However, as demonstrated by Zhou et al. [118], all rotation representations are discontinuous in four or fewer dimensions, i.e. these commonly used representations are not ideal for 3D rotation regression because of the discontinuity issue. To address this, they propose a continuous representation in six dimensions, which has been shown to outperform lower-dimensional representations.

In this section, several different rotation representations including rotation matrix (Section 3.3.1), Euler angles (Section 3.3.2), axis-angle (Section 3.3.3), unit quaternion (Section 3.3.4), and continuous rotation representation (Section 3.3.5), are reviewed in terms of their advantages and disadvantages in rotation regression. It should be noted that using different pose representations will only affect methods that directly regress 6-DoF poses from CNNs. The indirect methods (Section 2.3.2), which use 2D-3D correspondence information, can naturally get around the pose representation issue because of the PnP algorithm. Some latent representation methods

(Section 2.3.3) can also ignore this problem by finding the nearest neighbours or computing observation likelihoods instead of implementing regression-based algorithms.

3.3.1 Rotation Matrix

A rotation matrix $R \in \mathbb{R}^{n\times n}$ is used to perform a rotation in n-dimensional Euclidean space. It is characterised by two properties: it is orthogonal, and its determinant is equal to 1 [125]. Generally, the space of rotation matrices in n-dimensional space can be defined as:

$$SO(n) = \{R \in \mathbb{R}^{n\times n} : RR^T = R^T R = I, \det R = 1\} \tag{3.1}$$

Specifically, the group of all 3×3 matrices that meet the two properties is denoted as SO(3), where SO is the abbreviation of *special orthogonal*.

Regressing a 3×3 rotation matrix from CNNs is not trivial, due to the redundancy in the matrix's nine parameters, compared to other more compact representations like axis-angle and unit quaternion, which have fewer parameters to be optimised. In addition, an orthogonalisation process is essential for the rotation matrix, which can be seen as a constraint to network training through backpropagation. However, the rotation matrix representation provides a fundamental tool for the design of the loss function in the pose regression network. For example, in the work of Mahendran et al. [169], although the authors estimate the 6-DoF pose using CNNs with axis-angle and unit quaternion representations for simplicity, they construct a geodesic loss function (Eq. 3.2) based on the rotation matrix representation, which can be further adapted depending on the representations used in the network.

$$d(R_1, R_2) = \frac{\| \log(R_1 R_2^T) \|_F}{\sqrt{2}} \tag{3.2}$$

where R_1 and R_2 are two rotation matrices, log is the matrix logarithm, $\| \cdot \|_F$ is the Frobenius norm.

3.3.2 Euler Angles

Euler angles are one of representations used to describe a 3D rotation. They can represent any rotation of SO(3) using three ordered rotation angles (α, β, γ) about three axes [170].

The Euler angles representation requires three primary rotations combined to compute the 3D rotation matrices. These rotations are defined around three axes x, y, z as follows:

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix}, R_y(\beta) = \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix}, \\ R_z(\gamma) = \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.3}$$

These rotations can be combined in different sequences, e.g. $z-y-z$, $z-y-x$, $x-y-z$, etc. A subset of these sequences constitutes Cardan angles by strictly combining rotations about three orthogonal axes, for example, the $x-y-z$ or $z-y-x$ sequences. An example Cardan angles representation R can be obtained as follows:

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha) \tag{3.4}$$

These angles around distinct axes are often interpreted as yaw $(R_z(\gamma))$, pitch $(R_y(\beta))$, and roll $(R_x(\alpha))$. Yaw is the rotation around the z axis, similar to turning left or right; pitch is the rotation around the y axis, akin to nodding up and down; and roll describes the rotation around the object's x axis, like tilting side to side. The Cardan angles representation is intuitive for understanding an object's orientation, as it mimics the way humans describe rotations in everyday use.

As matrix multiplication is not commutative, the order in which the rotation of the

three consecutive elements is applied is critical. For example, a rotation R following the $z-y-x$ sequence will differ from rotations in other sequences, as demonstrated in Figure 3.12.

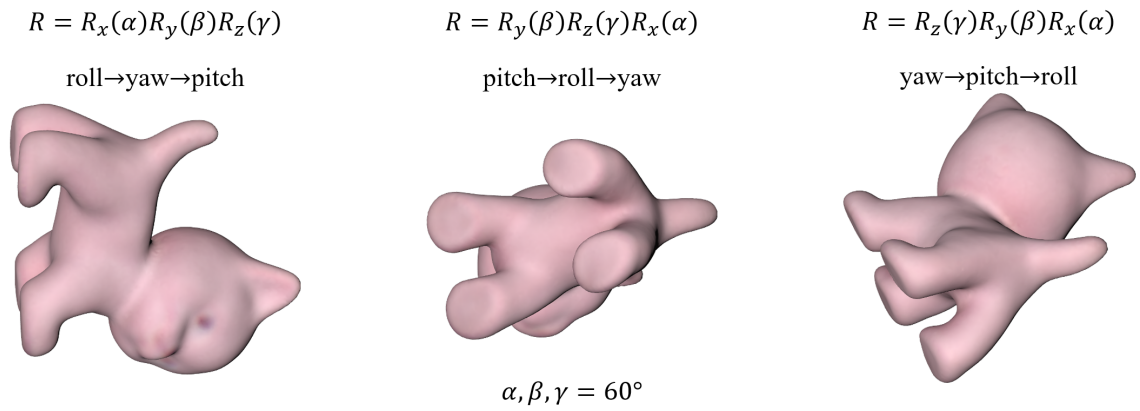


Figure 3.12: **Effects of the rotational orders in Euler angles representation.** The three angles (roll, yaw, pitch) are assigned at the same value but different sequences ($x-y-z$, $y-z-x$, and $z-y-x$), resulting in three different rotations. The 3D model of the Linemod cat object [1, 2] is rotated and rendered at the rotations which formed three different views. The 3D rendering is implemented using the Pyrender software [9].

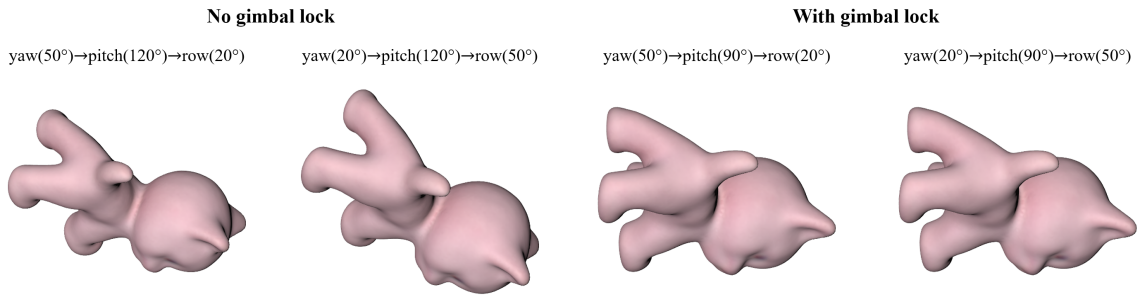


Figure 3.13: **Gimbal lock visualisation.** The Euler angles are assigned at different values in the $z-y-x$ sequence. The gimbal lock occurs when the pitch angle is locked at 90°, losing one degree of freedom. The 3D model comes from the Linemod dataset [1, 2], and the 3D rendering is implemented using the Pyrender software [9].

One problem of the Euler angles representation is gimbal lock, a phenomenon where two of the three gimbals align, leading to a loss of one degree of freedom. This limitation is demonstrated in Figure 3.13, using the yaw ($R_z(\gamma)$), pitch ($R_y(\beta)$), roll ($R_x(\alpha)$) sequence from Eq. 3.4. where, under the same pitch angle of 120°, different combinations of roll and yaw angles result in different rotations. However, when gimbal lock occurs, specifically at a pitch of 90°, the arbitrary combination of roll and yaw, such that their sum is constant, (e.g. $\alpha = 20°, \gamma = 50°$ and

$\alpha = 50°, \gamma = 20°$) will produce the same 3D rotation. Gimbal lock can have severe consequences in real-life applications, such as in aircraft flight control. For instance, if the roll and yaw gimbals become parallel, it will cause the loss of one degree of freedom, possibly compromising the capability to perform certain manoeuvres. Alternative representations such as unit quaternion and axis-angle, can avoid the gimbal lock problem, and provide a more efficient way for 3D rotation representation and regression.

3.3.3 Axis-angle

According to Euler's rotation theorem, any rotation of a rigid object in the 3D space is equivalent to a rotation about a fixed axis, which is commonly referred to as the axis-angle representation [171]. As the name suggests, this representation characterises a rotation in SO(3) as a combination of a unit vector e (the axis of rotation) and the angle of rotation θ about the axis. It is popular in many regression-based 6-DoF pose estimation approaches [116, 169, 172] for its representation simplicity. An axis-angle representation is defined as:

$$R(e, \theta) = \left(\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}, \theta \right) \tag{3.5}$$

However, the axis-angle representation has several disadvantages. For example, the implementation of rotation in the axis-angle representation requires application of the Rodrigues' rotation formula [123, 124, 125], and a superposition of two rotations, in that representation, is somewhat complicated. When $\theta = 0$, e can be arbitrary. Moreover, a many-to-one problem exists (see Eq. 3.6), though this is not unique to the axis-angle representation.

$$\begin{aligned} R(e, \theta) &= R(-e, -\theta) \\ R(e, \theta) &= R(e, 2k\pi + \theta), \forall k \in \mathbb{Z} \end{aligned} \tag{3.6}$$

3.3.4 Unit Quaternion

In the 6-DoF pose estimation, unit quaternion [173] is another popular representation for describing object rotation in the 3D space, which has been used in numerous state-of-the-art approaches [5, 111, 113, 174]. It is particularly useful for representing rotation because it avoids complexities and ambiguities associated with other representations like Euler angles. Also, different from the axis-angle representation, the unit quaternion representation gives a global reparameterisation of SO(3). One of the advantages of this representation is the simplicity with which it implements superposition of rotations (computed as multiplication of corresponding unit quaternion) when using quaternion algebra. A unit quaternion Q is defined as:

$$\begin{aligned} Q &= q_0 + q_1 i + q_2 j + q_3 k \quad q_i \in \mathbb{R}, i = 0, 1, 2, 3 \\ \|Q\| &= \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} = 1 \end{aligned} \tag{3.7}$$

where q_0 is the scalar component of Q, (q_1, q_2, q_3) is the vector component, and following formulas apply:

$$\begin{aligned} i^2 = j^2 = k^2 = ijk = -1, \\ ij = k = -ji, \\ jk = i = -kj \\ ki = j = -ik. \end{aligned} \tag{3.8}$$

Assuming that the rotation is given in an axis-angle representation as $R(e, \theta)$, the corresponding unit quaternion Q can be calculated as follows:

$$\begin{aligned} q_0 &= \cos\left(\frac{\theta}{2}\right) \\ q_1 &= e_x \sin\left(\frac{\theta}{2}\right) \\ q_2 &= e_y \sin\left(\frac{\theta}{2}\right) \\ q_3 &= e_z \sin\left(\frac{\theta}{2}\right) \end{aligned} \tag{3.9}$$

Despite its flexibility, the unit quaternion representation suffers from the problem of antipodal symmetry [175, 176], also known as "double cover", where Q and $-Q$ represent the same rotation, which means that it may end up with completely opposite values of unit quaternion representation for two identical rotations. To ensure uniqueness, some approaches, like that of Wu et al. [113], constrain the real component q_0 to be non-negative in the proposed network, restricting the rotation angle in the range of $(0, \pi)$. However, this does not completely resolve the issue, as similar rotations may still be far apart in this representation [176].

Furthermore, empirical results from Zhou et al. [118] suggest that all 3D rotation representations are discontinuous in real Euclidean spaces of four or fewer dimensions, and continuous representations are generally preferred. This implies that Euler angles, unit quaternion, and axis-angle representations may not be well suited for 3D rotation regression tasks. As an alternative, at least 5D or higher dimensions are recommended, which will be discussed in the next section.

3.3.5 Continuous Rotation Representation

As previously discussed in Section 3.3.1, regressing the rotation matrix representation in neural networks requires enforcing the special orthogonal property, which typically involves the Gram-Schmidt process [177]. This process can be challenging for backpropagation. Other representations such as unit quaternion, axis-angle, and Euler angles often suffer from the discontinuity problem, making them less suitable for network training. To address these challenges, several works have explored smoother rotation representations with higher dimensionality [118, 178, 179, 180]. The most representative one is proposed by Zhou et al. [118], which parameterises 3D rotation in six dimensions. This approach involves using the first two columns of the rotation matrix, forming a six-dimensional (6D) vector, as a continuous representation. This transformed vector allows effective reconstruction of the rotation matrix during network training with a process similar to Gram-Schmidt orthogonalisation. Given a rotation matrix $R \in \mathbb{R}^{3\times3}$ and a 6D representation $R_{6D} \in \mathbb{R}^6$, the

transformation between them can be expressed in Eq. 3.10:

$$
\begin{aligned}
R = \left(\begin{bmatrix} | & | & | \\ a_1 & a_2 & a_3 \\ | & | & | \end{bmatrix} \right) \longrightarrow R_{6D} = \begin{bmatrix} | & | \\ a_1 & a_2 \\ | & | \end{bmatrix}, \\
R_{6D} = \left(\begin{bmatrix} | & | \\ \hat{a}_1 & \hat{a}_2 \\ | & | \end{bmatrix} \right) \longrightarrow R = \begin{bmatrix} | & | & | \\ b_1 & b_2 & b_3 \\ | & | & | \end{bmatrix} \\
b_i = \begin{bmatrix} N(\hat{a}_1) & i = 1 \\ N(\hat{a}_2 - (b_1 \cdot \hat{a}_2) b_1) & i = 2 \\ b_1 \times b_2 & i = 3 \end{bmatrix}
\end{aligned}
\tag{3.10}
$$

where a_1, a_2, a_3 are the three columns of the rotation matrix R, N represents normalisation, $\times$ denotes cross product between two vectors. The two vectors $\hat{a}_1$ and $\hat{a}_2$ should not be parallel to uniquely recover rotation. The detailed derivations can be found in [118].

The continuous 6D rotation representation avoids issues of pose ambiguity and discontinuity, which is beneficial in training CNNs for regression of the 6-DoF pose. For example, PoseCNN [5] uses a unit quaternion representation but receives a certain number of errors at 180° when using the rotational error (RE) metric. Although the authors explain that these errors are attributed mainly to pose ambiguity in symmetric objects, the authors in [118] suggest that the discontinuity issue in unit quaternion might also contribute to such errors. They report similar errors of rotation when using unit quaternion (up to 179.93° when using the RE metric), whereas the continuous 6D rotation representation does not produce errors greater than 2°.

In summary, continuous rotation representations provide an unambiguous and compact way to represent the rotational component in 6-DoF pose estimation. Since they have been already adopted in multiple state-of-the-art approaches and produce promising results, including CosyPose [75] (winner of the BOP20) and GDR-Net [71]

(winner of the BOP Challenge 2022 [26]), it can be concluded as a preferred choice for pose regression tasks.

3.4 Autoencoders and Latent Space Representation

Autoencoders are a type of unsupervised deep neural network that learn to copy input data to output data, facilitated through a low-dimensional latent space representation. Typically, an autoencoder (depicted in Figure 3.14) consists of three main components: the encoder, the decoder, and the latent space. The encoder learns to encode the input data and maps it to a latent space representation, while the decoder maps the representation back to the original input data.

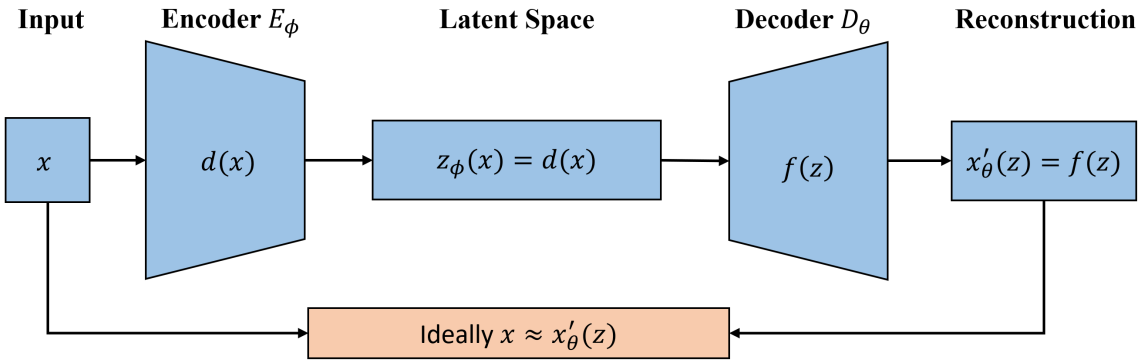


Figure 3.14: **Architecture of an autoencoder model.** An autoencoder model is trained to output $x'_\theta(z)$ that resembles the training data x from the latent code $z_\phi(x)$.

There exist various autoencoder models, including undercomplete autoencoder [181, 182, 183], sparse autoencoder [184, 185], contractive autoencoder [186], denoising autoencoder [62], variational autoencoder [63, 187], etc. Specifically, an undercomplete autoencoder, the most basic of these models, is trained to transform high-dimensional input data into low-dimensional latent space representation, and then to reconstruct the output data from this representation. Similar to other dimensionality reduction methods such as Principal Component Analysis (PCA) [188], it learns a compressed representation of the input that captures the most important features

while discarding redundant information. The training process involves minimising a reconstruction loss function, such as mean squared error or binary cross-entropy, between the input and the reconstructed output. However, in this case, the most salient features of the training data are less important than the reconstruction quality. The undercomplete autoencoder can be prone to overfitting, particularly when the network is allowed with too much capacity, as they learn an identity mapping function [79, 189].

To avoid the overfitting problem, more advanced models like contractive autoencoder, sparse autoencoder, and denoising autoencoder have been developed to learn more robust representations. The following sections delve into these variants. Section 3.4.1 introduces the denoising autoencoder, which is utilised in 6-DoF pose estimation methods using latent representation [65, 66], and is adopted in the proposed DALSR-Pose method (see Section 4.2). The variational autoencoder is the main focus, which is explained afterwards in Section 3.4.2. Although it is less similar to other autoencoder models mentioned above, it is the baseline model that is modified and used in the proposed CVML-Pose method (see Section 4.3). Furthermore, a variant of this, the conditional variational autoencoder, is covered in Section 3.4.3, which is the baseline model of the proposed CVAM-Pose method for multi-object pose estimation (see Section 4.4). Moreover, the concept of latent space representation is explored in Section 3.4.4.

3.4.1 Denoising Autoencoder

Inspired by the fact that humans can recognise partially occluded or corrupted images, Vincent et al. [62] propose the denoising autoencoder (DAE), an unsupervised autoencoder model capable of extracting robust latent space representations from partially corrupted inputs. Different from an undercomplete autoencoder that can learn an identity mapping between original input data and reconstructed output data, a DAE first alters the input data by adding noise as illustrated in Figure 3.15. The network is then trained to reconstruct the clean data from these corrupted in-

puts. By minimising the loss function between the reconstructed output and the original data (uncorrupted), the network learns to identify underlying patterns and features in the input data, as well as enabling effective noise removal.

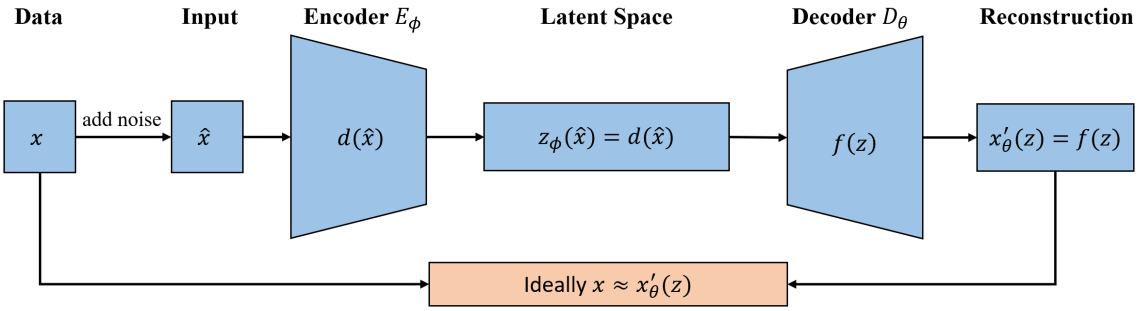


Figure 3.15: **Architecture of a denoising autoencoder model.** A denoising autoencoder model is trained to output $x'_\theta(z)$ that resembles the original data x from the latent code $z_\phi(\hat{x})$, where $z_\phi(\hat{x})$ comes from the corrupted input data $\hat{x}$ instead of x.

In the original DAE paper [62], noise is added by setting a fixed proportion of input values to 0, which shares a similar idea to the dropout technique proposed by Srivastava et al. [190]. By incorporating more complex salt-and-pepper noise, the DAE can be trained to successfully remove noise and reconstruct clean digits, subsequently learning a robust representation of the data in its latent space. This is demonstrated with a DAE model trained on the MNIST digits [163, 164], as shown in Figure 3.16.

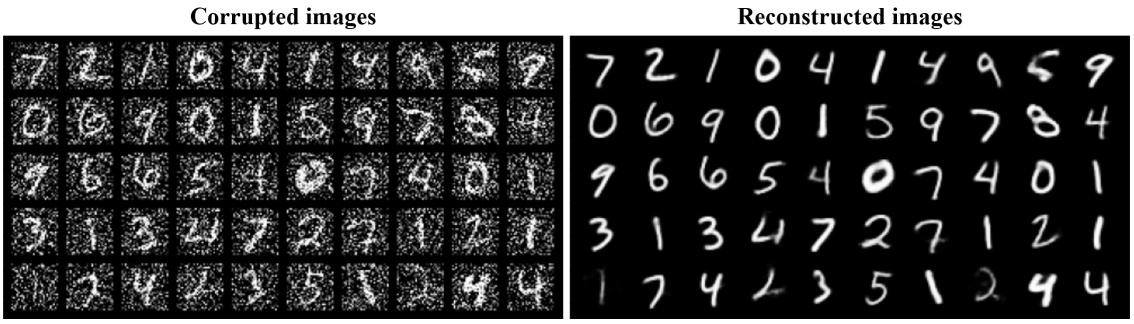


Figure 3.16: **Reconstruction on corrupted data using a denoising autoencoder.** The test digits from the MNIST dataset [163, 164] are corrupted with the salt-and-pepper noise, and the trained model can learn robust features and restore them to the original image.

In the context of object 6-DoF pose estimation, Sundermeyer et al. [65] train a DAE to encode the target object and reconstruct it with a clean background, using

strong data augmentation techniques on images of objects. This training approach forces the network to focus on the object's 3D rotation in the latent space, rather than other redundant information like background and illumination. The proposed DALSR-Pose method (Section 4.2) also trains a DAE model to implicitly learn an object's pose in the latent space. Although employing the same type of autoencoder network, the DALSR-Pose method demonstrates improved pose accuracy compared to existing latent representation methods, with the help of a more effective pose regression approach.

3.4.2 Variational Autoencoder

The variational autoencoder (VAE) is proposed in the context of generative models, which is different from other autoencoder models. The primary objective is to generate a new, typically highly dimensional, data point x (not available in the training data) with the generation process controlled by a low dimensional latent code z, which is randomly drawn from a distribution $p(z)$. This distribution is preferably selected in such a way that the corresponding sampling process is simple to implement.

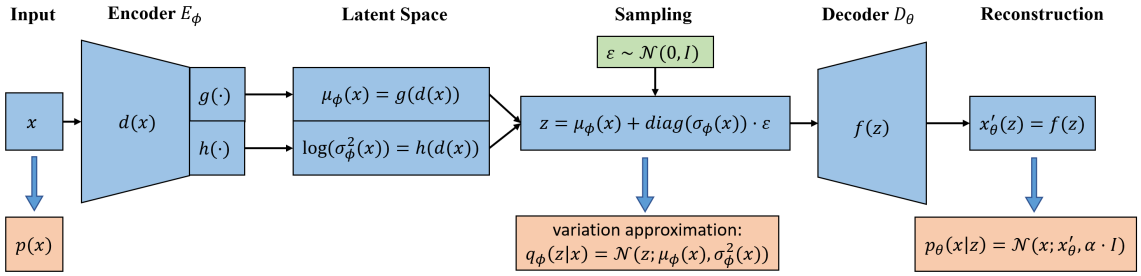


Figure 3.17: **Architecture of a variational autoencoder model.** A variational autoencoder model is trained to approximate the posterior $q_\phi(z|x)$ from the unknown distribution $p(x)$, and generate new data $x'_\theta(z)$ from $p_\theta(x|z)$ with a prior Gaussian distribution.

VAE approximates the unknown distribution $p(x)$ by $p_\theta(x) = \int p_\theta(x|z)p(z)dz$, where parameters θ are selected so x_i from the available training set $\mathcal{D}$ ($x_i \in \mathcal{D} = \{x_i\}_{i=1}^m$) are likely to be drawn from $p_\theta(x)$, i.e. following a maximum likelihood principle. However, computing $p_\theta(x) = \int p_\theta(x, z)dz$ is typically intractable. As detailed in [63,

187], a computationally viable approach replaces the highest likelihood optimisation objective with the evidence lower bound ($ELBO$), and substitutes the intractable posterior inference model $p_\theta(z|x)$ with a tractable variational approximation $q_\phi(z|x)$, where parameters ϕ control the approximation process.

The architecture of a typical VAE, as shown in Figure 3.17, starts with an encoder network $E_\phi(x)$ that estimates the optimal parameters $\mu_\phi(x)$ and $\log(\sigma^2_\phi(x))$ of the assumed posterior model $q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x)) \simeq p_\theta(z|x)$ with the prior $p(z) = \mathcal{N}(z; 0, I)$. Since sampling from $\mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x))$ is not differentiable, a reparameterization trick $z = \mu_\phi(x) + diag(\sigma_\phi(x)) \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$ ensures the sampling stage is differentiable. The decoder network $D_\theta(z)$ estimates the parameters θ of the observation model, assumed to be Gaussian with a diagonal covariance matrix $\alpha \cdot I$, $p_\theta(x|z) = \mathcal{N}(x; x'_\theta(z), \alpha \cdot I)$.
Assuming data points in $\mathcal{D}$ are independent and identically distributed, the $ELBO$ is given as:

$$ELBO(\phi, \theta, \mathcal{D}) = \sum_{x_i \in \mathcal{D}} \Bigg(\mathbb{E}_{q_\phi(z|x_i)} \log p_\theta(x_i|z) - D_{KL}(q_\phi(z|x_i)||p_\theta(z)) \Bigg) \leq \log p_\theta(\mathcal{D}) \quad (3.11)$$

and the parameters of the encoder and decoder in the VAE network in Figure 3.17 are computed using:

$$\hat{\theta}, \hat{\phi} = \arg\max_{\theta,\phi} ELBO(\phi, \theta, \mathcal{D}) \quad (3.12)$$

With the assumed distribution $p_\theta(x|z)$, $q_\phi(z|x)$, and $p_\theta(z)$, the Kullback–Leibler (KL) divergence $D_{KL}(q_\phi(z|x_i)||p_\theta(z))$ has a close form and the $ELBO$ in Eq. 3.12 can be estimated using:

$$ELBO \simeq -c \cdot \sum_{i=1}^{m} \Bigg(||x_i - x'_i||^2 - \alpha \cdot \sum_{j=1}^{n} \Big(1 + \log(\sigma^2_{ij}) - \mu^2_{ij} - \sigma^2_{ij} \Big) \Bigg) \quad (3.13)$$

where c is a positive constant, x_i is the input data, x'_i is the output reconstruction,

$x'_i = x'_\theta(z(x_i))$, μ_{ij} refers to the j element of the vector μ_i, σ^2_{ij} refers to the j element of the vector σ^2_i, $\mu_i = \mu_\phi(x_i)$, $\sigma^2_i = \sigma^2_\phi(x_i)$, m represents the number of data points in the training set, and n refers to the dimensionality of the latent space. Note that the scalar α weighs the KL divergence, which controls the regularisation (smoothness) of the latent space input data representation. The term $||x_i - x'_i||^2$ reflects the reconstruction fidelity between the input training image (data point) x_i and the corresponding output reconstructed image x'_i. A step-by-step derivation of the *ELBO* loss is provided in Appendix C.1. For details of VAE, please refer to [63, 187, 191, 192].

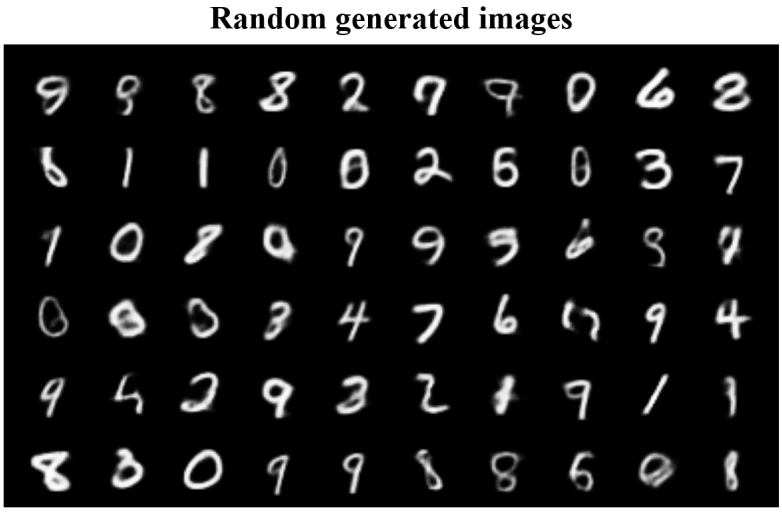


Figure 3.18: **Generate new digits from the Gaussian prior using a variational autoencoder.** During training, the model is trained to learn both the features and distribution of the MNIST digits dataset [163, 164]. During inference, the encoder network can be discarded and the decoder network can be treated as a generator to create new data from the prior $p(z) = \mathcal{N}(z; 0, I)$.

Compared to typical deep learning-based direct methods for pose estimation [5, 113, 116], VAE offers several advantages. For example, the autoencoder architecture can be trained to output a reconstruction image with a clean background, which means the latent space will focus on representing the object itself instead of the irrelevant features. Even with occluded objects, the decoder is trained to output the complete objects, enabling the model to handle occlusion effectively. As will be demonstrated in Section 5.3.2, the proposed CVML-Pose method using VAE, achieves comparable results to the state-of-the-art on the benchmark datasets, even

without using object 3D models. Furthermore, compared to AAE [65], a latent representation method using a DAE, the KL divergence with the reparameterization trick in VAE helps to regularise the latent space to combat overfitting. From the visualisation results shown in Figure 3.18, the trained VAE can reconstruct new digits from the probability distribution of its latent space. Therefore, it is postulated that the representation learned by a VAE can be adopted for the estimation of object attributes including pose, category, and topology.

3.4.3 Conditional Variational Autoencoder

A well-trained VAE is capable of generating new data from the prior distribution $p(z) = \mathcal{N}(z; 0, I)$, but a key limitation is its inability to control the specifics of the data generation process. As illustrated in Figure 3.18, a VAE trained on the MNIST dataset can produce high-quality digit images, but it cannot be instructed to generate a specific one. To solve this problem, Sohn et al. [64] propose a conditional variational autoencoder (CVAE), which extends the VAE framework to incorporate conditional parameters, thus enabling the generation of data with specific characteristics.

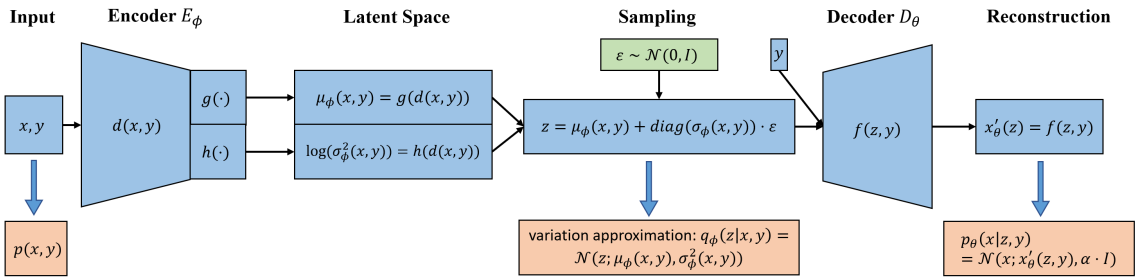


Figure 3.19: **Architecture of a conditional variational autoencoder model.** A conditional variational autoencoder model is trained to approximate the posterior $q_\phi(z|x,y)$ from the conditional distribution $p(x|y)$, and generate new data $x'_\theta(z)$ from $p_\theta(x|z,y)$ with a prior Gaussian distribution.

