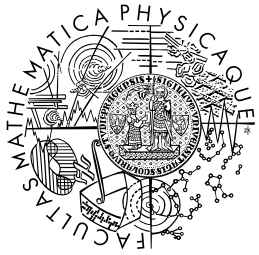

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Martin Cífka

6D Pose Estimation of Objects in Images

Department of Software and Computer Science Education

Supervisor of the master thesis: Dr. Ing. Josef Šivic

Study programme: Computer Science

Study branch: Visual Computing and Game Development

Prague 2024

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

I would like to express my sincere gratitude to my supervisor Dr. Ing. Josef Šivic and my advisor Ing. Vladimír Petrík Ph.D. for their valuable expertise, guidance, and patience during my work on this thesis, and for giving me the opportunity to participate in this field of research.

Next, I would like to thank Georgy Ponimatkin, Yann Labbé, and Méderic Fourmy for their advice during my work, and also all my colleagues from the Czech Institute of Informatics, Robotics and Cyberbetics (CIIRC CTU) for the possibility of being part of their friendly research environment.

Finally, I would like to thank my family for supporting me during my (sometimes winding) academic path that led me to this point.

Title: 6D Pose Estimation of Objects in Images

Author: Martin Cífka

Department: Department of Software and Computer Science Education

Supervisor: Dr. Ing. Josef Šivic

Abstract: The 6D pose estimation is an important computer vision task with applications in robotics, *e.g.* for manipulation or grasping, but also in computer graphics and augmented reality. Given an image, the task is to estimate the 3D rotation and 3D translation of the known object with respect to the camera. The task is even more challenging in an uncontrolled environment, *e.g.* when we do not have proper camera calibration. In that case, the focal length also needs to be estimated with the 6D pose. In this work, we address the issues of methods that work in such uncontrolled environments.

First, we focus on FocalPose, a state-of-the-art method for joint estimation of object 6D pose and camera focal length. We review the method and propose several improvements. These include (i) re-deriving and improving the 6D pose and focal length update rule, (ii) replacing the model retrieval method, and (iii) changing the distribution of 6D poses and focal lengths used for synthetic training data rendering. These changes lead to improved results compared to the state-of-the-art FocalPose method.

Second, to avoid often costly retraining of models for 6D pose estaimation, it is beneficial to consider methods with the ability to generalize to novel objects that have not been seen during training. These methods require a 2D bounding box and an object identity from a 3D object database to be known at inference time. Thus, a model-based method for novel object detection is also required. In particular, we investigate CNOS, a method for model-based segmentation of novel objects, that is widely used to solve the detection for 6D pose estimation. We provide an evaluation of average recall and average precision in its two stages: the proposal stage, which uses the Segment Anything Model to generate candidate object locations, and the matching stage, which uses DINOv2 feature similarity to assign confidence scores and object identities. We analyze the strengths and weaknesses of CNOS, and discuss the potential space for improvements.

Keywords: computer vision, 6D pose estimation, uncalibrated camera, novel object detection, visual recognition, deep learning

Contents

Introduction

Motivation

The object pose estimation is one of the classic problems dating back to the early days of computer vision [1, 2, 3]. The goal of the task is to estimate the pose of objects visible in the image, *i.e.* 3D rotation, and 3D translation, thus also referred to as *6D pose estimation*. It has many important applications; for example, an accurate pose of the object is essential in robot manipulation [4], grasping [5], guiding a robot using video demonstration [6], or object tracking [7]. Other applications include image overlaying in augmented reality, or 3D compositing in computer graphics [8, 9]. Typically, the 6D pose estimation requires knowledge of the 3D models of observed objects, as well as a calibrated camera. The task can be, however, considered also in an uncontrolled environment, where we may not know the camera calibration matrix and thus need to estimate not only the 6D pose, but also other parameters such as the camera focal length. This setting then allows inference on any image, *e.g.* image downloaded from the Internet. Another variant of the task is the scenario, where 3D models of observed objects are known only at inference time, but not during training, *i.e. 6D pose estimation of novel objects*. As training is typically costly and requires hours or days, avoiding retraining may be crucial in some applications.

Methods for solving all these problems typically also require known 2D locations (bounding boxes) of the objects in the images, and also the object identities (the specific models from the 3D object database). Object pose estimation is therefore highly interconnected with object detection, another important computer vision task, where the goal is to predict the 2D bounding box of an object in the image. To fully benefit from 6D pose estimation of novel objects, one must also consider the object detection in the same setup, requiring the method to be able to detect an object not seen during training, but where the 3D model of the object is available at test time.

Goals and contributions

The goal of this thesis is two-fold. First, we look into methods for joint estimation of object 6D pose and camera focal length. Here, the goal is to investigate the limitations of such methods and implement an improved method that addresses these limitations. In particular, we build on the FocalPose [10] method and introduce our version that results in better performance. We (i) re-derive and improve the 6D pose and focal length update rule that is used to obtain new parameter estimates, (ii) replace the method for retrieving the 3D model of the object seen in the image, and (iii) change the distribution of synthetic data used for network training. Second, we look at model-based methods for the detection of novel objects. Here, the goal is to identify limitations of current state-of-the-art methods and to explore the possibilities for addressing them. We focus on CNOS [11], a novel model-based method method which uses the Segment Anything Model [12] to generate object location proposals in the first stage and DINOv2 [13] to match the location proposals with the object 3D models in the

Figure 1: **Typical model-based 6D pose estimation pipeline.** Given an image of an object (left) and a database of 3D models, the task is to estimate camera-to-object 3D rotation and 3D translation. A typical pipeline consists of two stages. In the first stage (middle), a bounding box of the object is obtained and the 3D model of the object is retrieved from the database. Then in the second stage (right), the 6D pose of the object is estimated.

second stage. To evaluate CNOS and identify its limitations, we perform several experiments while mainly focusing on average recall in the first stage of CNOS and average precision in the second. We also look at how the average recall drops between the stages and hypothesize the reasons.

Challenges

The estimation of the the object pose without camera focal length is challenging for many reasons. First, adding focal length as an additional parameter increases the complexity of the optimization problem, as there are now 7 degrees of freedom. Next, there is an inevitable ambiguity between the focal length and the distance from the camera, as they both affect the object size in the image – an increase in focal length results in a bigger object projection size. The only information that can be directly used to resolve this ambiguity is the subtle change in object appearance when using different focal lengths, and also potentially the context of the entire scene in the image. Also, with calibrated setup, we can expect relatively stable environment, such as similar lighting conditions, all objects of one category having the same color or texture, etc. However, this may not be true in an uncalibrated "in the wild" setup. For example, images downloaded from the Internet can have a wide variety in illumination, object color and texture, and even the shape can change slightly; *e.g.* one car model can be manufactured in many different colors and with small customizable features. However, there are even more technical problems, such as the lack of large datasets with high-quality annotations, resulting in the need for synthetic training datasets. When estimating the 6D pose of novel objects, the main challenge lies in the model's ability to generalize to these new objects. The 3D models can be reconstructed, or, for example, provided by manufacturer, and can have large variations in shape,

texture, and color. Similar challenges apply for the model-based detection of novel objects, as the 3D model is needed for inference. Also, the multistage pipeline of separate detection and then estimation of 6D pose is more susceptible to errors, as they can accumulate during the process. The accurate object detection is thus very important for correct pose estimation.

Outline

In the Chapter 1 we discuss the related work to object pose estimation and detection. Then in Chapter 2 we focus on the pose estimation with unknown focal length, namely the FocalPose [10] method, and present our improved version FocalPose++ [14]. We experimentally demonstrate the benefits of our improved method. In Chapter 3 we investigate the detection of novel objects, an important problem related to the 6D pose estimation of novel objects, and perform several experiments evaluating the state-of-the-art CNOS method [11], while separately looking at its two stages. We conclude our work in Chapter 4 and suggest potential future work.

Publications

The work reported in Chapter 2 has been submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence as an extended version of [10]. The preprint of the submission is available at [14]. The overall setup of the problem and the original FocalPose method [10] are reviewed in Sec. 2.1 of this thesis. The extensions developed as part of this work, forming the new method FocalPose++, are described in Sec. 2.2, and the experimental results validating these extensions are reported in Sec. 2.3 of this thesis.

The work described in Chapter 3 is going to contribute to a manuscript on 6D pose estimation of novel objects in preparation for ECCV 2024.

1. Related work

1.1 6D pose estimation

The object pose estimation, being one of the oldest problems in computer vision [3, 2, 1], has a long history. Diverse approaches have been used to solve various variants of the task. In one of its basic forms, *6D pose estimation of rigid objects from single-view RGB images*, methods have been successfully using 2D-3D correspondences for object pose estimation, while relying on hand-crafted image features. Recovering the pose of the object is then typically performed by PnP and RANSAC [15, 16], or ICP [17, 18] in the case of RGB-D data. These methods either use local invariant features [1, 19, 20, 21], or directly regress the pose via template matching [22]. However, the former approach has problems with texture-less objects, while the latter struggles with partially occluded objects. These methods have also been revisited with convolutional neural networks (CNNs), using 2D features and keypoints [23, 24, 25, 26, 27, 28, 29, 30] or directly by finding 2D-3D correspondences [31, 32, 33]. Many recent state-of-the-art methods rely on the *render-and-compare* strategy [34, 35, 36, 37, 33], which aligns pixels of the input image with a rendering of the observed object using CNN. In this methodology, iterative refinement has been shown to be very effective. In addition, CosyPose [37] uses a bundle adjustment technique to predict consistent object poses in *multi-view* scenarios.

The render-and-compare strategy is also adaptable to other variants of the 6D pose estimation problem. *E.g.* in case of *uncalibrated camera*, state-of-the-art FocalPose [10] can be used to jointly estimate the focal length of the camera along with the pose of the object. It builds on CosyPose [37] and extends it to also predict the focal length. In theory, camera calibration techniques [38, 39, 40, 41, 42, 43, 44, 45] can also be used to estimate both the focal length and the 6D pose. However, these require finding image correspondences in multiple images using structured object patterns [46, 15, 45, 43], or identifying lines or vanishing points [44, 47, 45], which makes it often impractical. Other methods establish 2D-3D correspondences and use them to estimate the camera model and pose of the object. Wang *et al.* [48] modify Faster R-CNN to predict dense correspondence maps and use them to estimate both object pose and focal length. GP2C [49] extends this approach into two stages, where the first stage estimates the initial focal length and dense correspondences, and the second stage solves the PnPf problem using non-differentiable geometric optimization. To make the model trainable end-to-end, GCVNet [50] replaces the PnPf solver with a differentiable approximation. In Chapter 2, we build on the state-of-the-art FocalPose and extend it by improving the 6D pose and focal length update rule, replacing the model retrieval method, and changing the distribution of 6D poses and focal lengths used for rendering of synthetic training data.

6D pose of novel objects is another important variant of the problem, in which the 3D models of the tested objects are available at inference time, but not seen during training. Some methods estimate only the 3D orientation of the object [51, 52, 53, 54, 55, 56], or work only with category-level novel objects [57, 58, 59, 60, 61, 62, 63], *i.e.* they can only generalize to novel objects within a known

category (*e.g.* a new 3D model of mug). Other methods [64, 65, 66] estimate the full 6D pose; [64] predicts 2D-2D correspondences between the input image and rendered templates, obtains 3D information from the templates, and then solves the PnP problem. Methods such as [65, 66] rely on depth information, which may not always be available. The render-and-compare strategy can also be used in this case. To generalize to novel objects, MegaPose [67] extends [37] by training on a large-scale synthetic dataset rendered using more than 20K 3D models, using a classification-based network as a coarse estimator, and rendering additional viewpoints as input to the refiner alignment neural network. The latter allows the network to infer the anchor point of a novel 3D model. Lastly, few recent methods [68, 69] have also used the matching of image patch features, *e.g.* FoundPose [68] is leveraging the performance of the DINOv2 [13] model to obtain dense correspondences, while GigaPose [69] trains its own ViT [70] feature extractors and combines local and patch features to estimate different parameters.

1.2 Object detection

When detecting *known* objects, region-based convolution neural networks (R-CNNs) became one of the most widely used methods. In early works [71], these methods were using region proposal methods such as sliding windows [72] or selective search [73], and separately evaluated each proposal using a convolutional network. This has been improved using RoIPool [74, 75] for better speed and accuracy. Faster R-CNN [76] further improved this by adding a region proposal network (RPN) that identifies potential object candidates, and MaskRCNN [77] added a separate branch for object mask prediction. YOLO [78] introduced a one-stage approach for object detection, running significantly faster than the two-stage approaches, however, the accuracy was lower. RetinaNet [79] found that this accuracy drop was due to the foreground-background imbalance, and solved this by introducing the "focal loss", modifying the standard cross-entropy loss to focus more on hard misclassified examples. With the introduction of transformer networks, Vision Transformers (ViT) [70] have also been used for object detection and segmentation [80, 81, 82].

