GS-Pose: Generalizable Segmentation-Based 6D Object Pose Estimation With 3D Gaussian Splatting

Dingding Cai
Tampere University, Finland
dingding.cai@tuni.fi

Janne Heikkilä
University of Oulu, Finland
janne.heikkila@oulu.fi

Esa Rahtu
Tampere University, Finland
esa.rahtu@tuni.fi

Abstract

This paper introduces GS-Pose, a unified framework for localizing and estimating the 6D pose of novel objects. GS-Pose begins with a set of posed RGB images of a previously unseen object and builds three distinct representations stored in a database. At inference, GS-Pose operates sequentially by locating the object in the input image, estimating its initial 6D pose using a retrieval approach, and refining the pose with a render-and-compare method. The key insight is the application of the appropriate object representation at each stage of the process. In particular, for the refinement step, we leverage 3D Gaussian splatting, a novel differentiable rendering technique that offers high rendering speed and relatively low optimization time. Off-the-shelf toolchains and commodity hardware, such as mobile phones, can be used to capture new objects to be added to the database. Extensive evaluations on the LINEMOD and OnePose-LowTexture datasets demonstrate excellent performance, establishing the new state-of-the-art. The source code is publicly available at https://github.com/dingdingcai/GSPose.

1. Introduction

Acquiring the 3D orientation and 3D location of an object based on RGB images is a long-standing and important problem in computer vision and robotics. This 6D pose information is vital in applications that interact with the physical world, such as robotic manipulation [9, 10] and augmented reality [32, 48]. Popular pose estimation approaches are based on training instance-specific models, and they often assume the availability of an external object detector for detecting the object from input RGB images. While some works have proposed approaches to circumvent this problem [28, 36, 47], they often rely on high-fidelity 3D CAD models of the object, which can be expensive and time-consuming to acquire.

Ideally, a new object should be learned from a casually captured set of RGB reference images without requiring any expensive model parameter optimization. Recently, Liu *et al*. [30] introduced a method called Gen6D in this direction. Gen6D works by extracting 2D feature maps from the reference images, which are subsequently utilized for various sub-tasks, including object localization, initial pose estimation, and pose refinement. However, relying only on 2D representation often leads to sub-optimal performance. Alternatively, OnePose [51] and OnePose++ [15] explicitly reconstruct a 3D point cloud from the reference images via local feature matching. The 6D pose is obtained using 2D-3D correspondence matching between the test image and the reference point cloud. The practical challenge is to obtain an accurate 3D point cloud representation, particularly for texture-less and symmetric objects. Furthermore, both approaches still rely on an external object detector for cropping out the object of interest, limiting their applicability in real-world scenarios.

The key ingredient in 6D pose estimation is the object representation generated from the input images. Popular choices include 2D feature maps [30], 3D point clouds [15, 51], latent 3D models [41], and 3D CAD models [47], to name a few. Generally, each representation exhibits strengths in one aspect, *e.g*., object localization or fast initial 6D pose approximation, but performs poorly on other parts of the pipeline. With these insights, we propose a framework that applies multiple representations optimized for the three key steps: 1) object localization, 2) fast initial 6D pose estimation, and 3) iterative pose refinement. In particular, we leverage the recent advancements in so-called Foundation models and co-segmentation paradigms to construct powerful representations for object localization using only a handful of reference images. Secondly, we estimate a rough 6D pose using optimized template retrieval. Finally, the pose estimate is refined using an iterative render-and-compare technique. To this end, we rely on a novel inverse rendering method called 3D Gaussian Splatting (3DGS) [20], which represents a scene by many differentiable 3D Gaussian primitives with optimizable geometric and appearance properties. This explicit representation

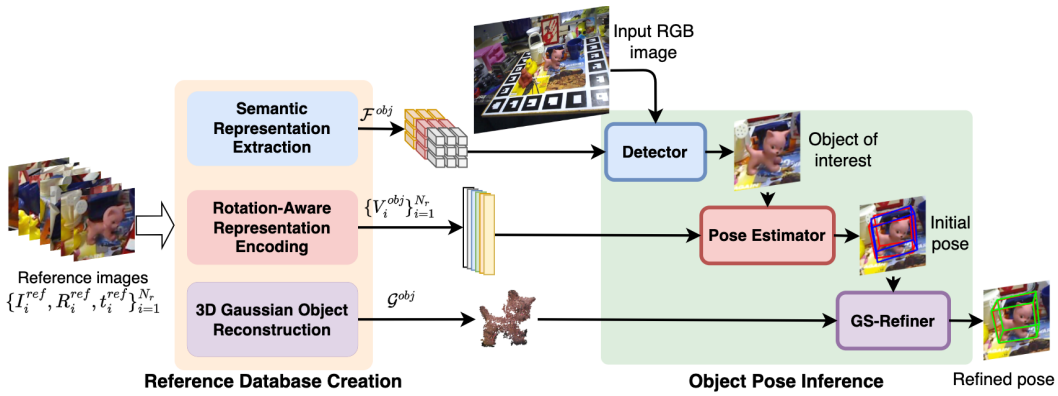

Figure 1. **Overview of GS-Pose**. GS-Pose involves two distinct phases to achieve pose estimation for a novel object, *i.e.*, reference database creation and object pose inference. The first phase operates offline and occurs only once per object to construct multiple representations of the object. These representations include an object semantic representation ($\mathcal{F}^{obj}$), a set of embedding vectors ($\{V_i^{obj}\}_{i=1}^{N_r}$), and a 3D Gaussian Object ($\mathcal{G}^{obj}$). During inference, GS-Pose first employs an object detector to detect the object in a query image using the semantic information $\mathcal{F}^{obj}$. Then, GS-Pose adopts a pose estimator to produce an initial pose (blue box) from the detection result with the embedding vectors $\{V_i^{obj}\}_{i=1}^{N_r}$, Finally, GS-Pose leverages a pose refinement module (GS-Refiner) with $\mathcal{G}^{obj}$ to obtain a refined pose (green box). We indicate the ground-truth pose in red.

enables real-time photorealistic rendering capabilities, ideal for 6D pose optimization.

We evaluate the proposed framework, called GS-Pose, on the LINEMOD [17] and OnePose-LowTexture [15] datasets and obtain new state-of-the-art results on both benchmarks. The contributions of our work are summarized as follows:

- We present an integrated framework for 3D CAD model-free 6D object pose estimation. For each stage, we propose an optimized representation obtained from a set of posed RGB images of newly added objects.
- We present a generalizable co-segmentation approach for extracting object segmentation masks jointly from the reference RGB images, facilitating representation learning.
- We present a robust 3D Gaussian splatting-based method for 6D object pose refinement.
- We experimentally confirm that the proposed framework achieves state-of-the-art performance on the LINEMOD and OnePose-LowTexture datasets.