CVAE operates on a principle similar to VAE but employs an additional condition y to approximate the conditional distribution $p(x|y)$. This approximation is represented as $p_\theta(x|y) = \int p_\theta(x|z,y)p(z|y)dz$, with parameters θ aimed at maximising the likelihood. Like VAE, directly computing $p_\theta(x|y) = \int p_\theta(x,z|y)dz$ is often in-

tractable. As a result, the optimisation objective of highest likelihood is replaced by $ELBO$ (see Eq. 3.12), and the complex posterior inference model $p(z|x,y)$ is approximated by a tractable variational approach $q_\phi(z|x,y)$, parameterised by ϕ.

In the standard CVAE architecture (see Figure 3.19), the encoder network $E_\phi(x,y)$ calculates the optimal parameters $\mu_\phi(x,y)$ and $\log(\sigma^2_\phi(x,y))$ for the approximate posterior model $q_\phi(z|x,y) = \mathcal{N}(z;\mu_\phi(x,y),\sigma^2_\phi(x,y)) \approx p(z|x,y)$, considering the prior $p(z) = \mathcal{N}(z;0,I)$. A reparameterization trick $z = \mu_\phi(x,y) + diag(\sigma_\phi(x,y)) \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0,I)$, is used to ensure differentiability during sampling from $\mathcal{N}(z;\mu_\phi(x,y),\sigma^2_\phi(x,y))$. The decoder network $D_\theta(z,y)$ then predicts the parameters θ for the observation model, which is assumed to be Gaussian with a diagonal covariance matrix $\alpha \cdot I$, $p_\theta(x|z,y) = \mathcal{N}(x;x'_\theta(z,y),\alpha \cdot I)$.

In the proposed CVAM-Pose method (Section 4.4), the CVAE model is trained with the same $ELBO$ loss as derived in Eq. 3.13. The network architecture is slightly modified to capture more high-level features, i.e. the additional condition y is embedded throughout the model. Compared to VAE, CVAE not only inherits the benefits of the autoencoder architecture and the regularised latent space, but also provides enhanced scalability and efficiency for the prediction of multi-object poses. As will be demonstrated in Section 5.3.3, the CVAM-Pose method outperforms the state-of-the-art latent representation methods of both multi-object version [66] and single-object version [65] on the challenging BOP version of the Linemod-Occluded benchmark dataset.

3.4.4 Latent Space Representation

The latent space of an autoencoder model is a lower-dimensional representation of the input data that captures its essential features. The learnt representation can be used for a variety of subtasks such as classification, regression, data visualisation, and data generation.

For example, when training a DAE on MNIST digits, the input data $\hat{x}$ are derived

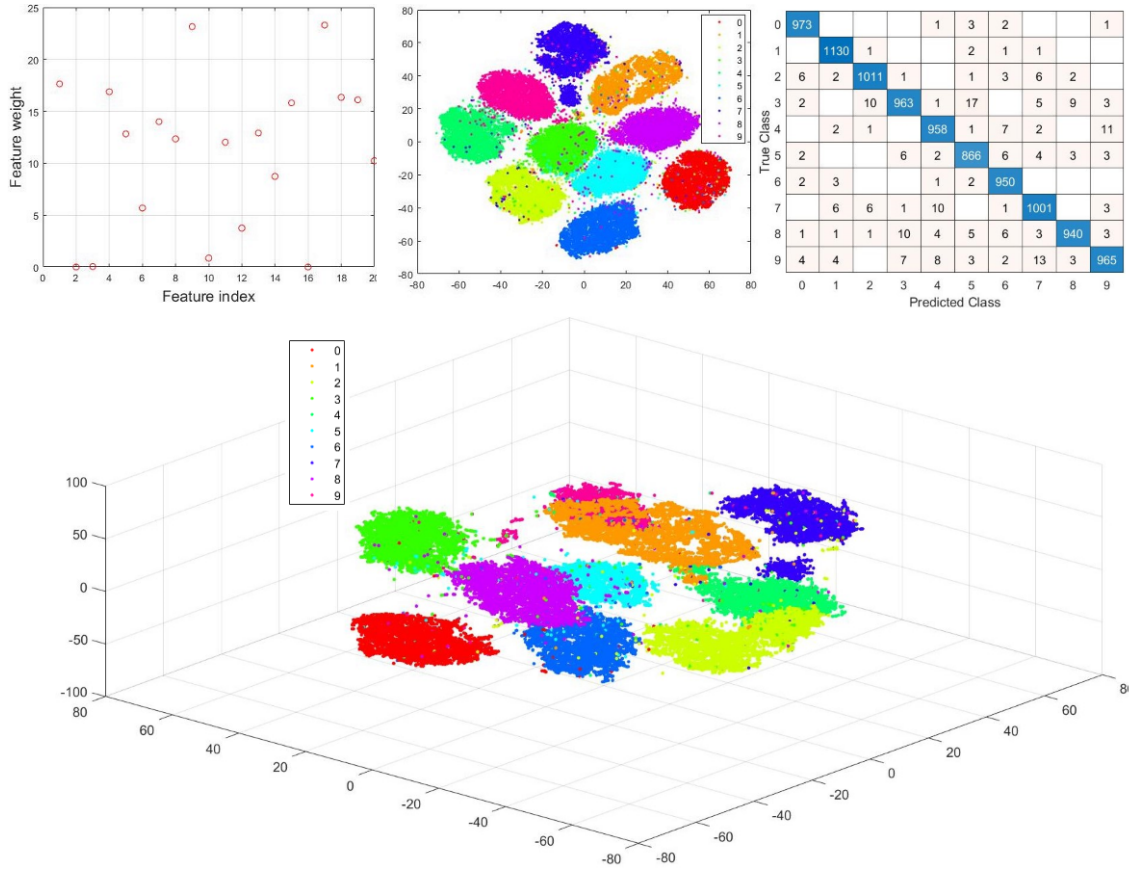


Figure 3.20: **Visualisation of the latent space in variational autoencoder.** The variational autoencoder is trained with MNIST handwritten digits [163, 164] and then applies a KNN classification on the latent space representation $\mu \in \mathbb{R}^{20}$. Upper left: feature weight based on NCA. Upper middle: 2D visualisation clusters based on t-SNE. Upper right: confusion matrix on MNIST test data. Lower Middle: 3D visualisation clusters based on t-SNE.

from corrupting the original data x before being passed to the encoder $E_\phi(x)$. This process results in intermediate representations $z_\phi(\hat{x})$, which are used for the decoder network $D_\theta(z)$ attempts to reconstruct x. Since the DAE can effectively remove noise from the corrupted input, it facilitates the capture of robust intermediate representations suitable for supervised classification [62]. During inference, the latent representation of the test instances can be used in various classification algorithms such as k-nearest neighbours (KNN) [59], support vector machine (SVM) [193], and decision tree (DT) [194]. Similarly, the VAE is also capable of learning robust representations from the input data. For instance, as shown in Figure 3.20, a VAE trained on the MNIST dataset can have its trained latent space visualised using the t-Distributed Stochastic Neighbor Embedding (t-SNE) algorithm [195]. The latent

mean vector $\mu \in \mathbb{R}^{20}$ can be further analysed with the Neighbourhood Component Analysis (NCA) feature selection algorithm [196]. In image classification tasks, the KNN algorithm can be used to find the closest data point and assign its label to the test instance. For data generation, as shown in Figure 3.18, the trained decoder in VAE can produce new data from the Gaussian distribution $p(z) = \mathcal{N}(z; 0, I)$. Regarding regression, a VAE can be trained to estimate the 2D rotation of digits, which will be explored later in Section 4.3.1 as a toy problem for 6-DoF pose estimation.

3.5 Pose Evaluation Metrics

Different from other computer vision tasks such as classification, the evaluation of object 6-DoF pose estimation presents unique challenges. The complexity of the pose itself necessitates specifically designed metrics. In this thesis, the pose evaluation metrics are categorised based on their dependency on the object's 3D model.

Most pose evaluation metrics such as average distance of model points (ADD(I)) [1], visible surface discrepancy (VSD) [7, 141, 197], maximum symmetry-aware distance (MSSD and MSPD) [7] are model dependent. They use a subset of model points from the original object's 3D model. For example, the widely adopted ADD metric measures the distance between sets of the 3D model points at the estimated pose and gt pose. This metric has inspired many model point-based loss functions as described in Section 2.3.1, and is employed for evaluation in many state-of-the-art methods [5, 35, 65, 67, 68, 70, 116, 126, 128]. Another category comprises model-independent metrics such as rotational error (RE) and translational error (TE), which have been used in various works [50, 66, 96, 111, 126, 139, 71, 198, 199]. The RE metric calculates the absolute error in the axis-angle rotation representation, while the TE metric calculates the Euclidean distance between the 3D translation vectors of objects.

The subsequent sections will revisit commonly used pose evaluation metrics. These include rotational error and translational error (Section 3.5.1), average distance of

model points (Section 3.5.2), visible surface discrepancy (Section 3.5.3), and maximum symmetry-aware distance (Section 3.5.4).

3.5.1 Rotational Error and Translational Error

The model-independent metrics, RE and TE, are often jointly used for evaluating 6-DoF pose estimation. The RE metric calculates the absolute error in the axis-angle rotation representations, while the TE metric calculates the L2 norm between translation vectors. Given an estimated pose $\hat{\mathbf{P}} = (\hat{\mathbf{R}}, \hat{\mathbf{T}})$ and the gt pose $\bar{\mathbf{P}} = (\bar{\mathbf{R}}, \bar{\mathbf{T}})$, the pose error can be measured by:

$$\begin{aligned} \text{RE} &= \arccos\left(\frac{\text{Tr}(\hat{\mathbf{R}}\bar{\mathbf{R}}^{-1}) - 1}{2}\right) \\ \text{TE} &= \|\hat{\mathbf{T}} - \bar{\mathbf{T}}\|_2 \end{aligned} \tag{3.14}$$

where arccos is the inverse of the cosine function, Tr is the matrix trace function.

The estimated pose is considered accurate if both the rotational and translational errors fall below specified thresholds. In the work by Drost et al. [96], the thresholds are set to 12° for RE and 10% of the 3D model diameter for translation. Choi and Christensen [198] adopt stricter criteria of 10° and 1 cm, respectively. Many other state of the art approaches [50, 66, 111, 126, 139] prefer more balanced criteria proposed by Shotton et al. [200], often referred to as the 5° 5 cm metric.

However, as highlighted in [197], for specific applications such as robotic grasping, the primary indicator of pose accuracy is the fitness of object surface alignment. In these scenarios, model-independent pose evaluation metrics like RE and TE may not be the preferred choice. Instead, model-dependent metrics that consider the alignment of object surfaces are more relevant for accurately assessing performance in tasks where precise pose estimation is crucial.

3.5.2 Average Distance of Model Points

The average distance of model points (ADD(I)) metric, proposed by Hinterstoisser et al. [1], is a widely used pose evaluation metric with two variants: ADD and ADI. The choice between these variants depends on whether the views of objects are distinguishable. The ADD metric calculates the mean of the point pair distances between the transformed 3D model points at the gt pose and those at the estimated pose, if all views are distinguishable (asymmetric objects). On the other hand, the ADI metric calculates the distances to the closest point when the object has indistinguishable views (symmetric objects). The estimated pose is considered correct when the average distance is less than a specific criterion of the 3D model's diameter (usually 10%). Given the estimated pose $\hat{\mathbf{P}}$ and the gt pose $\bar{\mathbf{P}}$, the point pair distance can be measured by:

$$\begin{aligned} \text{ADD} &= \frac{1}{m} \sum_{x \in M} ||\hat{\mathbf{P}}x - \bar{\mathbf{P}}x||_2 \\ \text{ADI} &= \frac{1}{m} \sum_{x_1 \in M} \min_{x_2 \in M} ||\hat{\mathbf{P}}x_1 - \bar{\mathbf{P}}x_2||_2 \end{aligned} \tag{3.15}$$

where m is the number of model points and M depicts the 3D model vertices.

The ADD(I) metric has significantly influenced many state-of-the-art 6-DoF pose estimation approaches, particularly in the development of model point-based loss functions. As noted in Section 2.3.1, Xiang et al. [5] imitate the ADD and ADI metrics and propose the two corresponding loss functions, PoseLoss and ShapeMatchLoss, respectively, to deal with both asymmetric and symmetric objects. The pose estimation network is trained to iteratively reduce the distance between 3D model points at the gt pose and the corresponding points at the estimated pose, which is identical to the ADD(I) metric.

For the evaluation of Linemod test data (see Table 5.2), following the state-of-the-art approaches [35, 65, 116, 126, 128], the criterion of the ADD(I) metric is established as 10% of the model diameter. The average recall $\text{AR}_{\text{ADD(I)}}$ given in percentage

points is calculated and reported based on the mean precision of the metric for all Linemod objects. This metric is also used to compare the pose estimation results based on the detection bounding box and the gt bounding box (see Table 5.7).

3.5.3 Visible Surface Discrepancy

The ADD(I), RE, and TE metrics are commonly used in many state-of-the-art methods for evaluation. However, these metrics can suffer from the pose ambiguity problem [174, 197, 201]. To address this, Hodan et al. [7] develop three ambiguity-invariant pose evaluation metrics: VSD, MSSD, and MSPD (MSSD and MSPD will be explained in the subsequent section). They are employed to calculate the final performance score AR_{score} for the methods that participated in the BOP Challenge, defined as $\text{AR}_{\text{score}} = (\text{AR}_{\text{VSD}} + \text{AR}_{\text{MSSD}} + \text{AR}_{\text{MSPD}})/3$. One of the main metrics is the VSD, and its error function is shown in Eq. 3.16, given an estimated pose $\hat{\mathbf{P}}$ and the gt pose $\bar{\mathbf{P}}$.

$$\text{e}_{\text{VSD}} = \text{avg}_{p \in \hat{V} \cup \bar{V}} \begin{cases} 0 & \text{if } p \in \hat{V} \cup \bar{V} \wedge |\hat{D}(p) - \bar{D}(p)| < \tau \\ 1 & \text{otherwise} \end{cases} \tag{3.16}$$

where $\hat{D}$ and $\bar{D}$ are the corresponding distance maps for the object 3D model M rendered in the estimated pose $\hat{\mathbf{P}}$ and the gt pose $\bar{\mathbf{P}}$. $\hat{V}$ and $\bar{V}$ are the corresponding visibility masks that can be obtained by comparing the distance maps to the distance map D_I of the test image I, i.e. a set of pixels p where the model M is visible in the image I. τ refers to tolerance to misalignment. The estimated 6-DoF pose is considered correct when the error is less than a specific criterion θ.

In the BOP Challenge 2018 [141] (also known as the SIXD Challenge), a fixed criterion of pose correctness was established, with the correctness threshold $\theta = 0.3$ and the misalignment tolerance $\tau = 20mm$. This criterion has been adopted by several state-of-the-art methods [65, 66, 68, 75, 158]. To fully evaluate the performance of the methods that participated in the BOP20, Hodan et al. [7] employed

multiple combinations of different values for θ and τ. Specifically, the tolerance for misalignment τ ranges from 5% to 50% of the 3D model's diameter with a step of 5%, and the correctness threshold θ ranges from 0.05 to 0.5 with a step of 0.05. The pose accuracy score $\mathrm{AR}_{\mathrm{VSD}}$ is calculated from the average recall rate based on these combinations.

For evaluation of the BOP version of the Linemod-Occluded and YCB-Video datasets (see Table 5.3 and 5.4), the same evaluation criterion proposed in the BOP20 is applied, calculating $\mathrm{AR}_{\mathrm{VSD}}$ based on the average recall of the VSD metric.

3.5.4 Maximum Symmetry-Aware Distance

In addition to VSD, the maximum symmetry-aware distance, including the maximum symmetry-aware surface distance (MSSD) and the maximum symmetry-aware projection distance (MSPD) are also proposed in the BOP20. MSSD is related to robot manipulation as robotic grasping is highly dependent on the maximum surface deviation in 3D. MSPD is suitable for augmented reality applications because it only evaluates the perceivable discrepancy in the projective space. To compare with the state-of-the-art methods, the same BOP20 evaluation criteria are applied, and the results based on the error functions, $\mathrm{e}_{\mathrm{MSSD}}$ and $\mathrm{e}_{\mathrm{MSPD}}$, are reported on the BOP version of the Linemod-Occluded and YCB-Video datasets (see Table 5.3 and 5.4), using the average recall rates $\mathrm{AR}_{\mathrm{MSSD}}$ and $\mathrm{AR}_{\mathrm{MSPD}}$ with different error thresholds θ, i.e. the pose is considered correct if the error function is smaller than the threshold, where θ_{MSSD} ranges from 5% to 50% of the 3D model diameter with a step of 5%, and θ_{MSPD} ranges from $5r$ to $50r$ with a step of $5r$, $r = w/640$, w is the width of image pixels.

Given the estimated pose $\hat{\mathbf{P}}$ and the gt pose $\bar{\mathbf{P}}$, the error functions $\mathrm{e}_{\mathrm{MSSD}}$ and $\mathrm{e}_{\mathrm{MSPD}}$ can be calculated by:

$$\begin{aligned} \mathrm{e}_{\mathrm{MSSD}} &= \min_{\mathbf{S}\in S_M} \max_{x\in M} ||\hat{\mathbf{P}}x - \bar{\mathbf{P}}\mathbf{S}x||_2 \\ \mathrm{e}_{\mathrm{MSPD}} &= \min_{\mathbf{S}\in S_M} \max_{x\in M} ||\mathrm{proj}(\hat{\mathbf{P}}x) - \mathrm{proj}(\bar{\mathbf{P}}\mathbf{S}x)||_2 \end{aligned} \tag{3.17}$$

where S_M is a set of global symmetry transformations of the target object, M denotes a set of model vertices, and the function proj() results in 2D projection in pixels. The global object symmetries can be obtained from the Hausdorff distance, the detailed derivations for which can be found in [7].

3.6 Summary

The tools and techniques outlined in this chapter, including datasets, pose representations, autoencoder models, and evaluation metrics, are closely connected to those discussed in the subsequent chapters. In Chapter 4, which focuses on the methodology, three proposed pose estimation methods, DALSR-Pose, CVML-Pose, and CVAM-Pose, are developed using the three autoencoder models, DAE, VAE, and CVAE, respectively.

Subsequent to the methodology description, in Chapter 5, these methods are evaluated across three benchmark pose datasets: Linemod, Linemod-Occluded, and YCB-Video. These three datasets are selected as the most representative for their comprehensive coverage of challenging scenarios in pose estimation tasks. They are also frequently used for evaluation of pose estimation algorithms. Other datasets, such as the remaining nine datasets from the BOP challenge, are not used mainly due to the prohibitive computational cost of running all simulations on additional datasets. This evaluation features a comprehensive comparison with the state-of-the-art methods, employing various evaluation metrics such as ADD(I), VSD, MSSD, and MSPD, to calculate the final performance scores on each dataset.

Additionally, different pose representations, including axis-angle, unit quaternion, and the continuous 6D representation, are employed in both the ablation experiments within the methodology chapter and the comparative experiments in the refinement chapter (Chapter 6). These pose representations play a crucial role in understanding and improving the proposed methods.

For detailed information about the methodology and the novelties of the proposed

methods, please refer to the subsequent chapter.

Chapter 4

Methodology

4.1 Introduction

Most vision-based 6-DoF pose estimation approaches typically rely on knowledge of object's 3D model, depth measurements, and often require time-consuming iterative refinement to improve accuracy. However, these can be seen as limiting factors for broader real-life applications. In this chapter, a unified autoencoder framework called Auto-Pose is proposed, which consists of three novel convolutional autoencoder-based methods for object 6-DoF pose estimation:

- **D**enoising **A**utoencoder Using **L**atent **S**pace **R**egression for Object 6-DoF Pose Estimation (DALSR-Pose)
- **C**onvolutional **V**ariational Autoencoder-Based **M**ulti-**L**evel Network for Object 6-DoF Pose Estimation (CVML-Pose)
- **C**onditional **V**ariational **A**utoencoder for **M**ulti-Object 6-DoF Pose Estimation (CVAM-Pose)

The key contribution of the proposed Auto-Pose framework is to implicitly learn object's 6-DoF pose from only colour images encoded in the latent space of the autoencoder without knowing the object's 3D model, depth information, or performing

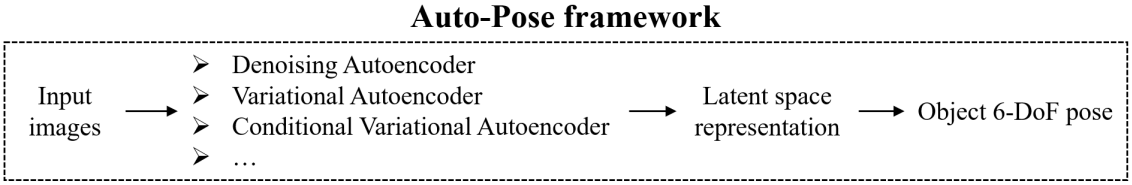


Figure 4.1: **Auto-Pose framework.** The Auto-Pose framework trains autoencoder networks to implicitly learn different styles of latent space representation from only colour images, and subsequently interpolates the learnt representation to object's 6-DoF pose using regression-based algorithms.

a post-refinement. The first DALSR-Pose method (Section 4.2), trains a denoising autoencoder (DAE) to encode the intermediate representations of an object in its latent space, and subsequently, the learnt representations are used to calculate 6-DoF pose through regression-based algorithms. The second CVML-Pose method (Section 4.3) published in [202], is developed based on DALSR-Pose, which utilises a completely different autoencoder architecture called variational autoencoder (VAE), to implicitly learn not only object's 6-DoF pose but also other attributes including, e.g. object category and shape topology, from the regularised latent space representation. The third CVAM-Pose method (Section 4.4), is an extension of the CVML-Pose method for multi-object pose estimation. This method combines the one-hot encoding technique with a conditional variational autoencoder (CVAE), enabling shared latent space across different objects, which demonstrates promising efficiency and scalability compared to competing approaches. To summarise, the novelties of the Auto-Pose framework are listed as follows:

- The framework does not require 3D models during inference, especially the CVML-Pose method, which significantly outperforms existing latent representation methods, and achieves comparable results with other state-of-the-art methods on the widely used Linemod [1, 2], Linemod-Occluded [3, 4], and YCB-Video [5, 6] benchmark datasets, without the use of depth measurement or post-refinement.

- The framework employs different autoencoder architectures, including a deterministic model (DAE), a generative model (VAE), and a conditional generative

model (CVAE), to efficiently characterise the 3D rotation and translation in the latent space. To the best of my knowledge, the Auto-Pose framework is the first to unify the latent space representation and the regression-based algorithms together to estimate the object's 6-DoF pose, as well as the first to use the regularised generative models' latent space, which outperforms the traditional latent representation methods across multiple datasets.

- The framework can cope with low-resolution images (typically 320-by-240 pixels, as used in the video demonstration) captured with inexpensive webcams, and can be operated fast enough[1] for real-time applications, due to not using iterative processing.

- The framework is also scalable to work well with both texture and texture-less objects, under challenging scenarios (detailed in Section 1.1), e.g. occlusion, truncation, and clutter.

The remainder of this chapter describes the details of the three proposed methods, DALSR-Pose, CVML-Pose, and CVAM-Pose, including their corresponding ablation experiments for parameter selection. An ablation test examines the system's performance by removing/fixing specific components to understand their contribution to the overall operation of the system. The reason for implementing ablation tests in the thesis is to select favourable design parameters for the proposed methods. The general procedure of our ablation tests begins by freezing some design parameters, then testing the effects of remaining parameters by selecting a range of values, sampling from that range, running simulations for different values within the selected range, and reporting the results. An example of such tests can be found in Section 4.3.4. The evaluation procedure, final results, and comparative analysis of these methods will be presented in Chapter 5.

[1] video available: https://ieeexplore.ieee.org/document/10040668.

4.2 DALSR-Pose: Denoising Autoencoder Using Latent Space Regression for Object 6-DoF Pose Estimation

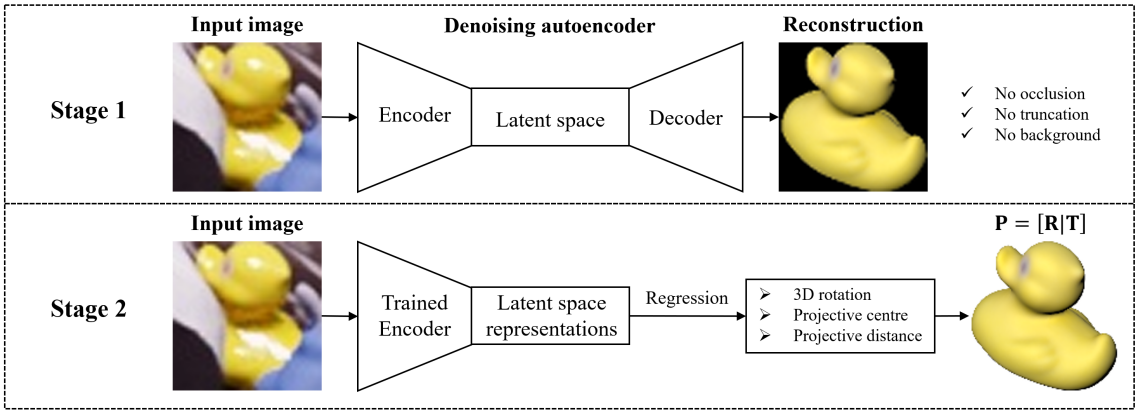


Figure 4.2: **DALSR-Pose pipeline.** The DALSR-Pose pipeline consists of two stages, the first stage is to use a DAE to capture an object's robust representation in the latent space. The second stage uses multiple regression-based algorithms to separately interpret the learnt representation to 3D rotation, 2D projective centre, and 2D projective distance, which forms the complete object 6-DoF pose.

Inspired by the Augmented Autoencoder (AAE) method [65], which learns implicit representations from rendered 3D model views using a denoising autoencoder (DAE), a novel DALSR-Pose method (Figure 4.2) is proposed. This method combines the DAE's latent space representation with a continuous regression strategy for estimating an object's 6-DoF pose. Differently from AAE, which discretises the 3D rotation into a finite number of instances using a lookup table (LUT), DALSR-Pose introduces a key novelty: performing regression directly on the latent space representation to compute a continuous object 6-DoF pose. To smoothly regress 3D rotation, a multilayer perception (MLP) is trained to estimate a continuous rotation representation that avoids discretisation of SO(3). For more accurate 3D translation estimation, another MLP is trained to regress the 2D projection of the object's centre. This approach differs from AAE, which simply uses the detection bounding box's centre as the 2D projection centre, and can be affected by heavy occlusion.

The DALSR-Pose method has been evaluated on the Linemod and the BOP version [7] of the Linemod-Occluded benchmark datasets. The method outperforms methods based on latent representation representation [65, 66] by enabling continuous latent space regression, and achieves results comparable to methods that use object's 3D model [67, 68]. Although DALSR-Pose still lacks accuracy compared to the leading state-of-the-art methods [35, 69, 70, 71, 72, 73, 74, 75], it facilitates the development of the later proposed CVML-Pose method.

The remainder of this section delves into the following details of DALSR-Pose, covering learning robust representation using denoising autoencoder (Section 4.2.1), pose regression in the latent space (Section 4.2.2), ablation tests of the method (Section 4.2.3), and initial findings (Section 4.2.4). Comprehensive evaluation procedure and results are presented in Chapter 5.

4.2.1 Learning Robust Representation using Denoising Autoencoder

To learn robust representation, a DAE network (see Section 3.4.1) is trained to encode images of the target object in its latent space and output the clean reconstruction. Different from the traditional process of adding noise, the input data x_i are assumed to inherently contain irrelevant information with respect to object 6-DoF pose, such as occlusion, truncation, and clutter. Consequently, the output data x_i' should present a complete and clean view of the object in the same pose. **Algorithm 1** presents the pseudo code for training the DAE network.

As depicted in Figure 4.3, a symmetrical autoencoder network is constructed to learn an intermediate representation of the object. This is similar to the one proposed in AAE (see Section 2.3.3 for details of the AAE method). The encoder network $E_\phi(x_i)$ takes the input data and generates a vector representation $z_i \in \mathbb{R}^{128}$ in the latent space, where the size of the latent space is the same as implemented in the AAE method, at 128. Subsequently, the decoder network $D_\theta(z_i)$ processes the latent space

Algorithm 1 Training the DAE Network

Require: Input data x, gt reconstruction data $\hat{x}$
Require: Encoder network E_ϕ
Require: Decoder network D_θ
Require: Mini-batch size m
Require: AdamW optimiser parameters

while not converged **do**
  Sample a mini-batch $\{x_i, \hat{x}_i\}_{i=1}^{m}$
  for each x_i in the mini-batch **do**
    $z_i \leftarrow E_\phi(x_i)$
    $x'_i \leftarrow D_\theta(z_i)$
  end for
  Compute the pixel-wise L2 loss:

$$\text{L2} = \sum_{i=1}^{m} ||x'_i - \hat{x}_i||^2$$

  Backpropagate the loss
  Update the encoder and decoder parameters ϕ, θ using AdamW optimiser
  Evaluate the loss on the validation set
  if validation loss does not improve for 50 epochs **then**
    Stop training to prevent overfitting
  end if
end while

representation z_i and outputs the reconstructed image x'_i. The training loss for the DAE network is a pixel-wise L2 loss (same to Eq. 2.3), which is measured between the output image x' and the ground truth (gt) reconstruction image $\hat{x}$. The loss function is iteratively minimised using the AdamW optimiser [203] with randomly selected mini-batches of 128 images. Additionally, as explained in Section 3.2.6, a set of validation data is used to prevent overfitting in training. The same loss function is applied to validation, indicating when the training should be terminated.

After training the denoising autoencoder, the learnt representation of the object can then be used for the estimation of the 6-DoF pose. In AAE, a method similarly using latent space representation, the authors employ an LUT to estimate the pose. However, the way they discretise the pose into a finite number of instances presents challenges, particularly when the training data cannot cover the entire 3D rotation group (denoted as SO(3)). On the contrary, the proposed pose regression strategy (detailed later in Section 4.2.2) enables smooth interpolation of the 3D rotation $\mathbf{R}$.

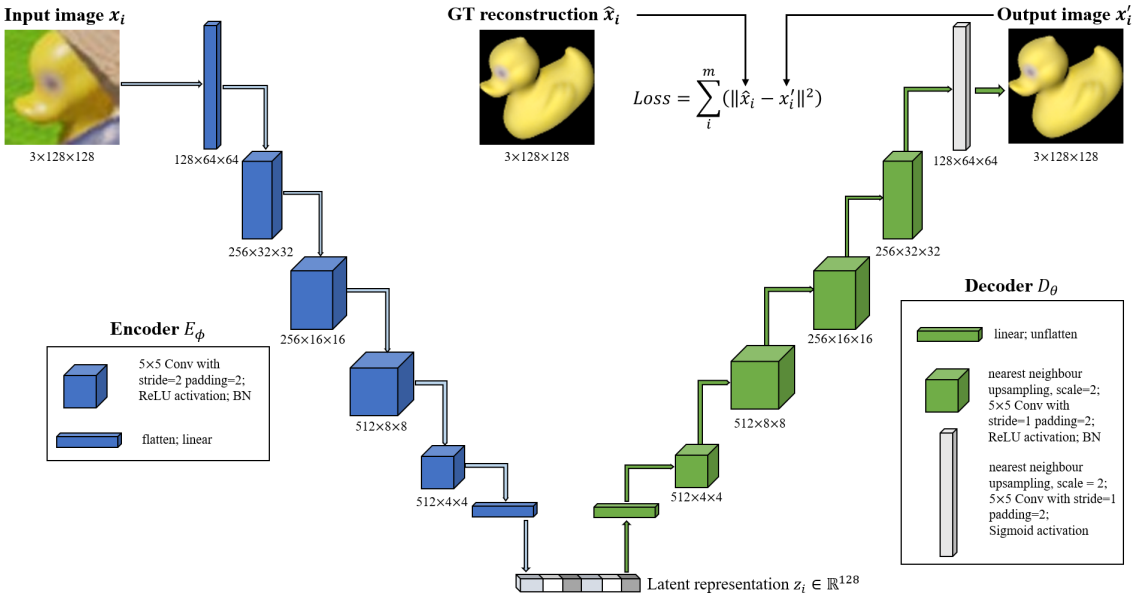


Figure 4.3: **The autoencoder architecture of DALSR-Pose.** ϕ represents all the parameters of the encoder network (the blue part) and θ represents all the parameters of the decoder network (the green part). The input image is first forwarded to the encoder which produces the latent variables. The decoder then outputs the reconstruction image from the variables. Both training and validation loss are the pixel-wise L2 loss.