Detection of novel objects has not received much attention until recently. Given an image, the task is to detect objects that have not been seen during training, but the 3D models are available at inference time. One of the closest research areas is class-agnostic instance detection and segmentation [83, 84, 85, 86, 87], where the task is to predict the bounding boxes or masks of *all* objects present in the image, but not their object identities. These networks have shown the ability to generalize to novel objects [88, 89]. Some other works have also approached tasks such as novel class discovery [90, 91]. In the area of novel object segmentation, UOIS-Net [92] is a two-stage detector commonly used in robotics. The first stage uses depth to estimate initial object masks, while the second stage refines them using RGB data. The RGB-D data are also used in [93], which learns feature embeddings and applies mean-shift clustering to discover and segment unseen objects. [94] avoids using depth information explicitly; however, it relies on information recovered from stereo images. Segment Anything Model (SAM) [12] introduced a powerful model for the new promptable segmentation task that is capable of segmenting any object in an RGB image using a user-given prompt,

but again without identifying the object model depicted in the image. To retrieve all objects in the image, a grid of points is used as the set of prompts. To obtain segmentation with object identities, ZeroPose [66] is using SAM to generate object proposals and then compares ImageBind features [95] of the cropped input image with features of pre-rendered object templates. CNOS [11] uses a similar approach, while improving the method by using DINOv2 [13] features, rendering photorealistic templates using BlenderProc [96], and also experimenting with FastSAM [97] for better run-time efficiency. In Chapter 3, we investigate the state-of-the-art CNOS method and evaluate its performance. We pay attention to the number of detections, the average recall, and the average precision separately in its two stages of generating object proposals and matching to object templates.

2. 6D pose estimation without camera calibration

In this chapter, we look at FocalPose [10], a state-of-the-art method for the 6D pose estimation without camera calibration. First, in Sec. 2.1 we recall the FocalPose method. Then in Sec. 2.2 we improve it by several modifications and evaluate this new approach in Sec. 2.3. Please note that this chapter is based on the results from our FocalPose++ [14] paper, an extended version of [10] that we submitted to IEEE Transactions and IEEE Transactions on Pattern Analysis and Machine Intelligence. The preprint of the paper is available online [14]. From the paper, we directly use figures 2.2, 2.3, 2.4, 2.5, and result tables 2.1, 2.2, 2.3, 2.4, 2.5, 2.6. The text of sections 2.2 and 2.3 is based on [14], but has been re-written.

2.1 Joint 6D pose and focal length estimation

In this section we recall the FocalPose [10] method. We present only crucial aspects of FocalPose required for understanding of our further modifications in the following sections. For other details such as loss function or training details, please refer to the original paper [10] or to our extended version [14].

FocalPose is a method for jointly estimating the 6D pose of an object and the camera focal length from a single RGB image of the object, given a known 3D model. It employs a *render-and-compare* strategy, where the parameters are iteratively refined by a neural network that compares the input image with the rendering of the object. This approach was originally introduced in DeepIM [34] for the 6D pose estimation task with a calibrated camera, and was further improved in CosyPose [37]. FocalPose builds on top of the CosyPose method and extends it with the ability to jointly estimate the focal length of the camera.

We now describe the FocalPose network architecture. Given an RGB image I, the 3D model $\mathcal{M}$ of the object, and the parameters $\theta^k = \{R^k, t^k, f^k\}$ (*i.e.* object 3D rotation R^k, translation t^k, and camera focal length f^k), our intention at iteration k is to estimate the new parameters θ^{k+1}. Using the 3D model, current parameters, and the renderer $\mathcal{R}$, FocalPose first creates a rendering $\mathcal{R}(\mathcal{M}, \theta^k)$ of the model. Then, the network F compares the rendered image with the input image I, and outputs the parameter updates $\Delta\theta_k$:

$$\Delta\theta_k = F(I, \mathcal{R}(\mathcal{M}, \theta^k)). \tag{2.1}$$

The predicted update $\Delta\theta_k$ consists of focal length update v_f^k, a translation update vector $[v_x^k, v_y^k, v_z^k]$, and two 3-vectors $v_{R,1}^k$ and $v_{R,2}^k$ for updating the rotation. However, the predicted values are not directly applied to the previous parameters estimates as simple additive or multiplicative factors. Instead, an update rule U is used to compute the new object pose and focal length:

$$\theta^{k+1} = U(\theta^k, \Delta\theta_k). \tag{2.2}$$

The update rule U is derived from the pinhole camera projection model considering the nonlinearities of the imaging process, and thus allowing for easier learning of the network F. The overview of the architecture is summarized in Fig. 2.1.

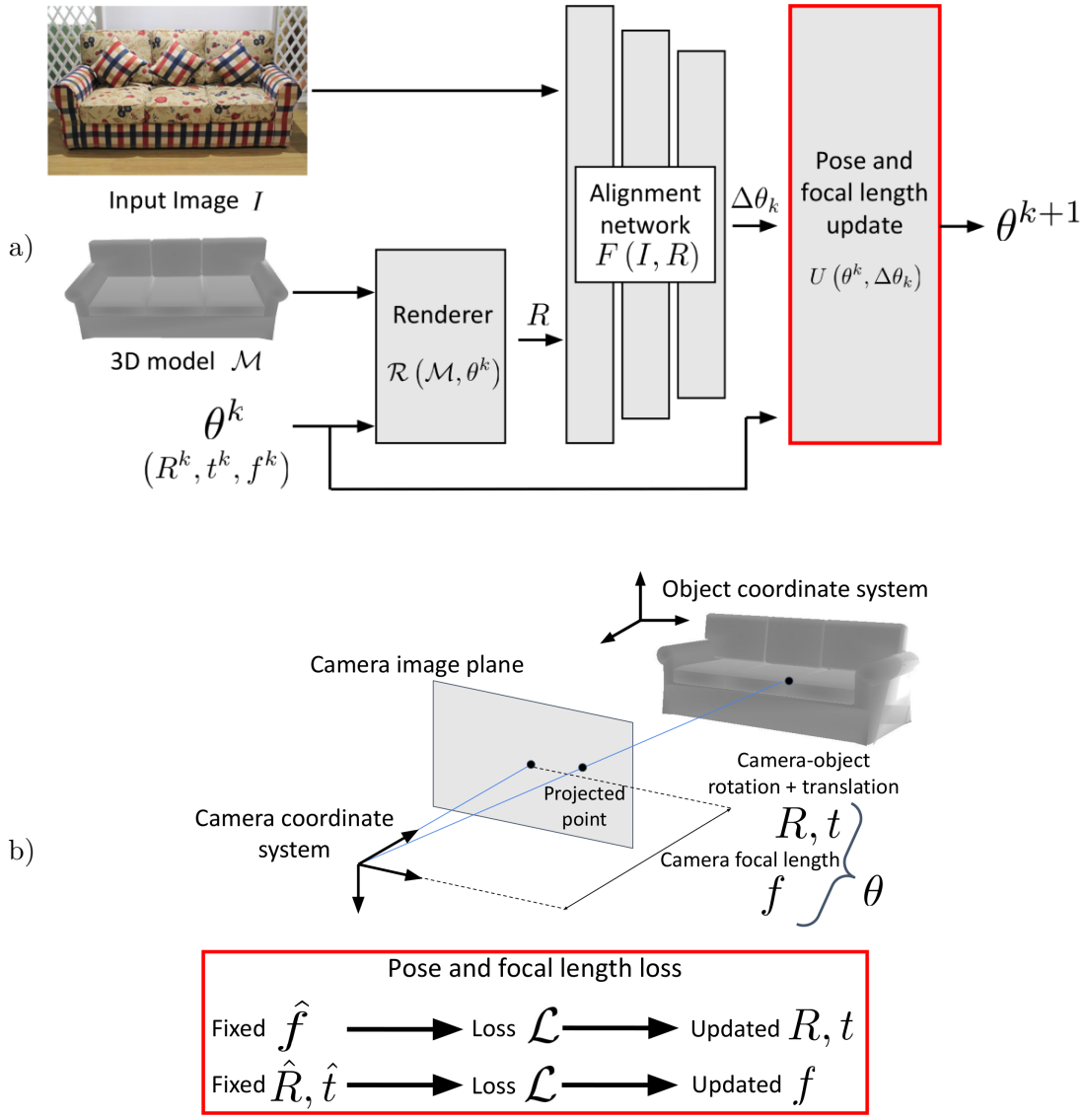

Figure 2.1: **FocalPose overview.** Figure from [10]. **a)** Given a 3D model $\mathcal{M}$ and previous parameter estimates θ^k (rotation R^k, translation t^k, and focal length f^k), a renderer R first renders an image $R(\mathcal{M}, \theta^k)$. Then, an alignment network F compares the rendering with an input RGB image I and predicts an update $\Delta\theta^k$. Finally, a non-linear update rule U is applied to the prediction $\Delta\theta^k$, obtaining new parameter estimates θ^{k+1}. **b)** Illustration of the image projection, given an object, its rotation R, translation t, and camera focal length f. The alignment network is trained using a novel loss function, disentangling the updates of the 6D pose and the focal length. When computing 6D pose loss, ground-truth value of the focal length is used and vice versa.

2.1.1 6D pose and focal length update rules

We now describe the update rule U used to update the focal length and the 6D pose of the object. FocalPose builds on the update rule from DeepIM [34] for 3D rotation and translation, and extends it with the update rule for focal length. For this purpose, a simplified pinhole camera model is assumed, with equal focal lengths for the x and y axes, *i.e.* $f_x^k = f_y^k = f^k$, and with the optical center set to the center of the image, *i.e.* $c_x = c_y = 0$. Also, we assume that the world coordinate system is placed at the object center.

Focal length update. The focal length must be strictly positive during the iterative process. This problem is solved by using an exponential function to the network output v_f^k, the new estimate is then obtained by multiplication with the focal length from the previous iteration:

$$f^{k+1} = e^{v_f^k} f^k. \tag{2.3}$$

6D pose update. When updating the 6D pose, we seek the new rotation R^{k+1}. FocalPose builds on the update rule from DeepIM [34], disentangling the update for rotation and translation. For rotation, the network F predicts two 3-vectors $v_{R,1}^k$ and $v_{R,2}^k$. FocalPose uses the two vectors to define a rotation matrix $R(v_{R,1}^k, v_{R,2}^k)$ using Gram-Schmidt orthogonalization. The resulting rotation matrix is then multiplied by the rotation matrix R^k from the previous iteration, resulting in a new rotation estimate:

$$R^{k+1} = R(v_{R,1}^k, v_{R,2}^k) R^k. \tag{2.4}$$

To obtain the updated translation $t^{k+1} = [x^{k+1}, y^{k+1}, z^{k+1}]$, the network F is trained such that output v_z^k represents the ratio of z-translations between the observed and rendered image:

$$z^{k+1} = v_z^k z^k. \tag{2.5}$$

For the x and y translations, the network is trained such that the outputs v_x^k, $v_y^k]$ represent the translation of the center of the object in the 2D image space, rather than the translation of the object in the 3D space. This results in the following update rule:

$$\begin{aligned} x^{k+1} &= \left(\frac{v_x^k}{f^{k+1}} + \frac{x^k}{z^k} \right) z^{k+1}, \\ y^{k+1} &= \left(\frac{v_y^k}{f^{k+1}} + \frac{y^k}{z^k} \right) z^{k+1}. \end{aligned} \tag{2.6}$$

For more details about the update rule, please refer to the Sec. 2.2.1, where we in detail explain and derive the update rule and introduce our improved version.

2.1.2 Full estimation pipeline

When estimating the object 6D pose and camera focal length, the alignment network F does not process the whole input image, instead it uses only the image cropped to the bounding box of an object. Also, we assume a known 3D model

of the detected object. Thus, before estimating the 6D pose and focal length, there are two additional stages. In the first stage, an object detector predicts the bounding box of the object. In the second stage, an instance classifier predicts the specific 3D model. Finally in the third step, given the cropped image, the network (iteratively) estimates the object 6D pose and camera focal length.

The first stage is the detection of objects using a Mask R-CNN [77] detector with pre-trained ResNet-50 [98] feature pyramid network backbone. The detector is trained using real images from the Pix3d/CompCars/StanfordCars datasets, while training a detector for each object class in the Pix3D dataset separately. The second stage uses finetuned DINO [99] model to predict an object instance from the cropped image. This model is trained again on real images separately for each object class. And the third stage of predicting the 6D pose and focal length actually uses two separately trained networks, coarse and refiner. Both networks use the same architecture, the ResNet-50 [98] backbone with prediction head for each output, but are trained separately. The coarse network is used only once for the prediction of rough parameter estimates, while the refiner is trained to repeatedly improve the parameters during several iterations. Both networks are trained using a novel loss function that disentangles the focal length and pose updates, *i.e.* the focal length is fixed to the ground-truth value when the loss for the 6D pose is computed and vice versa. See the FocalPose [10] paper or our extended version [14] for more details.