2. Related Works

Object-Specific Pose Estimation. Most existing pose estimation methods [2, 3, 6, 13, 18, 21, 27, 42, 49, 54, 58] are object-specific pose estimators, which are specialized for pre-defined objects and cannot generalize to previously unseen objects without retraining. Some of them [2, 3, 6, 21, 54, 58] directly regress the 6D pose parameters from RGB images by training deep neural networks on a large number of labeled images. While other approaches [6, 13, 18, 27, 40, 42, 49] establish 2D-3D correspondences between 2D images and 3D object models to estimate the 6D pose by solving the Perspective-n-Point (PnP) [24] problem. To relax the assumptions about each object instance, category-level methods [5, 7, 8, 55] have recently been proposed to handle unseen object instances of the same trained category by assuming that objects within the same category share similar shape priors. However, they are still incapable of estimating the object pose of unknown categories.

Generalizable Object Pose Estimation. This type of work [1, 15, 16, 30, 34, 37, 39, 47, 51, 52, 57] removes the requirement of the object specific-training and can perform pose estimation for previously unseen objects during inference. There are two mainstreams, *i.e.*, object model-based and object model-free. The model-based approaches [1, 47, 52] assume access to the 3D CAD models for rendering the object pose-conditioned images that are often utilized for template matching [1, 52, 59], pose refinement [19, 26], or correspondence establishment [47]. To avoid 3D CAD models, recent works [15, 30, 39, 51] resort to capturing object multi-view images with known poses as reference data for pose estimation. OnePose series [15, 51] utilize the posed RGB images to reconstruct 3D object point clouds and establish explicit 2D-3D correspondences between 2D query images and the reconstructed 3D point clouds to solve the 6D pose. However, reliance on correspondences becomes fragile when applied to objects with visual ambiguities, such as symmetry. Besides, the above methods often assume that the 2D object detection or segmentation mask is available given a query image. In contrast, Gen6D [30] leverages the labeled reference images to detect the object in query images, initialize its pose, and then construct a 3D feature volume for pose refinement, which is the first work to simultaneously satisfy the re-

quirements of being fully generalizable, model-free, and RGB-only. The follow-up works [39, 60] revisit the Gen6D pipeline and improve the performance and robustness in object localization and pose estimation.

2D Object Detection. Commonly used object detection methods [14, 45, 46] are category-specific detectors and cannot generalize to untrained categories. To tackle this issue, some approaches [25, 30, 38, 47, 61] leverage object reference images to detect previously unseen objects through template matching or feature correlation. However, they often show limited generalizability to new domains.

3D Object Representation. Most generalizable pose estimators [1, 28, 33, 36, 47] often assume that the 3D object representations are available, such as 3D CAD models. OnePose family [15, 51] explicitly reconstructs 3D object point clouds from object multi-view RGB images, which can easily fail with challenging symmetric or textureless objects. Moreover, LatentFusion [41] and Gen6D series [30, 39] utilize the 2D image features to build the 3D object feature volumes for pose refinement. In this work, we instead exploit the differentiable 3D Gaussian Splatting [20] technique to create 3D Gaussian Object representations for pose estimation. To the best of our knowledge, GS-Pose is the first work that leverages 3D Gaussian splatting for 6D object pose estimation.

3. Approach

This section presents GS-Pose for estimating the 6D pose of novel objects from RGB images. An overview of GS-Pose is provided in Fig. 1. GS-Pose operates in two distinct phases: object reference database creation and object pose inference. The creation phase, requiring RGB images of a novel object with known poses (*e.g.*, captured with commodity devices like mobile phones), is performed offline once per object. During inference, GS-Pose leverages the pre-built object reference database to facilitate the 6D pose estimation task in a cascaded manner. In the subsequent subsections, we first present the reference database creation process in Sec. 3.1. Next, we describe the pose inference workflow in Sec. 3.2. Finally, we present the objective functions for training GS-Pose in Sec. 3.3.

3.1. Reference Database Creation

This section describes the process for creating a reference database of a novel object based on its reference data. This database primarily consists of object semantic representation $\mathcal{F}^{obj}$, a set of 3D object rotation-aware embedding vectors $\{V_i^{obj}\}_{i=1}^{N_r}$, and a 3D Gaussian Object representation $\mathcal{G}^{obj}$, where N_r is the number of reference examples. The creation process involves three sub-steps: (1) semantic representation extraction, (2) 3D object rotation-aware representation encoding, and (3) 3D Gaussian Object (3DGO) model reconstruction, as depicted in Fig. 2. In the following paragraphs, we elaborate on each sub-step.

Semantic Representation Extraction. To enable GS-Pose for 2D object detection and segmentation, we first extract a set of feature representation tokens that can effectively capture the semantic information of the target object from reference images. We leverage DINOv2 [35] to extract these tokens from RGB images. Essentially, a Co-Segmenter is employed to segment the object from the background, ensuring that only relevant feature tokens within the object region are considered (see Fig. 2 top). Given N_r reference images, we first select N_k ($\ll N_r$) keyframes using farthest point sampling (FPS) [44] based on their corresponding 3D rotation labels. Then, we extract image feature tokens $\mathcal{F}^{fps} \in \mathbb{R}^{N_k \times L \times C}$ from these keyframes using DINOv2, where L and C denote the token number and feature dimension of each frame. Next, we feed these feature tokens into the proposed Co-Segmenter, consisting of a transformer-like module and a mask decoding head, to jointly predict the object segmentation masks. Specifically, we reshape the keyframe feature tokens as feature maps (denoted as $\hat{\mathcal{F}}^{fps}$), from which we sample a set of frame-wise center tokens $\hat{\mathcal{F}}_c^{fps} \in \mathbb{R}^{N_k \times C}$ located at the 2D center of these feature maps. Next, the transformer-like module takes $\mathcal{F}^{fps}$ and $\hat{\mathcal{F}}_c^{fps}$ as input and sequentially performs L_m stacked self- and cross-attention computations (see Fig. 3 top). The process can be formulated as