For the estimation of 3D translation $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$, authors in the AAE method use the detection bounding box centre as the 2D projective centre, which can be heavily affected by occlusion. Instead, the proposed DALSR-Pose method combines the latent space representation with the information from the detection bounding box, to accurately regress the real projective centre. The details of the pose regression procedure are described in Section 4.2.2.

4.2.2 Continuous Pose Regression in the Latent Space

Since the training data may not comprehensively cover the entire SO(3), a latent space regression approach is proposed instead of finding the most similar instance. This approach disentangles the estimation of 3D rotation $\mathbf{R}$ and 3D translation $\mathbf{T}$, employing supervised and unsupervised models. These models are used to approximate the mappings between pose representations (see Section 3.3 for details of the representations) and the latent space representation, ensuring favourable configurations of the method, which will be detailed later in the ablation tests.

For estimating 3D rotation $\mathbf{R} \in \mathrm{SO}(3)$, commonly used rotation representations such as unit quaternion, axis-angle, and the continuous 6D representation [118] are explored. Also, to see which interpolation method is effective, several regression models are compared including multilayer perception (MLP) [58], random forests (RF) [60, 61], and nearest neighbour search (NNS) [99]. To estimate 3D translation $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$, the 2D projection of object centre $P_c = (x_c, y_c)^T \in \mathbb{R}^2$ and the projective distance $T_z \in \mathbb{R}$ are predicted separately. This allows for the recovery of the first two elements of the 3D translation $\mathbf{T}$, i.e. T_x and T_y, using the pinhole camera equation (Eq. 4.2). The complete pose regression procedure is shown in Figure 4.4, for example, the continuous 6D rotation representation $R_{6D} \in \mathbb{R}^6$ is used as an intermediate representation for $\mathbf{R}$, and the regression models are trained to predict $R_{6D} \in \mathbb{R}^6$ from the learnt latent representation $z_i \in \mathbb{R}^{128}$. For estimating 2D projective distance $T_z^{x_i}$ of an object x_i, the learnt representation $z_i \in \mathbb{R}^{128}$ is concatenated with the bounding box's width w_i and height h_i, whereas the regression of 2D projective centre $P_c^{x_i}$ requires not only the bounding box dimensions, but also the top-left point P_{bbox} of the bounding box.

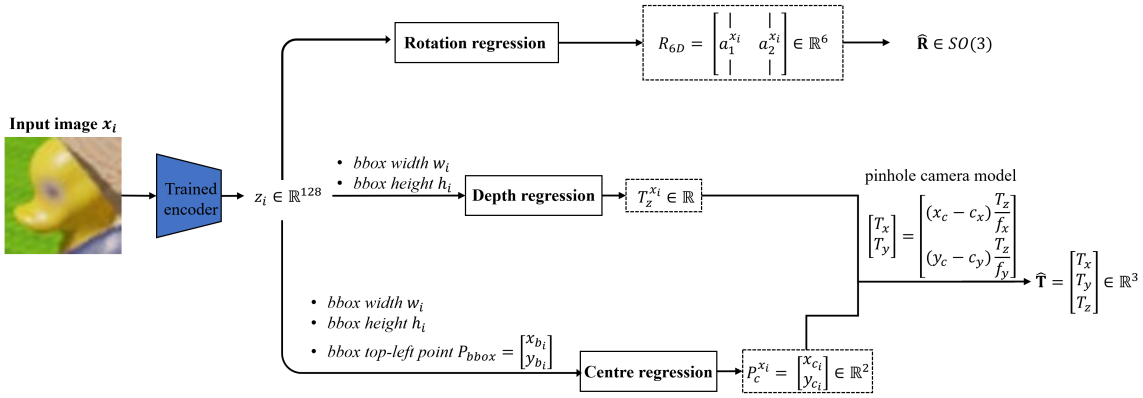


Figure 4.4: **Regress object's 6-DoF pose from the learnt latent space representation.** The pose estimation procedure is disentangled into rotation, depth, and centre regression. The rotation regression uses the learnt representations only, while the depth and centre regression both require additional information from the object bounding box, and the final translation is calculated based on the pinhole camera model (Eq. 4.2).

4.2.3 Ablation Tests

The design of the proposed DALSR-Pose method involves a number of ablation tests, including rotation regression and translation regression on the Linemod dataset, to obtain favourable configurations for pose regression, which is the main focus of this section. Based on the selected design parameters, the proposed DALSR-Pose method is then evaluated on both the Linemod and the BOP version of the Linemod-Occluded datasets, with its performance compared against current state-of-the-art methods. The detailed results of these evaluations, including comparisons and discussions, will be presented in Chapter 5.

In the ablation tests, all objects from the Linemod dataset are used. The performance of the ablation tests on rotation is measured using the average recall of the ADD(I) metric (see Section 3.5.2 for details of the metric). However, for ablation tests on translation estimation, the performance is assessed using the mean absolute error (MAE) metric [204] instead of the ADD(I) metric. This is because the ADD(I) metric depends on the designed threshold as explained in the previous section, while the MAE is a parameter-free metric, which makes it easier to interpret the translational error. All the results in these ablation tests are based on the gt bounding box of the Linemod test data to reduce the dependence of the results on the bounding box.

Rotation Regression

In the ablation tests for rotation estimation of the DALSR-Pose method, comprehensive experiments are conducted to determine the most effective combination of regression models and rotation representations. The experiments involve evaluating MLP, RF, and NNS, each regressing three different rotation representations: axis-angle, unit quaternion, and the continuous 6D representation. The performance of the corresponding experiments is reported in Table 4.1. To acquire an appropriate configuration for rotation, the performance score $AR_{ADD(I)}$ is calculated based

on the estimated rotation but the gt translation, to reduce the effects of incorrect translation estimation. This can help avoid a situation where the translation vector is wrongly predicted with a large error, no matter how accurate the rotation estimation is, the $AR_{ADD(I)}$ would still be low.

Model	**Rotation representation**	$AR_{ADD(I)}$
MLP	continuous representation	**91.32**
	unit quaternion representation	90.11
	axis-angle representation	85.82
RF	continuous representation	86.73
	unit quaternion representation	80.49
	axis-angle representation	57.69
NNS	any rotation representations	87.61

Table 4.1: **Ablation test on 3D rotation estimation.** The NNS algorithm does not require any rotation representations as the label because of the property of unsupervised learning.

The results of both regression models and rotation representations, as shown in Table 4.1, indicate that MLP consistently outperformed RF across all rotation representations, and the continuous 6D representation demonstrates superior performance compared to the axis-angle and unit quaternion representations in both MLP and RF models. Despite the NNS algorithm achieving a moderate performance, the discretisation of rotation lacks accuracy especially when the training data cannot cover the entire SO(3). Consequently, the NNS algorithm is not preferred for the estimation of 3D rotation in the final implementation of the method.

Translation Regression

In the process of estimating the translation vector **T** for the DALSR-Pose method, a comprehensive approach is adopted, similar to the one described in the PoseCNN method (detailed in Section 2.3.1). This involves separately estimating the 2D projection of the object centre $P_c = (x_c, y_c)^T \in \mathbb{R}^2$ and the projective distance $T_z \in \mathbb{R}$. In both cases, the latent space representation $z_i \in \mathbb{R}^{128}$ is concatenated with the spatial information of the bounding box, including its width w, height h, and/or top-left corner $P_{bbox} = (x_b, y_b)^T$.

Three regression-based models, including MLP, RF and KNN regressor, are trained to regress the concatenated representations to the target P_c and T_z. The strategy proposed in the AAE method is also compared, which directly uses the centre of the bounding box as the 2D projective centre. The performance of each model is calculated using the MAE metric (Eq. 4.1) across all objects in the Linemod dataset. For the 2D projective centre P_c, the error is measured in pixel differences ($\text{MAE}_{\text{pixel}}$), and for the projective distance T_z, the error is calculated in millimetres (MAE_{mm}).

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}|y_i - y_i'| \tag{4.1}$$

where m is the number of data points, y_i and y_i' refer to the gt value and the prediction.

Regression model	$\text{MAE}_{\text{pixel}}$
MLP	**1.61**
RF	6.81
KNN	6.11
AAE [65]	4.03

(a) **Projective centre**

Regression model	MAE_{mm}
MLP	36.23
RF	31.56
KNN	**25.91**

(b) **Projective distance**

Table 4.2: **Ablation tests on different regression models for the estimation of projective centre and projective distance.** The AAE method does not use any regression models, the method calculates the centre of the bounding box as the projective centre.

The results, as shown in Table 4.2 and visualised in Figure 4.5, indicate that MLP is particularly effective in precisely localising the 2D projective centre P_c, with relatively small errors (≈ 2 pixel), while the other models like KNN and RF can only achieve moderately accurate results. It is also noted that the AAE method performs better than KNN and RF, but still lacks accuracy compared to MLP. This suggests that the AAE's approach is acceptable under conditions of minimal occlusion. It would also be interesting to see whether the AAE's performance will significantly degrade when dealing with heavily occluded objects or inaccurate bounding boxes.

In terms of the results on 2D projective distance T_z, the KNN regressor achieves more accurate results than the two other regressors, this could be attributed to its

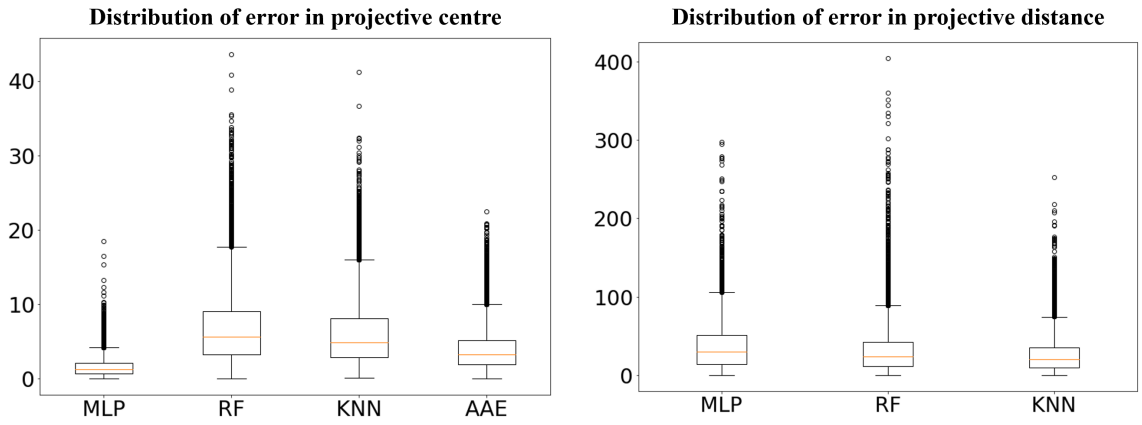


Figure 4.5: **Distribution of error in the estimation of projective centre and projective distance.** The left boxplot visualises the results presented in Table 4.2a, and the right one visualises the results presented in Table 4.2b.

inherent simplicity, which is less prone to overfitting, particularly with a well-chosen number of neighbours (k). In contrast, MLP and RF require fine-tuning of several parameters to avoid overfitting, such as the number of trees, maximum depth of the tree, and minimum number of samples at a leaf node. Notably, all these regression models have a certain margin of error in interpolating T_z. This might be linked to the nature of the training data, which comprise images of objects cropped from scenes, potentially leading to a loss of certain spatial information about the scene. Consequently, it would be difficult for the latent space to accurately perceive such information, like the projective distance T_z from the camera to the object.

4.2.4 Remarks

The ablation tests conducted for the DALSR-Pose method suggest that continuous regression of the latent space can achieve a better estimation of object's 6-DoF pose than the discretisation approach of SO(3). Therefore, the selected configuration of DALSR-Pose is to implicitly learn robust representations from colour images using a denoising autoencoder, and then combine the latent space representation and regression-based algorithms for the estimation of 3D rotation and translation. To estimate 3D rotation, an MLP regressor is trained to regress the continuous 6D representation. The 3D translation involves localising the projective centre and

predicting 2D object projective distance, using another MLP regressor and a KNN regressor, respectively. The final DALSR-Pose method is trained on 13 Linemod objects using the physically based rendering (PBR) images [7, 142], and evaluated on both the Linemod (see Table 5.2) and the BOP version of the Linemod-Occluded (see Table 5.3) datasets. The evaluation setup and the results can be found in Chapter 5.

To the best of my knowledge, the proposed DALSR-Pose method is the first to unify the latent space representation with continuous regression-based algorithms for 6-DoF pose estimation. As it will be demonstrated in the subsequent chapter, the DALSR-Pose method outperforms existing methods based on latent space representation [65, 66], and achieves comparable pose accuracy results to the state-of-the-art on the more challenging occlusion dataset. The method also opens the prospect for exploration of various autoencoder models which may learn different representations in the latent space, such as the variational autoencoder used in the CVML-Pose approach (Section 4.3), and conditional variational autoencoder used in the CVAM-Pose approach (Section 4.4).

4.3 CVML-Pose: Convolutional Variational Autoencoder Based Multi-Level Network for Object 6-DoF Pose Estimation

The CVML-Pose pipeline, as depicted in Figure 4.6, consists of two main modules: (i) CVML-AE, denoting the convolutional variational autoencoder (VAE) network tasked with abstracting regularised latent space representations from colour images, and (ii) MLP and KNN, used to interpolate the latent variables into object 6-DoF pose including, respectively, 3D rotation $\mathbf{R} \in \mathrm{SO}(3)$ and 3D translation $\mathbf{T} = (T_x, T_y, T_z)^T \in \mathbb{R}^3$. The first stage of the CVML-Pose method involves training the CVML-AE module to reconstruct images through a low-dimensional latent

representation. This requires the module to implicitly learn to represent images with a relatively small number of high-level features, which can then be used for downstream tasks. In the second step, the estimations of **R** and **T** are disentangled using MLP and KNN. For the estimation of **R**, a standard MLP is used to regress the latent variables to the continuous rotation representation [118]. To estimate **T**, the latent variables are regressed to the 2D projection of the object centre and the projective distance, using another MLP and a KNN regressor, respectively. To investigate whether the latent space represents other characteristics of objects, the method is trained on multiple objects simultaneously, with their respective test images processed through the trained network to generate corresponding latent variables. Analysis of these variables via clustering algorithms like t-SNE [195], indicates that it is possible to use the proposed architecture to infer other characteristics, such as object class and shape topology.

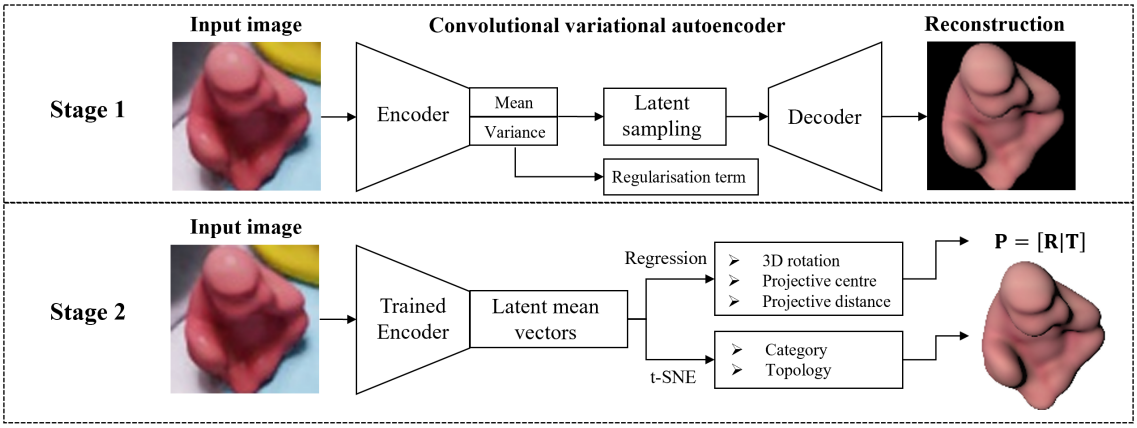


Figure 4.6: **CVML-Pose pipeline.** During training, the convolutional variational autoencoder captures object's latent representation in its latent space that is further interpreted to object 6-DoF pose, category, and topology using MLPs, KNN, and t-SNE.

The CVML-Pose method, published in [202], has been comprehensively evaluated on the Linemod, Linemod-Occluded (BOP version), and YCB-Video (BOP version) benchmark datasets. It is shown to significantly outperform methods based on the latent representation [65, 66] and those using 2D-3D model correspondence [67, 68]. The method also achieves comparable results to the leading methods [69, 70], but without the use of a 3D model or depth measurements. The latent space of CVML-

Pose, as demonstrated through clustering algorithms, effectively represents object category and topology. This opens up a prospect of integrated estimation of pose and other attributes (possibly also including surface finish or shape variations), which, with real-time processing due to the absence of iterative refinement, can facilitate real-world applications.

The remainder of this section will describe the following details of the method, including representing 2D rotation in the latent space (Section 4.3.1), 6-DoF pose estimation from latent space representation (Section 4.3.2), other characteristics in the latent space and visualisation (Section 4.3.3), ablation tests (Section 4.3.4), network parameters and training details (Section 4.3.5), and initial findings (Section 4.3.6). A thorough evaluation, discussion, and conclusion of the method will be presented in the subsequent chapter, focusing on the evaluation process and the results of the method.

4.3.1 Implicit Learning in 2D Rotation

Before exploring 6-DoF Pose Estimation using a VAE, a special case involving 2D rotation estimation is initially investigated with the VAE. This serves as a toy problem in understanding the capability of the VAE to encode specific rotational information within its latent space. This hypothesis is underpinned by the ability of the VAE's decoder to generate images that closely resemble the input [192], suggesting a potential for encoding representative information like rotation.

To test the hypothesis, a VAE network, similar but simpler to the one proposed in Figure 4.9, is trained to encode the rotated digits from the MATLAB Digits dataset [165], in the form of latent variables $\mu, \sigma^2 \in \mathbb{R}^{20}$, and output the reconstructed digits from the latent sample $z = \mu + diag(\sigma) \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. After training, the NNS algorithm is used to find the most similar mean vector $\mu \in \mathbb{R}^{20}$, and assign the corresponding 2D rotation $R_{2d} \in \mathbb{R}$ to each test instance. The visualised results, as illustrated in Figure 4.7, validate that the VAE is capable of

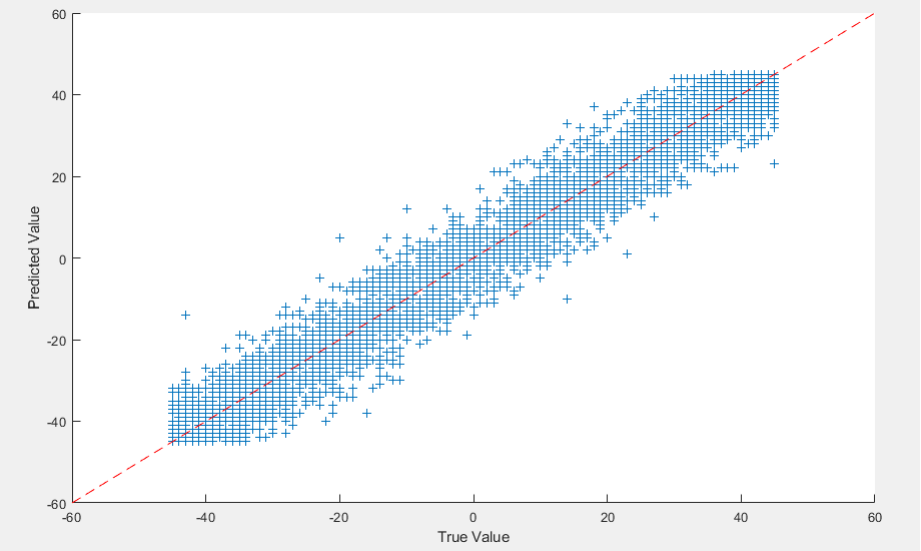


Figure 4.7: **Visualised results of 2D rotation estimation.** The 2D rotation predictions are visualised in a scatter plot, which shows the predicted rotation angles against the true angles.

representing 2D rotation in its latent space, which can be postulated that the more complicated 3D rotation could also be encoded in the space.

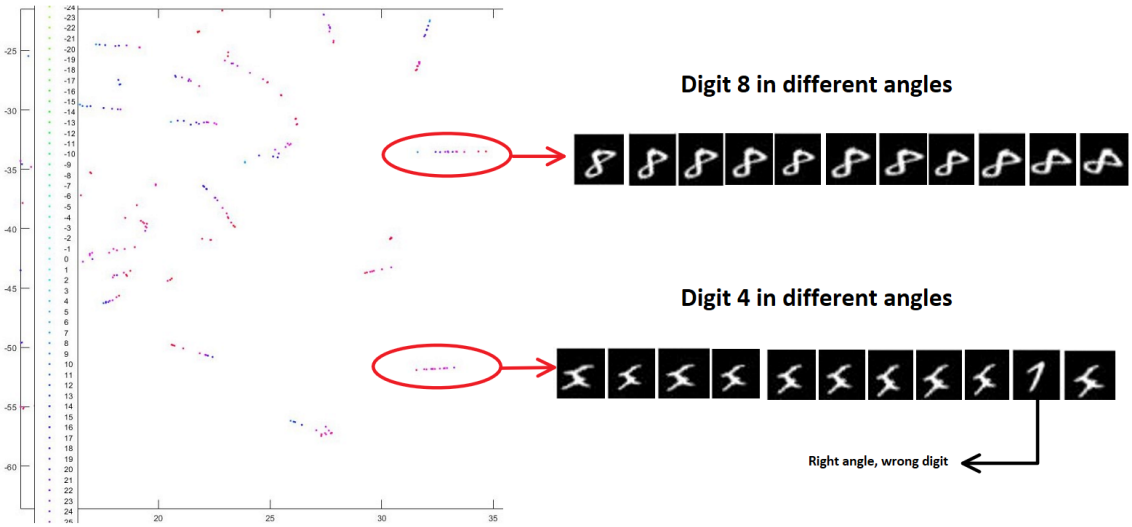


Figure 4.8: **t-SNE visualisation of different digits with different 2D rotations encoded in the latent space.** Each point represents a digit with the corresponding rotation angle. The rotated digits are generated from the MNIST dataset [163, 164].

Further analysis using the t-Distributed Stochastic Neighbor Embedding (t-SNE) algorithm [195], as demonstrated in Figure 4.8, shows the VAE's ability to encode different digits with different 2D rotations. This is evidenced by distinct clusters for different digits in the latent space, each having a linear relationship between the

rotation angles. For example, clusters for the digit "8" and "4", not only exhibit distinct rotation patterns within each group, but also clearly separate these two categories of digits. Such clustering indicates that the information about both the object's category and its rotation has been learnt in the latent space. Therefore, the results from the handwritten digits suggest that the VAE has the potential to concurrently regress the 2D rotation angle and classify the category of the digits. This also indicates that it is possible to extend this approach to include not only rotation estimation but also the prediction of other characteristics such as class and topology.

4.3.2 Regularised Latent Space Representation for 6-DoF Pose Estimation

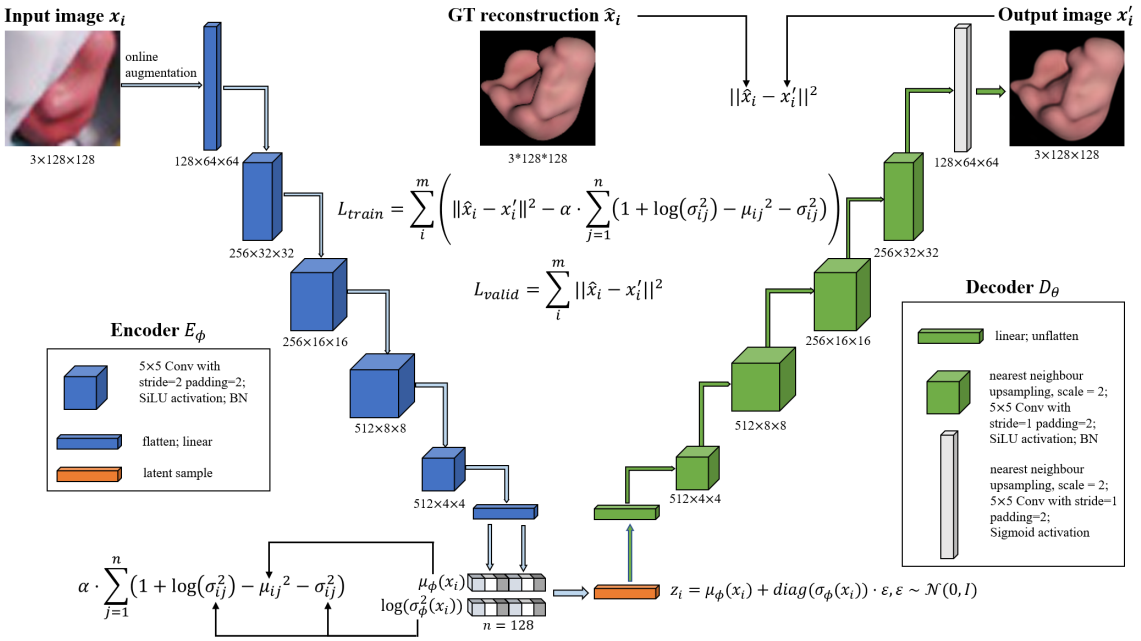


Figure 4.9: **The vanilla variational autoencoder (CVML-base) architecture.** ϕ represents all the parameters of the encoder network (the blue part) and θ represents all the parameters of the decoder network (the green part). The encoder can be replaced with more advanced networks like ResNet-18 and ResNet-34 backbones [115]. The input image is first forwarded to the encoder which produces the latent variables. The decoder then outputs the reconstruction image from the latent sampling. The training loss is modified from Eq. 3.13, and the validation loss L_{valid} is the pixel-wise L2 loss.

To implicitly learn an object's 6-DoF pose, the first part of the CVML-Pose is

implemented using a VAE network, which is trained to learn the characteristics of the object in its latent space. As depicted in Figure 4.9, the input data x_i is processed by the encoder $E_\phi(x_i)$ to produce two latent variables $(\mu_\phi(x_i), \sigma^2_\phi(x_i)) \in \mathbb{R}^n$, where n is the dimensionality of the latent space. An additional Kullback–Leibler (KL) divergence with weight α is used to regularise the latent space. At the sampling stage, the random sample process is implemented by the reparameterization trick $z_i = \mu_\phi(x_i) + diag(\sigma_\phi(x_i)) \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. After sampling, the decoder network $D_\theta(z_i)$ reconstructs the output image x'_i from z_i, and a pixel-wise L2 loss is calculated between the output image x'_i and the ground truth (gt) reconstruction image $\hat{x}_i$. This is also correlated with Eq. 3.13 derived in Section 3.4.2, that the term $||x_i - x'_i||^2$ in $ELBO$ is replaced by $||\hat{x}_i - x'_i||^2$, and the constant c is removed. **Algorithm 2** presents the pseudo code for training the VAE network.

In the provided code implementation[2], the training loss $L_{train} = -ELBO$ is iteratively minimised using the AdamW optimiser [203] with randomly selected mini-batches of 128 elements, i.e. m in Eq. 3.13 is replaced with 128. To prevent overfitting, the same pixel-wise L2 loss, excluding the KL divergence term, is used as the validation loss L_{valid}, which indicates whether the latent space has accumulated enough information from the training data.

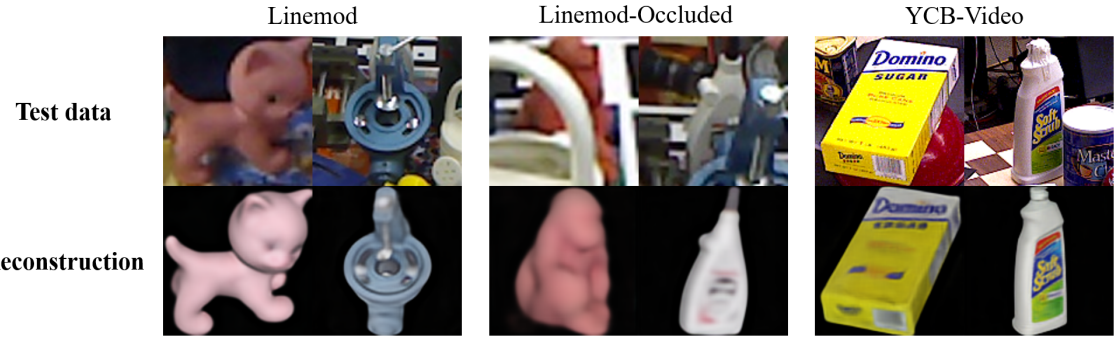


Figure 4.10: **Example reconstruction images from the trained VAE based on the test images.** The test images are taken from the Linemod [1, 2], Linemod-Occluded [3, 4], and YCB-Video [5, 6] datasets.

After training the VAE network, the latent space accumulates rich information about the object, possibly including SO(3). This can be evidenced by the clean reconstruc-

[2]code available: https://github.com/JZhao12/CVML-Pose.

Algorithm 2 Training the VAE Network

Require: Input data x, gt reconstruction data $\hat{x}$
Require: Encoder network E_ϕ
Require: Decoder network D_θ
Require: Mini-batch size m
Require: Latent space size n
Require: Auxiliary random variable $\epsilon \sim \mathcal{N}(0, I)$
Require: Regularisation weight α
Require: AdamW optimiser parameters

while not converged **do**
    Sample a mini-batch $\{x_i, \hat{x}_i\}_{i=1}^m$
    for each x_i in the mini-batch **do**
        $\mu_\phi(x_i), \sigma_\phi^2(x_i) \leftarrow E_\phi(x_i)$
        $z_i = \mu_\phi(x_i) + diag(\sigma_\phi(x_i)) \cdot \epsilon$
        $x_i' \leftarrow D_\theta(z_i)$
    end for
    Compute the $ELBO$ loss:

$$ELBO = \sum_{i=1}^{m} \left(||\hat{x}_i - x_i'||^2 - \alpha \cdot \sum_{j=1}^{n} \left(1 + \log\left(\sigma_{ij}^2\right) - \mu_{ij}^2 - \sigma_{ij}^2 \right) \right)$$

    Backpropagate the loss
    Update the encoder and decoder parameters ϕ, θ using AdamW optimiser
    Evaluate on the validation set using the L2 loss:

$$\text{L2} = \sum_{i=1}^{m} ||\hat{x}_i - x_i'||^2$$

    if validation loss does not improve for 50 epochs **then**
        Stop training to prevent overfitting
    end if
end while

tion images generated from the test data using the trained VAE. These reconstruction images not only preserve a complete view of the object with a clean background, but also minimise irreverent information such as occlusion and clutter, as shown in Figure 4.10. The test latent variables $(\mu_{test}, \sigma^2_{test}) \in \mathbb{R}^n$ are also robust again scaling, which does not affect the object orientation of the reconstruction images (see examples in Figure 4.11).