2.2 FocalPose++ improvements

We now describe our improvements of the FocalPose method. We make three modifications: First, in Sec. 2.2.1, we derive the translation part of the update rule and introduce our improved version. In Sec. 2.2.2, we change the distribution of the synthetic training data. In Sec. 2.2.3, we replace the model retrieval method. Then in Sec. 2.3 we evaluate our improved method, denoted as *FocalPose++*.

2.2.1 New update rule

In this section, we re-derive the translation equation of the 6D update rule, that is applied to the predictions of coarse and refiner alignment networks as presented in Sec. 2.1.1. However, we derive a new version of the update rule that also considers the change in focal length between iterations. In the end, we discuss the difference compared to the original update rule used in FocalPose [10].

For the derivation, we use a simplified pinhole camera model where the camera principal point is placed at the origin of the image coordinate system, *i.e.* $c_x = c_y = 0$. Also, we assume that focal lengths for x and y axes are equal, *i.e.* $f_x = f_y$. For simplicity, we place the world coordinate system in the center of the object. Then, using a standard camera projection equation in homogeneous coordinates, we project the center of the object $[x, y, z]$ coordinates in the camera frame to the point $[a, b]$ in the image coordinates:

$$\lambda \begin{pmatrix} a \\ b \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} . \tag{2.7}$$

This can be written as separate equations for x, y and z:

$$\begin{aligned} \lambda a &= fx\,, \\ \lambda b &= fy\,, \\ \lambda &= z\,, \end{aligned} \tag{2.8}$$

and by substitution of λ further simplified as:

$$\begin{aligned} a &= \frac{fx}{z}\,, \\ b &= \frac{fy}{z}\,. \end{aligned} \tag{2.9}$$

Let $[v_x^k, v_y^k, v_y^z]$ be the object translation update predicted by the alignment neural network F. We want v_z^k to represent the ratio of the camera-to-object depth between the object observed in the real image and the rendered image, thus we define the update rule for z^{k+1} as:

$$z^{k+1} = v_z^k z^k\,, \tag{2.10}$$

where z^k and z^{k+1} are the old and the new estimations of the z-translation respectively, and v_z^k is the corresponding network output for the z-translation at iteration k. Next, we want to define our update rule for x and y in such a way that the vector $[v_x^k,\, v_y^k]$ represents the translation correction of the projected object center measured in pixels. We can write this relationship as:

$$\begin{aligned} a^{k+1} &= a^k + v_x^k\,, \\ b^{k+1} &= b^k + v_y^k\,. \end{aligned} \tag{2.11}$$

To obtain the update rule, we substitute the image space coordinates a and b in eq. (2.11) by the expressions from eq. (2.9), resulting in:

$$\begin{aligned} \frac{f^{k+1}x^{k+1}}{z^{k+1}} &= \frac{f^k x^k}{z^k} + v_x^k\,, \\ \frac{f^{k+1}y^{k+1}}{z^{k+1}} &= \frac{f^k y^k}{z^k} + v_y^k\,. \end{aligned} \tag{2.12}$$

By rearranging the above equations, we derive the update rule equations for x^{k+1} and y^{k+1}:

$$\begin{aligned} x^{k+1} &= \left(\frac{v_x^k}{f^{k+1}} + \frac{f^k x^k}{f^{k+1} z^k}\right) z^{k+1}\,, \\ y^{k+1} &= \left(\frac{v_y^k}{f^{k+1}} + \frac{f^k y^k}{f^{k+1} z^k}\right) z^{k+1}\,, \end{aligned} \tag{2.13}$$

or equivalently with even more simplified expressions:

$$\begin{aligned} x^{k+1} &= \left(v_x^k + \frac{f^k x^k}{z^k}\right) \frac{z^{k+1}}{f^{k+1}}\,, \\ y^{k+1} &= \left(v_y^k + \frac{f^k y^k}{z^k}\right) \frac{z^{k+1}}{f^{k+1}}\,. \end{aligned} \tag{2.14}$$

Discussion. Our update rule in eq. 2.14 differs only slightly from the original FocalPose update rule for x and y in eq. (2.6). The right-hand sides of the equations are equivalent up to the multiplicative factor f^k/f^{k+1} for the terms x^k/z^k and y^k/z^k in parentheses. This can be seen more clearly from the less simplified eq. 2.13. From this point of view, the original update rule is an approximation of our new update rule: if the focal length does not change much between the iterations, the factor f^k/f^{k+1} will be close to 1 and could be ignored. In other words, the old translation update rule is equivalent to the one used in CosyPose [37] which considers focal length to be constant, while our update rule also incorporates the change of focal length between iterations. As shown in the ablation results (Sec. 2.3.4 and Table 2.6), the new update rule given by eq. (2.14) achieves slightly better performance.

2.2.2 Synthetic training data distributions

Existing image datasets with annotations for 6D object poses and camera focal lengths are notably limited in size. FocalPose [10] is evaluating its performance on 3 datasets – the CompCars [48], StanfordCars [48] and Pix3D [100] datasets (considering only *bed*, *chair*, *sofa* and *table* classes for Pix3D, and treating each class as separate dataset). Consequently, training models with such datasets is quite challenging. First, there are hundreds of different objects in the Pix3d-chair, CompCars, and StanfordCars datasets, and thus choosing an incorrect model can decrease the whole pipeline performance. Second, the available datasets have a very small number of training images. This is especially true for the Pix3D dataset, where each class contains only a few hundred images. To overcome this limitation, FocalPose trains the alignment neural network F using a combination of real and synthetically generated data. Synthetic data are created by randomly selecting the object model, the 6D pose, and the focal length, which are then utilized to render an image. Furthermore, the texture of the object is also randomized. To train the model, the synthetic dataset is then used together with the real dataset with 0.5% real-to-synth ratio. It is important to note that the object poses and focal lengths in the real data are not uniformly distributed; for instance, it is very rare to see objects like beds or cars being upside down. To respect this natural bias also in the synthetic dataset, we sample the object pose and focal length from a parametric distribution fitted to the real training data. This is in contrast to [10] which used a uniform distribution.

In detail, to model the distribution of 3D object rotations, we adopt the Bingham distribution [101], an antipodally symmetric probability distribution on the surface of a unit hypersphere, *i.e.* representing a 3D rotation as a unit quaternion. However, for any arbitrary unit quaternion q, both q and $-q$ represent the same rotation. Fortunately, the antipodal symmetry of the Bingham distribution solves this problem, making it suitable for characterizing the rotation distribution on the SO(3) group. Before rendering, we fit the distribution parameters to the real training dataset, *i.e.* orthogonal matrix $M \in \mathbb{R}^{4\times4}$ and diagonal matrix $Z = \text{diag}(z_1, z_2, z_3, 0) \in \mathbb{R}^{4\times4}$. During rendering, we sample from the distribution to obtain a random rotation. Both fitting and sampling are executed using the implementation from [102].

To represent the distribution of translations and focal length, we use two 2D

normal distributions. In particular, since both the focal length f and the z-translation affect the object scale after projection, we model the distribution of these two variables together. Furthermore, when we apply the logarithm function to both variables, they clearly conform to a 2D normal distribution. Thus, we fit a single 2D normal distribution to focal lengths and z-translations after applying the logarithm function, and apply the exponential function after sampling. The xy-translations exhibit properties of a 2D normal distribution, allowing us to fit the second normal distribution directly.

Fig. 2.2 visualizes 6D poses and focal lengths from the real Pix3D-sofa dataset together with our parametric distribution fitted to these real data, while keeping the same number of samples for both. Then in Sec. 2.3.2, we compare it with two other distributions (the uniform distribution used in [10], and our new nonparametric distribution) and demonstrate that introducing a bias towards the real dataset enhances the performance of the method.

2.2.3 Model retrieval method

After detecting an object in an image, we need to select the 3D model to which the detected object corresponds. As FocalPose trains Mask R-CNN for object detection, it could use its class prediction to retrieve the object ID. Instead, for better performance, it finetunes the DINO [99] model to retrieve the 3D model. In particular, it trains a simple logistic regression classifier on the DINO features extracted from the real training images cropped to the object bounding boxes. In addition, the classifier is trained five times with random initializations and soft voting is applied to obtain the object ID prediction. However, the accuracy of this method is only between 60 and 80% for the datasets on which it is evaluated (see Table 2.4). As the choice of the 3D model can potentially have a large effect on the performance of the whole pipeline, we replace it with a better architecture. Instead of training a classifier on DINO features, we train the ML-Decoder [103] model. In detail, the ML-Decoder is a classification head that replaces global average pooling in classification networks. It is attention-based, *i.e.* based on the classic transformer-decoder [104] architecture, however it introduces several modifications to improve scalability and efficiency. We use the network version with TResNet-L [105] as the backbone. Furthermore, we add 1000 synthetic images to each dataset to increase the performance of the model even more. We show the model retrieval ablation in Sec. 2.3.3. Note that further improving the model retrieval and the detector methods still has a potential to improve the performance of the entire FocalPose++ pipeline, as we show in Sec. 2.3.5.

2.3 Experiments

In this section, we first evaluate our improved FocalPose++ method, and then provide several ablations. In the last subsection, we provide an upper-bound for our method with ground-truth detections and object IDs. Following [10], we evaluate our method on the Pix3D [100] (*bed*, *chair*, *sofa* and *table* categories), CompCars [48] and StanfordCars [48] datasets using metrics for detection, rotation, translation, pose, focal length, and projection. All ablations are evaluated on the Pix3D-sofa dataset.

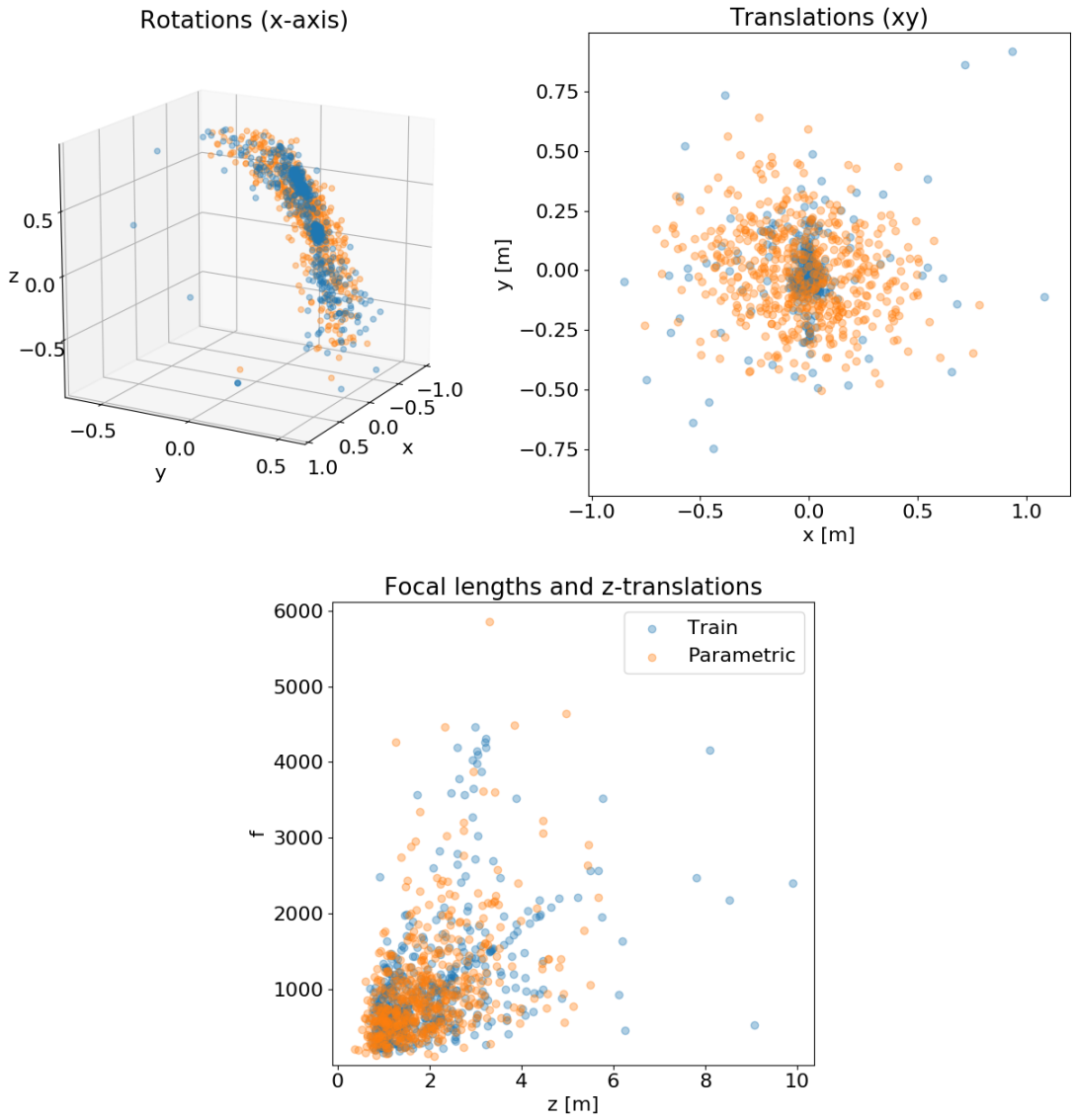

Figure 2.2: **Parametric distribution of the 6D poses and the focal lengths in the training data.** To visualize the parametric distribution, we plot the 6D poses and focal length from the Pix3D-sofa real training dataset (blue •) together with poses and focal length sampled from our distribution (orange •), while keeping the same number of samples for both. For rotation, we plot the unit x-axis vector multiplied by the sampled rotations. To visualize the translations and focal lengths, we keep the same approach as when sampling and plot the focal lengths with z-translations together, and the xy-translations in separate plot.