$$L_m \times \begin{cases} \mathcal{F}^{fps} = \text{SelfAttn}(F^{fps}) \in \mathbb{R}^{N_k \times L \times C} \\ \mathcal{F}^{fps} = \text{Reshape}(\mathcal{F}^{fps}) \in \mathbb{R}^{1 \times N_k L \times C} \\ \mathcal{F}^{fps} = \text{CrossAttn}(\mathcal{F}^{fps}, \hat{\mathcal{F}}_c^{fps}) \\ \mathcal{F}^{fps} = \text{SelfAttn}(\mathcal{F}^{fps}) \in \mathbb{R}^{1 \times N_k L \times C} \\ \mathcal{F}^{fps} = \text{Reshape}(\mathcal{F}^{fps}) \in \mathbb{R}^{N_k \times L \times C} \end{cases}, \quad (1)$$

where L_m is the depth of the module. The transformed $\mathcal{F}^{fps}$ is then fed into the mask decoding head (two 3×3 convolutional layers followed by an upsampling layer) to produce the keyframe segmentation masks. Finally, we extract the object-aware semantic feature tokens $\mathcal{F}^{obj}$ from the keyframe feature maps $\hat{\mathcal{F}}^{fps}$ using the predicted masks.

Rotation-Aware Representation Encoding. This step focuses on extracting the 3D object rotation-aware embedding vectors from reference images, allowing GS-Pose to estimate an initial pose through template retrieval. To achieve this, we first adopt an Obj-Segmenter to segment the object from each reference image and then utilize a Rotation-Aware Encoder (RA-Encoder) to extract an image-level embedding vector from the segmented image (see Fig. 2 middle). Obj-Segmenter includes the DINOv2 backbone, a transformer-like module, and a mask decoding head (identical to the one in Co-Segmenter). Concretely, Obj-Segmenter first extracts the DINOv2 feature tokens $F_i^{ref} \in \mathbb{R}^{L \times C}$ from the ith reference image. Then, the

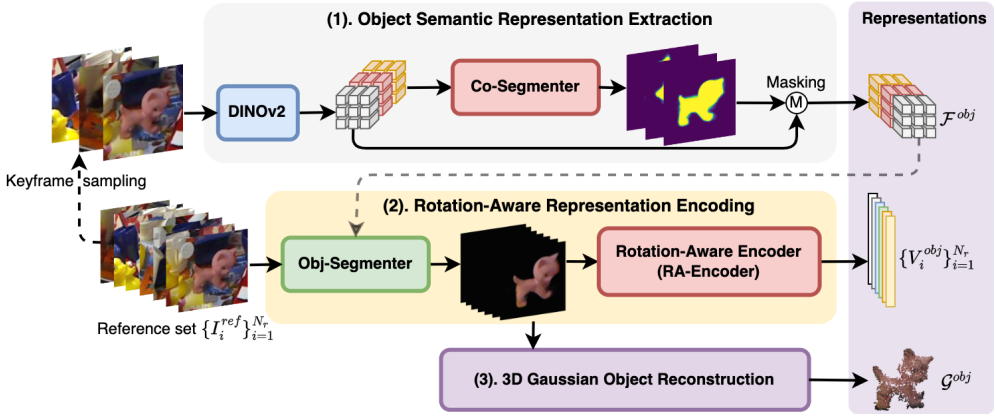

Figure 2. **Overview of the reference database creation process.** We begin by selecting a group of keyframes from reference images. (1). These keyframes are processed through DINOv2 and Co-Segmenter to jointly predict object segmentation masks, which are then utilized to extract the object semantic tokens ($\mathcal{F}^{obj}$) from the keyframe features. (2). Image-wise object segmentation is performed for all reference images $\{I_i^{ref}\}_{i=1}^{N_r}$ using an Obj-Segmenter with the obtained semantic information $\mathcal{F}^{obj}$. We then employ an RA-Encoder to extract the embedding vectors $\{V_i^{obj}\}_{i=1}^{N_r}$ from the segmented images. (3). Finally, we create a 3D Gaussian Object representation $\mathcal{G}^{obj}$ (viewed as a 3D point cloud for simplicity) using all segmented images with the known poses.

image feature tokens (F_i^{ref}) along with the object semantic tokens ($\mathcal{F}^{obj}$) are fed into the transformer-like module to perform L_m stacked self- and cross-attention computations (see Fig. 3 middle). This process can be formulated as

$$L_m \times \begin{cases} F_i^{ref} = \text{SelfAttn}(F_i^{ref}) \\ F_i^{ref} = \text{CrossAttn}(F_i^{ref}, \mathcal{F}^{obj}) \end{cases} . \quad (2)$$

Subsequently, the mask decoding head is utilized to produce a segmentation mask M_i^{ref} from the transformed image features F_i^{ref}. Finally, we extract an image-level representation vector $V_i^{ref} \in \mathbb{R}^{64}$ from the segmented image using RA-Encoder. RA-Encoder includes the DINOv2 backbone, four 3×3 convolutional layers with stride 2, a generalized average pooling layer, and a fully connected layer with an output dimension of 64 (see Fig. 3 bottom).

3D Gaussian Object Reconstruction. The last step is to create the 3DGO representation $\mathcal{G}^{obj}$ for pose refinement (see Fig. 2 bottom). 3D Gaussian Splatting [20] represents a 3D structure as a set of 3D Gaussians. Each 3D Gaussian is parameterized with a 3D coordinate $\mu \in \mathbb{R}^3$, a 3D rotation quaternion $r \in \mathbb{R}^4$, a scale vector $s \in \mathbb{R}^3$, an opacity factor $\alpha \in \mathbb{R}$, and spherical harmonics coefficients $h \in \mathbb{R}^k$, where k is the degrees of freedom. Consequently, the 3DGO model is represented as $\mathcal{G}^{obj} = \{\mu_i, r_i, s_i, \alpha_i, h_i\}_{i=1}^{U}$, where U is the number of 3D Gaussians. All segmented reference images with the known poses are utilized to build this 3DGO model. We kindly refer to [20] for more details.

3.2. Object Pose Inference

This section outlines the inference pipeline of GS-Pose, a cascaded process consisting of three core components.