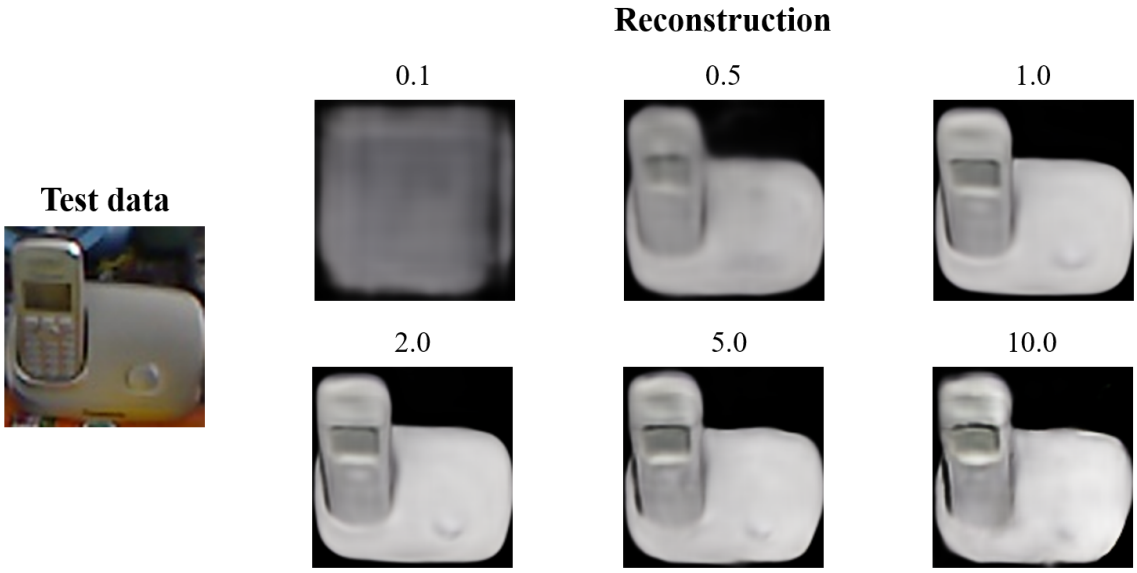


Figure 4.11: **Example reconstruction images based on different scaled latent variables.** The trained encoder network processes the test image and generates the test latent variables $(\mu_{test}, \sigma^2_{test}) \in \mathbb{R}^n$, and the trained decoder network outputs the reconstruction images based on the scaled latent variables with different factors $\in [0.1, 0.5, 1.0, 2.0, 5.0, 10.0]$. The test image is taken from the Linemod [1, 2] dataset.

The second part of the CVML-Pose method involves interpolating such information to 3D rotation and translation. Figure 4.12 depicts the complete procedure to estimate object 6-DoF pose using MLPs and KNN. To estimate the 3D rotation $\mathbf{R} \in \mathrm{SO}(3)$, a standard MLP (named the Rotation MLP) is used to regress the latent variable $\mu_{train} \in \mathbb{R}^{128}$ produced by the trained encoder, to a continuous 6D rotation representation $R_{6D} \in \mathbb{R}^6$ proposed by Zhou et al. [118]. This continuous representation has been demonstrated to be more effective than other commonly used representations such as unit quaternion and axis-angle, and has been successfully used by CosyPose [75] (winner of the BOP Challenge 2020 [7]) and GDR-Net [71] (winner of the BOP Challenge 2022 [26]). During training of the Rotation MLP, the

estimated rotation $\hat{\mathbf{R}}$ can be recovered from the estimated 6D representation R_{6D} (see Eq. 3.10 in Section 3.3.5), and an L1 loss (mean absolute error) is measured between the estimated rotation $\hat{\mathbf{R}}$ and the gt rotation $\bar{\mathbf{R}}$.

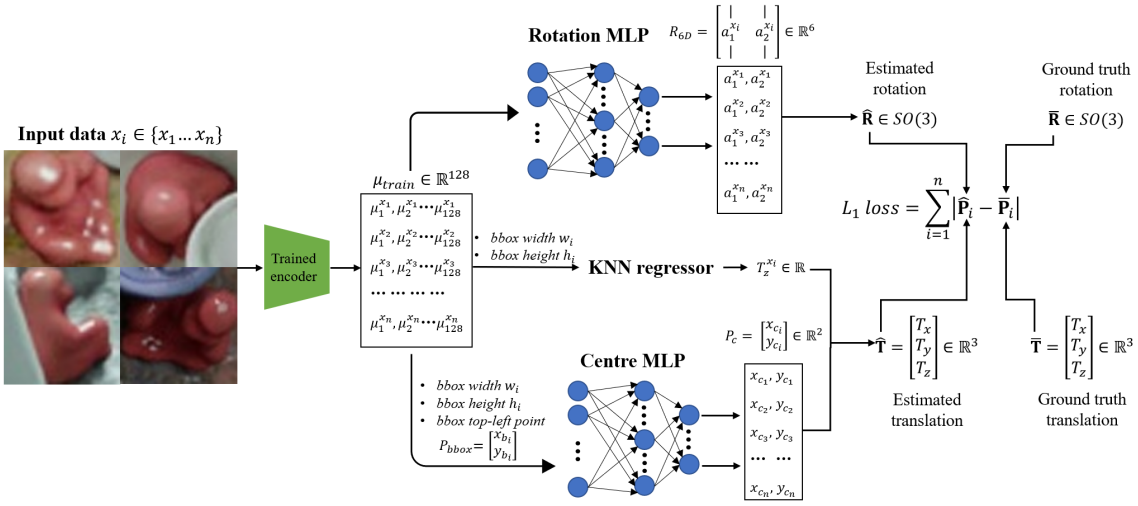


Figure 4.12: **From latent space representation to object's 6-DoF pose.** The trained latent variables $\mu_{train} \in \mathbb{R}^{128}$ are generated from the trained encoder network. The MLPs and KNN are utilised to regress the complete 6-DoF pose, and the training loss for MLPs is calculated between the estimated pose $\hat{\mathbf{P}}_i$ and the gt pose $\bar{\mathbf{P}}_i$, where the gt pose is provided by the Linemod PBR dataset [1, 2, 7].

For estimating 3D translation $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$, the CVML-Pose method adopts a similar approach to the one proposed by Xiang et al. [5]. This involves regressing the 2D projection of the object's centre $P_c = (x_c, y_c)^T \in \mathbb{R}^2$, and the 2D projective distance $T_z \in \mathbb{R}$, separately. The values of T_x and T_y are then determined using the pinhole camera model equations (Eq. 4.2 and 4.3).

Specifically, for the estimation of P_c, the latent variable $\mu_{train} \in \mathbb{R}^{128}$ is concatenated with prior information obtained from the gt bounding box, including its width w, height h, and top left corner $P_{bbox} = (x_b, y_b)^T$. The concatenated set of variables is then fed into another standard MLP, named Centre MLP, which is trained to regress these variables to the 2D projective centre P_c. In parallel, for the estimation of T_z, the same latent variable $\mu_{train} \in \mathbb{R}^{128}$ is concatenated with a different set of information, which only includes the gt bounding box's width w and height h. A KNN regressor is then trained to regress these variables to the projective distance T_z. The prediction of T_z is based on the local interpolation from the $k \in [1, 20]$

nearest neighbours within the training data. For each value of k, the corresponding validation error is calculated based on the complete 3D translation, instead of only on T_z. The favourable value of k is determined where the validation error is the smallest. Once T_z and the coordinates $(x_c, y_c)^T$ are known, the first two elements of $\mathbf{T}$, i.e. T_x and T_y, can be easily calculated based on the equations of the pinhole camera model shown in Eq. 4.3.

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} f_x \frac{T_x}{T_z} + c_x \\ f_y \frac{T_y}{T_z} + c_y \end{bmatrix} \tag{4.2}$$

$$\begin{bmatrix} T_x \\ T_y \end{bmatrix} = \begin{bmatrix} (x_c - c_x) \frac{T_z}{f_x} \\ (y_c - c_y) \frac{T_z}{f_y} \end{bmatrix} \tag{4.3}$$

where f_x and f_y denote the focal lengths, $(c_x, c_y)^T$ is the principal point.

To obtain favourable parameters of the method, extensive ablation tests are implemented. Advanced activation functions like SiLU [205] and ELU [206] are used to avoid zero gradients, instead of the commonly used ReLU [207]. Different network architectures are also investigated, for instance, the encoder part is replaced by more advanced convolutional neural networks (CNNs) such as ResNet-18 and ResNet-34 [115]. To distinguish different variants of the networks, the vanilla symmetric encoder-decoder network proposed in Figure 4.9 is termed CVML-base, and the ResNet-based autoencoder networks are named CVML-18 and CVML-34, respectively. These variants collectively form the CVML-AE module. To improve the generalisation ability of the method, the CVML-AE module is trained with selected online data augmentation techniques, detailed in Table 4.8. Further details about these ablation tests are explained in Section 4.3.4.

4.3.3 Other Characteristics in Latent Space

In the object characterisation experiments, the focus is on investigating whether the latent space of a VAE could represent and subsequently estimate other characteris-

tics of objects. In the experiments reported here, both shape topology and object class are investigated. Shape topology, as described by a genus, essentially counts the number of "holes" in an object [208]. For instance, a sphere has genus 0, and a torus has genus 1. For practical applications, such as in robotic manipulation, differentiating objects based on characteristics like the presence of a handle on a mug (which would alter its genus) could be important.

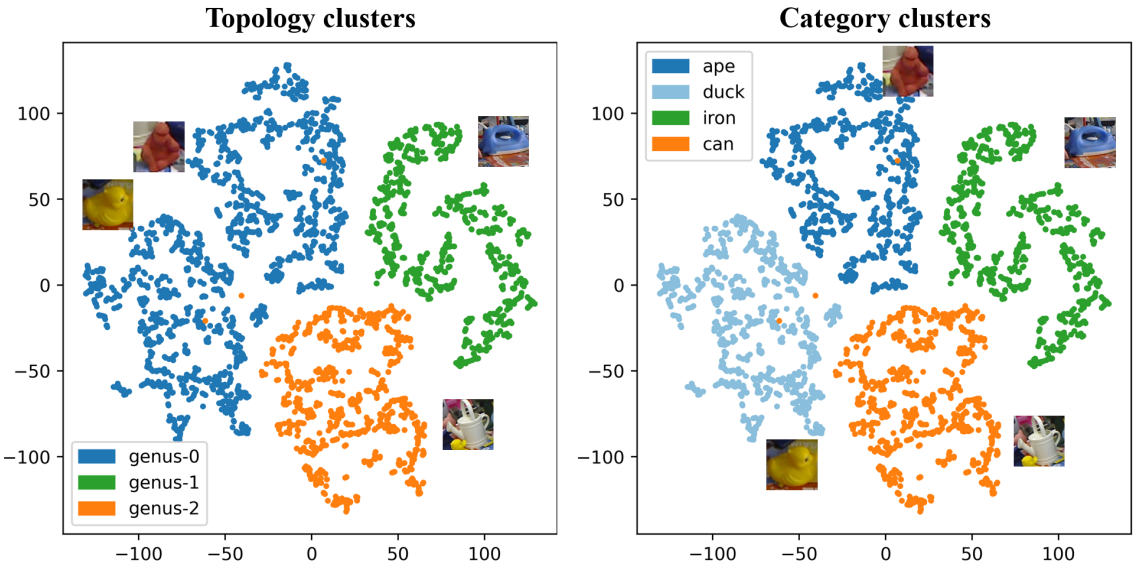


Figure 4.13: **Visualisation of the latent space using t-SNE for topology recognition (left) and object classification (right).** Images of objects are taken from the Linemod test data [1, 2].

To test the hypothesis proposed in Section 3.4.2 that VAE can learn other attributes, the CVML-base network is trained with images of four Linemod objects, each possessing different genera: the ape and duck (genus 0), the iron (genus 1), and the can (genus 2). The aim is to see if the network could encode these objects distinctively in its latent space, reflecting their differing topologies. Following the training procedure described in Section 4.3.2, the test images of these objects are processed through the trained CVML-base network to generate their latent variables. These variables are then visualised using the t-SNE algorithm [195]. Although t-SNE is not suitable for classification tasks, since new data can change the spatial distance of existing points, it serves as a useful tool for visual exploration of the latent space.

The t-SNE visualisation in Figure 4.13 shows distinct clusters corresponding to

different object characteristics. Notably, the latent variables for the ape and duck are closely grouped, indicating they share the same topology (both having genus 0). Based on the visualisation, it can be concluded that the trained latent space can possibly represent the objects' classes and topologies. It can be therefore speculated that the latent space in the CVML-AE module is capable of encoding a variety of objects' characteristics, which can be used in automatic object manipulation applications. However, more research is needed in that area.

4.3.4 Ablation Tests

To obtain effective configurations of the proposed CVML-Pose method, a large number of ablation tests are implemented. These tests evaluate various aspects, including network architecture, the dimensionality of the latent space, the weight factor of the KL regularisation term, activation functions for autoencoders and MLPs, and data augmentation techniques. Based on the selected design parameters, the proposed CVML-Pose method is evaluated on the Linemod, the BOP version of the Linemod-Occluded, and the BOP version of the YCB-Video benchmark datasets, and compared against the state-of-the-art, which will be illustrated in detail in Chapter 5.

Unlike the DALSR-Pose method, where all objects from the Linemod dataset are used in ablation tests, the CVML-Pose method focuses on four specific objects from the dataset for efficiency. These objects include the ape, driller, eggbox, and phone. The pose estimation results are evaluated using the commonly used ADD(I) metric, and the overall performance score is calculated using the average recall of the metric $AR_{ADD(I)}$. The results of these tests are comprehensively detailed in various tables (Table 4.3, 4.4, 4.5, 4.6, 4.7, and 4.9). It is also important to note that all results are calculated based on the gt bounding box of the test data to reduce the dependence of the results on the bounding box. For more details on the evaluation metrics and the complete evaluation procedure, please refer to Section 3.5 and Chapter 5, respectively.

Network Architecture

In the development of the CVML-Pose method, a well-structured VAE is crucial in encoding the 6-DoF pose of the object in the latent space. Comprehensive experiments are conducted on the autoencoder's design, and the results are presented in Table 4.3 and 4.4, investigating the structure of the encoder and decoder, and the overall autoencoder, respectively.

Network architecture	$AR_{ADD(I)}$
CVML-base	48.81
CVML-18	**55.26**
CVML-34	52.29

(a) **Encoder**

Upsampling approach	$AR_{ADD(I)}$
nearest neighbour	**48.81**
transposed convolution	48.04
bilinear	48.11

(b) **Decoder**

Table 4.3: **Ablation tests on the architecture of encoder and decoder.** The encoder network has three variants, and the decoder network can be incorporated with three different upsampling techniques.

The initial architecture, named CVML-base network (illustrated in Figure 4.9), follows a symmetric encoder-decoder design similar to the one proposed in the AAE method [65]. To test the benefit of alternative encoders, variants using ResNet-18 and ResNet-34 backbones [115], named CVML-18 and CVML-34 respectively, are explored to see how well each encoder network can abstract object pose information in the latent space. The results, as shown in Table 4.3a, indicate that with increasing depth of the encoder, the latent space acquires more 3D information about objects. However, the performance difference between CVML-18 and CVML-34 is small, suggesting a potential limitation in the decoder's capacity. It is possible that the decoder is not capable of taking full advantage of the rich latent information provided by the ResNet-34 encoder, in which case a deeper decoder network might help. It is also possible that available relevant pose information (based on the available training data) is well enough encapsulated (summed up) by the ResNet-18 encoder. Therefore, more complicated encoders, like ResNet-34, are unlikely to improve the results as there is nothing much to add (with respect to pose) to what the ResNet-18 captured.

Regarding the decoder, although it is not utilised in the final pose estimation stage, it plays an important role in interpreting the latent code during the VAE training. Several alternative decoders are assessed to determine if they can enhance the encoder's ability to abstract as much information as possible. This involves using alternative upsampling approaches like transposed convolution and bilinear upsampling. The results, as noted in Table 4.3b, suggest that the performance of the CVML-base network with nearest neighbour upsampling does not differ much from alternatives using transposed convolution and bilinear upsampling. This aligns with the expectation that the decoder network plays a relatively minor role in pose estimation. Moreover, the discussion in [209] highlights a significant issue with the transposed convolution, known as "checkerboard artefacts", which can potentially result in uneven overlaps on images. To avoid this problem, the authors recommend a resize-convolution approach, which involves applying nearest neighbour upsampling or bilinear upsampling to the feature map before convolution. In line with this approach, the CVML-base network is designed with the nearest neighbour upsampling, which not only avoids such artefacts, but also computes faster than the bilinear process.

Network architecture	$AR_{ADD(I)}$
CVML-base	48.81
CVML-base with additional intermediate layer	**49.49**
Network from AAE [65]	43.94

Table 4.4: **Ablation test on different autoencoder architectures.** The CVML-base with additional intermediate layer, integrates an MLP inside the latent space instead of a single layer. The network from AAE has only one layer shallower than the CVML-base network in both the encoder and decoder.

In addition, modifications to the overall autoencoder architecture are explored, such as adding an intermediate layer in the latent space, to see whether it facilitates a smoother representation, potentially improving the model's capability of capturing more information about the object. Also, to see how well the CVML-base network compares with other autoencoders proposed in existing pose estimation approaches, the DAE network from the AAE method [65] is converted to a VAE and trained

with the same data. Table 4.4 shows the performance score based on different autoencoder architectures. Notably, despite the CVML-base network having only 1 layer deeper than AAE in both its encoder and decoder, this marginal increase in depth boosts the pose estimation performance significantly. In the meantime, the introduction of an additional intermediate layer in the latent space appears to help build a slightly better pose representation of the object.

Consequently, the CVML-18 network is chosen as the final VAE network for the CVML-Pose method. The network incorporates nearest neighbour upsampling in its decoder, and an additional intermediate layer in the latent space, which is then used to compare against the state-of-the-art methods in subsequent evaluations.

Dimensionality of the Latent Space

In the configuration of the VAE, the dimensionality n of the latent space is a critical factor affecting the quality of the latent space representation. It determines the capacity of the latent space to capture and encode the essential information about objects.

In AAE, the authors implement ablation tests to find a proper n for the DAE network. In that case, the accuracy of pose estimation started to saturate at $n = 64$, and the best result is achieved with $n = 128$. However, their tests are limited to a range of $n \in [4, 8, 16, 32, 64, 128]$ and are conducted on a single object from the T-LESS dataset [151]. This raises questions about the pose estimation performance with $n > 128$, and on a broader range of objects, since a single object may not be sufficient to represent the entire dataset.

To determine an effective size of the latent space, extensive experiments are conducted on multiple objects from the Linemod dataset using the CVML-base network. Various dimensionalities are tested, including $n \in [1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]$. As indicated in Table 4.5, the best pose estimation accuracy can be achieved with $n = 128$. The results also demonstrate a general trend where the latent space can

Dimensionality of the latent space	$AR_{ADD(I)}$
$n = 1$	13.34
$n = 2$	19.39
$n = 4$	32.65
$n = 8$	35.34
$n = 16$	30.73
$n = 32$	39.27
$n = 64$	45.25
$n = 128$	**48.81**
$n = 256$	48.46
$n = 512$	48.59
$n = 1024$	45.99

Table 4.5: **Ablation test on the dimensionality of the latent space.** The CVML-base network is trained to encode the selected Linemod objects into a set of latent variables $\mu, \sigma^2 \in \mathbb{R}^n, n \in [1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]$, and the corresponding Rotation MLP and Centre MLP are trained to regress such variables to object 6-DoF pose.

generally acquire more 3D information about the objects with increasing capacity but finally saturate at $n = 128$. It is also worth noting that the accuracy dramatically goes down at $n = 1024$, suggesting that the latent space with extremely high dimensionality does not necessarily contribute to effective pose encoding, and may instead lead to diminished performance. Therefore, in the CVML-Pose method, the latent space is configured with a dimensionality of $n = 128$ for training all VAE networks.

Regularisation of the Latent Space

When training VAE, the KL regularisation term in the modified training loss L_{train} (see Eq. 3.13), i.e. $\alpha \cdot \sum_{j=1}^{n} \left(1 + \log\left(\sigma_{ij}^2\right) - {\mu_{ij}}^2 - \sigma_{ij}^2\right)$, directly affects the construction of the latent space $\mu_{ij}, \sigma_{ij}^2 \in \mathbb{R}^n$. To find a good balance of α that is capable of having both an informative latent representation of the input data and a good generalisation ability, the CVML-base network is trained with different values of $\alpha \in [0, 0.1, 0.5, 1, 1.5, 2]$. Table 4.6 demonstrates the pose estimation results of using different values of the regularisation term α.

Based on the reported results, the value for the regularisation weight α found in the

Regularisation weight	$AR_{ADD(I)}$
$\alpha = 0$	48.12
$\alpha = 0$, no reparameterization trick	46.67
$\alpha = 0.1$	**48.81**
$\alpha = 0.5$	44.20
$\alpha = 1$	41.50
$\alpha = 1.5$	42.00
$\alpha = 2$	39.38

Table 4.6: **Ablation test on different weighting α of the KL regularisation term.** There are two cases when $\alpha = 0$ to train the autoencoder network, the first case removes the constraint of the KL regularisation, and the second case removes the reparameterization trick which changes the type of the autoencoder.

experiments is 0.1, providing a sufficiently regularised latent space without overly constraining it. Notably, when $\alpha = 0$, the CVML-base network becomes unstable and difficult to train, requiring either a very small learning rate or the omission of the reparameterization trick. However, removing the reparameterization trick means the network is no longer a VAE, and it performs worse than the case with $\alpha = 0.1$ when generalising to the test images. On the other hand, with $\alpha = 1$, the latent space is greatly affected by the KL divergence, which seems to smooth the distribution too much. This excessive smoothing can result in the loss of information about the pose, as the network focuses on minimising the KL divergence over retaining distinctive features of the input data. Thus, $\alpha = 0.1$ is selected as a proper regularisation weight for training VAE.

Network Activation Function

Another fact that may affect the quality of the latent space is the activation function of the network. To avoid the well-known vanishing gradient problem [210], various alternative activation functions like Exponential Linear Unit (ELU) [206], Sigmoid Linear Unit (SiLU) [205], and Gaussian Error Linear Unit (GELU) [211] are considered instead of the commonly used Rectified Linear Unit (ReLU). The CVML-base network is trained using these activation functions, including in the MLPs, to evaluate their effectiveness in generating better representations of the pose.

Activation function	$AR_{ADD(I)}$
ReLU	47.08
ELU	48.81
SiLU	**49.93**
GELU	49.79

Table 4.7: **Ablation test on activation functions.** The vanilla CVML-base architecture uses the SiLU activation function as shown in Figure 4.9.

From the experimental results shown in Table 4.7, GELU and SiLU activation functions make nearly no difference in interpreting object 6-DoF poses, and they are better than ELU and ReLU. This aligns with previous research proposed by Hendrycks and Gimpel [211], which demonstrates that the GELU activation can achieve the lowest reconstruction error in a self-supervised deep autoencoder trained on the MNIST dataset, compared to ELU and ReLU. In terms of pose estimation performance, the best pose estimation performance is achieved by the SiLU activation and, therefore chosen as the final activation function for both VAE and MLP. The GELU activation function is also a strong candidate for training autoencoder models due to its similar performance level.

Data Augmentation

In the process of training VAE, online data augmentation plays a significant role in improving the robustness against varying object appearances. The ablation tests focus on evaluating the effectiveness of various augmentation techniques, as outlined in Table 4.8. The random change of brightness, contrast, and saturation is used to improve robustness to variation in environmental settings. The Gaussian blur is introduced to minimise the distortion problem of the test camera. The random scale and translation are adopted to improve robustness to variation in the estimated bounding box. The random erasing technique [212] is utilised to simulate more occlusion to objects' observations. To identify the impact of these online augmentation techniques on pose estimation performance, the CVML-base network is trained with images that are randomly transformed by these techniques with a probability of 30%.

Name of the augmentation	arguments	Value
ColorJitter	brightness	$[0.4, 2.3]$
	contrast	$[0.4, 2.3]$
	saturation	$[0.8, 1.2]$
GaussianBlur	kernel size	$(5, 5)$
	sigma	$[0.1, 1.2]$
RandomAffine	translate	$(-0.1, 0.1)$
	scale	$(0.9, 1.1)$
RandomErasing	scale	$(0.1, 0.2)$
	ratio	$(0.3, 3.0)$
	value	random

Table 4.8: **Online augmentation pipeline.** All the augmentations are implemented with a 30% possibility during training of the autoencoder network.

Table 4.9 shows the pose estimation results with different data augmentations. Based on the results, it can be concluded that training with ColorJitter, GaussianBlur, and RandomAffine slightly increases the accuracy of pose estimation. These techniques contribute to the network's ability to adapt to variations in lighting conditions, camera distortions, and incorrect detection, respectively. However, the network performs worse with RandomErasing augmentation. One possible reason is that the PBR images used in the Linemod dataset already incorporate a significant level of occlusion. Introducing additional occlusion through RandomErasing may lead to situations where an object is entirely occluded, leaving the network with insufficient visual information to predict the pose. This might cause the network to learn from instances without any information, leading to destructive training updates. Given these findings, all the online augmentation techniques except for RandomErasing, are used in training the VAE, which helps to improve the robustness of the method.

Online augmentation methods	$AR_{ADD(I)}$
No data augmentation	48.81
With ColorJitter	**49.85**
With GaussianBlur	**48.87**
With RandomAffine (scale)	**50.38**
With RandomAffine (translate)	**52.07**
With RandomErasing	48.57

Table 4.9: **Ablation test on data augmentation methods.** The RandomErasing technique is omitted in training the final CVML-18 network due to the worse results.

4.3.5 Network Parameters and Training Details

The proposed CVML-Pose method, which contains the CVML-AE module, MLPs, and KNN, is mainly implemented using PyTorch [213] and scikit-learn [204]. To train the CVML-AE module and MLPs, the AdamW optimiser [203] is used with betas $= (0.9, 0.999)$, eps $= 1e-8$, and weight decay $= 1e-2$. The learning rate is scheduled to be reduced when the validation error does not improve for a certain number of epochs i.e. the "patience" in Table 4.11, with a drop factor $d = 0.2$ on a plateau. The training process is set to terminate under certain conditions, for example when the model reaches its lowest learning rate and the validation loss stops dropping for a specified number of epochs, denoted as N, and $N = 50$ for the CVML-AE module, $N = 500$ for the MLPs. The batch size for the CVML-AE module is set to 128, while MLPs take all the inputs as a batch. The network parameters of the proposed CVML-AE module and MLPs are also reported in Table 4.10. The KNN regressor, implemented using scikit-learn, uses the "distance" weighting function, in which case closer neighbours of a query point will have a greater influence than neighbours which are further away. The training details of the CVML-AE module, the MLPs, and the KNN regressor are also shown in Table 4.11.

Networks	Number of parameters
CVML-base network	2.7774211×10^7
CVML-18 network	7.9082243×10^7
CVML-34 network	1.12682499×10^8
Rotation MLP	1.0966×10^4
Centre MLP	1.1338×10^4

Table 4.10: **Network parameters of the CVML-AE module and MLPs.**

Model name	training loss	validation loss	learning rate	patience
CVML-AE	L_{train}	L_{valid}	$[10^{-4}, 10^{-6}]$	50
MLPs	L1	L1	$[10^{-2}, 10^{-7}]$	500
KNN	Euclidean	L1	-	-

Table 4.11: **Training details of the CVML-AE module, MLPs, and KNN.** The term 'patience' here means how many epochs the learning rate would drop if the validation loss does not improve.

In terms of the training time, using the data described in Section 3.2.6, e.g. Linemod

PBR images, and the network configurations described in Section 4.3.4, two CVML-base networks can be trained in parallel in approximately 8 hours on a single NVIDIA Geforce RTX 3090. CVML-18 takes about 12 hours, and CVML-34 takes around 20 hours on the same device. All MLPs are trained in parallel on the same device, which takes roughly 12 hours. The KNN regressor can be trained in less than 2 minutes on the Intel® Core™ i9-7900X CPU. More details on the implementation of the CVML-Pose method can be found at: https://github.com/JZhao12/CVML-Pose.

4.3.6 Remarks

Based on the comprehensive ablation tests detailed in Section 4.3.4, it can be concluded that the modified VAE's latent space can generalise more effectively than the DAE network for object 6-DoF pose estimation. The favourable configuration involves using the CVML-18 network as the autoencoder model, with nearest neighbour upsampling in the decoder network, and an additional intermediate layer in the latent space. Key hyperparameters of the network include the latent space dimensionality $n = 128$, the regularisation $\alpha = 0.1$, and the SiLU activation function. The network is also trained with several data augmentation techniques including ColorJitter, GaussianBlur, and RandomAffine, to enhance its robustness and generalisation capability.

The CVML-Pose method is trained to implicitly learn regularised latent space representations from colour images, subsequently employing MLPs and KNN to interpolate the complete 6-DoF pose from the learnt representations. For Linemod objects, the method is trained with corresponding PBR images, and evaluated on both the Linemod (see Table 5.2) and the BOP version of the Linemod-Occluded (see Table 5.3) test images. For YCB-Video objects, the method is trained with a combination of PBR, non-PBR synthetic, and real images, and evaluated on the BOP version of the YCB-Video (see Table 5.4). The evaluation details, results, discussions, as well as conclusions can be found in Chapter 5.

To the best of my knowledge, the CVML-Pose method is the first to unify a generative model's latent space and the regression-based algorithms to estimate continuous pose representations, specifically, using the VAE's latent space. As it will be demonstrated in the next chapter, the CVML-Pose method significantly outperforms methods based on the autoencoder's latent space [65, 66] and 2D-3D model correspondence [67, 68], and achieves competitive pose accuracy to the most state-of-the-art methods [69, 70] across multiple benchmark datasets. Moreover, the use of regularised latent representation shows promising results, which also facilitates the subsequently proposed multi-object pose estimation method (Section 4.4) using a conditional variational autoencoder.

4.4 CVAM-Pose: Conditional Variational Autoencoder for Multi-Object 6-DoF Pose Estimation

The CVAM-Pose pipeline (Figure 4.14) contains two main stages of estimating the complete 6-DoF poses for multiple objects, without using 3D information, depth measurements, or iterative post-refinement. The first stage is to learn multi-object representations by training a label-embedded conditional variational autoencoder (CVAE), to implicitly learn to represent multiple objects in a single latent space through the encode-decode process. Unlike the original CVAE, the proposed label-embedded CVAE is trained with the layer-wise one-hot encoding technique, where the encoded categorical labels are represented as complete feature maps which exist in every convolutional layer. The second stage involves applying regression-based algorithms, such as MLP and RF, which are trained with the one-hot encoded label vectors as well, to regress the learnt multi-object latent representations to objects' 6-DoF poses. The proposed CVAM-Pose method extends the CVML-Pose method with multi-object prediction, improving both scalability and computational efficiency, but maintains almost the same performance level as the CVML-Pose

method.

This work has been evaluated on the BOP version of the Linemod-Occluded benchmark dataset, which outperforms not only latent representation methods with both single-object and multi-object prediction [65, 66], but also methods using 3D model information [67, 68], and achieves comparable results to the state-of-the-art [69, 70]. The remainder of this section describes the following details of the method including learning multi-object representations using conditional variational autoencoder (Section 4.4.1), multi-object pose regression (Section 4.4.2), ablation tests of the method (Section 4.4.3), and initial findings (Section 4.4.4). The evaluation procedure and results will be explained in Section 5.2 and 5.3.3.

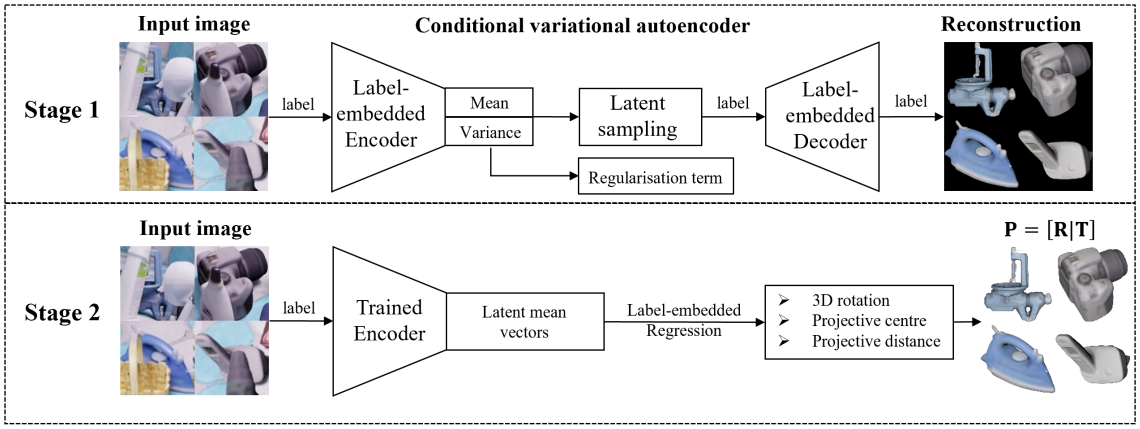


Figure 4.14: **CVAM-Pose pipeline.** During training, the CVAE network captures both input images of objects and their corresponding label conditions in its latent space, which can be further interpreted to multi-object 6-DoF poses.

4.4.1 Learning Multi-Object Representation using Conditional Variational Autoencoder

To effectively learn regularised multi-object representation, a CVAE network (see Section 3.4.3 for details of the network) is trained to encode images of different objects in a single latent space and output the corresponding clean reconstruction. A critical feature of this approach is the embedded categorical labels throughout the whole network, enabling a well-organised space where different objects do not affect each other. **Algorithm 3** presents the pseudo code for training the CVAE network.