2.3.1 FocalPose++ Evaluation

We compare our method with FocalPose [10] and other state-of-the-art methods [48, 49, 50]. The results for the three datasets are reported in Table 2.1, and for the individual Pix3D classes in Table 2.2. We observe improved results with our modifications, validating our approach. In Fig. 2.3 and Fig. 2.4, we present qualitative results for Pix3D, CompCars and Stanford cars datasets. For additional qualitative results, refer to Sec. A.1 in the attachments.

Evaluation criteria. For the evaluation, we use the same metrics for rotation, translation, pose, focal length, and projection as FocalPose. We briefly describe the metrics here. For more details and exact equations, please refer to FocalPose [10] or to our extended version [14] of the paper.

- Rotation is evaluated using the rotation error $MedErr_R$ and the rotation accuracy $Acc_{R\frac{\pi}{6}}$. The error measures the geometric distance between the estimated and the ground-truth rotation, while the accuracy $Acc_{R\frac{\pi}{6}}$ reports the percentage of images with the rotation error below $\frac{\pi}{6}$.
- The translation error $MedErr_t$ measures the Euclidean distance between the estimated and the ground-truth translation vector. Furthermore, the translation error is normalized by the norm of the ground-truth translation.
- The pose error $MedErr_{R,t}$ is calculated as the average Euclidean distance between the 3D points of the object model transformed with the ground-truth and the estimated 6D pose. The pose error is normalized by the ratio between the diagonal of the ground-truth bounding box and the diagonal of the image.
- The focal length error $MedErr_f$ measures the difference between the ground truth and the estimated focal length in absolute value, normalized by the ground-truth focal length.
- Finally, projection metrics consist of the projection error $MedErr_P$ and the projection accuracy $Acc_{P_{0.1}}$. Here, $MedErr_P$ measures the average reprojection error of the points of the 3D model, *i.e.* the Euclidean distance of the points projected to the 2D image space using the estimated 6D pose and focal length, and the points projected using the ground-truth parameters. The projection error is normalized by the diagonal of the ground-truth bounding box. The projection accuracy $Acc_{P_{0.1}}$ reports the percentage of images in which the re-projection error is less than 0.1.

All error metrics are computed as median values for all images in the corresponding test dataset. Following FocalPose, we also report the detection accuracy $Acc_{D_{0.5}}$, *i.e.* the percentage of images where the intersection over union of the ground-truth and the predicted bounding box is above 0.5. However, as we did not change the detection method, we report the same values as FocalPose.

Pix3D dataset. For the Pix3D dataset, we report the average for the bed, char, sofa and table classes in Table 2.1 (top). We significantly outperform other methods [48, 49, 50] in 5 of the 8 metrics. The most significant improvement

is in the projection error (36% relative error reduction), translation error (30% reduction), focal length and pose errors (22% reduction), but also in rotation error (14% reduction). Compared to the original FocalPose [10], our method improves all metrics with the most notable difference in projection and rotation errors.

CompCars and Stanford cars. Similarly to the Pix3D dataset, we also observe improved results for the CompCars and StanfordCars datasets. The results are reported in Table 2.1 (middle, bottom). It is important to note that these two datasets differ from Pix3D, as they contain hundreds of distinct 3D models. However, they also contain more training images. Compared to [48, 49, 50] on StanfordCars, our method shows significant improvements in translation, pose, and focal length errors (almost a 50% relative error reduction). Furthermore, we achieve a 36% reduction in projection error and a 21% reduction in rotation error. On CompCars, we demonstrate improvements in projection error (30% relative error reduction), translation, pose, and focal length errors (approximately 20% reduction), along with a 10% reduction in rotation error. Once again, our method outperforms FocalPose [10] in all the reported metrics.

Per class results on the Pix3D dataset We present the per-class results of the Pix3D dataset (bed, chair, table, sofa) in Table 2.2. Once again, we outperform previous methods in 5 out of the 8 reported metrics when evaluating on the **bed**, **sofa**, and **table** classes, with relative error reductions ranging from 12% to 56%. Notably, for the **chair** class, we observe improvements primarily in rotation and projection. Interestingly, when compared to FocalPose [10], we perform slightly worse in translation, pose, and focal length errors, although we still outperform other methods by 3%-10%.

Limitations We examine the typical failure modes of [14] and validate whether the results change using our improved approach. We show the failure modes examples in Fig. 2.5. First, FocalPose typically fails on symmetric objects (such as tables and stools). Although these failures are often not apparent in the qualitative results, they significantly contribute to the rotation and projection error metrics. As our strategy does not account for object symmetries, we do not observe any improvement. Next, due to the iterative nature of the approach, we may get stuck in local minima during the refinement process. Although we did not explicitly attempt to address this issue, we observed improved results for our approach. We attribute this to our parametric distribution of synthetic training data, which enables our model to avoid uncommon object poses, such as beds and cars turned upside down, *etc*. The third common failure mode involves the incorrect identification of the object model. Utilizing the ML-Decoder [103] as the instance classifier significantly improves our ability to retrieve the correct model with higher accuracy. Nevertheless, we observe that the results do not change significantly concerning pose and focal length errors, as FocalPose demonstrates robust performance even when working with an approximate 3D model. For additional failure mode examples, refer to Sec. A.1 in the attachments.

Dataset	Method	**Detection**	**Rotation**		**Trans.**	**Pose**	**Focal**	**Projection**	
		$Acc_{D_{0.5}}$	$MedErr_R$ $\cdot 1$	$Acc_{R\frac{\pi}{6}}$	$MedErr_t$ $\cdot 10^1$	$MedErr_{R,t}$ $\cdot 10^1$	$MedErr_f$ $\cdot 10^1$	$MedErr_P$ $\cdot 10^2$	$Acc_{P_{0.1}}$
Pix3D	[48]	96.0%	7.25	87.8%	2.52	1.76	2.41	6.33	71.5%
	[49]-LF	96.2%	6.92	88.4%	1.85	1.30	1.72	3.85	85.5%
	[49]-BB	**97.7%**	6.89	**90.8%**	1.94	1.30	1.75	3.66	**88.0%**
	[10] FocalPose	95.5%	4.92	84.1%	1.49	1.09	1.53	2.97	79.2%
	[14] Ours	95.5%	**4.19**	85.1%	**1.31**	**0.99**	**1.34**	**2.34**	81.5%
CompCars	[48]	**98.9%**	5.24	97.6%	3.30	2.35	3.23	7.85	73.7%
	[49]-LF	98.8%	5.23	97.9%	2.61	1.86	2.97	4.21	95.1%
	[49]-BB	**98.9%**	4.87	98.1%	2.55	1.84	2.95	3.87	**95.7%**
	[50]-TwoStep	-	4.37	98.1%	3.22	1.90	3.79	4.54	90.2%
	[50]-GCVNet	-	3.99	98.4%	3.18	1.89	3.76	4.31	90.5%
	[10] FocalPose	98.2%	3.99	98.4%	2.35	1.67	2.65	2.95	93.0%
	[14] Ours	98.2%	**3.61**	**98.5%**	**1.96**	**1.44**	**2.37**	**2.70**	94.2%
StanfordCars	[48]	99.6%	5.43	98.0%	2.33	1.80	2.34	7.46	76.4%
	[49]-LF	**99.6%**	5.38	**98.3%**	1.93	1.51	2.01	3.72	96.2%
	[49]-BB	**99.6%**	5.24	**98.3%**	1.92	1.47	2.07	3.25	**96.5%**
	[50]-TwoStep	-	5.09	97.5%	2.29	1.52	2.52	3.78	93.6%
	[50]-GCVNet	-	4.92	97.5%	2.20	1.46	2.43	3.65	94.6%
	[10] FocalPose	99.5%	4.44	95.1%	1.00	0.84	1.09	2.55	93.8%
	[14] Ours	99.5%	**3.87**	96.2%	**0.94**	**0.76**	**1.04**	**2.07**	95.1%

Table 2.1: **Comparison with the state-of-the-art for 6D pose and focal length estimation** on the Pix3D, CompCars and Stanford cars datasets. The best result for each metric and dataset among directly comparable methods is highlighted in **bold**. Compared to other methods, our FocalPose++ approach clearly outperforms other methods [48, 49, 50] in all error metrics, *i.e.* in 5 of the 8 metrics reported on all three datasets. The relative reduction in the median error ranges from 10% to 50%, validating the FocalPose approach and our improvements.

2.3.2 Ablation of different synthetic data distributions

To render a single image for a synthetic dataset, we need to sample the 6D pose of the rendered object and the focal length of the camera. As synthetic datasets make up a large part of all training data, the choice of distribution can make a significant difference in overall performance. To evaluate the effect, we compare the results of models trained with synthetic data using the parametric distribution (employed in our approach) against two alternatives: a uniform distribution as used in [10], and a novel non-parametric distribution. We explain the details of these two alternatives in the following paragraphs. We keep the same ratio of synthetic and real data while training. Our intuition is that sampling from a distribution closer to the distribution of object poses and focal length in the real dataset can increase the performance. In Table 2.3 we show that both of our new distributions perform better. The parametric distribution works best, achieving significantly better results.

Class	Method	Detection	Rotation		Trans.	Pose	Focal	Projection	
		$Acc_{D_{0.5}}$	$MedErr_R$ $\cdot 1$	$Acc_{R\frac{\pi}{6}}$	$MedErr_t$ $\cdot 10^1$	$MedErr_{R,t}$ $\cdot 10^1$	$MedErr_f$ $\cdot 10^1$	$MedErr_P$ $\cdot 10^2$	$Acc_{P_{0.1}}$
bed	[48]	98.4%	5.82	95.3%	1.95	1.56	2.22	6.05	74.9%
	[49] LF	99.0%	5.13	96.3%	1.41	1.04	1.43	3.52	90.6%
	[49] BB	**99.5%**	5.40	**97.9%**	1.66	1.17	1.59	3.55	**93.2%**
	[10] FocalPose	98.4%	3.16	91.6%	1.28	0.93	1.28	1.91	88.9%
	[14] Ours	98.4%	**2.74**	93.2%	**1.15**	**0.78**	**1.21**	**1.53**	90.0%
chair	[48]	94.9%	7.52	88.0%	2.69	1.58	1.98	6.04	75.3%
	[49]-LF	95.2%	7.52	88.8%	1.92	1.21	1.62	3.41	88.2%
	[49]-BB	**97.3%**	6.95	**91.0%**	1.68	1.08	1.58	3.24	**90.9%**
	[10] FocalPose	91.8%	3.56	85.4%	**1.49**	**0.94**	**1.36**	1.73	79.3%
	[14] Ours	91.8%	**3.49**	87.5%	1.63	1.02	1.51	**1.69**	82.5%
sofa	[48]	96.5%	4.73	94.8%	2.28	1.62	2.42	4.33	82.2%
	[49] LF	96.5%	4.49	95.0%	1.92	1.33	1.79	2.56	93.7%
	[49] BB	**98.3%**	4.40	**97.0%**	1.63	1.16	1.73	2.13	**95.6%**
	[10] FocalPose	96.9%	2.98	97.6%	1.29	0.83	1.36	1.52	93.9%
	[14] Ours	96.9%	**2.77**	95.6%	**1.14**	**0.75**	**1.18**	**1.19**	95.4%
table	[48]	94.0%	10.94	72.9%	3.16	2.28	3.03	8.90	53.6%
	[49] LF	94.0%	10.53	73.5%	2.16	1.62	2.05	5.92	69.5%
	[49] BB	**95.7%**	10.80	**77.2%**	2.81	1.78	2.10	5.74	**72.4%**
	[10] FocalPose	94.9%	9.98	61.8%	1.90	1.68	2.13	6.72	54.7%
	[14] Ours	94.9%	**7.75**	64.1%	**1.33**	**1.42**	**1.45**	**4.95**	58.1%

Table 2.2: **Comparison with the state-of-the-art for 6D pose and focal length estimation** on the Pix3D dataset split by class. The best result for each metric and dataset among directly comparable methods is highlighted in **bold**.

Uniform Distribution. The object pose and focal length can be sampled by using a uniform distribution very easily. In our case, we sample the rotation uniformly in SO(3) space. The 3D position of the object is sampled uniformly in the interval $(0.0, 1.0)$ meters for x, $(-0.5, 0.5)$ meters for y. The interval for z is $(0.8, 3.0)$ meters for CompCars and Stanford cars datasets, and $(0.8, 2.4)$ meters for Pix3D. The camera focal length is sampled uniformly within (200, 1000) pixels, covering the range of focal lengths from all three datasets.