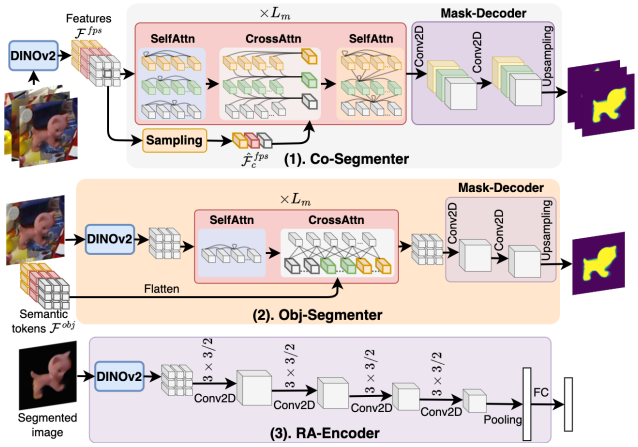

Figure 3. **(1). Co-Segmenter** includes a transformer-like module and a mask decoder to produce the co-segmentation masks. **(2). Obj-Segmenter** consists of the DINOv2 backbone, a transformer-like module, and a mask decoder to predict the object mask. **(3). RA-Encoder** contains the DINOv2 backbone, four 3×3 2D convolutional (Conv2D) layers with stride 2, a generalizable average pooling layer, and a fully connected (FC) layer.

Firstly, GS-Pose employs an object detector for detection. Secondly, GS-Pose obtains an initial pose using a pose estimator based on the detection. Finally, a 3D Gaussian Splatting-based pose refinement module (GS-Refiner) is adopted to optimize the initial pose. Fig. 4 illustrates these components, and we describe each one in detail below.

Detector. We leverage a segmentation-based detector to localize the target object (see Fig. 4 top). The detector consists of an Obj-Segmenter (as described in Sec. 3.1) and a

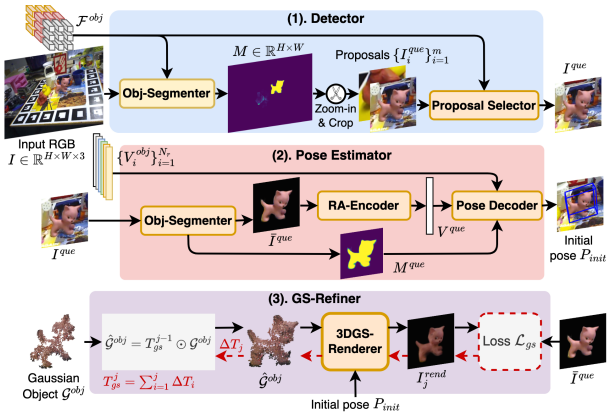

Figure 4. **(1). Detector** first employs an Obj-Segmenter to produce a mask from the input image using the semantic information ($\mathcal{F}^{obj}$). Then, connected components are computed from the predicted mask to generate proposals, which are further processed by a proposal selector to determine the final output. **(2). Pose Estimator** utilizes an Obj-Segmenter to predict an object mask M^{que} ($\mathcal{F}^{obj}$ is omitted for clarity). An embedding vector V^{que} is then extracted from the segmented image using RA-Encoder, followed by a pose decoder for estimating an initial pose (P_{init}) using both V^{que} and M^{que}. **(3). GS-Refiner** starts by applying an optimizable transformation T_{gs}^{j-1} to the 3D coordinates of the 3D Gaussian Object (3DGO) $\mathcal{G}^{obj}$, where $j \geq 1$ is the refinement step. Then, the 3D Gaussian Splatting-based renderer (3DGS-Renderer) generates an RGB image (I_j^{rend}) using the initial pose (P_{init}) and the transformed 3DGO ($\hat{\mathcal{G}}^{obj}$). Finally, the gradient ΔT_i is used to update the transformation parameter T_{gs}^j, minimizing the difference ($\mathcal{L}_{gs}$) between the rendered and the segmented images.

proposal selector. Specifically, given an input image, we first apply Obj-Segmenter to predict a segmentation mask, from which we generate a set of mask proposals $\{M_i^{que}\}_{i=1}^m$ by finding the connected components, where m represents the number of proposals. Subsequently, a set of object-centric RGB images $\{I_i^{que}\}_{i=1}^m$ are cropped from the input image using the 2D bounding boxes derived from these mask proposals. Next, we feed these RGB images into the proposal selector to obtain the final detection result. Within the proposal selector, we first extract the DINOv2 feature tokens $\{F_i^{que} \in \mathbb{R}^{L\times C}\}_{i=1}^m$ from these cropped images and then compute the image-level cosine similarities between these image features and the object semantic representation $\mathcal{F}^{obj}$. We select the one with the highest similarity score as the output, denoted as I^{que}.

Pose Estimator. In the second stage, we estimate an initial pose using a template retrieval-based pose estimator (see Fig. 4 middle). This pose estimator is comprised of an Obj-Segmenter (identical to the one in Detector), an RA-Encoder (as described in Sec. 3.1), and a pose decoder. We first obtain a segmentation mask M^{que} using Obj-Segmenter as well as an image-level representation vector $V^{que} \in \mathbb{R}^{64}$ using RA-Encoder from the detection. We then input M^{que} and V^{que} into the pose decoder to compute an initial 6D pose $P_{init} = [R_{init}, t_{init}]$. More specifically, the pose decoder first computes the cosine similarity scores $\{c_i = ||V^{que}|| \cdot ||V_i^{obj}||\}_{i=1}^{N_r}$ between the query vector V^{que} and the reference vectors within the set of 3D object rotation-aware representations $\{V_i^{obj}\}_{i=1}^{N_r}$. Consequently, the reference rotation matrix R_j^{ref} with the highest similarity score is retrieved as the initial 3D rotation estimate (R_{init}), where j denotes the index of the closest reference template. We then analytically infer the initial 3D translation t_{init} using the query mask (M^{que}) and the jth reference mask (M_j^{ref}). Specifically, we calculate a relative scale factor $\delta_s \in \mathbb{R}$ and a relative 2D center offset ratio $\Delta_{xy} \in \mathbb{R}^2$ between M^{que} and M_j^{ref} as follows:

$$\begin{cases} \delta_s & = \sqrt{Area(M^{que})/Area(M_j^{ref})}, \\ \Delta_{xy} & = (C_{bbox}(M^{que}) - C_{bbox}(M_j^{ref}))/S, \end{cases} \tag{3}$$

where $Area(M) = \sum_{j=0}^{S}\sum_{i=0}^{S} M_{[i,j]}$ denotes the mask area, S is the mask scale, and $C_{bbox}(M)$ denotes the 2D center of the bounding box tightly surrounding the mask M. We then compute the distance $t_z^{que} \in \mathbb{R}$ and the 2D center $P_{xy}^{que} \in \mathbb{R}^2$ of the object in the query image by

$$t_z^{que} = t_z^{ref} S/\delta_s/S_{box}^{que}, \quad P_{xy}^{que} = S_{box}^{que}\Delta_{xy} + C_{box}^{que}, \tag{4}$$

where t_z^{ref} is the pre-computed z-axis distance of the object in the reference image (more details provided in the supplementary materials), C_{box}^{que} and S_{box}^{que} are the 2D center and scale of the 2D object bounding box predicted from the original input image. Finally, we obtain the initial translation estimate by

$$t_{init} = t_z^{que} K^{-1} \bar{P}_{xy}^{que},$$

where $\bar{P}_{xy}^{que} \in \mathbb{R}^3$ is the homogeneous form of P_{xy}^{que}, and K denotes the camera intrinsic matrix.