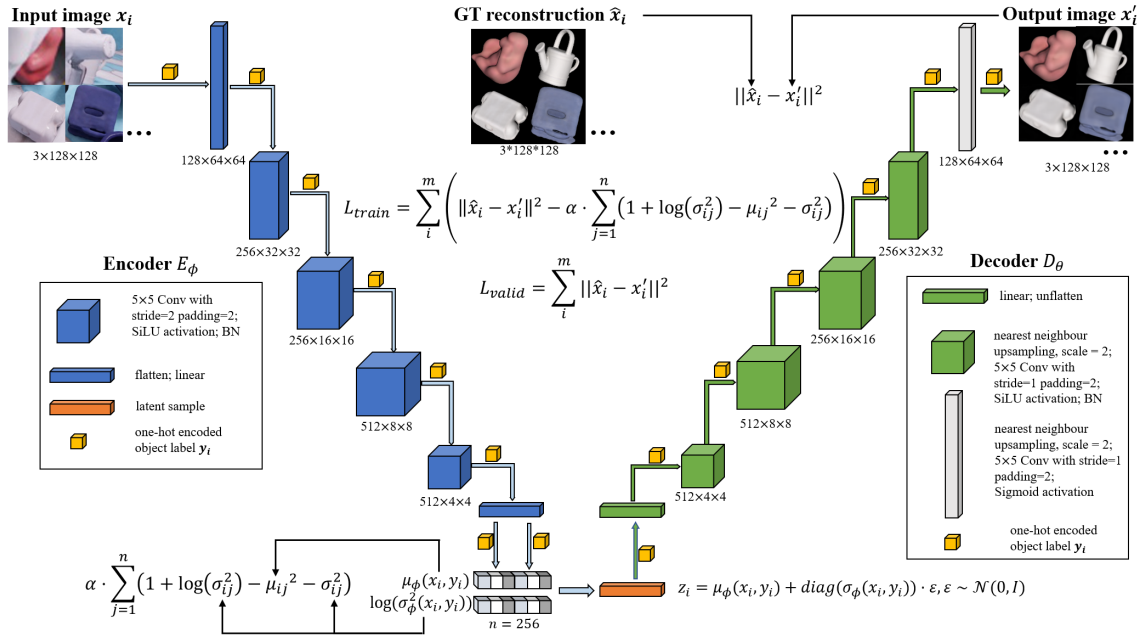


Figure 4.15: **The modified conditional variational autoencoder architecture.** ϕ represents all the parameters of the encoder network (the blue part) and θ represents all the parameters of the decoder network (the green part). The input images of objects are combined with their one-hot encoded label y_i first, and then forwarded to the encoder where the label exists in every convolution layer, which produces the conditioned latent variables. The decoder outputs the corresponding reconstruction images from the latent sampling, where the same label y_i is also applied in each layer.

As depicted in Figure 4.15, the encoder network $E_\phi(x_i, y_i)$ processes both input image x_i and its embedding label y_i. Specifically, the conditional variable is embedded at every convolution layer (layer-wise), until the final latent variables $(\mu_\phi(x_i, y_i), \sigma_\phi^2(x_i, y_i)) \in \mathbb{R}^n$ are obtained in the latent space. After sampling through reparameterization trick $z_i = \mu_\phi(x_i, y_i) + diag(\sigma_\phi(x_i, y_i)) \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, the decoder network $D_\theta(z_i)$ outputs the reconstruction image x_i' from the sampled variables z_i, where the label embedding also exists in every convolution layer in the decoder. The training loss stays the same as the one used in the proposed CVML-Pose method, i.e. a pixel-wise L2 loss, computed between the output image x_i' and the gt reconstruction image $\hat{x}_i$, and a weighted KL divergence, regularising the latent space.

Based on the implemented ablation tests in the CVML-Pose method (see Section 4.3.4), it can be drawn on the experiences to obtain the approximate good configurations for the CVAM-Pose method. For example, to abstract more infor-

Algorithm 3 Training the label-embedded CVAE Network

Require: Input data x, gt reconstruction data $\hat{x}$, one-hot encoded categories y
Require: Encoder network E_ϕ
Require: Decoder network D_θ
Require: Mini-batch size m
Require: Latent space size n
Require: Auxiliary random variable $\epsilon \sim \mathcal{N}(0, I)$
Require: Regularisation weight α
Require: AdamW optimiser parameters

while not converged **do**
  Sample a mini-batch $\{x_i, y_i, \hat{x}_i\}_{i=1}^m$
  for each (x_i, y_i) in the mini-batch **do**
    $\mu_\phi(x_i, y_i), \sigma_\phi^2(x_i, y_i) \leftarrow E_\phi(x_i, y_i)$
    $z_i = \mu_\phi(x_i, y_i) + diag(\sigma_\phi(x_i, y_i)) \cdot \epsilon$
    $x'_i \leftarrow D_\theta(z_i, y_i)$
  end for
  Compute the $ELBO$ loss:

$$ELBO = \sum_{i=1}^{m} \left(||\hat{x}_i - x'_i||^2 - \alpha \cdot \sum_{j=1}^{n} \left(1 + \log\left(\sigma_{ij}^2\right) - \mu_{ij}^2 - \sigma_{ij}^2 \right) \right)$$

  Backpropagate the loss
  Update the encoder and decoder parameters ϕ, θ using AdamW optimiser
  Evaluate on the validation set using the L2 loss:

$$\text{L2} = \sum_{i=1}^{m} ||\hat{x}_i - x'_i||^2$$

  if validation loss does not improve for 50 epochs **then**
    Stop training to prevent overfitting
  end if
end while

mation in the latent space, the encoder part in Figure 4.15 can be replaced by a ResNet-18 backbone network. In this case, the conditional label y_i is embedded in every residual block instead of every convolution layer, simplifying the embedding process. The hyperparameters of the autoencoder network can be also fine-tuned, for instance, since the network needs to increase its capability of learning multi-object representations, the dimensionality n of the latent space can be expanded from 128 to higher values, e.g. 256 or 512. Also, to see whether the use of the layer-wise one-hot encoding technique would help to abstract more high-level features of the object, the original CVAE network is compared with the network shown in Figure 4.15, where in the original CVAE, the conditional labels only exist at the first convolutional layer in both encoder and decoder. Therefore, to obtain the desirable structure of the CVAE network, the ResNet-based CVAE, the one proposed in Figure 4.15, and the original CVAE are compared against, their results are reported in Section 4.4.3.

4.4.2 Multi-Object Pose Regression

After training the CVAE network, the autoencoder's latent space accumulates robust multi-object representation. The subsequent stage of the CVAM-Pose method involves a pose regression strategy, which is similar to the one used in the DALSR-Pose and CVML-Pose methods, with an adaptation to handle the multi-object scenario. This pose regression strategy, as detailed in Figure 4.16, incorporates the additional one-hot encoded object labels, to facilitate precise pose estimation for multiple objects.

For estimating the 3D rotation $\mathbf{R} \in \mathrm{SO}(3)$, an MLP regressor is trained to regress the learnt multi-object latent representations to the continuous 6D rotation representation $R_{6D} \in \mathbb{R}^6$ [118]. The 3D translation $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$ is predicted separately, through the estimation of 2D projective centre $P_c = (x_c, y_c)^T \in \mathbb{R}^2$ and 2D projective distance $T_z \in \mathbb{R}$. The complete translation vector can be finally calculated based on the projective camera model (Eq. 4.2 and 4.3).

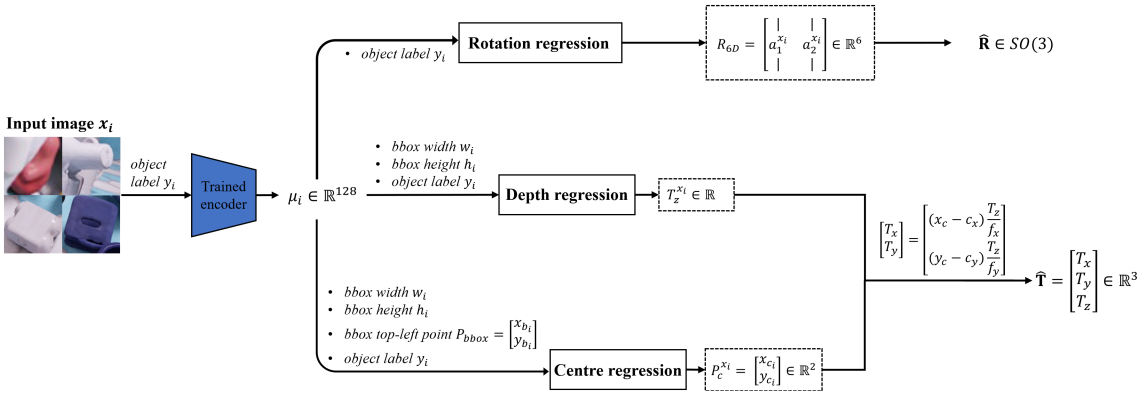


Figure 4.16: **Regress multi-object 6-DoF poses from the learnt multi-object latent representations.** The pose regression procedure is similar to DALSR-Pose and CVAM-Pose, the only difference is all the representations of objects are regressed together instead of per object per regressor. The one-hot encoded object label y_i is concatenated with the learnt representations as well, which helps the regressor to learn multi-object poses.

In the practical implementations of the pose regression, the implemented KNN regressor does not work well with the multi-object estimation of 2D projective distance T_z, since the effectiveness of the algorithm can be limited by a large number of data points, i.e. the algorithm requires calculating the distance between the query instance and every other instance, which can be computationally complex for large datasets, as well as the requirement of large system memory. Therefore, the KNN regressor is replaced by a random forest (RF) regressor, which is typically more suitable for handling large datasets because of the building of multiple decision trees to efficiently process a large number of data. The validation data are also used to obtain favourable hyperparameters of the RF regressor, for example, the number of trees in the forest and maximum depth of the tree, where the validation error is calculated based on the complete 3D translation instead of just T_z, and the final hyperparameters of the RF can be determined when the validation error is the smallest.

4.4.3 Ablation Tests on Label Embedding

As mentioned in Section 4.4.1, an ablation experiment is implemented to investigate the effectiveness of the proposed layer-wise one-hot encoding technique in the CVAE

network. An additional experiment is also conducted to see whether the technique can help with multi-object pose regression. The ablation results are evaluated on the BOP version of the Linemod-Occluded benchmark dataset, using the BOP Challenge metrics, including VSD, MSSD, and MSPD (see Section 3.5.3 and 3.5.4 for details of the metrics). Unlike previous tests implemented in the DALSR-Pose and CVML-Pose methods that use the gt bounding box, results reported here are calculated based on the bounding box from the pretrained Mask-RCNN detector [214]. The complete evaluation procedure and the results of the CVAM-Pose method are illustrated in detail in Chapter 5.

Metric	original CVAE	CVAE in Figure 4.16	ResNet-based CVAE
AR_{VSD}	0.256	0.275	0.309
AR_{MSSD}	0.255	0.288	0.318
AR_{MSPD}	0.630	0.644	0.709
AR_{score}	0.380	0.402	0.445

Table 4.12: **Comparison between different CVAE networks.** Each column represents the three metrics of a method on all Linemod-Occluded objects, and the final performance is calculated by $AR_{score} = (AR_{VSD} + AR_{MSSD} + AR_{MSPD})/3$.

In the original implementation of the CVAE network [64], the label condition only incorporates low-level features, i.e. the feature map of images before being sent into the first convolutional layer in the encoder. This can be less effective for learning distinct multi-object representations because, with the increasing depth of the network, the label-conditioned features learned at the very beginning may not be evident in the latent space. To investigate this, the original CVAE network is trained to compare with the proposed layer-wise one-hot encoded CVAE network. Additionally, a variant of the CVAE network using the ResNet-18 backbone, where the conditional labels are incorporated at every residual block (the convolution layers inside the block are not embedded with the labels), is also examined. The results, presented in Table 4.12, indicate that the layer-wise one-hot encoding technique significantly improves the network's capacity to capture high-level features related to object pose. Furthermore, employing a more advanced encoder network, like ResNet-18, is also useful for abstracting useful information in the latent space.

Metric	with labels	without labels
AR_{VSD}	0.275	0.251
AR_{MSSD}	0.288	0.243
AR_{MSPD}	0.644	0.554
AR_{score}	0.402	0.349

Table 4.13: **Effects of using one-hot encoded labels in pose regression.** Each column represents the three metrics of a method on all Linemod-Occluded objects, and the final performance is calculated by $AR_{score} = (AR_{VSD} + AR_{MSSD} + AR_{MSPD})/3$.

Another aspect of the tests involves determining whether the learnt representations can be effectively regressed to multi-object 6-DoF poses without the need for one-hot encoded label conditions. For this purpose, the same regression models, including MLPs and RF, are trained with different input data configurations. The results are presented in Table 4.13, showing that with the label conditions, the pose regressors can result in better pose estimation accuracy. However, it is also observed that even without explicit label conditions, the learnt representations intrinsically contain label information. These findings suggest that the CVAE network is capable of encoding relevant multi-object poses in the latent space, as well as the importance of the proposed layer-wise one-hot encoding technique.

4.4.4 Remarks

Based on the results of the ablation tests reported in Section 4.4.3, it can be concluded that the proposed CVAE network with layer-wise one-hot encoding network is possible to learn multi-object 6-DoF poses. The favourable configuration of the method is to use the ResNet-based CVAE network as the autoencoder model, and trained with selected hyperparameters including the dimensionality $n = 256$, regularisation factor $\alpha = 0.1$, SiLU activation, and data augmentation (see Section 4.3.4). Like the previous proposed methods, the ResNet-based CVAE network is trained to implicitly learn regularised latent space representations from colour images of multiple objects, subsequently employing MLPs and RF to interpolate the objects' poses from the learnt representations. For Linemod-Occluded objects, the final

CVAM-Pose method is trained on 8 objects with the corresponding PBR images, and evaluated on the BOP version of the Linemod-Occluded dataset (see Table 5.3). The evaluation details can be found in Chapter 5. For the results reported in the thesis, CVAM-Pose was trained with 8 different objects. Experiments with larger numbers of objects were also conducted but not reported. It was observed that increasing the number of objects in the CVAM-Pose method beyond 15 would lead to a decrease in pose estimation accuracy.

To the best of my knowledge, the CVAM-Pose is the first to combine a conditional variational autoencoder with continuous regression algorithms to estimate multi-object 6-DoF poses. The proposed layer-wise one-hot encoding technique also increases the capability of learning more high-level features/representations. As it will be demonstrated later in Section 5.3.3, the CVAM-Pose method shows promising results on the challenging occlusion benchmark dataset, and archives almost the same pose accuracy as the proposed CVML-Pose method, but significantly improves the scalability and computational efficiency with multi-object predictions in a single-encoder-single-decoder architecture.

4.5 Summary

This chapter mainly focuses on the methodology part of the thesis, starting with an overview of the Auto-Pose framework, as well as the novelties of the three autoencoder-based methods, DALSR-Pose, CVML-Pose, and CVAM-Pose. The proposed methods aim to achieve state-of-the-art pose estimation performance without using an object's 3D model, depth sensor, or iterative refinement process. Each of these methods employs a workflow of initially learning the latent space representations from the selected autoencoder model (explained in Chapter 3), followed by the interpolation of the objects' poses using regression-based algorithms. The primary difference between the methods is the type of latent space, where the DALSR-Pose method trains the DAE model for a robust latent representation, the CVML-Pose

method forms a regularised latent space using the VAE model, and the CVAM-Pose method obtains not only regularised but also constrained multi-object representations from the CVAE model. A series of ablation experiments, e.g. the dimensionality of the latent space and rotation representations, have been also conducted to obtain favourable configurations of the methods.

Given that the evaluation procedures for the three proposed methods are similar, owing to their shared idea of using the autoencoder's latent space, the methods are assessed together in the later evaluation chapter (Chapter 5). The comprehensive comparison with the state-of-the-art methods will also demonstrate their advantages on the more challenging datasets with occlusion and clutter. For more details on the evaluation pipeline, results, and discussion of the methods, please see the next chapter.

Chapter 5

Evaluation and Results

5.1 Introduction

This chapter details the comprehensive evaluation process, results, discussions, and conclusions for the three autoencoder-based 6-DoF pose estimation methods presented earlier. As illustrated in the previous chapter, the proposed methods are benchmarked in two ways. The first involves performing a series of ablation tests to determine favourable configurations of the methods, which have been concisely illustrated and summarised in Section 4.2.3, 4.3.4, and 4.4.3. The second way is to follow the evaluation methodologies proposed in the state-of-the-art approaches and the BOP Challenge [7, 26, 141], which is the main focus of this chapter.

The contents of this chapter are organised as follows: description of the complete evaluation pipeline (Section 5.2), comprehensive results and detailed discussions (Section 5.3), and concluding remarks (Section 5.4).

5.2 Evaluation Pipeline

This section introduces the complete evaluation procedure for the three methods, all of which are based on the autoencoder architecture. Given the similarities in their inference processes, the inference procedure of the CVML-Pose method (illus-

trated in Figure 5.1) is chosen as a representative example. The evaluation of these methods is illustrated from two key perspectives, including the evaluation setup of the methods (Section 5.2.1), and the evaluation of the adopted object detectors (Section 5.2.2).

5.2.1 Evaluation Setup

To facilitate an intuitive comparison with the state-of-the-art methods, the evaluation setup for the proposed methods on the Linemod dataset [1, 2] and the BOP version datasets [3, 4, 5, 6, 7] are different, based on the selected data and metrics.

The original Linemod dataset, as elaborated in Section 3.2.1, is split into training (15%) and test sets (85%) by Brachmann et al. [3], and the bowl and driller objects (2 out of 15) are omitted due to improper 3D models. Following the evaluation strategy adopted in many state-of-the-art methods [3, 35, 50, 65, 67, 70, 119, 126, 128], the proposed DALSR-Pose and CVML-Pose methods are evaluated on the test set of the remaining 13 Linemod objects. The performance score $AR_{ADD(I)}$ is calculated based on the average recall rate of the ADD(I) metric (details in Section 3.5.2). The two methods are compared against current leading methods [35, 65, 67, 68, 70, 72, 74, 119], with results presented in Table 5.2.

For evaluation of the Linemod-Occluded [3, 4] and the YCB-Video [5, 6] datasets, the standard protocol of the BOP Challenge is used. As explained in Section 3.2.2 and 3.2.3, the challenge organisers manually select a subset of the original test data for evaluation, known as the BOP version, by removing low-quality images with inaccurate ground truth (gt) poses. This data selection also speeds up the evaluation process. Participants of the BOP Challenge must use the three evaluation metrics (VSD, MSSD, and MSPD) proposed by Hodan et al. [7] (details in Section 3.5.3 and 3.5.4), which standardise comparisons. For the BOP version of the Linemod-Occluded test data, the three proposed methods are compared against the state-of-the-art methods participated in the challenge [35, 65, 66, 67, 68, 69, 70, 71, 73, 75, 119],

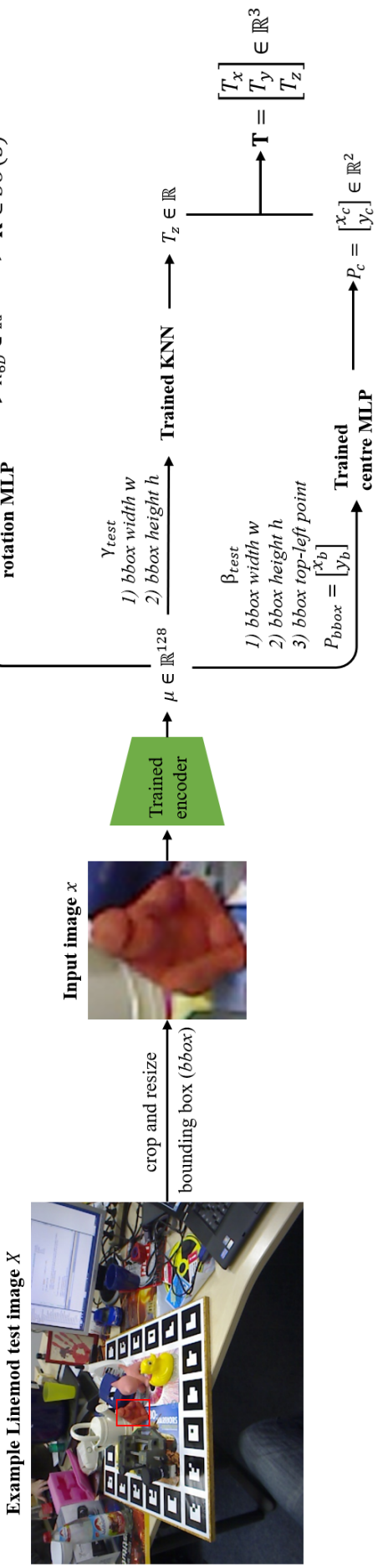


Figure 5.1: **Inference procedure of the CVML-Pose method.** During inference, the input test image x can be obtained from the original test image X using the crop-and-resize strategy (see Section 3.2.6) with the bounding box from object detection models, e.g. a Mask R-CNN detector. Then the test image x is fed to the trained encoder which generates the test latent variables $\mu \in \mathbb{R}^{128}$. Finally, the trained MLP and KNN regressor can predict the object's 3D rotation $\mathbf{R} \in \mathrm{SO}(3)$ and 3D translation $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$ from the latent variables.

and their comprehensive results are reported in Table 5.3. The final performance score for each method is calculated based on the average of the three metrics: $\text{AR}_{\text{score}} = (\text{AR}_{\text{VSD}} + \text{AR}_{\text{MSSD}} + \text{AR}_{\text{MSPD}})/3$. The same protocol is applied to the evaluation of the BOP version of the YCB-Video test data, and the CVML-Pose method is compared against these leading methods [65, 66, 67, 68, 69, 70, 71, 73, 75, 76].

For the latest and comprehensive results, please refer to the BOP Challenge leaderboard[1]. Details on the adopted evaluation metrics can be found in Section 3.5. It is also noted that, while the proposed methods do not require object 3D models for inference, as outlined in previous chapters, the evaluation metrics used, including ADD(I), VSD, MSSD, and MSPD, do necessitate object's model points for accurate assessment.

5.2.2 Evaluation of Object Detection

For evaluation, an object detector needs to be built to enable the proposed methods to operate on the test scene images. For instance, a Mask-RCNN detector [214], pretrained by the CosyPose method [75], can be utilised to detect objects of interest in test images. As depicted in Figure 5.1, with the crop-and-resize strategy described in Section 3.2.6, the test images of the target object can be obtained based on the detection bounding box from the Mask-RCNN detector. Subsequently, the cropped and resized images are fed into the trained encoder network, generating a set of test latent variables. Finally, these variables are forwarded to the trained pose regressors to predict the complete 6-DoF pose of the test objects. The number of detected test instances (using the pretrained Mask-RCNN detector) and the gt test instances, for both the Linemod test data and the BOP version of the Linemod-Occluded test data, is shown in Table 5.1.

For real-time demonstration, a video is also provided online[2], showcasing real-time

[1]Accessible at: https://bop.felk.cvut.cz/leaderboards/.
[2]video available: https://ieeexplore.ieee.org/document/10040668.

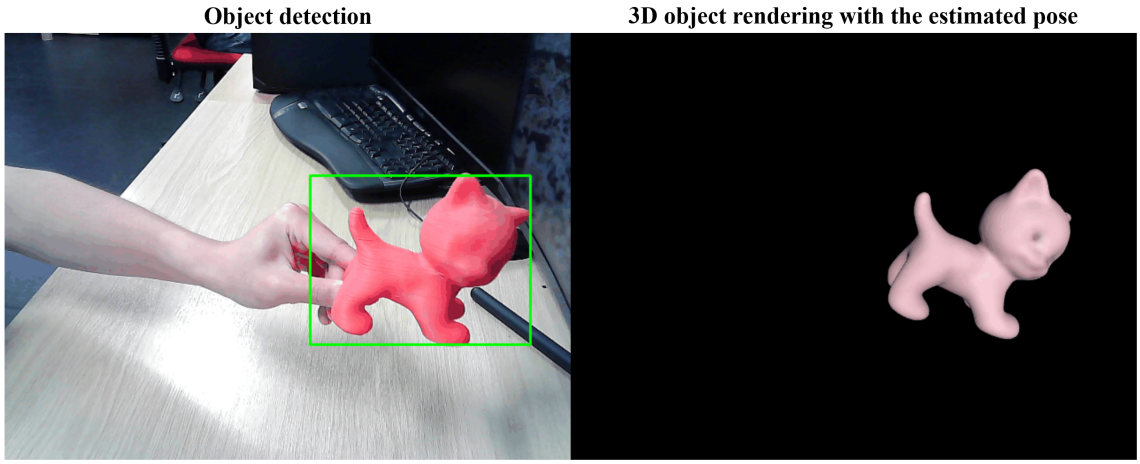
	Linemod		BOP Linemod-Occluded	
Objects	gt	Mask-RCNN	gt	Mask-RCNN
ape	1050	1021	182	137
benchvise	1031	1031	-	-
cam	1020	954	-	-
can	1016	1015	199	177
cat	1002	1000	189	130
driller	1009	1006	200	185
duck	1065	1064	182	175
eggbox	1065	974	180	103
glue	1036	1033	142	126
holepuncher	1051	1050	200	198
iron	979	979	-	-
lamp	1042	1042	-	-
phone	1041	1021	-	-

Table 5.1: **Number of test instance in the Linemod [1, 2] and the BOP version [7] of the Linemod-Occluded [3, 4] datasets.** The gt objects are cropped from the gt bounding box, while the Mask-RCNN objects are cropped based on the Mask-RCNN detector. The Linemod-Occluded dataset has only 8 objects as described in Section 3.2.2.

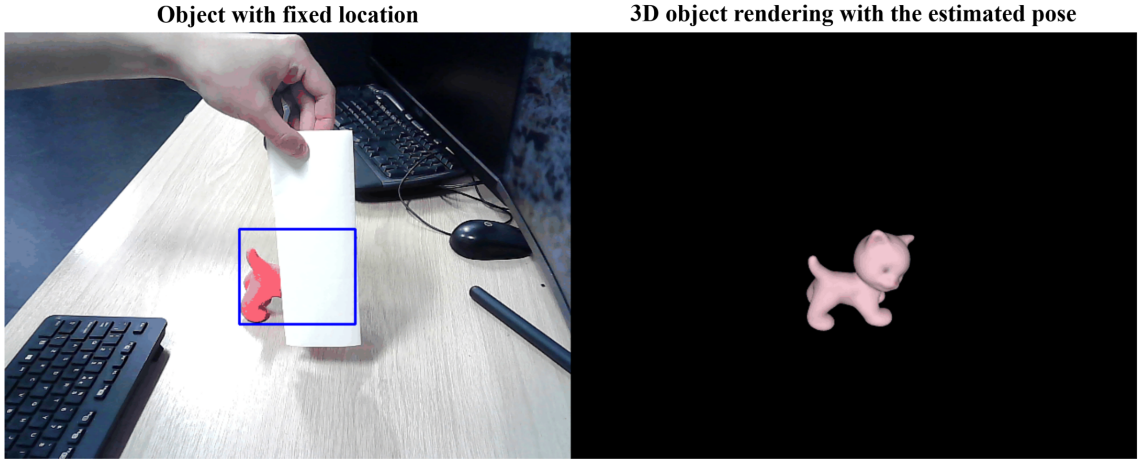
pose estimation under complex scenarios using the proposed CVML-Pose method. The video exhibits the method's effectiveness in handling different levels of object occlusions and cluttered scenes, with low-resolution images captured using an inexpensive webcam, and without specific requirements for illumination or any constraints on the object's motion. The webcam intrinsic parameters are estimated using the checkerboard-based procedure from MATLAB[3] [215], with an average reprojection error of 0.47 pixels. The parameters are used to remove radial distortion and calculate the translation vector of the pose. Example screenshots of the video are displayed in Figure 5.2. An Inertial Measurement Unit (IMU) sensor is used as a reference to calculate errors, which is shown in Appendix B.

To evaluate the contribution of the object detector to the 6-DoF pose estimation task, a comparison is made between the pose estimation results based on the gt bounding box and those based on the Mask-RCNN detector, as shown in Table 5.7. Additionally, to investigate how training data could affect detection models and, in turn, influence pose estimation accuracy, two models are used for evaluation, includ-

[3]UCLan academic licence.



(a) **First scenario: free motion.** The target object is unrestrictedly moved with different levels of 3D rotation and translation. It is detected by a pretrained Mask-RCNN detector, and the green bounding box shows the result of the detection.



(b) **Second scenario: heavy occlusion.** The target object is fixed in the area of the blue bounding box, and the pose is still estimated for each frame independently. The white paper is used to gradually introduce occlusions to the object, and when the object gets partially or even heavily occluded, the pose can be still estimated correctly, which shows robustness against occlusions.

Figure 5.2: **Real-time demonstration of the proposed CVML-Pose method in different scenarios.** The Linemod cat object [1, 2] is selected as the target object, and its 3D model is rendered in real time at the pose that is estimated from the proposed CVML-Pose algorithm. If the rendering starts to move to another pose, that means the method cannot cope with such scenarios. It is noted that the CVML-Pose does not require an object 3D model for inference, the model is used just for visualisation of the pose estimation results.

ing the pretrained Mask-RCNN detector, and another detector called YOLOX [216] pretrained by [71]. The performance of these models in both object detection and pose estimation is reported in Table 5.8.

5.3 Results and Discussions

This section presents the results and detailed discussions for the proposed methods, starting with the DALSR-Pose method (Section 5.3.1), followed by the CVML-Pose method (Section 5.3.2) and the CVAM-Pose method (Section 5.3.3). Subsequent sections include a comparative analysis of different types of deep learning-based methods (Section 5.3.4), performance against occlusion (Section 5.3.5), as well as the impact of using different object detectors (Section 5.3.6).

The main results of the proposed methods are reported in Table 5.2, 5.3, and 5.4, where they are compared against a range of state-of-the-art methods, including AAE [65], AAE-ICP (AAE with ICP refinement [146]), Multi-Path [66] (an extended version of AAE for multi-object pose estimation), SSD6D [119], CosyPose [75], GDR-Net [71], DPOD [67], Pix2Pose [68], CDPN [70], CDPNv2 (an improved version of CDPN), EPOS [69], PVNet [35], SurfEmb [73], RNNPose [72], ZebraPose [76], and YOLOv5-6D [74]. Many of these competitive methods have been published at top conferences like ICCV[4], ECCV[5], and CVPR[6], or awarded in the BOP Challenge [7, 26, 141]. For example, AAE was awarded the Best Paper at ECCV 2018, as well as the best open source and the fastest method at the BOP Challenge 2019 (BOP19). SSD6D was presented at ICCV 2017, DPOD at ICCV 2019, and Multi-Path at CVPR 2020. PVNet, EPOS, SurfEmb, ZebraPose, and RNNPose were presented at CVPR 2019, CVPR 2020, CVPR 2022, CVPR 2022, and CVPR 2022, respectively, with PVNet also being featured in an oral presentation. ZebraPose also reports the results in the latest BOP Challenge 2023 [217]. CosyPose, pub-

[4]The International Conference on Computer Vision
[5]The European Conference on Computer Vision
[6]The IEEE / CVF Computer Vision and Pattern Recognition Conference

lished at ECCV 2020, won the BOP Challenge 2020 (BOP20). GDR-Net, published at CVPR 2021, won the BOP Challenge 2022 (BOP22). Pix2Pose was featured at ICCV 2019 and distinguished as the best method on the YCB-Video and RU-APC [155] datasets at the BOP19. CDPN, published at ICCV 2019, was the top RGB method at the BOP19, and its extended version, CDPNv2, was the winner on the HomebrewedDB dataset [153] at the BOP20. The methods listed in the tables are categorised based on the criteria outlined in Section 2.3, where the three proposed methods are categorised as the latent representation method.

Additional results of CVML-Pose on individual Linemod-Occluded objects are also presented in Table 5.5. The performance score for individual objects, AR_{object}, is calculated from the average recalls across the three metrics: AR_{VSD}, AR_{MSSD}, and AR_{MSPD}. The average value, denoted as Avg, is identical to the reported main results in Table 5.3. The results for occlusion are shown in Section 5.3.5, which assess how the visibility of objects in the scene images influences the accuracy of pose estimation.