Nonparametric Distribution. We propose a nonparametric distribution as follows: We save all training data points, *i.e.* samples consisting of rotation matrix, translation vector and focal length, and estimate hyperparameters δ_R, δ_x, δ_y, δ_z and δ_f that determine maximum perturbation factors (described later in this paragraph). When sampling from the distribution, a random training data point is selected and perturbed. The rotation is adjusted by composing with a rotation given by a randomly sampled axis and a random angle $\alpha \in [0, \delta_R]$. Next, similarly as in the parametric distribution, we observe that z-translation and focal length are highly dependent; therefore, we entangle the perturbation of the focal length and the z-translation, and treat the xy-translation separately. For each of the two pairs of variables, we sample a random 2D vector within a unit circle and scale

Distribution	$MedErr_R$	$MedErr_t \cdot 10$	$MedErr_f \cdot 10$
a. Uniform	2.95	1.27	1.34
b. Nonparametric	2.82	**1.14**	1.24
c. Parametric	**2.77**	**1.14**	**1.18**

Table 2.3: **Ablation of different synthetic training data distributions on Pix3D sofa.** Each image in the synthetic training dataset contains one object rendered in a random 6D pose using a random focal length. We try three different distributions for sampling the 6D pose and the focal length and show that the use of distribution biased towards the real datasets (b. and c.) significantly improves model performance. The parametric distribution (c.) performs the best, while the uniform distribution (a.) used in [10] performs the worst.

the vector by factors (δ_z, δ_f) and (δ_x, δ_y) respectively. In other words, we sample a random vector within an ellipse with the axes determined by the aforementioned hyperparameter pairs. The obtained vector is then used as an additive perturbation factor for the selected sample. To estimate the hyperparameters (δ_x, δ_y), we first compute the Euclidean distance of each data point to its nearest neighbor (considering only x and y translation values of each sample) and then take the 95 percentile of all distances. We repeat the same procedure for the z-translations and the focal lengths to obtain (δ_z, δ_f). The hyper-parameter δ_R is computed in the same fashion, except that the angle between the two rotations is used to compute the distance.

2.3.3 Ablation of different model retrieval methods

FocalPose showed that its approach is reasonably effective, even when only a similar or approximate 3D model is used during inference. To verify how much this affects the performance of the full estimation pipeline, we replace the original classifier learned on top of the DINO [99] features with a better method. We use ML-Decoder [103], an attention-based classification head with a pre-trained TResNet-L [103] network as a backbone. While the DINO classifier was trained only on the real dataset, we added 1000 synthetic images to the training data. We measure the classification accuracy on all three datasets, and report the results for the previously used DINO classifier, ML-Decoder trained only on the real dataset, and separately ML-Decoder trained on data with the added synthetic images (denoted as "ML-Decoder+1k"). The results are reported in Table 2.4. Note that adding the synthetic data helps only in the case of Pix3D dataset that contains a very low number of real training images. Next, we run the evaluation for the full 6D pose estimation pipeline and report the results in Table 2.5. Although the accuracy with the ML-Decoder is significantly higher, the performance of the full estimation pipeline does not change much. This confirms that the FocalPose architecture is quite robust and works well, as long as the 3D model shapes are similar enough.

Dataset	Pix3D Acc	CompCars Acc	Stanford Acc
[99] DINO	62.1%	79.0%	71.2%
[103] ML-Decoder	72.8%	93.3%	**95.1%**
[103] ML-Decoder+1k	**77.6%**	**93.5%**	94.7%

Table 2.4: **Model retrieval accuracy.** Using classifier on DINO [99] features to retrieve an object model yields quite a low accuracy; therefore, we replace the classifier with ML-Decoder [103] using a pre-trained TResNet-L [105] network as the backbone. On all three datasets, the ML-Decoder significantly improves retrieval accuracy. To increase the accuracy even more, we add 1000 images from the synthetic datasets to the training data (denoted as "ML-Decoder+1k"). While on CompCars and Stanford cars datasets the accuracy does not change much with added data, it helps a lot in the case of the Pix3D dataset where the number of real training images is very low.

Retrieval Method	$MedErr_R$	$MedErr_t \cdot 10$	$MedErr_f \cdot 10$
[99] DINO	3.00	**1.13**	1.20
[103] ML-Decoder+1k	**2.77**	1.14	**1.18**

Table 2.5: **6D pose estimation with improved model retrieval on Pix3D sofa.** We evaluate our model with the DINO [99] model as the instance classifier and compare it with the ML-Decoder [103] classification head using a pre-trained TResNet-L [105] network as the backbone. We observe that although the retrieval accuracy is much better for the ML-Decoder (see Table 2.4), it has only a moderate effect when incorporated into the entire 6D pose estimation pipeline even when trained with added synthetic data ("ML-Decoder+1k").

2.3.4 Ablation of the update rule

Building on FocalPose [10], we refined the x and y components of the 6D pose update rule. Here, we compare the evaluation results with the new and the old update rule from [10]. As shown in Table 2.6, the new update rule achieves slightly better results. For a detailed derivation of the update rule and a discussion of the differences compared to [10], please refer to the Sec. 2.2.1.

2.3.5 Upper-bound with ground-truth detections and object instances

Finally, we evaluate our method using ground-truth detections and object instances, *i.e.* providing an upper bound to our pipeline. The results on the Pix3d-sofa dataset are shown in Fig. 2.3.5. We report results of our full estimation pipeline compared to the results using the ground truth bounding boxes, the ground truth object instances, and both combined. We also show results from the original FocalPose [10] as a reference. Although our improvements are significant, the results show that there is still potential for improvements using better detection and classification methods.

Update Rule	$MedErr_R$	$MedErr_t \cdot 10$	$MedErr_f \cdot 10$
[10] FocalPose	2.81	1.16	1.19
[14] Ours	**2.77**	**1.14**	**1.18**

Table 2.6: **Update rule ablation on Pix3D sofa.** When training our model with the new update rule, it performs slightly better compared to the original update rule used in [10].

Figure 2.3: **Pix3D qualitative results.** We show several qualitative examples of FocalPose++. In each row, we show one example with the input image (left), ground-truth object pose and focal length (center) and our prediction (right). For the ground-truth and our prediction, we show the input image overlayed with rendering of the object in corresponding object pose and camera focal length. Notice the precise alignment of objects when handling large perspective effects (rows 3, 5, 6). In some cases (row 8), our prediction is even better than the manual object annotation.

Figure 2.4: **Example qualitative results on the CompCars** (rows 1-5) **and Stanford cars** (rows 6-10) **datasets.**

Figure 2.5: **FocalPose main failure modes** are: (a) symmetric objects, (b) local minima, and (c) incorrect 3D models identified by the object detector.

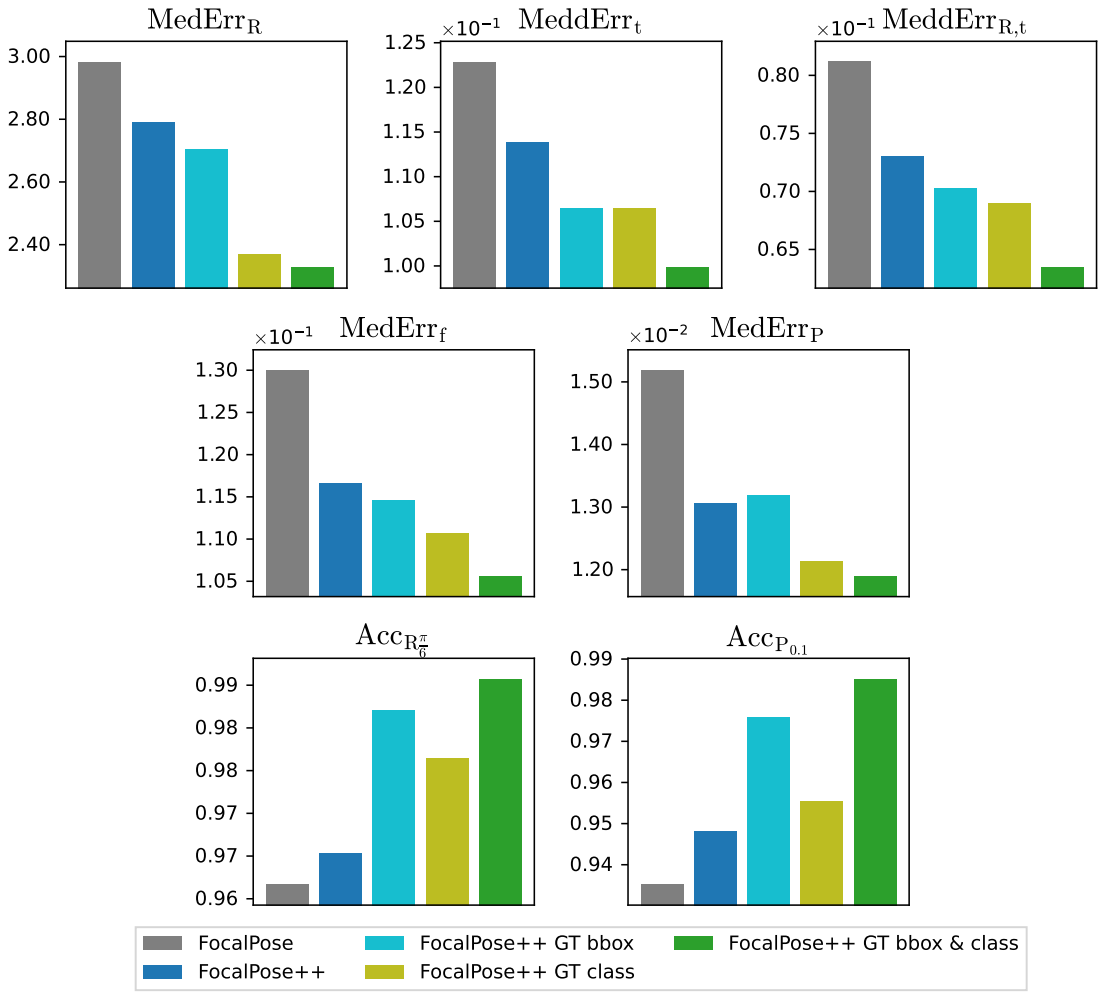

Figure 2.6: **FocalPose++ upper-bound with ground-truth detections and object instances.** Using ground-truth detections (*FocalPose++ GT bbox*), ground-truth classes (*FocalPose++ GT class*), and both combined (*FocalPose++ GT bbox & class*), we evaluate the performance of our FocalPose++ model on the Pix3D-sofa dataset. The results show that with better detection and classification methods, our method can achieve even better results in all reported metrics. First two rows show the performance on error metrics (lower is better), the third row reports rotation and projection accuracies (higher is better).

3. Detection for 6D pose estimation of unseen objects

One of the limitations of existing 6D pose estimation methods is their assumption of a known 3D model already during the training phase. However, in the case of novel objects, *i.e.* objects unseen during training, this results in model retraining, which is often costly and requires hours or days to generate synthetic data and train the model. For example, MegaPose [67], one of the current state-of-the-art methods, solves this as follows. First, the model is trained on a large-scale dataset of synthetic images rendered using more than 20K models. Second, a classification-based network is used for the coarse 6D pose estimation phase. And finally, the refiner renders multiple viewpoints during the refiner step, allowing the network to infer the anchor point of the 3D model. However, MegaPose [67], as well as other state-of-the-art methods [68, 106, 69, 107], requires as input the bounding box of the object and the identity of the 3D model depicted in the image. This highlights the importance of another computer vision problem, the *detection of unseen objects*, on which we will focus in this chapter. Given an RGB image and a set of 3D object models not seen during training, the task is to estimate the bounding boxes and object identities of all objects, corresponding to the given 3D models, visible in the image. State-of-the-art methods (CNOS [11], ZeroPose [66]) use the Segment Anything Model (SAM) [12] in the first stage to generate object proposals, and then select only relevant objects by measuring the similarity of the masked objects and rendered 3D model templates using visual features in the second stage.

Here, we focus on CNOS [11] and its two stages separately and perform several experiments to understand the strengths and weaknesses of these methods. For the first stage, we measure the average recall versus the number of detections using different SAM parameter settings. CNOS also experiments with FastSAM [97], a YOLOv8-based [108] promptable segmentation model that has better run-time efficiency. However, in our work we focus only on the CNOS version with the Segment Anything model [12]. For the second stage, we look at the scoring problems using DINOv2 [13] feature similarity, and show the scoring upper and lower bound. Also, we look at the difference between evaluation with and without object identities, and the difference between using modal or amodal bounding boxes, *i.e.* considering only the visible part of the object or the whole object with occluded areas for the bounding box. In addition, we measure the change in average precision and recall *between* the two stages. For evaluation, we use the BOP Toolkit from the BOP Challenge [109]. The toolkit reports several precision and recall metrics, following the setup of the COCO Object Detection Challenge [110]. This chapter is structured as follows. In Sec. 3.1 describe in detail the SAM and CNOS methods. Next, in Sec. 3.2 we explain the metrics and our evaluation setup. Finally, in Sec. 3.3 we conduct our experiments evaluating CNOS performance in its proposal and matching stages.