GS-Refiner. The initial pose estimate is further refined by leveraging the 3D object representation $\mathcal{G}^{obj}$ through an iterative render-and-compare optimization procedure. This pose refinement stage, termed GS-Refiner, utilizes differentiable 3D Gaussian Splatting-based rendering [20], which facilitates the optimization of a learnable transformation T_{gs} to minimize the discrepancy between the rendered object and the observed query image. Formally, the optimal transformation T_{gs}^* is obtained by minimizing the following objective:

$$T_{gs}^* = \arg\min_{T_{gs}} \mathcal{L}_{gs}(\mathcal{R}_{gs}(T_{gs} \odot \mathcal{G}^{obj}, P_{init}), \bar{I}^{que}). \tag{5}$$

Here, $\mathcal{R}_{gs}$ denotes the differentiable rendering function, $T_{gs} \in SE(3)$ represents the learnable transformation parameters, $\odot$ indicates applying a rigid transformation to the

3D coordinates of $\mathcal{G}^{obj}$, and $\bar{I}^{que}$ is the segmented query image. The loss function $\mathcal{L}_{gs}$ is defined as a combination of the losses based on the image structural similarity (SSIM) and Multi-Scale SSIM [56]:

$$\mathcal{L}_{gs} = \mathcal{L}_{D-SSIM} + \mathcal{L}_{D-MSSIM} \tag{6}$$

The optimization is initialized with an identity transformation and iteratively updates T^*_{gs} using the AdamW optimizer [31] with the cosine annealing learning rate schedule, starting from 5×10^{-3} over a maximum of N_{gs} iterations with 10 warm-up steps. An early-stopping strategy is employed when the refinement loss converges to a predefined threshold η. The final refined pose is obtained as $P = P_{init}T^*_{gs}$.

3.3. Training Objective Functions

We employ the Binary Cross Entropy (BCE) loss to train both Co-Segmenter ($\mathcal{L}_{coseg}$) and Obj-Segmenter ($\mathcal{L}_{objseg}$) for pixel-wise segmentation prediction, *i.e.*,

$$\mathcal{L}_{coseg} = \mathcal{L}_{BCE}(\mathcal{M}, \bar{\mathcal{M}}), \quad \mathcal{L}_{objseg} = \mathcal{L}_{BCE}(M, \bar{M}), \tag{7}$$

where $\mathcal{M}$ and $\bar{\mathcal{M}}$ separately denote the predicted and ground truth group-wise segmentation masks, M and $\bar{M}$ are the predicted and ground-truth frame-wise segmentation masks, respectively. Additionally, we adopt the Negative Log-Likelihood (NLL) loss to train RA-Encoder for learning the 3D object rotation-aware representation, defined as:

$$\mathcal{L}_{rot} = -log\frac{exp(||V^{que}|| \cdot ||V_p||/\tau)}{\sum_{j=1}^{N_s} exp(||V^{que}|| \cdot ||V_j||/\tau)}, \tag{8}$$

where N_s is the number of the reference samples in a batch, V^{que} and V_j are the representation vectors of the query and the jth reference samples, respectively, τ is the temperature, and p is the index of the positive training sample determined by measuring the geodesic distance of the 3D rotation matrices, calculated as:

$$p = \underset{0 \le j \le N_s}{\arg\min} \ \arccos \frac{trace(R^{que}R_j^T) - 1}{2}, \tag{9}$$

where R^{que} is the ground truth rotation matrix of the query sample, and R_j is for the jth training sample. Consequently, the entire network is optimized through a combined loss in an end-to-end manner,

$$\mathcal{L}_{total} = \lambda_c\mathcal{L}_{coseg} + \lambda_o\mathcal{L}_{objseg} + \lambda_r\mathcal{L}_{rot}, \tag{10}$$

where $\lambda_{\{c,o,r\}}$ represent the balance weights.

4. Experiments

Datasets. We utilize the synthetic MegaPose dataset [22] for training and the real-world datasets LINEMOD [17] and OnePose-LowTexture [15] for evaluation. The MegaPose dataset was generated using BlendProc [11] and 1000 diverse objects from the Google Scanned Objects dataset [12] and includes one million synthetic RGB images. The LINEMOD dataset [17] contains 13 objects and is commonly used for 6D object pose evaluation. Following [15, 30, 39, 51], the training split of LINEMOD is selected as reference data, while the testing split is used for evaluation. OnePose-LowTexture [15] is a challenging dataset with low-texture or texture-less objects, from which eight scanned objects are utilized for evaluation. Each object was captured by two video sequences with different backgrounds. We follow OnePose++ [15] and select the first video as the reference and the other as query data.

Baseline Methods. For comparison, we assess GS-Pose against several state-of-the-art methods: Gen6D [30], Cas6D [39], OnePose [51], OnePose++ [15], and MFOS [23]. They take RGB reference images of novel objects with known poses as input to define the object coordinate system and then estimate the 6D pose of these objects from query images without retraining the network parameters.

Metrics. We adopt the widely used ADD [17] metric that measures the average distance between 3D points after being transformed by the ground truth and predicted poses. The ADDS metric is used for symmetric objects, which measures the average distance to the closest point instead of the ground truth point. Following the protocol[17], we report the average recall rate of ADD(S) within 10% of the object diameter, denoted as **ADD(S)@0.1d**. We also compute the 2D projection errors of the points after being transformed by the ground truth and predicted poses. We report the average recall rate within 5 pixels, denoted as **Proj@5pix**. In addition, the mean Average Precision (**mAP**)[0.5:0.95](%) [29] is reported for evaluating the 2D object localization performance.