	Latent representation methods				**Direct method**	**Indirect methods**					
Objects	DALSR-Pose	CVML-Pose	AAE [65]	AAE-ICP [65]	SSD6D [119]	DPOD [67]	Pix2Pose [68]	CDPN [70]	PVNet [35]	RNNPose [72]	YOLOv5-6D [74]
ape	16.85	25.07	4.18	24.35	2.6	37.22	58.1	64.38	43.62	88.19	87.81
benchvise	52.09	48.50	22.85	89.13	15.1	66.76	91.0	97.77	99.90	100.0	100.0
cam	23.79	26.73	32.91	82.10	6.1	24.22	60.9	91.67	86.86	98.04	97.45
can	40.20	48.57	37.03	70.82	27.3	52.57	84.4	95.87	95.47	99.31	99.31
cat	25.90	35.70	18.68	72.18	9.3	32.36	65.0	83.83	79.34	96.41	96.21
driller	45.43	46.02	24.81	44.87	12.0	66.60	76.3	96.23	96.43	99.70	99.11
duck	13.44	24.81	5.86	54.63	1.3	26.12	43.8	66.76	52.58	89.30	86.57
eggbox	86.34	85.73	81.00	96.62	2.8	73.35	96.8	99.72	99.15	99.53	100.0
glue	46.18	55.28	46.17	94.18	3.4	74.49	79.4	99.61	95.66	99.71	100.0
holepuncher	24.10	23.71	18.20	51.25	3.1	24.50	74.8	85.82	81.92	97.43	95.34
iron	53.83	52.20	35.05	77.86	14.6	85.02	83.4	97.85	98.88	100.0	99.19
lamp	54.70	56.91	61.15	86.31	11.4	57.26	82.0	97.89	99.33	99.81	100.0
phone	24.00	31.15	36.27	86.24	9.7	29.08	45.0	90.75	92.41	98.39	97.89
$AR_{ADD(I)}$	38.99	43.11	32.63	71.58	9.1	50	72.38	89.86	86.27	97.37	96.84

Table 5.2: **Results on the Linemod dataset [1, 2].** Results of other methods are taken from [35, 65, 67, 68, 70, 72, 74]. Each column represents the ADD(I) metric on a single object, and the performance score $AR_{ADD(I)}$ is the average across all objects. The glue and eggbox objects are considered symmetric, and the results of the bowl and cup objects are not included due to the reported problems with the 3D models as discussed in Section 3.2.1. The results of other state-of-the-art methods, including Multi-Path [65], EPOS [69], CosyPose [75], GDR-Net [71], and SurfEmb [73] are not reported here because they do not provide results on the dataset.

	Latent representation methods						Direct methods			Indirect methods						
Metric	DALSR-Pose	CVML-Pose	CVAM-Pose	AAE [65]	AAE-ICP [65]	Multi-Path [66]	SSD6D [119]	CosyPose [75]	GDR-Net [71]	DPOD [67]	Pix2Pose [68]	CDPN [70]	CDPNv2 [70]	EPOS [69]	PVNet [35]	SurfEmb [73]
AR_{VSD}	0.298	0.331	0.309	0.090	0.208	0.150	0.047	0.480	0.549	0.101	0.233	0.393	0.445	0.389	0.428	0.497
AR_{MSSD}	0.306	0.347	0.318	0.095	0.218	0.153	0.083	0.606	0.701	0.126	0.307	0.537	0.612	0.501	0.543	0.640
AR_{MSPD}	0.679	0.714	0.709	0.254	0.285	0.346	0.285	0.812	0.887	0.278	0.550	0.779	0.815	0.750	0.754	0.851
AR_{score}	0.428	0.464	0.445	0.146	0.237	0.217	0.139	0.633	0.713	0.169	0.363	0.569	0.624	0.547	0.575	0.663

Table 5.3: **Results on the BOP version [7] of the Linemod-Occluded dataset [3, 4].** Results of other methods are taken from the BOP Challenge leaderboard on Linemod-Occluded[7]. Each column represents the results of the VSD, MSSD, and MSPD metrics on all objects, and the performance score is calculated by $AR_{score} = (AR_{VSD} + AR_{MSSD} + AR_{MSPD})/3$. The DALSR-Pose, CVML-Pose, CVAM-Pose, SSD6D, CosyPose, GDR-Net, Pix2Pose, CDPN, CDPNv2, EPOS, PVNet, and SurfEmb, are trained with the Linemod PBR images. Please also note that the Pix2Pose method has two entries, the old version has $AR_{score} = 0.281$, and the more advanced version which has $AR_{score} = 0.363$ is chosen for a fair comparison.

	Latent representation methods				Direct methods		Indirect methods						
Metric	CVML-Pose	AAE [65]	AAE-ICP [65]	Multi-Path [66]	CosyPose [75]	GDR-Net [71]	DPOD [67]	Pix2Pose [68]	CDPN [70]	CDPNv2 [70]	EPOS [69]	SurfEmb [73]	ZebraPose [76]
AR_{VSD}	0.500	0.399	0.460	0.279	0.772	0.760	0.196	0.372	0.329	0.396	0.626	0.548	0.770
AR_{MSSD}	0.592	0.428	0.581	0.297	0.842	0.846	0.216	0.429	0.478	0.570	0.677	0.620	0.842
AR_{MSPD}	0.537	0.510	0.475	0.292	0.850	0.869	0.256	0.571	0.564	0.631	0.783	0.773	0.872
AR_{score}	0.543	0.446	0.505	0.289	0.821	0.825	0.222	0.457	0.457	0.532	0.696	0.647	0.828

Table 5.4: **Results on the BOP version [7] of the YCB-Video dataset [5, 6].** Results of other methods are taken from the BOP Challenge leaderboard on YCB-Video[8]. The SSD6D [119] and PVNet [35] methods are not included in the table because they do not submit the YCB-Video results to the challenge.

5.3.1 DALSR-Pose

Table 5.2 and 5.3 report the results of the proposed DALSR-Pose method on the Linemod dataset and the BOP version of the Linemod-Occluded dataset.

On the Linemod dataset (Table 5.2), DALSR-Pose outperforms AAE, a method that similarly uses latent space representation, by an overall margin of 6.4% using the ADD(I) metric ($AR_{ADD(I)}$), where the method also demonstrates higher pose accuracy across most objects. Compared to SSD6D, a method that does not explicitly access 3D models, DALSR-Pose is superior to SSD6D by a large margin of 29.9% in $AR_{ADD(I)}$. However, when compared to the indirect methods such as DPOD, Pix2Pose, CDPN, and PVNet (detailed in Section 2.3.2), DALSR-Pose does not perform as well. This is a common observation among the latent representation methods; for example, even with depth images and model point-based refinement, the AAE-ICP method still lacks pose accuracy, performing 18.3% and 14.7% worse in $AR_{ADD(I)}$ compared to CDPN and PVNet.

When tested on the more challenging BOP version of the Linemod-Occluded dataset (Table 5.3), DALSR-Pose is better than AAE, Multi-Path, and SSD6D, by considerable margins of 28.2%, 21.1%, and 28.9% on average using the three BOP metrics (AR_{score}), respectively. It even outperforms AAE-ICP by a large margin of 19.1% in AR_{score}, despite not using depth information and iterative post-refinement. Compared to the indirect methods, DALSR-Pose can even catch up with methods that explicitly use 3D models to build 2D-3D dense correspondence, such as DPOD and Pix2Pose, showing improvements of 25.9% and 6.5% in AR_{score}, respectively. However, it still falls short of the accuracy achieved by leading methods like CosyPose, GDR-Net, and SurfEmb.

Compared to AAE, the proposed continuous pose regression proves to be more effective. Rather than using a lookup table (LUT) technique, training an MLP regressor with the continuous 6D representation [118] allows for smoother 3D rotation estimation. The smooth interpolation of the 3D translation is done by localising the

object's 2D projective centre using another MLP regressor, and predicting the projective distance using a KNN regressor, which also outperforms AAE that directly uses the bounding box centre as the projective centre.

5.3.2 CVML-Pose

Table 5.2, 5.3 and 5.4, separately report the results of the proposed CVML-Pose method on the Linemod dataset, the BOP version of the Linemod-Occluded dataset, and the BOP version of the YCB-Video dataset, and compare with the state-of-the-art methods. Additional results on individual objects from Linemod-Occluded are presented in Table 5.5.

On the Linemod dataset, CVML-Pose demonstrates higher pose accuracy compared to the DALSR-Pose, AAE, and SSD6D methods, with respective margins of 4.1%, 10.5%, and 34.0% on average using the ADD(I) metric ($AR_{ADD(I)}$). Notably, these methods, like CVML-Pose, do not explicitly use the object's 3D model during training. CVML-Pose even marginally outperforms AAE-ICP on certain objects, such as the ape (0.7%) and driller (1.2%). It is worth noting that the performance of AAE-ICP can be limited, particularly in scenarios where depth measurement is not available under certain lighting conditions, e.g. excessively bright light or extremely low light. Furthermore, the ICP refinement employed in AAE-ICP cannot guarantee real-time processing due to its iterative process. Similar to DALSR-Pose, the CVML-Pose method falls short in comparison to the indirect methods, but achieves slightly better results than DPOD on several objects, including cam (2.5%), cat (3.3%), and eggbox (12.2%).

When tested on the two challenging BOP datasets, the proposed CVML-Pose method demonstrates significant advantages over the state-of-the-art methods. For example, on the Linemod-Occluded dataset, CVML-Pose outperforms all the latent representation methods, including AAE, AAE-ICP, and Multi-Path, by margins of 31.8%, 22.7%, and 24.7% on average using the three BOP metrics (AR_{score}). It also achieves

significantly better results than one of the deep learning-based direct methods, i.e. SSD6D (32.5% higher in AR_{score}). Compared to the indirect methods, CVML-Pose surpasses DPOD and Pix2Pose by certain margins of 29.5% and 10.1% in AR_{score}. In individual metrics, CVML-Pose can achieve comparable results to EPOS (5.8% gap in AR_{VSD}) and PVNet (4.0% gap in AR_{MSPD}).

On the YCB-Video dataset, CVML-Pose performs even better. For instance, the method still dominates in the category of latent representation methods. Compared to the indirect methods, CVML-Pose outperforms most of them, including DPOD, Pix2Pose, CDPN, and CDPNv2, by various margins of 32.1%, 8.6%, 8.6%, and 1.1% in AR_{score}, respectively. In individual metrics, CVML-Pose achieves competitive results compared to SurfEmb, with only a 4.8% gap in AR_{VSD} and a 2.8% gap in AR_{MSSD}. It also achieves comparable results to EPOS (an 8.5% gap in AR_{MSSD}). In particular, it is dramatically better than CDPN and CDPNv2, for which it was worse than either of them on the Linemod-Occluded dataset. This also happens on the AAE method, where it is significantly worse than the DPOD, Pix2Pose, CDPN, and CDPNv2 methods on the Linemod-Occluded data, e.g. by margins of 2.3%, 21.7%, 42.3%, and 47.8% in AR_{score}, but achieves comparable results to them on the YCB-Video data, e.g. 1.1% difference compared to CDPN.

The results across all datasets indicate that the regularised latent space and continuous pose regression approach of CVML-Pose is more effective than latent representation methods like AAE. Rather than using a denoising autoencoder (DAE), a modified variational autoencoder (VAE) can capture robust and regularised representations of objects. Additionally, the continuous pose regression, as opposed to LUT-based approaches, is more suitable as it avoids the significant errors caused by pose discretisation during inference. The continuous regression on the 2D projection of the object centre and the projective distance also reduces the effects caused by incorrect detection bounding boxes in heavily occluded scenarios.

In terms of individual objects, as shown in Table 5.5, the eggbox object exhibits

Object	ape	can	cat	driller	duck	eggbox	glue	holepuncher	Avg
AR_{VSD}	0.329	0.407	0.291	0.322	0.455	0.124	0.275	0.448	0.331
AR_{MSSD}	0.325	0.449	0.298	0.410	0.412	0.066	0.329	0.486	0.347
AR_{MSPD}	0.828	0.728	0.812	0.599	0.811	0.406	0.733	0.797	0.714
AR_{object}	0.503	0.528	0.467	0.444	0.559	0.199	0.446	0.577	0.464

Table 5.5: **Results of CVML-Pose on individual objects of the Linemod-Occluded dataset.**

much lower accuracy (26.5% worse in AR_{object}) compared to Avg, which might be associated with object symmetries, i.e. the pose ambiguity problem. To improve pose accuracy, especially for symmetrical objects, the proposed CVML-Pose method could be extended to estimate the distribution of potential poses through random sampling in the latent space, thereby better accommodating variances induced by object symmetries. Another observation is that among the three evaluation metrics, MSPD demonstrates considerably higher accuracy than the other two, e.g. 38.3% higher than VSD on average. As explained in [7], this might be because the MSPD metric does not account for alignment along the optical axis, which is significant when evaluating perspective images.

5.3.3 CVAM-Pose

Table 5.3 reports the results of the proposed CVAM-Pose method on the BOP version of the Linemod-Occluded benchmark dataset, which demonstrates that state-of-the-art performance for multi-object pose estimation can be achieved using a conditional generative model. In the category of latent representation methods, CVAM-Pose exhibits higher pose accuracy than AAE, AAE-ICP, and Multi-Path, outperforming them by substantial margins of 29.9%, 20.8%, and 22.8% on average using the three BOP metrics (AR_{score}). It also surpasses SSD6D, DPOD, and Pix2Pose by margins of 30.6%, 27.6% and 8.2% in AR_{score}, respectively, and achieves comparable results to EPOS and PVNet, with only a 4.1% and 4.5% gap in AR_{MSPD}.

Compared to Multi-Path, a multi-object method using latent space representation, the proposed learning of constrained representations achieves higher pose estimation

accuracy (22.8% better in AR_{score}). Unlike Multi-Path, which consists of multiple decoder networks and requires a significant amount of GPU memory for building multiple reconstructions, especially when dealing with numerous objects, the proposed layer-wise one-hot encoded CVAE network in CVAM-Pose efficiently learns constrained representations through a single-encoder-single-decoder network, which is more computationally efficient. Additionally, CVAM-Pose avoids the "per object per network" training strategy common in single-object prediction methods like AAE and AAE-ICP, enhancing the scalability of the method. This strategy is also applied to most of the methods (except CosyPose, EPOS, and SurfEmb) reported in Table 5.3, which demonstrate the advantages of scalability and efficiency of the proposed CVAM-Pose method. Moreover, similar to the DALSR-Pose and CVML-Pose methods, CVAM-Pose benefits from the use of continuous regression in the latent space over the typical LUT technique, which boosts the pose accuracy.

Comparing CVAM-Pose with its prototype method, CVML-Pose, a slight decrease in pose accuracy (1.9% worse in AR_{score}) is observed. While this reduction is relatively small, it suggests a need for further refinements to the method. Another finding is that, the Multi-Path method shows better performance (7.1% higher in AR_{score}) than its prototype method (AAE) on the Linemod-Occluded dataset, but performs significantly worse (15.7% lower in AR_{score}) on the YCB-Video dataset, as reported in Table 5.3 and 5.4. This difference may be attributed to the varying number of objects trained in the Multi-Path method. For example, the Multi-Path method can be scalable with 8 Linemod-Occluded objects, but might be sensitive to the increasing number of objects, i.e. 21 objects in the YCB-Video dataset. It indicates a potential scalability issue that merits future investigation for CVAM-Pose.

In future research, to achieve state-of-the-art pose accuracy, a certain number of systematic ablation tests will be implemented, including exploring different network architecture, dimensionality, and the capacity for handling a varying number of objects. Moreover, just like the CVML-Pose method, further investigations will delve into object characterisation and real-time processing capabilities.

5.3.4 Comparative Analysis of Deep Learning-based Methods

This section comprehensively analyses different categories of deep learning-based methods (see Section 2.3 for details of the methods) reported in Table 5.2, 5.3 and 5.4.

By analysing the results on the Linemod dataset (Table 5.2), latent representation methods typically exhibit lower performance than the deep learning-based indirect methods. For example, despite using additional depth information and model point-based refinement, AAE-ICP still falls short in accuracy ($AR_{ADD(I)}$) on several Linemod objects including ape (12.9%), driller (21.7%), and iron (7.2%) compared to DPOD. It is also not as good as other indirect methods such as Pix2Pose, CDPN, PVNet, and YOLOv5-6D. This shortfall may be attributed to the fact that indirect methods, although not directly using 3D model points during training, benefit from the 2D-3D correspondence mapping that provides sufficient 3D information, and facilitates promising pose estimation results. Another notable disadvantage of these typical latent representation methods like AAE, AAE-ICP, and Multi-Path is the discretisation of the pose, resulting in coarse pose estimation that often requires a post-refinement process. The issue of discretisation also affects the SSD6D method, leading to a notable reduction in pose accuracy.

Conversely, on the BOP version of the Linemod-Occluded dataset (Table 5.3), the latent representation methods demonstrate significantly better results, particularly in comparison to DPOD and Pix2Pose. The typical cases are the proposed DALSR-Pose and CVML-Pose methods, which rank better on the more challenging occlusion data, while they show only moderate ranks on the Linemod data (mild occlusion). A possible reason for this is that, approaches like DPOD and Pix2Pose, which predict pixel-wise dense correspondence, perform well in scenarios with relatively complete object visibility, i.e. objects in the Linemod dataset, as they can always obtain enough 2D-3D dense predictions. However, when objects are heavily occluded, i.e.

objects in the Linemod-Occluded dataset, their performance starts to decline due to an insufficient number of points available for assessing correspondence. On the contrary, the proposed methods benefit from the autoencoder architectures that can robustly handle object occlusion and truncation by reconstructing complete objects from occluded views. The continuous regression strategy in the latent space does not discretise the pose, which further enhances robustness compared to the LUT-based approaches such as AAE and Multi-Path.

By analysing the results on the BOP version of the YCB-Video dataset (Table 5.4), the latent representation methods, for example, AAE, can even achieve significantly better results than DPOD, and catch up with Pix2Pose and CDPN. This can be postulated that the provided YCB-Video training data are more suitable for latent representation methods than the Linemod training data. As illustrated in Section 3.2.4, the YCB-Video training set consists of PBR, non-PBR synthetic, and real images, which provides a larger volume of data than Linemod (see Figure 3.11 for distribution of the datasets). As the latent representation methods do not count on the object's 3D model or established model correspondence, training with more data may result in a more informative latent space. On the opposite, methods relying on 2D-3D correspondence might not benefit as much from a larger data volume, as their focus is more on model correspondence.

It is worth noting that on all three benchmark datasets, the overall performance scores of the proposed three methods still don't perform as well as the leading methods including CosyPose, GDR-Net, CDPN, CDPNv2, RNNPose, YOLOv5-6D, EPOS, SurfEmb, ZebraPose, and PVNet. These leading methods improve their accuracy through various techniques, for example, CosyPose directly utilises the object's 3D model with iterative refinement in training that increases the pose estimation accuracy. CDPN proposes a Scale-Invariant Feature Transform (SITE) algorithm [70] on local image patches, which significantly improves the results of the 3D translation estimation. The extended version CDPNv2 uses a better network architecture with domain randomisation technique [65] to improve robustness

to occlusion. EPOS addresses the many-to-one mapping problem associated with symmetric objects by estimating multiple potential predefined pixel-wise correspondences, rather than limiting to single correspondence. PVNet uses a voting-based Perspective-n-Point (PnP) algorithm with Random Sample Consensus (RANSAC), where the voting hypotheses warrant robust 2D keypoint localisation and naturally deal with occlusions. Although these methods are generally better than the proposed methods in pose accuracy, they all require 3D models for either setting up the 2D-3D correspondence or training with the model point-based loss function, while the unique contribution of the proposed methods lies in addressing the 6-DoF pose estimation problem without using object's 3D model, depth measurement, and post-refinement, offering a novel solution in scenarios where such data is unavailable.

5.3.5 Performance Against Occlusion

The experiments on occlusion are conducted using the proposed CVAM-Pose method on the BOP version of Linemod-Occluded. Performance against occlusion is illustrated in Figure 5.3, which shows box plots quantifying the distribution of the MSPD metric (AR_{MSPD}) across different visibility rates from 10% to 100%. It is evident that as visibility increases (occlusion decreases), the median of pose estimation accuracy improves and eventually achieves a value of 0.78. Even under heavy (20%-30% visibility) and mild (50%-60% visibility) occlusions, the median of our pose estimation accuracy achieves a value of 0.53 and 0.72, respectively, indicating its robustness against challenging occlusion scenarios.

To compare the pose regression strategy with the LUT-based approaches described in [65, 66], further experiments are conducted on the BOP version of Linemod-Occluded and the results are averaged on all objects. The LUT technique assigns the rotation and projective distance from the most similar instance to the test instance, and utilises the centre of the bounding box as the 2D projective centre. This approach may lead to inaccuracies, particularly with heavily occluded objects or imprecise bounding boxes. In our analysis, the results for 3D rotation are re-

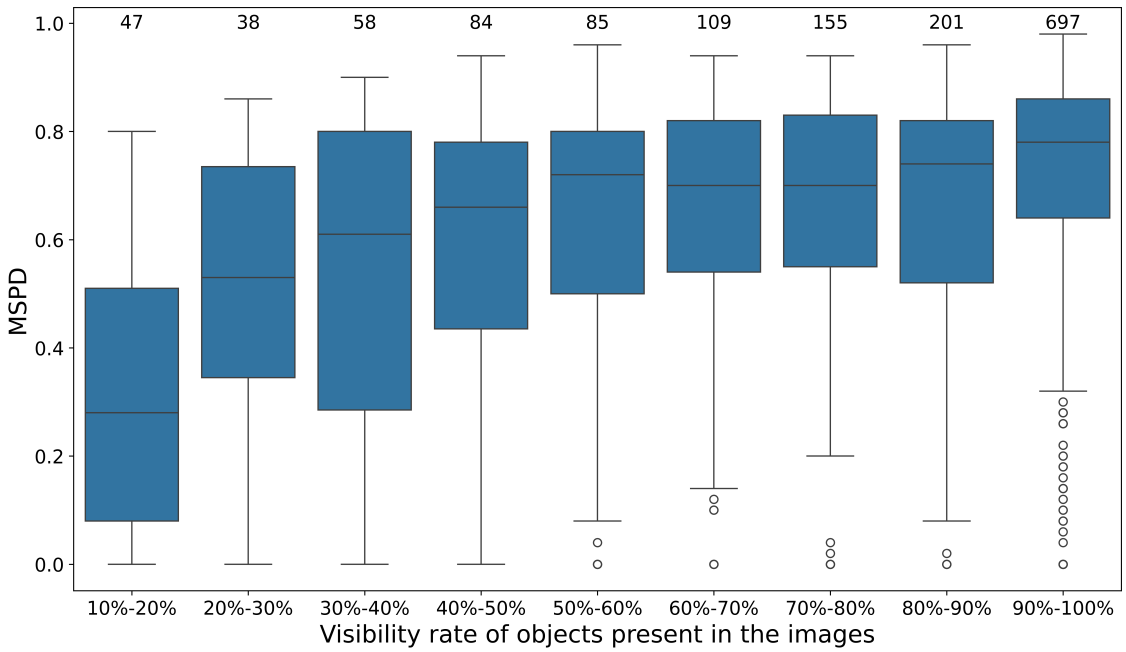


Figure 5.3: **Box plots of the MSPD metric as a function of the objects' visibility rate.** The number of data instances for each visibility rate is shown above each box. Please note that for better visualisation, the MSPD metric is calculated using thresholds ranging from 1 to 50 with a step of 1, instead of using the thresholds (from 5 to 50 with a step of 5) defined in the BOP Challenge.

ported using the MSPD metric, while results for projective centre and distance are evaluated using the mean absolute error (MAE) metric. The choice of MAE over MSPD is due to its parameter-free nature, which simplifies the interpretation of translational errors, as opposed to MSPD that depends on predefined thresholds as outlined in [7].

As shown in Table 5.6, the proposed continuous pose regression strategy demonstrates better results than using the LUT technique in estimating 3D rotation, 2D projective centre, and 2D projective distance, e.g. our method achieves smaller errors in distance measurement (improved by approximately 2% when computed in relation to the gt average object's distance in the test set). This can be attributed to the avoidance of the pose discretisation problem inherent in the LUT technique, particularly when the training data do not cover the entire SO(3). The performance of centre prediction is further illustrated in Figure 5.4, which presents box plots quantifying the distribution of errors ($\text{MAE}_{\text{pixel}}$). It is evident that the median error

in our method is consistently lower than that produced by the LUT technique across various visibility rates. The LUT method can also generate noticeable outlier errors in centre prediction, as high as 27 pixels.

Rotation	AR_{MSPD} ↑
LUT	0.666
Ours	**0.709**

Centre	MAE_{pixel} ↓
LUT	4.064
Ours	**2.913**

Distance	MAE_{mm} ↓
LUT	60.981
Ours	**43.278**

Table 5.6: **Comparison between LUT and our proposed pose regression strategy.** The experiments are implemented on the estimation of 3D rotation, 2D projective centre, and 2D projective distance.

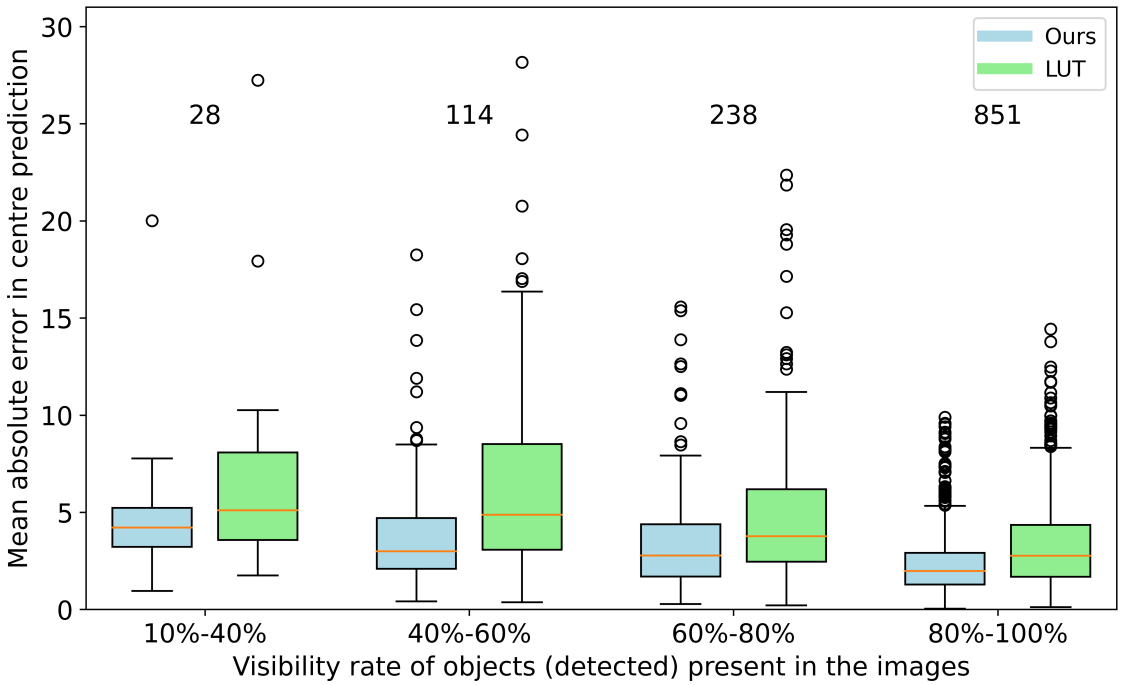


Figure 5.4: **Box plots of the MAE metric as a function of the objects' visibility rates.** The number of data instances for each rate is shown above each pair of boxes.

Figure 5.5 visualises pose estimation results on two randomly selected images from the Linemod-Occluded dataset, with poses estimated using CVAM-Pose. The target objects, including ape, cat, driller, duck, eggbox, glue, holepuncher, and iron, are rendered based on the estimated poses and reprojected onto the original test images. Correct estimations are represented by aligned reprojection masks, e.g. the cat object in the first image, while misaligned masks indicate incorrect estimations, e.g. the eggbox object in the first image.

Figure 5.5: **Example visualisation of the estimated poses using CVAM-Pose.** The rendering process uses the Pyrender software [9].

5.3.6 Object Detection

Table 5.7 compares the performance of the CVML-Pose method using the gt bounding box with its performance using the detected bounding box from the pretrained Mask-RCNN detector. It can be noted that the method exhibits significant improvements with the gt bounding box on both the Linemod dataset (11.2% better in $AR_{ADD(I)}$) and the BOP version of the Linemod-Occluded dataset (7.0% better in $AR_{ADD(I)}$).

	Linemod		**BOP Linemod-Occluded**	
Objects	gt	Mask-RCNN	gt	Mask-RCNN
ape	33.71	25.07	13.74	8.76
benchvise	69.06	48.50	-	-
cam	45.88	26.73	-	-
can	58.76	48.57	31.66	18.08
cat	41.72	35.70	14.81	13.08
driller	62.14	46.02	36.00	22.16
duck	34.74	24.81	22.53	14.29
eggbox	95.02	85.73	36.67	29.13
glue	57.63	55.28	41.55	36.51
holepuncher	34.44	23.71	27.00	25.76
iron	56.08	52.20	-	-
lamp	69.58	56.91	-	-
phone	46.69	31.15	-	-
$AR_{ADD(I)}$	54.27	43.11	28.00	20.97

Table 5.7: **Comparison between pose estimation results of using the gt and the Mask-RCNN bounding box.** The CVML-Pose method is chosen as the baseline model. The number of test instances in each dataset can be found in Table 5.1.

To investigate whether a more effective detection model can yield similar improvements in pose estimation, the YOLOX detector [216], pretrained by [71] and awarded as the best 2D detection method at the BOP Challenge 2022 [26], is also employed. Following the same process of using the Mask-RCNN detector, the pretrained YOLOX detector is used to identify and crop objects of interest from the test images, subsequently providing detection bounding boxes for the estimation of 3D translation $\mathbf{T}$. The results of 2D object detection and pose estimation using the YOLOX detector on the BOP version of the Linemod-Occluded and YCB-Video

datasets are presented in Table 5.8. It is clear to see that the YOLOX detector consistently outperforms the Mask-RCNN detector in both object detection and pose estimation tasks across both datasets. For example, on the Linemod-Occluded dataset, YOLOX surpasses Mask-RCNN by 12.9% and 4.7% when evaluated using AP and AR_{score}, respectively. On the YCB-Video dataset, YOLOX performs 10.7% and 4.1% better than Mask-RCNN when training and testing with different types of data. This gap also increases to 19.2% and 8.9% when using PBR data. This highlights the importance of accurate object detection in pose estimation performance, and further work should look to improve this aspect of the pipeline.

Test data	Linemod-Occluded				YCB-Video			
Detector	Mask-RCNN		YOLOX		Mask-RCNN		YOLOX	
Training data for detectors	PBR	PBR+real	PBR	PBR+real	PBR	PBR+synt+real	PBR	PBR+real
AP for object detection	0.566	0.566	0.695	0.695	0.594	0.745	0.786	0.852
AR_{score} for pose estimation	0.464	0.464	0.511	0.511	0.473	0.543	0.562	0.584

Table 5.8: **Effects of using different object detectors and training data.** Both object detection and pose estimation methods are evaluated on the BOP version [7] of Linemod-Occluded [3, 4] and YCB-Video [5, 6] data. For object detection, the results are evaluated with the metrics of the COCO object detection challenge [218], and the final performance score AP calculates the average over 10 different Intersection over Union (IoU) values (IoU $= 0.5, 0.55, 0.6, ..., 0.95$). For pose estimation, the results are evaluated with the three main metrics of the BOP Challenge, which are VSD, MSSD, and MSPD, and the final performance is calculated by $AR_{score} = (AR_{VSD} + AR_{MSSD} + AR_{MSPD})/3$. For more comprehensive comparison and results, please refer to [26] and the challenge leaderboard: https://bop.felk.cvut.cz/leaderboards/.