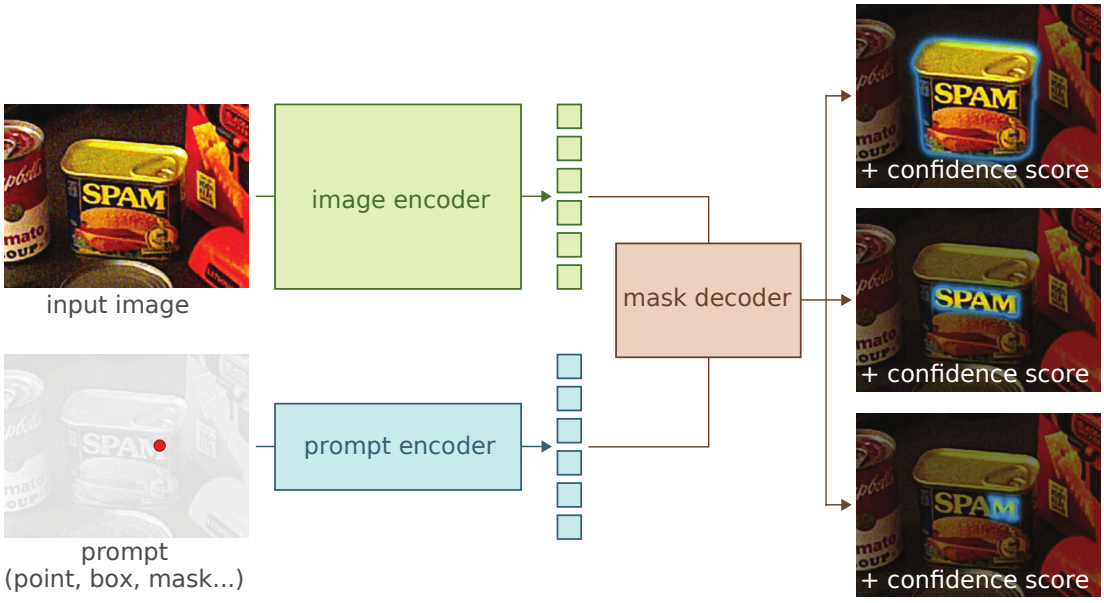

Figure 3.1: **Overview of Segment Anything Model.** Given an image and a prompt (2D point, bounding box, text description, mask *etc.*) as input, SAM provides multiple segmentation masks and corresponding confidence scores as output. First, an image embedding is obtained using *image encoder*, and prompt embedding using *prompt encoder*. Then, a *mask decoder* takes the two embeddings as an input, and returns multiple object segmentation masks with their confidence score. The model returns multiple outputs to deal with prompt ambiguity – *e.g.* a single point can refer either to whole object, its part, or its subpart.

3.1 Segment Anything Model and CNOS

In this section, we first explain the details of the Segment Anything Model [12] and then CNOS [11] (CAD-based Novel Object Segmentation), a method that uses SAM to generate object proposals.

3.1.1 Segment Anything Model (SAM)

SAM is a recent model that outputs multiple object masks given a user-given input prompt, *e.g.* a 2D point, bounding box, text description, or a mask. At a high level, SAM creates an image embedding using *image encoder*, prompt embedding using *prompt encoder*, and then uses both embeddings to obtain an object mask with *mask decoder*. In this promptable segmentation task, SAM aims to return a valid mask, given any prompt. The architecture is visualized in Fig. 3.1. The image encoder can be any network that produces an image embedding of the desired size. In particular, SAM uses an MAE [111] pre-trained Vision Transformer [70] that produces an embedding of size $64 \times 64 \times 256$. The prompt encoder differs according to the prompt type. For the *sparse prompts* (point, bounding box, and text), the embedding is a 256-dimensional vector. A point is represented as a positional encoding [112], while a bounding box is represented as a pair of positional encodings of the top-left and bottom-right corners of the bounding box. In addition, the encodings are summed with learned embeddings representing the specific prompt type, allowing us to distinguish between the simple point prompt

Figure 3.2: **Example of image segmentation using SAM.** Given an input image and a grid of point prompts (left), the Segment Anything Model retrieves the object mask for every point of the grid. The masks are then filtered by stability and confidence score, and duplicate masks are removed by NMS, resulting in the final segmentation of the image (right).

and the two corners of bounding box. For text prompts, the CLIP [113] text encoder is used. SAM also supports *dense prompts* (*e.g.* masks), which are processed with a convolution neural network that produces an output of the same size as the image embedding. The mask decoder is designed to be lightweight and fast, so a large set of prompts can be used to obtain segmentation of the entire image. The decoder takes the image embedding and the prompt embeddings and produces a fixed number of masks and confidence scores (*i.e.*, estimated IoUs) for each prompt. This is an important detail, as it deals with a prompt ambiguity – it may not be clear what is *the object* we are looking for, as a point may correspond to the whole object, its part, or only a subpart. The authors observed that 3 outputs are usually sufficient to cover most of the object parts.

In our work, we focus on the so-called *segment everything* mode, where a regular *grid of points* is used as a set of prompts. In this case, the image decoder runs only once, while the mask decoder runs once per each point. An example of such image segmentation is visualized in Fig. 3.2. By default, SAM uses a grid 32×32 that produces 3072 masks in total (three for each point). To obtain only high-quality masks, only *confident* and *stable* masks are selected. A mask is considered confident if the mask confidence score is above a given threshold, and stable if thresholding of the soft mask at $0.5 - \delta$ and $0.5 + \delta$ results in similar binary masks (*i.e.* intersection over union is above the given stability threshold). Finally, duplicate masks are removed by non-maximum suppression (NMS).

3.1.2 CNOS

To perform object detection, one must not only propose object locations, but also select only detection of objects that we are interested in. In addition, an object ID and confidence score need to be assigned to each detection. In our scenario, we are interested in model-based detection of novel objects, *i.e.* we know the 3D models of the objects we are looking for only during inference time. One of the state-of-the-art methods for this task is CNOS [11], which uses SAM to generate object proposals, and then compares the segmented objects with pre-rendered

images of the 3D models.

The method can be divided into an onboarding stage and two detection stages. In the onboarding stage, templates of 3D models are rendered from $V = 42$ viewpoints, defined by subdivision of a regular icosahedron. The detection phase is visualized in Fig. 3.3. In the first detection stage, the input image is segmented using SAM, specifically, by the "segment everything" mode using a grid of points, as described in the previous section. The parameters of SAM, such as the number of points, stability, confidence, and NMS thresholds, can be adjusted to obtain a suitable segmentation. Then in the second detection stage, DINOv2 [13] features are extracted from each segmented part of the image and then compared using cosine similarity with the features extracted from all $V = 42$ templates of each 3D model. This results in a tensor of size $V \times N_{\text{models}} \times N_{\text{detections}}$. This tensor is aggregated over the V templates, using the mean of the top-5 scores for each 3D model as an aggregation function, obtaining a $N_{\text{models}} \times N_{\text{detections}}$ similarity matrix. As a final step, the max and argmax functions are applied over the dimension with the different 3D models, yielding a score and object ID for each detection.

3.2 Experimental setup and evaluation metrics

As mentioned previously, we use one of the standard sets of metrics for object detection, although we slightly modify the settings for our experiments. Specifically, we use the BOP toolkit from BOP Challenge[109], a benchmark used for the evaluation of object detection and 6D pose estimation methods. The challenge follows the metrics of the COCO Object Detection Challenge [110] using `Pycocotools` Python package. However, the details of the metrics may not be clear at first glance and the information in the respective papers is quite sparse. We take this opportunity to explain the metrics in detail, as is needed for our experiments and discussion of our results. We pay attention only to the details of *object detection*, *i.e.* we do not consider metrics for the 6D pose estimation here. The BOP Challenge also includes an object segmentation task; however, from the object detection task, the evaluation differs only in the computation of intersection over union using segmentation masks instead of bounding boxes. Thus, we also leave out the details for this task.

3.2.1 Basic definitions

We recall some of the most important definitions and concepts in the object detection task here, considering a set of detection bounding boxes with assigned scores and object IDs, and a set of ground-truth bounding boxes with assigned object IDs.

Intersection over union. For each prediction, one must decide whether it was correct, *i.e.* the bounding box corresponds to some ground truth object. In classification tasks this is a straightforward problem, but in object detection this may not be so obvious as the boxes may be more or less aligned. To overcome this, the similarity between two bounding boxes is measured using the *intersection*

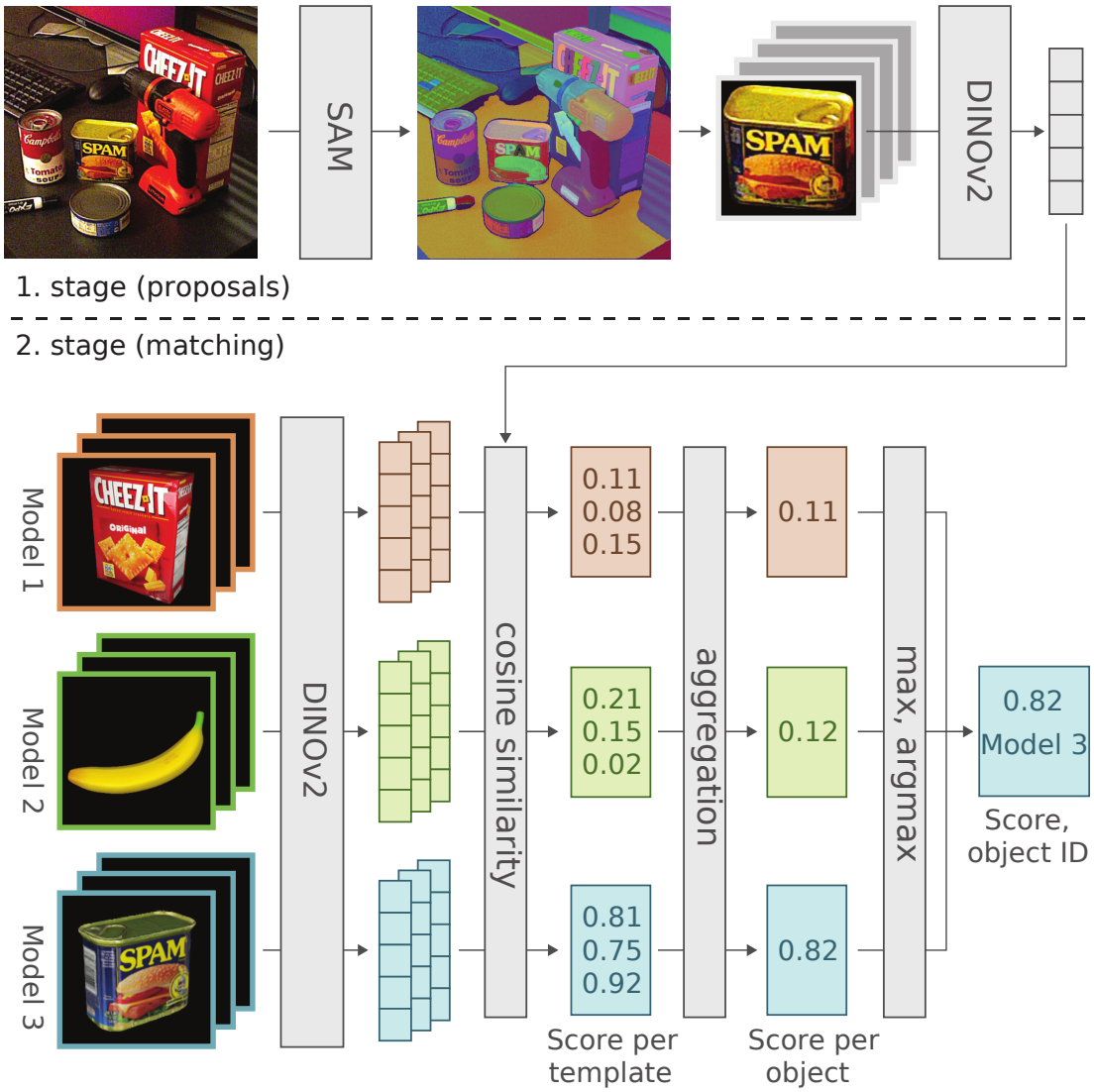

Figure 3.3: **Overview of CNOS.** CNOS can be divided into 2 stages. In the first stage, segmentation masks from SAM are used to generate object proposals. In the second stage, DINOv2 features are used to compare segmented objects with rendered object templates. Each detection is compared with all V templates of each 3D model using the cosine similarity of the extracted features. Then, the scores are aggregated across all V views (using the mean of top-5 scores) to get a score per model for each detection. Finally, max and argmax functions are applied to obtain a confidence score and ID of the best-matching 3D model.

over union (IoU):

$$\text{IoU}(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{3.1}$$

where A and B are the bounding boxes. Given an IoU threshold t, detection D is accepted as correct with respect to ground truth G, if $\text{IoU}(G, D) > t$, and if they have the same object ID.