Configurations. In our experiments, we set the hyperparameters: $N_k = 8$, $N_{gs} = 400$, $L_m = 4$, $\eta = 1 \times 10^{-4}$, $\tau = 0.1$, $\lambda_c = 1$, $\lambda_r = 1$, $\lambda_o = 1$, $N_s = 32$, unless

Method	Type	cat	duck	bvise	cam	driller	Avg.
SRPN-P [25]	BBox	11.85	1.62	18.94	2.44	8.91	8.76
SRPN [25]	BBox	9.72	4.56	22.47	13.43	10.97	12.23
SRPN-D [25]	BBox	22.97	1.85	49.14	17.76	18.89	22.12
OSOP [47]	BBox	32.10	34.81	26.68	24.33	21.36	27.86
Gen6D [30]	BBox	76.99	42.15	63.33	72.92	48.78	60.83
Cas6D [39]	BBox	79.46	67.44	66.32	76.39	59.35	69.79
LocPoseNet [60]	BBox	81.68	61.80	**79.45**	**80.50**	68.31	74.35
GS-Pose (ours)	Mask	69.14	80.07	66.46	75.51	73.08	72.85
GS-Pose (ours)	BBox	**84.44**	**86.88**	71.76	79.04	**80.60**	**80.54**

Table 1. Quantitative results of the 2D object localization on **LINEMOD [17]** regarding the mAP@[0.5:0.95](%) metric. "Type" indicates the detection type in the form of either bounding boxes or segmentation masks. GS-Pose derives the minimum 2D object bounding box from the mask prediction for comparison. We highlight the best in **Bold**.

Method	YOLOv5	ape	bwise	cam	can	cat	driller	duck	ebox*	glue*	holep.	iron	lamp	phone	Avg.
		ADD(S)@0.1d													
Gen6D [30]		-	62.1	45.6	-	40.9	48.8	16.2	-	-	-	-	-	-	-
Gen6D [30]†		-	77.0	66.7	-	60.7	67.4	40.5	98.3	87.8	-	-	89.8	-	-
Cas6D [39]†		-	86.3	70.1	-	60.6	84.8	51.3	98.8	88.5	-	-	93.4	-	-
OSOP [47]		26.1	55.6	36.2	52.2	42.5	49.6	22.2	72.4	52.3	18.6	72.3	27.9	39.6	43.6
OnePose [51]	✓	11.8	92.6	88.1	77.2	47.9	74.5	34.2	71.3	37.5	54.9	89.2	87.6	60.6	63.6
OnePose++ [15]	✓	31.2	97.3	88.0	89.2	70.4	92.5	42.3	**99.7**	48.0	69.7	97.4	97.8	76.0	76.9
MFOS [23]	✓	47.2	73.5	87.5	85.7	80.2	92.4	60.8	99.6	69.7	**93.5**	82.4	95.8	51.0	78.4
PoseMatcher [4]	✓	59.2	98.1	93.4	96.0	88.0	**98.4**	54.1	97.8	91.5	73.4	97.9	98.1	**92.1**	87.5
GS-Pose (ours)		59.6	99.6	96.0	97.6	**88.9**	95.1	74.9	99.3	92.2	86.8	98.2	96.7	80.7	89.7
GS-Pose (ours)	✓	**71.0**	**99.8**	**98.2**	**97.7**	86.7	96.2	**77.2**	99.6	**98.4**	87.4	**99.2**	**98.9**	85.0	**92.0**
		Proj@5pix													
Gen6D [30]†		-	82.5	90.8	-	96.1	72.4	79.7	97.8	96.2	-	-	91.6	-	-
Cas6D [39]†		-	93.4	96.3	-	**99.0**	**95.0**	93.5	98.3	**98.8**	-	-	96.9	-	-
OnePose [51]	✓	35.2	94.4	96.8	87.4	77.2	76.0	73.0	89.9	55.1	79.1	92.4	88.9	69.4	78.1
OnePose++ [15]	✓	97.3	**99.6**	**99.6**	**99.2**	98.7	93.1	97.7	**98.7**	51.8	98.6	98.9	**98.8**	**94.5**	94.3
GS-Pose (ours)		77.5	98.9	98.4	97.6	97.6	92.3	97.7	97.3	91.3	96.5	98.9	90.9	91.9	94.4
GS-Pose (ours)	✓	**97.9**	98.9	99.1	97.6	98.9	93.7	**97.8**	97.1	97.4	**98.8**	**99.6**	94.2	93.8	**97.3**

Table 2. Quantitative results on **LINEMOD [17]** regarding the ADD(S)@0.1d and Proj@5pix metrics. ✓ indicates using the external YOLOv5 [53] as the object detector. * indicates symmetric objects. † indicates that the method includes a subset of objects of LINEMOD as training data. We highlight the best in **bold**. "-" indicates unavailable results.

Method	GTBox	Toy.	Tea.	Cat.	Cam.	Shin.	Molie.	David	Marse.	Avg.
PVNet [43]		12.3	90.0	68.1	67.6	95.6	57.3	49.6	61.3	62.7
Gen6D [30]		55.5	40.0	70.0	42.2	62.7	16.6	15.8	8.1	38.9
OnePose [51]	✓	65.6	89.0	39.7	90.9	87.9	31.2	42.7	30.4	59.7
OnePose++ [15]	✓	**89.5**	**99.1**	97.2	**92.6**	98.5	79.5	**97.2**	57.6	88.9
GS-Pose $_{\text{init}}$		55.0	75.7	82.6	69.7	95.1	63.4	65.7	57.5	70.6
GS-Pose (ours)		89.3	86.7	**100.0**	90.2	**99.3**	**95.9**	91.7	**83.6**	**92.1**

Table 3. Quantitative results on each object in **OnePose-LowTexure [15]** regarding the ADD(S)@0.1d metric. "$_{\text{init}}$" indicates the initial pose estimation results of GS-Pose. "GTBox" indicates the ground truth 2D object bounding boxes. We highlight the best in **bold**.

otherwise specified. We use the AdamW [31] solver with the cosine annealing learning rate schedule, starting from 1×10^{-4} to 1×10^{-6}, to train our framework for 100,000 steps on an Nvidia RTX3090 GPU with batch size 2.

4.1. Object Detection

Experiment Setups. Given a set of object-centric RGB images as reference data, the task is to localize the object of interest in query images without fine-tuning the model parameters.