Also, as previously discussed in Section 3.2.4, Table 3.1 and 3.2 exemplify the importance of training data in pose estimation. Although the PBR synthetic images provide good enough generalisation for many methods to perform better on the BOP version of the Linemod-Occluded test data, the standard synthetic images (without the PBR technique) and real images of the YCB-Video dataset are also beneficial for training, as they can provide good enough generalisation ability as well as the PBR images. This is also reflected in the object detection task as shown in Table 5.8. For instance, on the Linemod-Occluded dataset, regardless of the object detector used, the results of using PBR synthetic images make no difference from the results of using a combination of PBR and real images, in both object detection and pose

estimation tasks. In particular, the baseline pose estimation model (CVML-Pose) does not benefit from the detection models trained on the PBR and real data. Conversely, when testing on the YCB-Video dataset, the combination of multiple types of data significantly improves the results on both object detection (15.1% and 6.6% higher in AP) and pose estimation (7% and 2.2% higher in AR_{score}), which matches the results demonstrated in Table 3.2. This implies that the choice of training data should be tailored to the specific dataset. For the Linemod-Occluded dataset, PBR images alone may be sufficient for computational efficiency. For the YCB-Video dataset, a diverse mix of image types, including PBR, non-PBR synthetic, and real, are recommended for methods to achieve higher pose accuracy.

5.4 Summary

Extensive experiments with the proposed pose estimation methods have shown promising results on challenging, publicly available datasets. On the Linemod test data, the DALSR-Pose and CVML-Pose methods rank moderately compared to the state-of-the-art methods, but outperform the AAE method by margins of 6.4% and 10.5% in $AR_{ADD(I)}$, respectively.

In more complex scenarios, such as those involving object occlusion and cluttered backgrounds on the BOP test data, the proposed methods demonstrate significant advantages. Specifically, on the BOP version of the Linemod-Occluded dataset, all three proposed methods surpass AAE by considerable margins of 28.2%, 31.8%, and 29.9% in AR_{score}. Even compared to AAE-ICP, which uses time-consuming ICP refinement, their pose accuracies are notably higher by margins of 19.1%, 22.7%, and 20.8%. Although the proposed methods do not match the accuracy of the leading methods like CosyPose, CDPN, CDPNv2, EPOS, and PVNet, which either use 3D models in training or acquire prior information from the models, they still achieve more promising results than DPOD and Pix2Pose that rely on model correspondence information, e.g. 25.9%, 29.5%, and 27.6% better than DPOD in AR_{score}.

On the BOP version of the YCB-Video dataset, the proposed CVML-Pose method outperforms a variety of state-of-the-art methods, including AAE (9.7%), AAE-ICP (3.8%), Multi-Path (25.8%), DPOD (32.1%), Pix2Pose (8.6%), CDPN (8.6%), and CDPNv2 (1.1%), in AR_{score}, although still having lower accuracy than Cosy-Pose (27.8%), GDR-Net (28.2%), EPOS (15.3%), SurfEmb (10.4%), and ZebraPose (28.5%), which do not use latent space representations.

The datasets used in the thesis consist of colour images, and therefore the developed methods operate on colour information. However, for the results reported on the Linemod and Linemod-Occluded datasets, which consist of texture-less objects, colour information is not critical; instead, the shape is the key feature. Therefore, it is expected that the proposed methods should perform well if the input images are grayscale. For the YCB-Video dataset, texture is an important feature, but it is still anticipated that the grayscale texture may be sufficient. Though, in all cases, the methods would require retraining on grayscale images since the input tensor would have a different dimensionality compared to colour images.

The results also highlight the importance of object detection quality in pose estimation accuracy. For example, using the gt bounding box on the Linemod and the BOP version of the Linemod-Occluded datasets significantly improves the pose estimation accuracy of the CVML-Pose method by a margin of 11.2% and 7.0% in $AR_{ADD(I)}$, compared to when it is using the bounding box from the Mask-RCNN detector. Similarly, on the BOP version of the Linemod-Occluded and YCB-Video datasets, using a better YOLOX detection model results in better performance on both tasks, i.e. the YOLOX detector outperforms the Mask-RCNN detector by certain margins of 12.9% and 19.2% in AP in object detection task, 4.7% and 8.9% in AR_{score} in pose estimation task.

Furthermore, the selection of training data plays a crucial role in both object detection and pose estimation model performance. On the BOP version of the Linemod-Occluded dataset, using only PBR images is sufficient for achieving generalisable

results in both object detection and pose estimation tasks, compared to when the models are trained with PBR and real images. However, on the BOP version of the YCB-Video dataset, a combination of PBR, non-PBR synthetic, and real images significantly improves results over using only PBR images in both tasks, by margins of 15.1% and 6.6% in AP for object detection, and 7% and 2.2% in AR_{score} for pose estimation. This is also identical to the remarks made in Section 3.2.4.

Chapter 6

Impacts of Pose Refinements

6.1 Introduction

In state-of-the-art 6-DoF pose estimation methods, pose accuracy is often improved through various refinement techniques, which typically involve training and inference with object 3D models and depth measurements. One common refinement approach requires model point-based loss functions, such as the Shape-Match loss proposed by [5] (refer to Eq. 2.1). During training, the loss functions keep minimising the distance between corresponding 3D model points transformed at the estimated and ground truth (gt) poses. Another similar refinement technique [75, 111] involves training networks to iteratively match the image rendered from a 3D model at its estimated pose with the observed input image. These techniques are collectively named learning-based refinement as they employ CNNs to iteratively update the pose or its representation based on the 3D model points.

Except for learning-based refinement, post-refinement techniques are often employed during inference in many methods. The iterative closest point (ICP) algorithm [146] is one of the most commonly used techniques, which estimates an optimal rigid transformation that aligns two sets of point clouds by iteratively minimising the distance between corresponding points in the two sets. The initial point clouds can

be retrieved from object 3D models and transformed at the estimated pose, and the target point clouds are typically generated from the depth data. While ICP is a benchmark in pose refinement from point clouds and is used in several state-of-the-art methods [5, 65, 66, 68, 96, 119], it has limitations, such as a tendency to get stuck in local minima, especially when the initial point cloud is close enough to the target point cloud, leading to premature convergence. In situations where two point clouds have very little overlap or are substantially distant from each other, the algorithm frequently results in numerous misalignments. Some efficient ICP variants [147, 219, 220] have been proposed to mitigate such issues.

In the three proposed methods (illustrated in Chapter 4), although object 3D models and depth information are not required to achieve state-of-the-art performance, two optional refinement processes are introduced here, to examine their impacts on pose estimation. These include training with 3D model points (Section 6.2) and inference with ICP refinement (Section 6.3), with detailed discussions and remarks in Section 6.4. The proposed CVML-Pose method (detailed in Section 4.3) is chosen as the baseline method, and evaluated on the Linemod dataset [1, 2] using the gt bounding box. Since the estimation of rotation and translation are disentangled in the method, Section 6.2 focuses solely on the refinement of 3D rotation. The ICP refinement of the complete pose is later introduced in Section 6.3 as this refinement process is applied after the initial estimation of the complete pose by the proposed method. The evaluation procedure follows the procedure described in Chapter 5, with the performance score $\text{AR}_{\text{ADD(I)}}$ calculated based on the ADD(I) metric and averaged across all Linemod objects. Additionally, it is important to stress that the proposed methods do not require pose refinement. Training with 3D model points does not improve the pose estimation accuracy of the proposed CVML-Pose method. While ICP does improve the pose estimation accuracy by 36.3% higher in $\text{AR}_{\text{ADD(I)}}$ on the Linemod dataset, it is not considered real-time as defined on pages 21 and 22.

6.2 Training with 3D Model Points

In the CVML-Pose method, the 3D rotation is estimated using a multilayer perceptron (rotation MLP) with the continuous 6D representation [118]. The L1 loss (mean absolute error) is measured between the estimated rotation $\hat{\mathbf{R}} \in \mathrm{SO}(3)$ and the gt rotation $\bar{\mathbf{R}} \in \mathrm{SO}(3)$. To refine the estimated rotation with object 3D model points, the point-based loss function proposed by the EfficientPose method [116] (Eq. 2.2) is modified to the following Eq. 6.1.

$$\begin{aligned} loss_{asym} &= \frac{1}{m} \sum_{x \in M} ||\mathrm{Rot}(\tilde{\mathbf{r}}, x) - \mathrm{Rot}(\mathbf{r}, x)||_2 \\ loss_{sym} &= \frac{1}{m} \sum_{x_1 \in M} \min_{x_2 \in M} ||\mathrm{Rot}(\tilde{\mathbf{r}}, x_1) - \mathrm{Rot}(\mathbf{r}, x_2)||_2 \end{aligned} \tag{6.1}$$

where $\mathrm{Rot}(\tilde{\mathbf{r}}, x)$ and $\mathrm{Rot}(\mathbf{r}, x)$ are the model points transformed from the estimated axis-angle representation $\tilde{\mathbf{r}}$ and the gt axis-angle representation $\mathbf{r}$, using the Rodrigues' rotation formula [123, 124, 125]. The axis-angle representation can be either obtained directly from the regression output or converted from the continuous 6D rotation representation $R_{6D} \in \mathbb{R}^6$.

Given that the continuous 6D representation cannot be directly applied as a rigid transformation on object 3D model points, two approaches are implemented. The first approach converts the obtained 6D rotation representation $R_{6D} \in \mathbb{R}^6$ to other representations, such as axis-angle representation, which is directly applicable to the 3D model points. Alternatively, following the EfficientPose method, the MLP can be trained to regress the axis-angle representation directly, using the Rodrigues' rotation formula for applying the rotation to the model points.

For evaluation, the performance score $\mathrm{AR}_{\mathrm{ADD(I)}}$ is calculated based solely on the estimated rotation, to reduce the impact of any inaccuracies in translation estimation. The results, as shown in Table 6.1, indicate that the online refinement with object 3D models improves pose estimation accuracy (2.9% higher in $\mathrm{AR}_{\mathrm{ADD(I)}}$)

Methods	Rotation representation	$AR_{ADD(I)}$
CVML-Pose without 3D models	axis-angle	94.39
CVML-Pose with 3D models	axis-angle	97.33
CVML-Pose without 3D models	continuous 6D representation	**97.97**
CVML-Pose with 3D models	continuous 6D representation	97.87

Table 6.1: **Results of training with 3D model points based on the CVML-Pose method.** Each row represents the CVML-Pose method with a specific rotation representation, and training with or without object 3D models.

when regressing the discontinuous axis-angle representation. However, the best results are achieved without using the model point-based refinement. Notably, when the rotation MLP is trained to regress the continuous 6D representation with 3D model points, the results are nearly identical (0.1% difference in $AR_{ADD(I)}$) to those obtained without using the model points, which shows the effectiveness of the continuous rotation regression implemented in the method.

6.3 Test with ICP refinement

In addition to learning-based refinement, another common practice in pose estimation is to refine with images obtained from depth sensors. In the CVML-Pose method, this can be achieved through the ICP registration with depth images and object 3D model points. Unlike the previous section, the ICP algorithm here refines the complete rigid transformation $\mathbf{H}$ that includes both 3D rotation $\mathbf{R}$ and 3D translation $\mathbf{T}$. Following a similar approach as in the AAE method [65], the point-to-plane ICP variant [219, 221] is used for pose refinement. Typically, the source point cloud is derived from the rendered depth image based on the estimated rigid transformation $\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix}$, the object's 3D model, and the camera intrinsic matrix. The target point cloud, meanwhile, is obtained from the depth image of the test scene. The adopted ICP algorithm first finds the closest point $\mathbf{q}$ in the target cloud for each point $\mathbf{p}$ in the source cloud, and then iteratively minimises an energy function $E(\mathbf{H})$ to get the best possible transformation $\mathbf{H}$. The energy

function $E(\mathbf{H})$ is defined as follows:

$$E(\mathbf{H}) = \sum_{(\mathbf{p},\mathbf{q})\in C} ((\mathbf{p} - \mathbf{H}\mathbf{q}) \cdot \mathbf{n}_\mathbf{p})^2 \tag{6.2}$$

where C is the found correspondence set, $\mathbf{n}_\mathbf{p}$ is the computed surface normal of each $\mathbf{p}$.

In CVML-Pose, the 3D translation $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$ is disentangled into the estimation of projective centre $P_c = (x_c, y_c)^T \in \mathbb{R}^2$ and projective distance $T_z \in \mathbb{R}$. This regression approach, similar to DALSR-Pose (detailed in Section 4.2.3), has shown promising results for localising P_c. However, the estimation of T_z exhibits notable errors in all regression models (as shown in Table 4.2b). Since T_x and T_y are calculated from T_z, the errors can accumulate on the complete translation vector $\mathbf{T} = \begin{pmatrix} T_x & T_y & T_z \end{pmatrix}^T \in \mathbb{R}^3$. If the ICP refinement can improve the accuracy of T_z with the help of depth measurement, the errors in 3D translation can be significantly reduced.

Method	$\text{AR}_{\text{ADD(I)}}$	MAE_{mm}	$\text{Time}_{\text{(ms)}}$
CVML-Pose without ICP	54.27	21.94	**4.76**
CVML-Pose with ICP	**90.53**	**6.58**	357.68

Table 6.2: **Results of testing with ICP refinement based on the CVML-Pose method.** Each row represents the CVML-Pose method testing with or without ICP refinement, and the results are calculated from two different metrics. The $\text{AR}_{\text{ADD(I)}}$ is calculated on the complete 6-DoF pose, while the MAE_{mm} is calculated only on the distance T_z. The $\text{Time}_{\text{(ms)}}$ indicates the average inference time per object per image.

There are two different metrics used for evaluation. The first performance score $\text{AR}_{\text{ADD(I)}}$ is calculated based on the complete estimated pose, and the second score MAE_{mm} focuses only on the projective distance T_z. Additionally, a measurement of the pose estimation speed, denoted as $\text{Time}_{\text{(ms)}}$, measures the average inference time for a single object from a single image, which consists of the time taken for the encoder, MLP, KNN, and ICP processing (if used). As presented in Table 6.2, the original CVML-Pose method suffers from the inaccurate estimation of T_z, resulting

in a certain number of misalignments when measuring the model distance using the ADD(I) metric. With object 3D model points and depth data, the accuracy of T_z improves by approximately 1% when computed in relation to the gt average object's distance in the test set. The pose estimation accuracy also improves significantly (36.3% higher in $\text{AR}_{\text{ADD(I)}}$). However, it is important to note that the ICP refinement leads to a considerable reduction in inference speed, making it 352.9 milliseconds slower compared to the method without the refinement.

6.4 Summary

The experimental results in Table 6.1 suggest that online refinement with object 3D models is not necessary for the CVML-Pose method, considering accuracy, efficiency, and cost. In terms of accuracy, training with the continuous 6D rotation representation already achieves satisfactory precision. For the sake of computational efficiency, training with 3D models demands more GPU memory and extends training time. Additionally, it would be difficult to build an accurate 3D model for every possible object of interest in real-life applications, given the cost associated with 3D scanning and the need for further refinements. Consequently, it is expected that the methods proposed in this thesis should achieve state-of-the-art performance without explicitly accessing object 3D models.

From the experimental results demonstrated in Table 6.2, it can be concluded that ICP refinement can solve the problem encountered in RGB-based pose estimation methods, particularly in improving 3D translation accuracy. However, for efficiency, the ICP refinement would be on the losing side due to its iterative processing. The average processing time for CVML-Pose is approximately 5 milliseconds per object, while ICP refinement takes around 350 milliseconds, extending the time by a factor of 70. The processing time of ICP can be even longer in extreme cases, such as convergence to local minima or extensive closest point search. Due to the nature of iterative processing, it would be challenging to facilitate real-time applications as the

processing time is unpredictable. Taking into account both manpower and financial costs, the use of 3D models and depth sensors would increase them, especially for the construction of 3D models, which may require high-performance instruments at expensive prices. While depth cameras are not extremely expensive, they can be affected by various noises in the context of random variations of brightness or colour and material of objects [34]. They can also exhibit motion blur when there's movement of the object or the camera, which leads to either overestimating or underestimating the depth. Therefore, given these considerations, particularly the impact on speed and cost, the ICP refinement technique is not a favourable option for the proposed pose estimation methods.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis addresses one of the key challenges in computer vision: finding an object's 6-DoF pose in real time without requiring the object's 3D model. The proposed methods demonstrate that state-of-the-art performance can be achieved using only data from a monoscopic camera, eliminating the need for 3D models, depth measurements, or iterative post-refinement. The main contribution of the reported research is the proposed use of autoencoder networks and the construction of regularised latent space representations from 2D images. Subsequently, the learnt latent space representations are interpolated to continuous pose representations, and processed with supervised learning methods for fast and accurate pose estimation. Different configurations of the methods were systematically evaluated using extensive ablation tests, leading to a favourable selection of parameters.

The proposed methods achieve promising pose estimation results compared to state-of-the-art methods, without the use of 3D models and depth measurements, which are extensively used in other approaches. The proposed methods outperform other existing methods that utilise latent space representation on various challenging textured, texture-less, cluttered, and occluded datasets. For instance, the proposed

DALSR-Pose method surpasses AAE, AAE-ICP, and Multi-Path in the AR_{score} benchmarking score, with considerable margins of 28.2%, 19.1%, and 21.1%, respectively. This superior performance is largely attributed to the smooth interpretation of the learnt latent space representations through continuous pose regression. Similarly, the proposed CVML-Pose method demonstrates remarkable results, outperforming the same three methods by significant margins of 31.8%, 22.7%, and 24.7%, respectively. This can be mainly attributed to the construction of robust latent space representation through the use of regularised learning. Owing to the implicit learning process, the autoencoder architecture can handle object occlusions and cluttered scenes, and does not need prior knowledge of the object's 3D model or post-refinement, e.g. using ICP. Additionally, the latent space visualisation shows potential for the CVML-Pose method to be extended to a multi-purpose object characterisation. Furthermore, an online video demonstration[1] also exhibits that the proposed CVML-Pose method copes well with low-resolution images captured with an inexpensive webcam. The absence of iterative post-refinement ensures predictable processing times, for example, it takes only approximately 5 milliseconds to estimate the pose of a single object using the CVML-Pose method, facilitating real-time implementation, which is crucial for practical applications. The CVAM-Pose method, in particular, demonstrates the scalability of a single latent space to be expanded to multi-object representations without compromising pose accuracy (1.9% worse than the proposed CVML-Pose in AR_{score}). It is expected that the latent representation could be further leveraged to infer other attributes of the object such as material, surface finish, and deformations, which are also important for specific tasks such as object grasping.

7.2 Novelty

This thesis addresses a classical yet still not fully solved problem in computer vision, i.e. estimation of the 6-DoF pose of rigid objects. The focus of this work

[1] video available: https://ieeexplore.ieee.org/document/10040668.

is on 6-DoF pose estimation in real time from a single colour image, without access to 3D object models, depth sensors, or iterative post-refinement techniques. Building upon existing results, the novel contributions of this thesis address the limitations of current methods and effectively handle several challenging scenarios involving object occlusion, truncation, and clutter. The thesis uniquely bridges the gap between deep learning-based latent space representation methods and classical machine learning-based regression algorithms, opening up novel prospects for integrated pose estimation. The novelties of each method are listed as follows:

- DALSR-Pose incorporates deep learning-based latent space representation with regression algorithms to interpolate an object's 6-DoF pose. The main contribution is the proposed continuous pose regression from the learnt latent space representation, using supervised learning algorithms. On the Linemod [1, 2] and the BOP version [7] of the Linemod-Occluded [3, 4] benchmark datasets, DALSR-Pose outperforms AAE (a latent representation method) by a margin of 6.4% using the $AR_{ADD(I)}$ metric, and 28.2% using the AR_{score} benchmarking score. It also achieves moderate performance compared to state-of-the-art methods that explicitly access object 3D model information. To the best of my knowledge, DALSR-Pose is the first method to unify latent space representation and continuous pose regression for 6-DoF pose estimation.

- The CVML-Pose method innovatively uses a variational autoencoder, i.e. a generative model, to address the problem of 6-DoF pose estimation. The primary contribution is in the proposed use of a regularised latent space representation. The regularised representation is subsequently interpolated to the continuous pose representation using regression-based algorithms, and clustered in both category and topology using t-SNE. Comprehensive ablation tests and systematic evaluation procedures, demonstrate the superiority of the CVML-Pose method over existing state-of-the-art methods using latent space representation on the challenging benchmark datasets. For example, on the BOP version of the Linemod-Occluded dataset, CVML-Pose outperforms

AAE by a large margin of 31.8% in AR_{score}. On the BOP version of the YCB-Video dataset [5, 6], although CVML-Pose cannot perform as well as the most state-of-the-art methods, e.g. 28.2% and 28.5% worse than [71, 76] in AR_{score}, it outperforms not only latent representation methods [65, 66] but also some methods using 3D models [67, 68, 70] by different margins, e.g. 32.1% better than [67] in AR_{score}. The video demonstration[2] also shows the ability to seamlessly estimate object 6-DoF pose without any iterative post-refinement in real time, and handle both object occlusions and cluttered scenes on low-resolution images from a webcam. This work has been published in [202]. To the best of my knowledge, it is the first method that combines the latent space of a generative model with continuous pose regression in the field of object 6-DoF pose estimation.

- The CVAM-Pose method extends the CVML-Pose method from single-object to multi-object pose estimation. The primary contribution is the use of regularised and constrained representations in a conditional variational autoencoder's latent space. The method introduces a novel layer-wise one-hot encoding technique to embed conditional labels into every convolutional layer/block of the autoencoder network, learning high-level representations. The learnt multi-object representations are then interpolated to the continuous pose representations through the regression-based algorithms. Extensive evaluations on the challenging BOP version of the Linemod-Occluded dataset demonstrate the scalability and efficiency of CVAM-Pose, which nearly matches the pose accuracy of CVML-Pose (a narrow margin of 1.9% in AR_{score}), and achieves comparable results to the state-of-the-art methods using 3D models. To the best of my knowledge, CVAM-Pose is the first method that combines conditioned latent space representation with continuous regression for multi-object pose estimation.

[2] video available: https://ieeexplore.ieee.org/document/10040668.

7.3 Limitations

This thesis introduces three novel 6-DoF pose estimation methods, each offering unique contributions to the field. Despite their promising results in accurately and efficiently estimating object poses under complex environments, it is crucial to acknowledge certain limitations that may affect the applicability and reliability of the methods in real-world scenarios. While these limitations do not diminish the novelties of the methods, they are important considerations for future development and practical implementation. The potential limitations are listed as follows:

- **Computational Efficiency and Scalability:** The "per object per network" training strategy of the single-object pose estimation methods (DALSR-Pose and CVML-Pose) requires substantial computational resources. Although the proposed CVAM-Pose method has partially addressed this issue, the scalability of the method still necessitates further analysis.

- **Pose Ambiguity Problem:** Unlike methods specifically designed for symmetrical objects, the proposed autoencoder-based methods effectively handle object symmetry in the latent space, even though they were not explicitly designed for this purpose. This inherent capability to handle symmetry, despite not being a primary focus, enhances the robustness of the methods. To further improve pose accuracy, particularly for symmetrical objects, the methods can be extended to estimate the distribution of potential poses through random sampling in the latent space, thereby accommodating variances induced by object symmetries.

- **Non-Rigid Object Handling:** The proposed methods are designed under the assumption that the objects of interest are rigid. However, real-world scenarios often involve articulated or deformable objects whose pose estimation is inherently more complex due to variable shapes. The current proposed methods do not accommodate these variations, potentially leading to an inaccurate

estimation of the pose.

- **Accuracy of Pose Estimation:** While the CVML-Pose method stands out as the most accurate among the proposed methods, it still exhibits a certain margin compared to the leading state-of-the-art methods that utilise object 3D models and depth measurements. For example, it is 10.4% worse than [73] on the YCB-Video dataset using AR_{score}. This gap highlights potential improvements in the development of more precise pose estimation algorithms.

7.4 Future Work

This thesis presents a novel challenge in object 6-DoF pose estimation, i.e. estimating the pose from a single colour image, while eliminating the need for 3D models, depth sensors, or iterative post-refinement techniques. To solve this challenging problem, three novel methods are proposed, which achieve comparable results to the state-of-the-art methods that extensively depend on such resources. These methods not only address the inherent challenges in pose estimation, such as occlusion and clutter, but also open up new perspectives for further exploration, for example, object characterisation. As research never ends, the field of object 6-DoF pose estimation continues to evolve with numerous unresolved challenges, which include the estimation for deformable objects [222, 223, 224] and unseen objects [225, 226, 227], as well as dealing with multiple objects [228, 229, 230], in a few-shot [231, 232, 233, 234] or zero-shot [235, 236] manner. Other relevant fields like object recognition [237, 238], detection [26], and segmentation [239, 240, 241] also contribute to the development of 6-DoF pose estimation.

In terms of the proposed Auto-Pose framework, three main directions are envisaged for an expanded version in the future. First, a more comprehensive object characterisation including shape, shape deformations, material, pose stability, and surface finish will be investigated. The sensitivity of the method to various practical limitations of an image acquisition system, including nonlinear lens distortion and

motion blur, will also be evaluated. Second, the results reported here are based on a synthetic training dataset. The results reported in the literature for the state-of-the-art methods suggested that high-quality synthetic training data is a good surrogate for real training images. This conjecture will be extensively tested for the proposed methods. Finally, it has been shown that the inaccuracies in object detection can significantly impact performance, and that further work on object detection would be beneficial. To this end, the possibility of an end-to-end pose estimation algorithm, including object detection, latent space and pose estimation in a single learning process, will be explored.

In line with these future research directions, a recent research project from the UCLan Undergraduate Research Internship Programme (UURIP)[3], started to build a data collection system for deformable objects' 6-DoF pose estimation. The system consists of several hardware components, including a turntable, a light box covered in ArUco markers obtained from ARToolKitPlus [242] and an RGB-D sensor (Microsoft Kinect v1) attached to a tripod. Future development of the system will also involve more advanced sensors like Intel RealSense [78]. Additionally, a high-resolution 3D scanner (Artec Eva) and its accompanying software are used for the creation of object 3D models. This is a challenging problem because of object deformation and articulation. Although the system is still under development, it attempts to solve the most advanced model-free 6-DoF pose estimation problem, closely following the direction of future research.

[3]The programme started on 01/06/2023 and ended on 28/07/2023.

Appendix A

Supplementary Figures

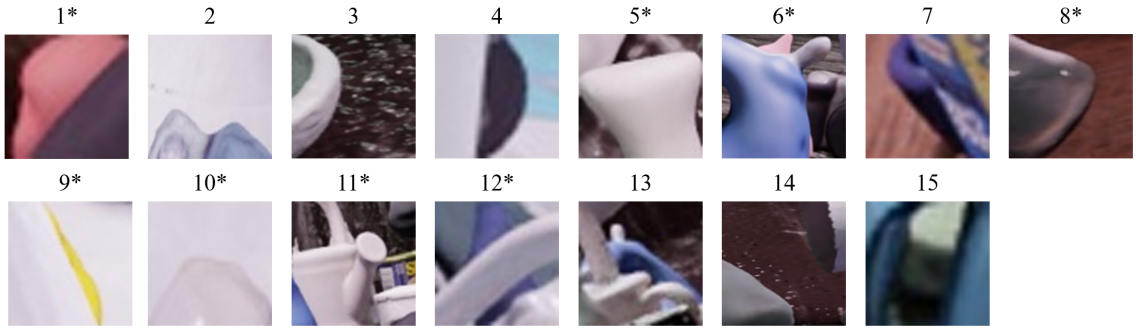


Figure A.1: **Example bad cases for Linemod objects [1, 2].** Objects with the $*$ symbol refer to those also contained in the Linemod-Occluded dataset [3, 4].

Appendix B

Test with IMU

We additionally use an Inertial Measurement Unit (IMU) sensor as a reference to calculate errors in pose estimation. Figure B.1 shows an example of how these errors are calculated using the IMU sensor[1]. The target Linemod cat object is attached to the sensor, and they are moved together. The object's pose is estimated using the proposed CVML-Pose method. The coordinates of the object and the sensor are aligned, and their rotations are represented using the axis-angle representation (see Section 3.3.3 for details of the representation). For simplicity, the rotations for both sensors are set to 0 for the first frame, and relative rotations are estimated for the subsequent frames. The error between CVML-Pose and IMU for images shown in Figure B.1 is calculated as 2.85° using the RE metric (see Section 3.5.1 for details of the metric). A total of 21 images were captured, with the error distribution shown in Figure B.2.

[1]Product name: FDISystems DETA-10

Figure B.1: **Example of using the IMU sensor to calculate errors.**

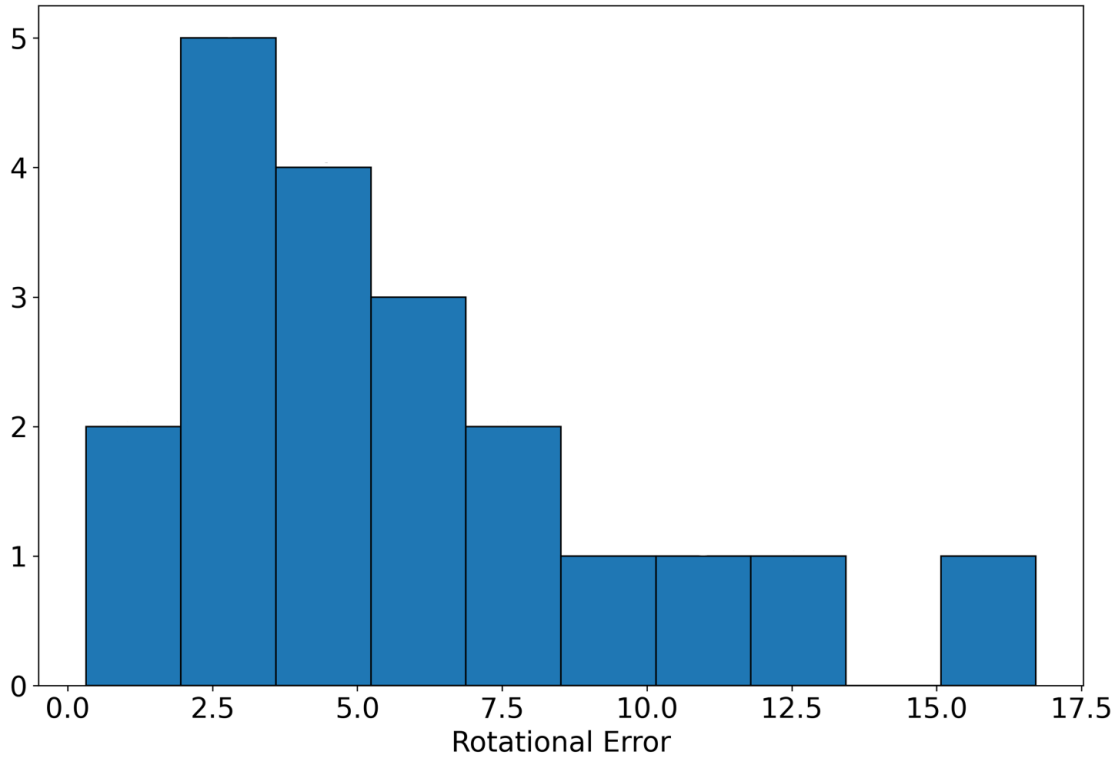


Figure B.2: **Distribution of error using the IMU sensor.**

Appendix C

Mathematical Derivation

C.1 Derivation of the Evidence Lower Bound in Variational Autoencoder

The objective of a variational autoencoder (VAE) is to maximise the likelihood $\log p_\theta(x)$ of the input data x. Given the latent space z, it learns a joint probability $\log p_\theta(x, z)$:

$$\log p_\theta(x) = \log \int p_\theta(x, z) dz \tag{C.1}$$

The true posterior $p_\theta(z|x)$ is typically intractable, so a tractable variational approximation $q_\phi(z|x)$ is introduced:

$$\log p_\theta(x) = \log \int p_\theta(x, z) \frac{q_\phi(z|x)}{q_\phi(z|x)} dz \tag{C.2}$$

The integral can be expressed as an expectation under the variational approximation:

$$\log p_\theta(x) = \log \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(x, z)}{q_\phi(z|x)} \right] \tag{C.3}$$

Apply Jensen's inequality to move the log inside the expectation for a lower bound:

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)} \left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)} \right] \tag{C.4}$$

Decompose the logarithm:

$$\mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x,z) - \log q_\phi(z|x) \right] \tag{C.5}$$

Expand the joint probability $\log p_\theta(x,z)$ into the log likelihood $\log p_\theta(x|z)$ and the log prior $\log p_\theta(z)$:

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(z)] - \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x)] \tag{C.6}$$

which contains the Kullback–Leibler (KL) divergence $D_{KL}(q_\phi(z|x)\|p_\theta(z))$:

$$D_{KL}(q_\phi(z|x)\|p_\theta(z)) = \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x)] - \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(z)] \tag{C.7}$$

Eventually the evidence lower bound ($ELBO$) can be derived:

$$ELBO = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)\|p_\theta(z)) \tag{C.8}$$

where ϕ represents the encoder network with the learning parameter in the encoder, θ represents the learning parameter in the decoder.