True positive, true negative, false positive, false negative. After making the decision about the detections and wheter they are correct, we can distinguish between the following terms:

- *True positive (TP)* – a correct detection (has been matched to some ground-truth).
- *False positive (FP)* – an incorrect detection (has not been matched to any ground-truth), not corresponding to an object.
- *False negative (FN)* – an undetected object (ground-truth that has not been matched to any detection).
- *True negative (TN)* – not applicable for object detection, as it would correspond to anything that is not an object and has not been detected.

Precision and recall. Given an IoU threshold t and matching between ground truths and detections, we compute the number of true positives (TP), false positives (FP) and false negatives (FN) in all test images. Then, precision and recall are computed as:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{3.2}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{3.3}$$

I.e. precision is number of true positives divided by the number of all detections, and recall is the number of true positives divided by number of all ground-truth detections.

3.2.2 Detection matching

When evaluating predictions, the set of predicted detections has to be matched against the set of ground-truth detections. This is done for each object separately; *i.e.* a predicted detection cannot be matched to a ground-truth detection with a different object ID. Each detection can have assigned at most one matching ground-truth and vice versa. In general, given an IoU threshold t and ground-truth G, we want to find a detection D that satisfies $\text{IoU}(G, D) > t$ and has not been used so far. There may be multiple detections that satisfy this; therefore, we select the one with the highest score that has not yet been matched. If no detection satisfies the condition, the ground-truth remains unmatched and counts as a false negative.

3.2.3 Metrics

The BOP Toolkit reports several precision and recall metrics. They differ by selection of the IoU thresholds, the maximum number of detections, and the object area range considered. We describe these three parameters in the following paragraphs. Then in Table 3.1 we summarize all the BOP metrics.

Maximum detections. The number of detections in each image is limited by the number M. This is in fact a limit for each image and each object ID, *i.e.* in each image there can be at most M detections for each object. By default $M = 100$ is used, and in two other cases values of 1 and 10 are used for recall computation.

IoU threshold. Unless specified otherwise, BOP Toolkit evaluates metrics using 10 equally spaced IoU thresholds between 0.5 and 0.95 and averages the results. In two other cases, single thresholds of 0.5 and 0.75 are used for the average precision computation.

Area range. Depending on the selected area range, predicted and ground-truth detections that do not fit the specified area range are ignored. This can be used to evaluate how a method performs on objects of different sizes. The area can be either *small* (at most 1024 pixels), *medium* (between 1024 and 9216 pixels) or *large* (at least 9216 pixels), or we can consider *all* detection regardless of the area. However, the filtering of detections also depends on the matching process – if the predicted detection is matched to some ground-truth detection, it will be ignored if and only if the respective ground-truth is ignored, regardless of the detection area. The ignored ground-truth detections are used as the last option for matching, *i.e.* they will not be used unless there is no other option.

Average recall

For a given IoU threshold t, we compute the true positives (TP), false positives (FP), and false negatives (FN) considering all detections in all images. Then the recall for this threshold is computed as presented in eq. 3.3. When multiple IoU thresholds are specified or the test dataset contains multiple object identities, the recall is computed separately for each of them, and the AR metric is computed as the average of all the recall values. The average recall (between 0 and 1) indicates what portion of objects has been successfully retrieved.

Average precision

The computation of the average precision is more complicated. The reported AP metric is, in fact, an approximation of the area under the precision-recall (PR) curve. First, the detections are sorted by their score (regardless of the image in which they have been detected). Then, precision and recall are computed using top-k detections for k between 1 and the number of all detections. *I.e.* when testing on a dataset of N images, where each image has exactly M detections, we get $N \cdot M$ detections in total and thus the same number of precision and recall scores, computed using the first k detections for $k \in \{1, ..., N \cdot M\}$. These

Score	Score type	IoU threshold	Area range	Max. det.
AP	PR-AUC	0.5:0.95	all	100
AP_{50}	PR-AUC	0.5	all	100
AP_{75}	PR-AUC	0.75	all	100
AP_S	PR-AUC	0.5:0.95	small	100
AP_M	PR-AUC	0.5:0.95	medium	100
AP_L	PR-AUC	0.5:0.95	large	100
AR_1	recall	0.5:0.95	all	1
AR_{10}	recall	0.5:0.95	all	10
AR_{100}	recall	0.5:0.95	all	100
AR_S	recall	0.5:0.95	small	100
AR_M	recall	0.5:0.95	medium	100
AR_L	recall	0.5:0.95	large	100

Table 3.1: **An overview of the default BOP metrics.** The different average precision (area under the precision-recall curve – "PR-AUC") and average recall metrics differ by the IoU thresholds, the considered object area range, and the maximum number of detections per object in each image.

precision values are smoothed and plotted on a graph with their respective recall values. Smoothing replaces the precision at each point on the PR curve with the maximum precision of all points with the same or higher recall, resulting in monotonically decreasing PR curve. The final AP metric is then computed as an approximation of the area under the curve by taking an average of precision values from the PR curve at equally spaced recall thresholds between 0 and 1. This is illustrated in Fig. 3.4. The average precision thus summarizes both precision and recall in a single value between 0 and 1 (higher is better). However, the standard "precision" metric (also known as the positive predictive value) is not reported in the BOP metrics. For multiple IoU thresholds or multiple object identities, the same approach is used as when computing average recall: AP values are computed separately and then averaged across all objects and thresholds.

3.3 Experiments

As explained before, methods such as CNOS [11] follow the two-stage approach, where the first stage generates proposals and the second stage selects only relevant detections. We note that this two-stage distinction is very important with respect to recall and precision. In the first stage, the focus on recall is crucial, since any object not proposed in this stage cannot be detected later. Although the number of proposals is, in fact, not important in the first stage because recall does not penalize false positives, it is obvious that we want to keep it as low as possible. Having too many detections can become a problem in the second stage, where

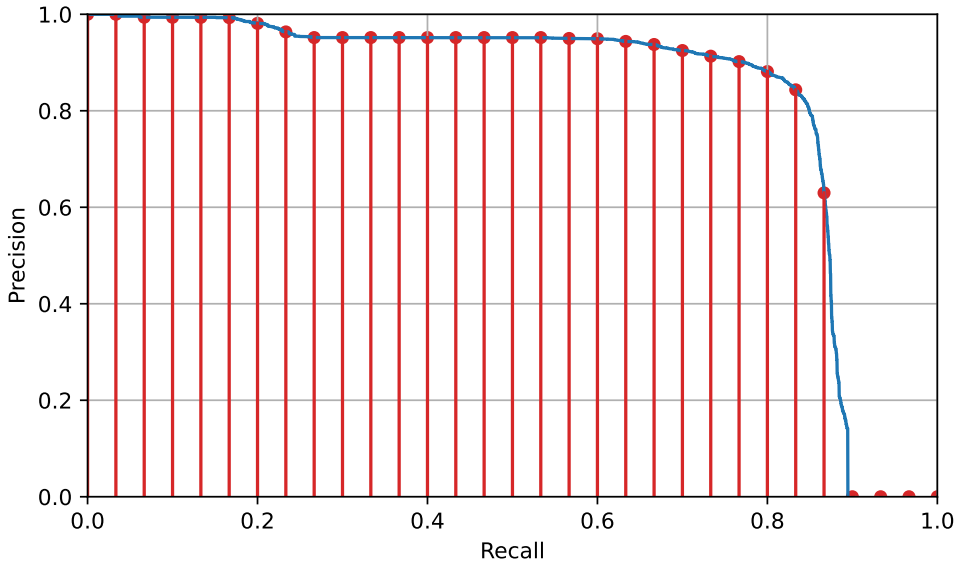

Figure 3.4: **Illustration of the AP metric calculation.** Given the PR curve (blue), the AP metric is computed as an approximation of the area under the curve. This is done by computing an average of precision values at equally spaced recall thresholds (red).

we want to select the best proposals by assigning them a high score and correct object ID. The purpose of the second stage is to increase the precision without losing too much recall at the same time. Too many detections can only increase the number of false positives and possibly reduce precision.

Keeping this in mind, we perform the following experiments. In Sec. 3.3.2 we evaluate the average recall of SAM proposals using different parameter settings, measure the average number of detections per image, and compare SAM with simple proposal baselines [73, 114]. In Sec. 3.3.3 we look at how the average precision and recall change between the two stages using different parameter settings. Sec. 3.3.4 evaluates the second stage of CNOS, *i.e.* the scoring using the similarity of DINOv2 [13] features, and provides the scoring upper and lower bounds. Then, in Sec. 3.3.5 and Sec. 3.3.6 we perform two more evaluation experiments. First, we show how the performance changes when also using the object identity predictions from CNOS. Second, we show the difference when using amodal instead of modal masks. We explain our evaluation setup in the following Sec. 3.3.1.

3.3.1 Evaluation setup

For our experiments, we use modified BOP [109] evaluation settings. First, since the SAM [12] model predicts only object masks and not object identities, we use the object proposal evaluation setup of Pycocotools, in which all detections are treated as having the same object ID. Second, we do not limit the number of detections per image. Instead, we run the evaluation with all detection. The motivation for this setting is the nature of methods such as CNOS [11], where the potentionally large set of SAM detections is used as a set of object proposals

that is filtered in later stages, producing only a small subset of the proposals at the end. Third, we use ground-truth *modal* bounding boxes instead of *amodal* ones, *i.e.* bounding boxes that cover only visible parts of the objects, without considering object occlusions. This decision comes from the nature of SAM, since its segmentation does not contain occluded areas. In the following sections, we perform experiments using all previously stated settings, unless otherwise specified. For example, in Sec. 3.3.6 we compare CNOS performance using modal and amodal bounding boxes. In such cases, we explicitly state which evaluation settings have been used. Also note that when reporting the average precision for the first CNOS stage, we use the confidence of the SAM as the score for evaluation.

3.3.2 Recall and the number of detections

To evaluate the effect of choosing different SAM [12] parameters, we conduct the following experiment. Using the *segment everything* mode, we retrieve object masks and compute the bounding boxes. Starting with a baseline setting that keeps a high portion of the masks (confidence and stability thresholding turned off, and NMS IoU threshold at 0.95), we change individual parameters separately and measure the average recall and average number of masks per image. We experiment with the number of points per side (8, 11, 16, 23, 32, 45, 64), stability and confidence thresholds (0.95, 0.85, 0.75, 0.5, 0.0) and NMS IoU thresholds (0.6, 0.7, 0.8, 0.9, 0.95, 1.0). We do the same evaluation with two different settings that change all three parameters at once, and also with the setting that CNOS is using (stability threshold 0.95, confidence threshold 0.88). Furthermore, we compare SAM with two simple baselines: Felzenszwalb and Huttenlocher's segmentation algorithm [114] (again using a few different parameter settings) and selective search [73]. The results are shown in Fig. 3.5-top. With well-selected parameters, [114] is able to maintain a low number of detections, but the average recall is below 20%. Selective search improves recall by a factor of 2, *i.e.* reach about 40% average recall, but has almost 10 times more masks (more than 1000 per image). SAM performs substantially better, reaching an average recall of around 80%, while keeping the average number of detections roughly between 100 and 400, depending on the specific setting.

Focusing now on the SAM results only (Fig. 3.5-bottom), the original CNOS settings is reaching quite a low number of detections per image, but the average recall is lower by approximately 5 to 10 percentage points compared to other settings. We can see that all three parameters affect the performance, and thus tuning the parameters to achieve better recall is desirable. The NMS IoU has the most steep curve, indicating a potential benefit for recall with a relatively small increase in the number of detections. On the other hand, an increasing number of points per side generates too many masks without significant recall improvement. Relaxing the stability and confidence thresholds may be beneficial when changed carefully.