Results on LINEMOD. We report the quantitative results of the 2D object detection on LINEMOD [17] regarding the mAP@[0.5:0.95](%) metric in Table 1. We primarily compare GS-Pose against Gen6D [30], Cas6D [39], and LocPoseNet [60], which are the most similar works. Overall, GS-Pose achieves 72.85% mAP and 80.54% mAP in terms of 2D segmentation masks and the mask-induced 2D bounding boxes, respectively. Our segmentation-based detection approach outperforms all baseline methods. It is worth noting that all methods include a subset of held-out objects in LINEMOD as training data and evaluate on the other 5 selected objects, except OSOP [47] and GS-Pose.

4.2. Object Pose Estimation

Experiment Setups. Given a set of reference RGB images of a novel object with known poses, the task is to estimate the 6D pose of the object in query images without fine-tuning the network parameters. We conduct experiments under two settings: 1) pose estimation without pre-existing 2D bounding boxes and 2) pose estimation within pre-existing 2D bounding boxes. The latter involves estimating the pose from cropped object-centric images acquired either using the YOLOv5 detector [53] (Table 2) or by projecting the 3D object bounding boxes using ground truth poses (Table 3).

Results on LINEMOD. Table 2 shows the quantitative results in terms of the ADD(S)@0.1d and Proj@5pix metrics. Overall, GS-Pose achieves impressive 89.7% ADD(S)@0.1d and 94.4% Proj@5pix recalls on average, outperforming all baseline approaches. When using the 2D detection results predicted by YOLOv5 [53], as in OnePose [51] and OnePose++ [15], GS-Pose further improves the ADD(S)@0.1d metric to 92.0% and Proj@5pix to 97.3%, setting new state-of-the-art performance on LINEMOD. This advantage is largely attributed to the

Variant	ADD(S) @0.1d
w/o proposal selector	88.95
w/o $\mathcal{L}_{D-SSIM}$	89.44
w/o $\mathcal{L}_{D-MSSIM}$	89.29
GS-Pose (ours)	**90.86**

Table 4. Ablation studies on the **LINEMOD** subset regarding different variants.

Method	Number of reference images (N_r)					
	8	16	32	64	128	All (∼180)
Gen6D [30]	-	29.07	49.41	-	-	62.45
Cas6D [39]	-	32.43	53.90	-	-	70.72
OnePose++ [15]	-	31.38	54.98	-	-	78.10
GS-Pose (ours)	49.39	**62.50**	**74.50**	85.81	89.00	**90.86**

Table 5. Results on the **LINEMOD** subset regarding the varying number of reference images in terms of the ADD(S)@0.1d metric.

Maximum refinement steps (N_{gs})	100	200	300	400	500
ADD(S)@0.1d	85.58	90.31	90.70	90.86	90.82
Runtime (ms)	851	936	950	958	966

Table 6. Results on the LINEMOD subset in terms of the ADD(S)@0.1d metric. The refinement process automatically terminates when the loss converges.

low-textured or symmetric objects (*e.g.*, *ape, duck, glue*), where the correspondence-based methods like OnePose [51], OnePose++[15], MFOS [23], and PoseMatcher [4] inherently struggle.

Results on OnePose-LowTexture. We further compare GS-Pose against the baselines [15, 30, 51] on OnePose-LowTexture [15]. In addition, we also include PVNet [43], which trains a single network per object using approximately 5000 rendered images. Table 3 reports the quantitative results regarding ADD(S)@0.1d and shows new state-of-the-art performance (92.1%) achieved by GS-Pose. The keypoint-based approach OnePose [51] obtains an average recall of 59.7%, which lags behind our initial result (70.6%) by about 10% and our refined result (92.1%) by over 30%. OnePose relies on local feature matching to establish the keypoint-based 2D-3D correspondences, making it unreliable for low-textured or texture-less objects in this dataset. To alleviate this, OnePose++ [15] employs the keypoint-free LoFTR [50] for feature matching and significantly improves the result to 88.9%. Even though OnePose++ necessitates ground-truth 2D object bounding boxes for evaluation, GS-Pose still outperforms it using our built-in detector. Compared to the object-specific pose estimator PVNet [43], GS-Pose outperforms it by a substantial margin.

4.3. Additional Experiments

We conduct additional experiments on the LINEMOD subset and report the results in Tab. 4, Tab. 5, and Tab. 6.

Ablation studies. To assess the efficacy of the connected component-based proposal selector in object detection, we remove it from our object detector and then utilize the 2D bounding box derived from the entire segmentation mask as the output. As a result, the ADD(S)@0.1d metric decreases by about 2%, indicating the proposal selector's efficacy. Besides, when either $\mathcal{L}_{D-SSIM}$ or $\mathcal{L}_{D-MSSIM}$ is removed from GS-Refiner, the performance decreases, indicating that both terms contribute positively to the pose refinement.

Number of reference images. GS-Pose consistently achieves better performance with more reference images. When using only 32 reference images, GS-Pose obtains 74.5% recall, already comparable to or even outperforming the results achieved by the baseline methods using all reference images.

Maximum refinement steps. As expected, GS-Pose achieves consistently better performance with more refinement steps, reaching saturation at up to 400 steps. The refinement process terminates when the loss converges, thus resulting in a nonlinear increase in runtime with more steps.

Runtime. GS-Pose takes about one second to process a single RGB image (with resolution 480×640) on a desktop with an AMD 835 Ryen 3970X CPU and an Nvidia RTX3090 GPU, in which ∼0.16s for object detection, ∼0.01s for pose initialization, and ∼0.96s for refinement. GS-Pose employs an iterative, gradient-based optimization process for pose refinement, which improves accuracy but at the cost of computational efficiency. In future work, we plan to explore more efficient optimization algorithms, such as the Levenberg-Marquardt algorithm, to accelerate the pose refinement process for GS-Pose.

5. Discussion and Conclusion

This work presents GS-Pose, an integrated framework for estimating the 6D pose of novel objects in RGB images. GS-Pose leverages multiple representations of newly added objects to facilitate cascaded subtasks: object detection, initial pose estimation, and pose refinement. GS-Pose is trained once using synthetic RGB images and evaluated on two real-world datasets, LINEMOD and OnePose-LowTexture. The experimental results demonstrate that GS-Pose achieves state-of-the-art performance on the benchmark datasets and shows promising generalizability to new datasets. However, objects with thin or slender structures may pose challenges for GS-Pose due to poor segmentation. Future work could be extending GS-Pose for 6D pose tracking of unseen objects.

6. Acknowledgment

The work was supported by the Academy of Finland projects 353139 and 362409. We also acknowledge CSC - IT Center for Science, Finland, for computational resources.