C.2 Transformation Between Rotation Representations

The transformation functions between different rotation representations are listed below:

Rotation Matrix to Euler Angles

Given a rotation matrix R, the Euler angles (α, β, γ) can be calculated as:

$$\beta = \arctan 2\left(\sqrt{R_{13}^2 + R_{23}^2}, R_{33}\right)$$
$$\alpha = \arctan 2\left(R_{23}/\cos(\beta), R_{33}/\cos(\beta)\right)$$
$$\gamma = \arctan 2\left(R_{12}/\cos(\beta), R_{11}/\cos(\beta)\right)$$

Euler Angles to Rotation Matrix

Given Euler angles (α, β, γ), the rotation matrix R can be calculated as:

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha)$$

Rotation Matrix and Axis-angle

Given a rotation matrix R, the axis $e = (e_x, e_y, e_z)$ and angle θ can be calculated as:

$$\theta = \arccos\left(\frac{\mathrm{Tr}(R) - 1}{2}\right)$$
$$e = \frac{1}{2\sin(\theta)}\begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix}$$

Axis-Angle to Rotation Matrix

Given an axis $e = (e_x, e_y, e_z)$ and an angle θ, the rotation matrix R can be calculated using the Rodrigues' rotation formula:

$$R = \mathbf{I} + \sin(\theta)\mathbf{K} + (1 - \cos(\theta))\mathbf{K}^2$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{K}$ is the skew-symmetric matrix of e:

$$\mathbf{K} = \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 0 \end{bmatrix}$$

Rotation Matrix to Quaternion

Given a rotation matrix R, the quaternion $Q = q_0 + q_1 i + q_2 j + q_3 k$ can be calculated as:

$$\begin{aligned} q_0 &= \frac{1}{2}\sqrt{1 + R_{11} + R_{22} + R_{33}} \\ q_1 &= \frac{1}{4q_0}(R_{32} - R_{23}) \\ q_2 &= \frac{1}{4q_0}(R_{13} - R_{31}) \\ q_3 &= \frac{1}{4q_0}(R_{21} - R_{12}) \end{aligned}$$

Quaternion to Rotation Matrix

Given a unit quaternion $Q = q_0 + q_1 i + q_2 j + q_3 k$, the rotation matrix R can be calculated as:

$$\begin{aligned} R_{11} &= 1 - 2q_2^2 - 2q_3^2 & R_{12} &= 2q_1q_2 - 2q_3q_0 & R_{13} &= 2q_1q_3 + 2q_2q_0, \\ R_{21} &= 2q_1q_2 + 2q_3q_0 & R_{22} &= 1 - 2q_1^2 - 2q_3^2 & R_{23} &= 2q_2q_3 - 2q_1q_0, \\ R_{31} &= 2q_1q_3 - 2q_2q_0 & R_{32} &= 2q_2q_3 + 2q_1q_0 & R_{33} &= 1 - 2q_1^2 - 2q_2^2 \end{aligned}$$

Bibliography

[1] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part I 11*, pages 548–562. Springer, 2013.

[2] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Database., 2012. URL https://campar.in.tum.de/Main/StefanHinterstoisser.

[3] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*, pages 536–551. Springer, 2014.

[4] Eric Brachmann. 6D Object Pose Estimation using 3D Object Coordinates [Data], 2020. URL https://doi.org/10.11588/data/V4MUMX.

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

[6] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. The

ycb-video dataset., 2018. URL https://rse-lab.cs.washington.edu/projects/posecnn/.

[7] Tomas Hodan, Martin Sundermeyer, Bertram Drost, Yann Labbe, Eric Brachmann, Frank Michel, Carsten Rother, and Jiri Matas. Bop challenge 2020 on 6d object localization. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 577–594. Springer, 2020.

[8] Tomas Hodan, Martin Sundermeyer, Bertram Drost, Yann Labbe, Eric Brachmann, Frank Michel, Carsten Rother, and Jiri Matas. Datasets., 2020. URL https://bop.felk.cvut.cz/datasets/.

[9] Matthew Matl. Pyrender: Easy-to-use gltf 2.0-compliant opengl renderer for visualization of 3d scenes., 2018. URL https://github.com/mmatl/pyrender.

[10] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3614–3631, 2020.

[11] Adane Nega Tarekegn, Mario Giacobini, and Krzysztof Michalak. A review of methods for imbalanced multi-label classification. *Pattern Recognition*, 118: 107965, 2021.

[12] Aria Salari, Abtin Djavadifar, Xiangrui Liu, and Homayoun Najjaran. Object recognition datasets and challenges: A review. *Neurocomputing*, 495:129–152, 2022.

[13] Xiaoxu Li, Xiaochen Yang, Zhanyu Ma, and Jing-Hao Xue. Deep metric learning for few-shot image classification: A review of recent developments. *Pattern Recognition*, page 109381, 2023.

[14] Wenguan Wang, Qiuxia Lai, Huazhu Fu, Jianbing Shen, Haibin Ling, and Ruigang Yang. Salient object detection in the deep learning era: An in-depth

survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (6):3239–3259, 2021.

[15] Dingwen Zhang, Junwei Han, Gong Cheng, and Ming-Hsuan Yang. Weakly supervised object localization and detection: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5866–5885, 2021.

[16] Gabriel Huang, Issam Laradji, David Vazquez, Simon Lacoste-Julien, and Pau Rodriguez. A survey of self-supervised and few-shot object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4071–4089, 2022.

[17] Gong Cheng, Xiang Yuan, Xiwen Yao, Kebing Yan, Qinghua Zeng, Xingxing Xie, and Junwei Han. Towards large-scale small object detection: Survey and benchmarks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[18] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023.

[19] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44 (7):3523–3542, 2021.

[20] Yujian Mo, Yan Wu, Xinneng Yang, Feilin Liu, and Yujun Liao. Review the state-of-the-art technologies of semantic segmentation based on deep learning. *Neurocomputing*, 493:626–646, 2022.

[21] Tianfei Zhou, Fatih Porikli, David J Crandall, Luc Van Gool, and Wenguan Wang. A survey on deep learning technique for video segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7099–7122, 2022.

[22] Wei Shen, Zelin Peng, Xuehui Wang, Huayu Wang, Jiazhong Cen, Dongsheng

Jiang, Lingxi Xie, Xiaokang Yang, and Q Tian. A survey on label-efficient deep image segmentation: Bridging the gap between weak supervision and dense prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[23] Caner Sahin, Guillermo Garcia-Hernando, Juil Sock, and Tae-Kyun Kim. A review on object pose recovery: From 3d bounding box detectors to full 6d pose estimators. *Image and Vision Computing*, 96:103898, 2020.

[24] Sabera Hoque, Md Yasir Arafat, Shuxiang Xu, Ananda Maiti, and Yuchen Wei. A comprehensive review on 3d object detection and 6d pose estimation with deep learning. *IEEE Access*, 9:143746–143770, 2021.

[25] Guoguang Du, Kai Wang, Shiguo Lian, and Kaiyong Zhao. Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review. *Artificial Intelligence Review*, 54(3): 1677–1734, 2021.

[26] Martin Sundermeyer, Tomáš Hodaň, Yann Labbe, Gu Wang, Eric Brachmann, Bertram Drost, Carsten Rother, and Jiří Matas. Bop challenge 2022 on detection, segmentation and pose estimation of specific rigid objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2784–2793, 2023.

[27] Nikolaus Correll, Kostas E Bekris, Dmitry Berenson, Oliver Brock, Albert Causo, Kris Hauser, Kei Okada, Alberto Rodriguez, Joseph M Romano, and Peter R Wurman. Analysis and observations from the first amazon picking challenge. *IEEE Transactions on Automation Science and Engineering*, 15(1): 172–188, 2016.

[28] Carlos Hernandez, Mukunda Bharatheesha, Wilson Ko, Hans Gaiser, Jethro Tan, Kanter van Deurzen, Maarten de Vries, Bas Van Mil, Jeff van Egmond, Ruben Burger, et al. Team delft's robot winner of the amazon picking challenge

2016. In *RoboCup 2016: Robot World Cup XX 20*, pages 613–624. Springer, 2017.

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[30] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[31] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[32] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 778–792. Springer, 2010.

[33] Tomáš Hodaň, Xenophon Zabulis, Manolis Lourakis, Štěpán Obdržálek, and Jiří Matas. Detection and fine 3d pose estimation of texture-less objects in rgb-d images. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4421–4428. IEEE, 2015.

[34] Azmi Haider and Hagit Hel-Or. What can we learn from depth camera sensor noise? *Sensors*, 22(14):5448, 2022.

[35] Sida Peng, Xiaowei Zhou, Yuan Liu, Haotong Lin, Qixing Huang, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof object pose estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):3212–3223, 2020.

[36] Xinke Deng, Yu Xiang, Arsalan Mousavian, Clemens Eppner, Timothy Bretl, and Dieter Fox. Self-supervised 6d object pose estimation for robot manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3665–3671. IEEE, 2020.

[37] Antti Hietanen, Jyrki Latokartano, Alessandro Foi, Roel Pieters, Ville Kyrki, Minna Lanz, and Joni-Kristian Kämäräinen. Benchmarking pose estimation for robot manipulation. *Robotics and Autonomous Systems*, 143:103810, 2021.

[38] Munkhtulga Byambaa, Gou Koutaki, and Lodoiravsal Choimaa. 6d pose estimation of transparent object from single rgb image for robotic manipulation. *IEEE Access*, 10:114897–114906, 2022.

[39] Myung-Hwan Jeon, Jeongyun Kim, Jee-Hwan Ryu, and Ayoung Kim. Ambiguity-aware multi-object pose optimization for visually-assisted robot manipulation. *IEEE Robotics and Automation Letters*, 8(1):137–144, 2022.

[40] Kai Chen, Rui Cao, Stephen James, Yichuan Li, Yun-Hui Liu, Pieter Abbeel, and Qi Dou. Sim-to-real 6d object pose estimation via iterative self-training for robotic bin picking. In *European Conference on Computer Vision*, pages 533–550. Springer, 2022.

[41] 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. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3343–3352, 2019.

[42] Di Wu, Zhaoyong Zhuang, Canqun Xiang, Wenbin Zou, and Xia Li. 6d-vnet: End-to-end 6-dof vehicle pose estimation from monocular rgb images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[43] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings*

of the IEEE conference on Computer Vision and Pattern Recognition, pages 7074–7082, 2017.

[44] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *European Conference on Computer Vision*, pages 644–660. Springer, 2020.

[45] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition workshops*, pages 1000–1001, 2020.

[46] David Joseph Tan, Nassir Navab, and Federico Tombari. 6d object pose estimation with depth images: A seamless approach for robotic interaction and augmented reality. *arXiv preprint arXiv:1709.01459*, 2017.

[47] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. Onepose: One-shot object pose estimation without cad models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6825–6834, 2022.

[48] 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*, pages 298–315. Springer, 2022.

[49] Fulin Tang, Yihong Wu, Xiaohui Hou, and Haibin Ling. 3d mapping and 6d pose computation for real time augmented reality on cylindrical objects. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(9):2887–2899, 2019.

[50] 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 conference on computer vision and pattern recognition*, pages 3364–3372, 2016.

[51] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker, Alberto Rodriguez, and Jianxiong Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 1386–1383. IEEE, 2017.

[52] Masakazu Yoshimura, Murilo M Marinho, Kanako Harada, and Mamoru Mitsuishi. Single-shot pose estimation of surgical robot instruments' shafts from monocular endoscopic images. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9960–9966. IEEE, 2020.

[53] Nicholas Greene, Wenkai Luo, and Peter Kazanzides. dvpose: Automated data collection and dataset for 6d pose estimation of robotic surgical instruments. In *2023 International Symposium on Medical Robotics (ISMR)*, pages 1–7. IEEE, 2023.

[54] Ziqi Lu, Yihao Zhang, Kevin Doherty, Odin Severinsen, Ethan Yang, and John Leonard. Slam-supported self-training for 6d object pose estimation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2833–2840. IEEE, 2022.

[55] Jeremy Ma and Joel W Burdick. A probabilistic framework for stereo-vision based 3d object search with 6d pose estimation. In *2010 IEEE International Conference on Robotics and Automation*, pages 2036–2042. IEEE, 2010.

[56] Antonio Criminisi and Jamie Shotton. *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.

[57] Simon DiMaio, Mike Hanuschik, and Usha Kreaden. The da vinci surgical system. *Surgical robotics: systems applications and visions*, pages 199–217, 2011.

[58] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. Cambridge University Press, 2023. https://D2L.ai.

[59] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[60] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[61] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63:3–42, 2006.

[62] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[63] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[64] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

[65] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, and Rudolph Triebel. Augmented autoencoders: Implicit 3d orientation learning for 6d object detection. *International Journal of Computer Vision*, 128:714–729, 2020.

[66] 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*, pages 13916–13925, 2020.

[67] 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*, pages 1941–1950, 2019.

[68] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7668–7677, 2019.

[69] 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*, pages 11703–11712, 2020.

[70] 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*, pages 7678–7687, 2019.

[71] 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*, pages 16611–16621, 2021.

[72] 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*, pages 14880–14890, 2022.

[73] 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*, pages 6749–6758, 2022.

[74] Christiaan GA Viviers, Lena Filatova, Maurice Termeer, Peter HN de With,

and Fons van der Sommen. Advancing 6-dof instrument pose estimation in variable x-ray imaging geometries. *IEEE Transactions on Image Processing*, 33:2462–2476, 2024.

[75] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, pages 574–591. Springer, 2020.

[76] 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*, pages 6738–6748, 2022.

[77] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE multimedia*, 19 (2):4–10, 2012.

[78] Vladimir Tadic, Akos Odry, Istvan Kecskes, Ervin Burkus, Zoltan Kiraly, and Peter Odry. Application of intel realsense cameras for depth image generation in robotics. *WSEAS Transac. Comput*, 18:2224–2872, 2019.

[79] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[80] Tomáš Hodaň. Pose estimation of specific rigid objects. *PhD Thesis, Czech Technical University in Prague*, 2021.

[81] Ju Il Sock. Active recognition and domain adaptation for 6d object pose estimation. *PhD Thesis, Imperial College London*, 2020.

[82] Zhaoxin Fan, Yazhi Zhu, Yulin He, Qi Sun, Hongyan Liu, and Jun He. Deep learning on monocular object pose detection and tracking: A comprehensive overview. *ACM Computing Surveys*, 55(4):1–40, 2022.

[83] Rui Zeng, Yuhui Wen, Wang Zhao, and Yong-Jin Liu. View planning in robot

active vision: A survey of systems, algorithms, and applications. *Computational Visual Media*, 6:225–245, 2020.

[84] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, 1986.

[85] David G Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial intelligence*, 31(3):355–395, 1987.

[86] Yehezkel Lamdan and Haim J Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *1988 Second International Conference on Computer Vision*, pages 238–239. IEEE Computer Society, 1988.

[87] David G Lowe et al. Fitting parameterized three-dimensional models to images. *IEEE transactions on pattern analysis and machine intelligence*, 13(5): 441–450, 1991.

[88] Dima Damen, Pished Bunnun, Andrew Calway, and Walterio W Mayol-Cuevas. Real-time learning and detection of 3d texture-less objects: A scalable approach. In *BMVC*, 2012.

[89] Tomá Hodan, Dima Damen, Walterio Mayol-Cuevas, and Jirí Matas. Efficient texture-less object detection for augmented reality guidance. In *2015 IEEE International Symposium on Mixed and Augmented Reality Workshops*, pages 81–86. IEEE, 2015.

[90] Piotr Dollár and C Lawrence Zitnick. Structured forests for fast edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1841–1848, 2013.

[91] Jeffrey S. Beis and David G. Lowe. Indexing without invariants in 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1000–1015, 1999.

[92] Martin A Fischler and Robert C Bolles. Random sample consensus: a

paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[93] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155–166, 2009.

[94] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[95] Manolis Lourakis and Xenophon Zabulis. Model-based pose estimation for rigid objects. In *International conference on computer vision systems*, pages 83–92. Springer, 2013.

[96] 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*, pages 998–1005. Ieee, 2010.

[97] Vincent Lepetit, Julien Pilet, and Pascal Fua. Point matching as a classification problem for fast and robust object pose estimation. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II. IEEE, 2004.

[98] Vincent Lepetit, Pascal Lagger, and Pascal Fua. Randomized trees for real-time keypoint recognition. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 775–781. IEEE, 2005.

[99] Shree K Nayar, Sameer A Nene, and Hiroshi Murase. Real-time 100 object recognition system. In *Proceedings of IEEE international conference on robotics and automation*, volume 3, pages 2321–2325. IEEE, 1996.

[100] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural computation*, 9(7):1545–1588, 1997.

[101] 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*, 66(3):231–259, 2006.

[102] Manuel Martinez, Alvaro Collet, and Siddhartha S Srinivasa. Moped: A scalable and low latency object recognition and pose estimation system. In *2010 IEEE International Conference on Robotics and Automation*, pages 2043–2049. IEEE, 2010.

[103] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 international conference on computer vision*, pages 858–865. IEEE, 2011.

[104] Zhe Cao, Yaser Sheikh, and Natasha Kholgade Banerjee. Real-time scalable 6dof pose estimation for textureless objects. In *2016 IEEE International conference on Robotics and Automation (ICRA)*, pages 2441–2448. IEEE, 2016.

[105] Reyes Rios-Cabrera and Tinne Tuytelaars. Discriminatively trained templates for 3d object detection: A real time scalable approach. In *Proceedings of the IEEE international conference on computer vision*, pages 2048–2055, 2013.

[106] Wadim Kehl, Federico Tombari, Nassir Navab, Slobodan Ilic, and Vincent Lepetit. Hashmod: A hashing method for scalable 3d object detection. *BMVC*, 2016.

[107] Deping Li, Hanyun Wang, Ning Liu, Xiaoming Wang, and Jin Xu. 3d object recognition and pose estimation from point cloud using stably observed point pair feature. *IEEE Access*, 8:44335–44345, 2020.

[108] Chi-Yi Tsai and Shu-Hsiang Tsai. Simultaneous 3d object recognition and pose estimation based on rgb-d images. *IEEE Access*, 6:28859–28869, 2018.

[109] Ales Leonardis and Horst Bischof. Dealing with occlusions in the eigenspace approach. In *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 453–458. IEEE, 1996.

[110] Hiroshi Murase and Shree K Nayar. Visual learning and recognition of 3-d objects from appearance. *International journal of computer vision*, 14(1): 5–24, 1995.

[111] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. *International Journal of Computer Vision*, 128:657–678, 2020.

[112] Gert-Jan Van den Braak, Cedric Nugteren, Bart Mesman, and Henk Corporaal. Fast hough transform on gpus: Exploration of algorithm trade-offs. In *Advanced Concepts for Intelligent Vision Systems: 13th International Conference, ACIVS 2011, Ghent, Belgium, August 22-25, 2011. Proceedings 13*, pages 611–622. Springer, 2011.

[113] Jimmy Wu, Bolei Zhou, Rebecca Russell, Vincent Kee, Syler Wagner, Mitchell Hebert, Antonio Torralba, and David MS Johnson. Real-time object pose estimation with pose interpreter networks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6798–6805. IEEE, 2018.

[114] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 472–480, 2017.

[115] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[116] Yannick Bukschat and Marcus Vetter. Efficientpose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach. *arXiv preprint arXiv:2011.04307*, 2020.

[117] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

[118] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.

[119] 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 international conference on computer vision*, pages 1521–1529, 2017.

[120] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016.

[121] Xinke Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking. *IEEE Transactions on Robotics*, 37(5):1328–1342, 2021.

[122] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5974–5983, 2017.

[123] Jian S Dai. Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections. *Mechanism and Machine Theory*, 92:144–152, 2015.

[124] Roger W Brockett. Robotic manipulators and the product of exponentials formula. In *Mathematical Theory of Networks and Systems: Proceedings of the MTNS-83 International Symposium Beer Sheva, Israel, June 20–24, 1983*, pages 120–129. Springer, 2005.

[125] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[126] 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 international conference on computer vision*, pages 3828–3836, 2017.

[127] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[128] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 292–301, 2018.

[129] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[130] Ultralytics. Yolov5: A state-of-the-art real-time object detection system, 2021. URL https://github.com/ultralytics/yolov5. Visited on 19/06/24.

[131] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pages 483–499. Springer, 2016.

[132] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134, 2018.

[133] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017.

[134] Peter Kontschieder, Samuel Bulò, Antonio Criminisi, Pushmeet Kohli, Marcello Pelillo, and Horst Bischof. Context-sensitive decision forests for object detection. *Advances in neural information processing systems*, 25, 2012.

[135] Daniel Barath and Jiří Matas. Graph-cut ransac. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6733–6741, 2018.

[136] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[137] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404: 132306, 2020.

[138] Kevin Murphy and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Sequential Monte Carlo methods in practice*, pages 499–515. Springer, 2001.

[139] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3109–3118, 2015.

[140] Sergey Zakharov, Wadim Kehl, Benjamin Planche, Andreas Hutter, and Slobodan Ilic. 3d object instance recognition and pose estimation using triplet loss with dynamic margin. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 552–559. IEEE, 2017.

[141] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent-

Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. Bop: Benchmark for 6d object pose estimation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018.

[142] 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)*, pages 66–70. IEEE, 2019.

[143] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Dmitry Olefir, Tomas Hodan, Youssef Zidan, Mohamad Elbadrawy, Markus Knauer, Harinandan Katam, and Ahsan Lodhi. Blenderproc: Reducing the reality gap with photorealistic rendering. In *International Conference on Robotics: Sciene and Systems, RSS 2020*, 2020.

[144] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003.

[145] George Terzakis and Manolis Lourakis. A consistently fast and globally optimal solution to the perspective-n-point problem. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 478–494. Springer, 2020.

[146] PJ Besl and Neil D McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 14(02):239–256, 1992.

[147] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE, 2001.

[148] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel,

and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.

[149] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. *IEEE Robotics & Automation Magazine*, 22(3):36–52, 2015.

[150] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3):261–268, 2017.

[151] Tomáš Hodan, Pavel Haluza, Štepán Obdržálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 880–888. IEEE, 2017.

[152] 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 international conference on computer vision workshops*, pages 2200–2208, 2017.

[153] Roman Kaskman, Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[154] Stephen Tyree, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Jeffrey Smith, and Stan Birchfield. 6-dof pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems*

(IROS), pages 13081–13088. IEEE, 2022.

[155] Colin Rennie, Rahul Shome, Kostas E Bekris, and Alberto F De Souza. A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place. *IEEE Robotics and Automation Letters*, 1(2):1179–1185, 2016.

[156] Andreas Doumanoglou, Rigas Kouskouridas, Sotiris Malassiotis, and Tae-Kyun Kim. Recovering 6d object pose and predicting next-best-view in the crowd. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3583–3592, 2016.

[157] Alykhan Tejani, Danhang Tang, Rigas Kouskouridas, and Tae-Kyun Kim. Latent-class hough forests for 3d object detection and pose estimation. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 462–477. Springer, 2014.

[158] Joel Vidal, Chyi-Yeu Lin, Xavier Lladó, and Robert Martí. A method for 6d pose estimation of free-form rigid objects using point pair features on range data. *Sensors*, 18(8):2678, 2018.

[159] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 969–977, 2018.

[160] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10285–10295, 2019.

[161] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and

Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[162] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 39(06):1137–1149, 2017.

[163] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86 (11):2278–2324, 1998.

[164] Yann LeCun and Corinna Cortes. Mnist digits classification dataset., 1998. URL https://keras.io/api/datasets/mnist/.

[165] Inc. The MathWorks. Data sets for deep learning., 1994-2024. URL https://uk.mathworks.com/help/deeplearning/ug/data-sets-for-deep-learning.html.

[166] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012.

[167] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Nyu depth dataset v2., 2012. URL https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html.

[168] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

[169] Siddharth Mahendran, Haider Ali, and René Vidal. 3d pose regression using convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2174–2182, 2017.

[170] Eric W. Weisstein. Euler angles, 1999-2023. URL https://mathworld.wolfram.com/EulerAngles.html. Visited on 27/07/23.

[171] Carlo Tomasi. Vector representation of rotations. *Computer Science*, 527:2–4, 2013.

[172] Siddharth Mahendran, Haider Ali, and Rene Vidal. A mixed classification-regression framework for 3d pose estimation from 2d images. *BMVC*, 2018.

[173] Erik B Dam, Martin Koch, and Martin Lillholm. *Quaternions, interpolation and animation*, volume 2. Citeseer, 1998.

[174] 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*, pages 6841–6850, 2019.

[175] Simon L Altmann. *Rotations, quaternions, and double groups*. Courier Corporation, 2005.

[176] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Learning 3-d object orientation from images. In *2009 IEEE International conference on robotics and automation*, pages 794–800. IEEE, 2009.

[177] Steven J Leon, Åke Björck, and Walter Gander. Gram-schmidt orthogonalization: 100 years and more. *Numerical Linear Algebra with Applications*, 20 (3):492–532, 2013.

[178] Valentin Peretroukhin, Matthew Giamou, David M. Rosen, W. Nicholas Greene, Nicholas Roy, and Jonathan Kelly. A Smooth Representation of SO(3) for Deep Rotation Learning with Uncertainty. In *Proceedings of Robotics: Science and Systems (RSS'20)*, Jul. 12–16 2020.

[179] Jake Levinson, Carlos Esteves, Kefan Chen, Noah Snavely, Angjoo Kanazawa, Afshin Rostamizadeh, and Ameesh Makadia. An analysis of svd for deep

rotation estimation. *Advances in Neural Information Processing Systems*, 33: 22554–22565, 2020.

[180] Jiayi Chen, Yingda Yin, Tolga Birdal, Baoquan Chen, Leonidas J Guibas, and He Wang. Projective manifold gradient layer for deep rotation regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6646–6655, 2022.

[181] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

[182] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[183] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning internal representations by error propagation, 1985.

[184] Marc'Aurelio Ranzato, Y-Lan Boureau, Yann Cun, et al. Sparse feature learning for deep belief networks. *Advances in neural information processing systems*, 20, 2007.

[185] Alireza Makhzani and Brendan Frey. K-sparse autoencoders. In *International Conference on Learning Representations*, 2014.

[186] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th international conference on international conference on machine learning*, pages 833–840, 2011.

[187] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[188] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

[189] Lilian Weng. From autoencoder to beta-vae. *lilianweng.github.io*, 2018. URL https://lilianweng.github.io/posts/2018-08-12-vae/.

[190] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[191] Partha Ghosh, Medhi SM Sajjadi, Antonio Vergari, and Michael Black. From variational to deterministic autoencoders. In *8th International Conference on Learning Representations*, 2020.

[192] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[193] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[194] Wei-Yin Loh. Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(1):14–23, 2011.

[195] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[196] Wei Yang, Kuanquan Wang, and Wangmeng Zuo. Neighborhood component feature selection for high-dimensional data. *J. Comput.*, 7(1):161–168, 2012.

[197] Tomáš Hodaň, Jiří Matas, and Štěpán Obdržálek. On evaluation of 6d object pose estimation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*, pages 606–619. Springer, 2016.

[198] Changhyun Choi and Henrik I Christensen. 3d pose estimation of daily objects using an rgb-d camera. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3342–3349. IEEE, 2012.

[199] Yan Di, Fabian Manhardt, Gu Wang, Xiangyang Ji, Nassir Navab, and Fed-

erico Tombari. So-pose: Exploiting self-occlusion for direct 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12396–12405, 2021.

[200] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2937, 2013.

[201] Haowen Deng, Mai Bui, Nassir Navab, Leonidas Guibas, Slobodan Ilic, and Tolga Birdal. Deep bingham networks: Dealing with uncertainty and ambiguity in pose estimation. *International Journal of Computer Vision*, 130(7): 1627–1654, 2022.

[202] Jianyu Zhao, Edward Sanderson, and Bogdan J Matuszewski. Cvml-pose: Convolutional vae based multi-level network for object 3d pose estimation. *IEEE Access*, 11:13830–13845, 2023.

[203] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[204] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[205] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.

[206] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[207] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[208] Patrick Popescu-Pampu et al. *What is the Genus?*, volume 2162. Springer, 2016.

[209] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

[210] Jason Brownlee. How to fix the vanishing gradients problem using the relu., 2020. URL https://machinelearningmastery.com/how-to-fix-vanishing-gradients-using-the-rectified-linear-activation-function/. Visited on 15/12/23.

[211] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[212] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, pages 13001–13008, 2020.

[213] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS 2017 Workshop on Autodiff*, 2017.

[214] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[215] The MathWorks Inc. Evaluating the accuracy of single camera calibration., 1994-2024. URL https://uk.mathworks.com/help/vision/ug/

evaluating-the-accuracy-of-single-camera-calibration.html. Visited on 25/06/24.

[216] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021.

[217] Tomas Hodan, Martin Sundermeyer, Yann Labbe, Van Nguyen Nguyen, Gu Wang, Eric Brachmann, Bertram Drost, Vincent Lepetit, Carsten Rother, and Jiri Matas. Bop challenge 2023 on detection segmentation and pose estimation of seen and unseen rigid objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5610–5619, 2024.

[218] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[219] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.

[220] Kok-Lim Low. Linear least-squares optimization for point-to-plane icp surface registration. *Chapel Hill, University of North Carolina*, 4(10):1–3, 2004.

[221] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[222] Liu Liu, Han Xue, Wenqiang Xu, Haoyuan Fu, and Cewu Lu. Toward real-world category-level articulation pose estimation. *IEEE Transactions on Image Processing*, 31:1072–1083, 2022.

[223] Liu Liu, Qi Wu, Zhendong Xue, Sucheng Qian, and Rui Li. Reaper: Articulated object 6d pose estimation with deep reinforcement learning. In *2023*

IEEE International Conference on Image Processing (ICIP), pages 21–25. IEEE, 2023.

[224] Lixin Yang, Kailin Li, Xinyu Zhan, Jun Lv, Wenqiang Xu, Jiefeng Li, and Cewu Lu. Artiboost: Boosting articulated 3d hand-object pose estimation via online exploration and synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2750–2760, 2022.

[225] Wanli Peng, Jianhang Yan, Hongtao Wen, and Yi Sun. Self-supervised category-level 6d object pose estimation with deep implicit shape representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2082–2090, 2022.

[226] Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. Megapose: 6d pose estimation of novel objects via render & compare. In *Conference on Robot Learning*, pages 715–725. PMLR, 2023.

[227] Dingding Cai, Janne Heikkilä, and Esa Rahtu. Ove6d: Object viewpoint encoding for depth-based 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6803–6813, 2022.

[228] Lahav Lipson, Zachary Teed, Ankit Goyal, and Jia Deng. Coupled iterative refinement for 6d multi-object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6728–6737, 2022.

[229] 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*, pages 1060–1070. PMLR, 2023.

[230] 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*, pages 392–406. Springer, 2022.

[231] Pedro Castro and Tae-Kyun Kim. Posematcher: One-shot 6d object pose estimation by deep feature matching. *arXiv preprint arXiv:2304.01382*, 2023.

[232] Ning Gao, Vien Anh Ngo, Hanna Ziesche, and Gerhard Neumann. Sa6d: Self-adaptive few-shot 6d pose estimator for novel and occluded objects. In *7th Annual Conference on Robot Learning*, 2023.

[233] Yang Xiao, Vincent Lepetit, and Renaud Marlet. Few-shot object detection and viewpoint estimation for objects in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3090–3106, 2022.

[234] Yisheng He, Yao Wang, Haoqiang Fan, Jian Sun, and Qifeng Chen. Fs6d: Few-shot 6d pose estimation of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6814–6824, 2022.

[235] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Zero-shot category-level object pose estimation. In *European Conference on Computer Vision*, pages 516–532. Springer, 2022.

[236] Zhiwen Fan, Panwang Pan, Peihao Wang, Yifan Jiang, Dejia Xu, Hanwen Jiang, and Zhangyang Wang. Pope: 6-dof promptable pose estimation of any object, in any scene, with one reference. *arXiv preprint arXiv:2305.15727*, 2023.

[237] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[238] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor

Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.

[239] Meng-Hao Guo, Cheng-Ze Lu, Qibin Hou, Zhengning Liu, Ming-Ming Cheng, and Shi-Min Hu. Segnext: Rethinking convolutional attention design for semantic segmentation. *Advances in Neural Information Processing Systems*, 35:1140–1156, 2022.

[240] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

[241] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19830–19843, 2023.

[242] Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. 2007.