References

[1] 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. 2, 3

[2] Dingding Cai, Janne Heikkilä, and Esa Rahtu. Sc6d: Symmetry-agnostic and correspondence-free 6d object pose estimation. In *2022 International Conference on 3D Vision (3DV)*, pages 536–546. IEEE, 2022. 2

[3] Dingding Cai, Janne Heikkilä, and Esa Rahtu. Msda: Monocular self-supervised domain adaptation for 6d object pose estimation. In *Scandinavian Conference on Image Analysis*, pages 467–481. Springer, 2023. 2

[4] Pedro Castro and Tae-Kyun Kim. Posematcher: One-shot 6d object pose estimation by deep feature matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2148–2157, 2023. 7, 8

[5] Dengsheng Chen, Jun Li, Zheng Wang, and Kai Xu. Learning canonical shape space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11973–11982, 2020. 2

[6] Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2781–2790, 2022. 2

[7] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, Linlin Shen, and Ales Leonardis. Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1581–1590, 2021. 2

[8] Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. In *European Conference on Computer Vision*, pages 139–156. Springer, 2020. 2

[9] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *The international journal of robotics research*, 30(10):1284–1306, 2011. 1

[10] 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. 1

[11] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *arXiv preprint arXiv:1911.01911*, 2019. 6

[12] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Michael Hickman, Krista Reymann, Thomas Barlow McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560, 2022. 6

[13] Rasmus Laurvig Haugaard and Anders Glent Buch. Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. *arXiv preprint arXiv:2111.13489*, 2021. 2

[14] 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. 3

[15] Xingyi He, Jiaming Sun, Yuang Wang, Di Huang, Hujun Bao, and Xiaowei Zhou. Onepose++: Keypoint-free one-shot object pose estimation without cad models. *Advances in Neural Information Processing Systems*, 35:35103–35115, 2022. 1, 2, 3, 6, 7, 8

[16] 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. 2

[17] 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 *Asian conference on computer vision*, pages 548–562. Springer, 2012. 2, 6, 7

[18] 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. 2

[19] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M Kitani. Repose: Fast 6d object pose refinement via deep texture rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3303–3312, 2021. 2

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 1, 3, 4, 5

[21] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020. 2

[22] 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. *arXiv preprint arXiv:2212.06870*, 2022. 6

[23] JongMin Lee, Yohann Cabon, Romain Brégier, Sungjoo Yoo, and Jerome Revaud. Mfos: Model-free & one-shot object pose estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2911–2919, 2024. 6, 7, 8

[24] 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, 2009. 2

[25] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8971–8980, 2018. 3, 6

[26] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018. 2

[27] 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. 2

[28] Jiehong Lin, Lihua Liu, Dekun Lu, and Kui Jia. Sam-6d: Segment anything model meets zero-shot 6d object pose estimation. *arXiv preprint arXiv:2311.15707*, 2023. 1, 3

[29] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[30] 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. 1, 2, 3, 6, 7, 8

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6, 7

[32] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2015. 1

[33] Van Nguyen Nguyen, Yinlin Hu, Yang Xiao, Mathieu Salzmann, and Vincent Lepetit. Templates for 3d object pose estimation revisited: Generalization to new objects and robustness to occlusions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6771–6780, 2022. 3

[34] Van Nguyen Nguyen, Thibault Groueix, Mathieu Salzmann, and Vincent Lepetit. Gigapose: Fast and robust novel object pose estimation via one correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9903–9913, 2024. 2

[35] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 3

[36] Evin Pınar Örnek, Yann Labbé, Bugra Tekin, Lingni Ma, Cem Keskin, Christian Forster, and Tomas Hodan. Foundpose: Unseen object pose estimation with foundation features. *arXiv preprint arXiv:2311.18809*, 2023. 1, 3

[37] Evin Pınar Örnek, Yann Labbé, Bugra Tekin, Lingni Ma, Cem Keskin, Christian Forster, and Tomas Hodan. Foundpose: Unseen object pose estimation with foundation features. In *European Conference on Computer Vision*, pages 163–182. Springer, 2025. 2

[38] Anton Osokin, Denis Sumin, and Vasily Lomakin. Os2d: One-stage one-shot object detection by matching anchor features. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 635–652. Springer, 2020. 3

[39] Panwang Pan, Zhiwen Fan, Brandon Y Feng, Peihao Wang, Chenxin Li, and Zhangyang Wang. Learning to estimate 6dof pose from limited data: A few-shot, generalizable approach using rgb images. *arXiv preprint arXiv:2306.07598*, 2023. 2, 3, 6, 7, 8

[40] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[41] Keunhong Park, Arsalan Mousavian, Yu Xiang, and Dieter Fox. Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1, 3

[42] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019. 2

[43] 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*, 2020. 7, 8

[44] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 3

[45] 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. 3

[46] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015. 3

[47] Ivan Shugurov, Fu Li, Benjamin Busam, and Slobodan Ilic. Osop: A multi-stage one shot object pose estimation framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6835–6844, 2022. 1, 2, 3, 6, 7

[48] Yongzhi Su, Jason Rambach, Nareg Minaskan, Paul Lesur, Alain Pagani, and Didier Stricker. Deep multi-state object pose estimation for augmented reality assembly. In *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pages 222–227. IEEE, 2019. 1

[49] 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. *arXiv preprint arXiv:2203.09418*, 2022. 2

[50] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 8

[51] 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. 1, 2, 3, 6, 7, 8

[52] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *The European Conference on Computer Vision (ECCV)*, 2018. 2

[53] Ultralytics. Yolov5: Real-time object detection, 2023. 7

[54] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. GDR-Net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16611–16621, 2021. 2

[55] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019. 2

[56] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, pages 1398–1402. Ieee, 2003. 6

[57] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024. 2

[58] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018. 2

[59] Yang Xiao, Xuchong Qiu, Pierre-Alain Langlois, Mathieu Aubry, and Renaud Marlet. Pose from shape: Deep pose estimation for arbitrary 3d objects. *arXiv preprint arXiv:1906.05105*, 2019. 2

[60] Chen Zhao, Yinlin Hu, and Mathieu Salzmann. Locposenet: Robust location prior for unseen object pose estimation. *arXiv preprint arXiv:2211.16290*, 2022. 3, 6, 7

[61] Yizhou Zhao, Xun Guo, and Yan Lu. Semantic-aligned fusion transformer for one-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7601–7611, 2022. 3