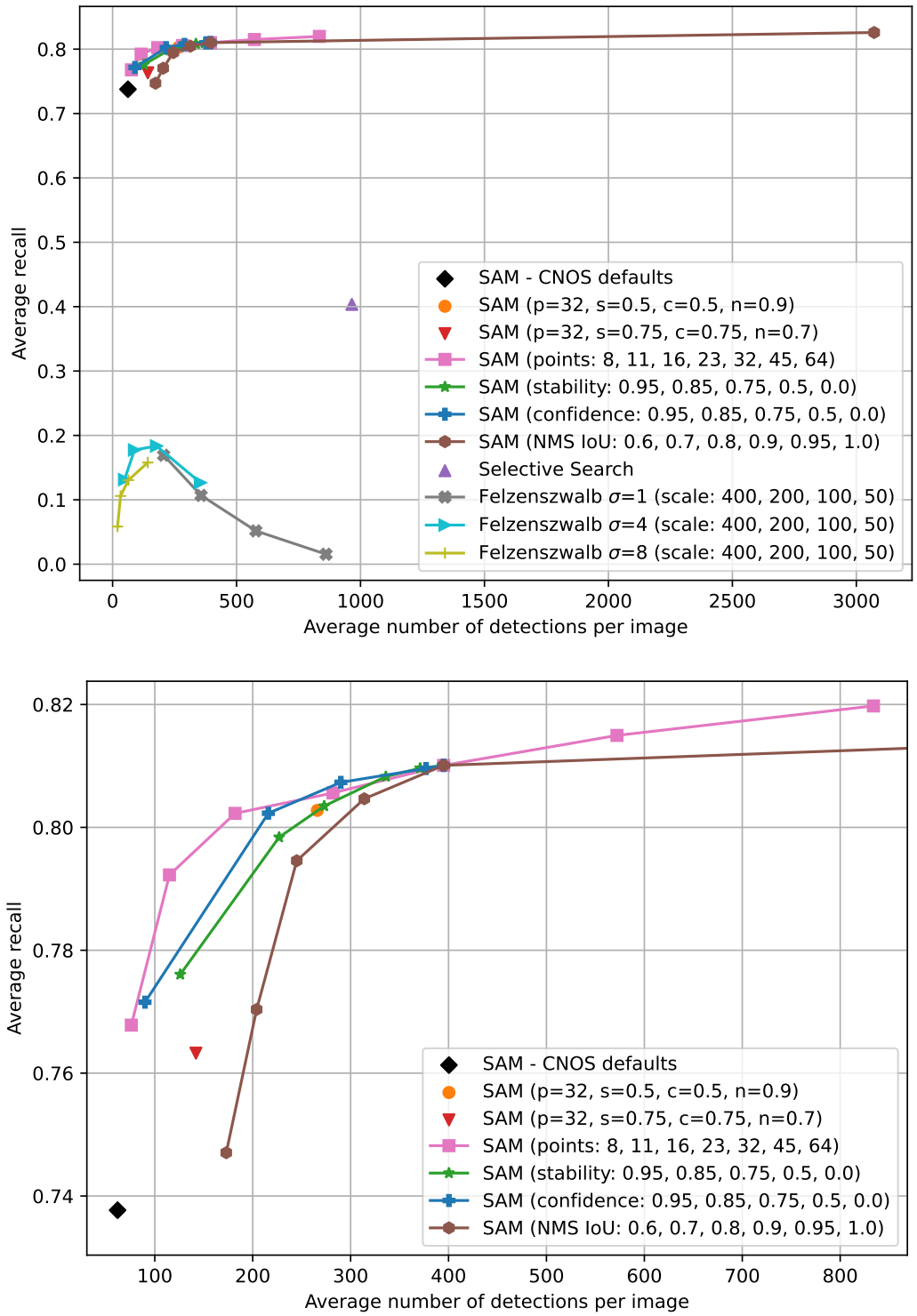

Figure 3.5: **Comparison of object proposal methods.** We evaluate SAM [12], Felzenszwalb and Huttenlocher's segmentation algorithm [114], and selective search [73] with different parameters on average recall and average number of masks per image (top). Focusing on SAM (bottom), we evaluate effect on performance when separately changing stability (s), confidence (c) and NMS IoU thresholds (n), and number of points per side (p). The parameter values in the legend correspond to the points on the curves from left to right. Also, we compare it with two different settings combining the parameters, and the settings CNOS is using by default.

3.3.3 Precision and recall of SAM and CNOS

In the second stage, we want to select only the correct detections from all the proposals. This process should increase precision while maintaining high recall. This is done by assigning each detection a confidence score. We remind that CNOS assigns the score by comparing DINOv2 [13] features of segmented objects and pre-rendered templates (while also assigning each detection an object ID).

We now focus only on scoring, while keeping the evaluation setup that ignores object identity assignments, and perform the following experiment. Starting from the default CNOS setting of the SAM, we change individual parameters and observe how the AP and AR metrics are changing. Running both the SAM (first stage only) and the full CNOS pipeline, we try to observe how the performance after the first stage relates to the performance after the second stage. Fig. 3.6 shows the results with only the first stage on the left and with both stages on the right. Compared to defaults, we can see that recall in the first stage of CNOS can be improved by changing the parameters. In particular, the most effect can be seen when changing the non-maximum suppression threshold, and also the stability threshold. On the other hand, changing the confidence score has almost no effect on the SAM results. Note that even though the recall changes are small, we could expect a larger recall improvement when combining the settings. However, this does not necessarily translate into a better score after the second stage. As seen in Fig. 3.6 (right), we were able to slightly improve the score only in two cases (when changing the NMS threshold to 0.9 ●, and confidence threshold to 0.9 ✚); others have lower both AP and AR.

3.3.4 Second stage of CNOS and scoring

CNOS is computing the scores as cosine similarity using DINOv2 [13] features between pre-rendered object templates and segmented object from the input image. To evaluate how well the DINOv2 features perform in this second stage, we take the CNOS predictions and change the scores. First, to provide a lower bound on the average precision under which the methods should not drop, we evaluate the predictions using random scores. To provide an upper bound, we change the scores of each detection to the maximum IoU with all ground truths, obtaining "oracle" results. We perform the experiment first for SAM-only predictions, then for the full CNOS pipeline both with and without considering the object ID predictions, and show the results in Table 3.2. We observe that the AP metric of CNOS is quite high with respect to the upper bound, but can still be improved by approximately 10 percentage points. In other words, with current recall, CNOS performance is at 85% between our lower and upper bounds when ignoring object identification assignments, and at 80% when the assigned object IDs are considered.

Our oracle results show that the upper bound for AP is actually the value of AR metric. This is expected, even though we think that the names AP (average precision) and AR (average recall) are a bit confusing when used next to each other, as the former summarizes the whole PR curve considering both precision and recall, while the latter is computed only using the recall. Nevertheless, it is easy to see why this relationship should be true. First, re-scoring cannot change the recall, as long as we use all the detections during evaluation. This is because

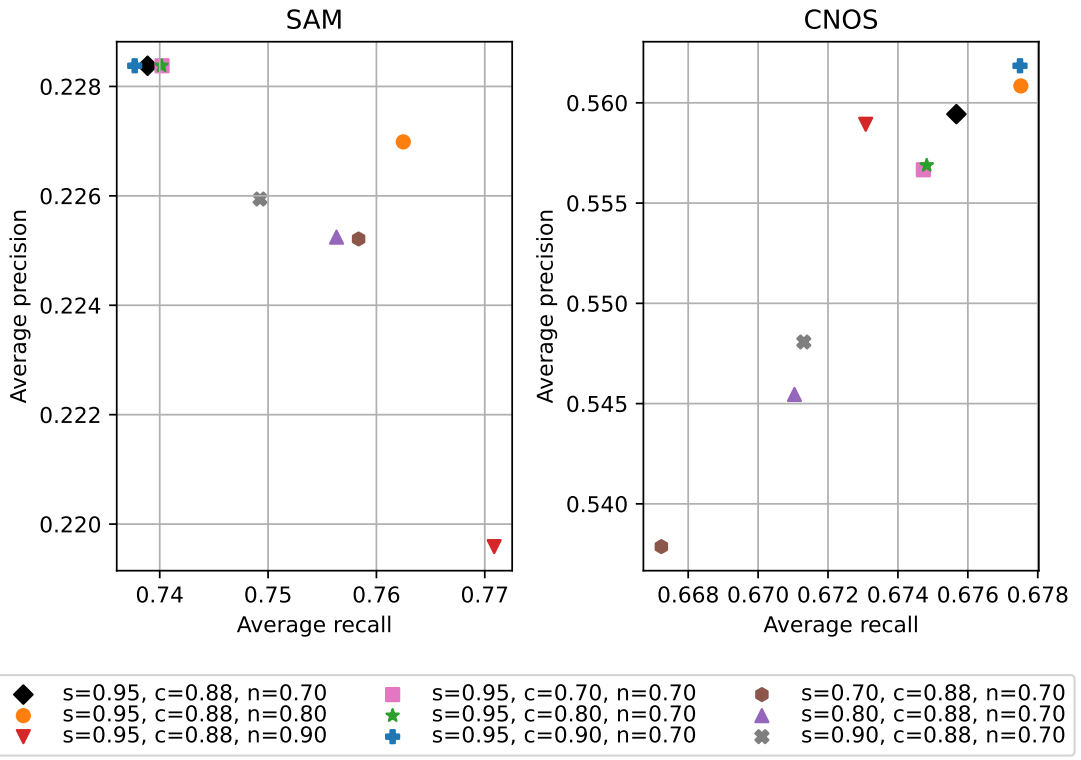

Figure 3.6: **Average precision and recall of SAM and CNOS.** We compare the average precision and recall of the first stage of CNOS (SAM only – left) and both CNOS stages (right). As expected, the second stage significantly increases precision, but also reduces recall due to filtering out some of the correct detections. We evaluate the stages using several different SAM settings, starting from the default settings (black diamond) and changing individual parameters – stability (s), confidence (c), and NMS IoU (n) thresholds.

Method	Using object IDs	Scoring	AP	AR
SAM only	No	Oracle	0.841	0.842
		SAM confidence score	0.271	
		Random	0.058	
CNOS	No	Oracle	0.762	0.762
		DINOv2 cosine similarity	0.659	
		Random	0.073	
CNOS	Yes	Oracle	0.726	0.728
		DINOv2 cosine similarity	0.622	
		Random	0.217	

Table 3.2: **Scoring upper and lower bounds.** We take the detections of SAM and CNOS using default parameters and evaluate them using different scoring methods, providing upper and lower bounds for the AP metric. The upper bound is obtained using simple "oracle" scoring, where each detection has a score equal to the IoU with the best matching ground truth detection. The lower bound is obtained using random scores between 0 and 1. For CNOS, we run the evaluation both with and without considering the object identity predictions.

recall does not count false positives. Second, because the AP metric is computed as the (approximated) area under the PR curve and the recall cannot change, the area under the curve can be at most a rectangle with width equal to the recall and height equal to 1 (perfect precision). Thus, the AP is bounded by the AR. Our simple "oracle" approach ensures that (almost) all detections identified as true positives have higher scores than false negatives, maximizing the area.

In Fig. 3.7 we visualize 4 examples of CNOS detections from the YCB-Video [25] dataset with their corresponding scores. We split the detections from each image into two sets and visualize them separately; one image shows detections with a confidence score of at least 0.5 and the other image shows detections with scores below 0.5. We do not show the confidence scores for the latter, as it contains a large number of masks, which makes it difficult to visualize. This also shows one problem of CNOS: In practical applications, the user would need to set a score threshold to filter out unconfident detections, however, we can see that in some cases even a good detection has a relatively low score *e.g.* 0.6. This can be due to various reasons, for example, object occlusions. Thus, the threshold would need to be set to a relatively low value. For the first example, our threshold 0.5 works well. In the second example, we can see a lot of false positives just above our threshold. The problem is that the confidence score obtained using the DINOv2 feature similarity lacks the notion of *objectness*, *i.e.* whether a detection actually corresponds to an object or not. In example 2, false positives on the chessboard have been assigned an object ID corresponding to the wooden block from the YCB-Video dataset. This is simply because the detections also have a wooden texture, making the DINOv2 features similar. However, this scoring completely ignores that there is, in fact, no object in the segmented part of the image. Examples 3 and 4 show scenarios where CNOS was unable to retrieve some objects. In example 3 this is due to a low confidence score (the bowl) and in example 4 because SAM retrieved only subparts, but not the whole object (the

drill). For additional qualitative results, see Sec. A.2 in the attachments.

3.3.5 Evaluation with the object ID assignments

Another important factor in the second stage of CNOS is the assignment of object identities. If we predict a really precise bounding box but the object ID will not match the one of the corresponding ground-truth, it will not be counted as a successful detection. Let us recall that CNOS assigns the object IDs by applying the argmax function on the similarity matrix over the dimension with different 3D models; *i.e.* it is again dependent on the DINOv2 feature similarity. To measure the effect of incorrect object ID predictions, we compare our previous evaluation results that ignored object ID assignments with results obtained using an evaluation setup that considers them. Following the SAM settings used in Sec. 3.3.3, we change individual SAM parameters and plot the results in Fig. 3.8, using filled markers for evaluation ignoring object IDs, and non-filled markers for evaluation with object IDs. We can see that on average both AP and AR drop approximately by 5 percentage points when the object IDs are considered, and the difference does not change much between various parameter settings. Therefore, object ID assignments still make a significant difference in the evaluation, although they have a smaller effect than the the confidence score assignments.

3.3.6 Evaluation with amodal masks

Bounding boxes or masks predicted by CNOS are, as already mentioned, of the modal type, *i.e.* covering only the visible part of the object. This is because the masks are produced by segmentation using SAM. On the other hand, BOP is using amodal bounding boxes by default, *i.e.* covering also occluded parts of the object. For the 6D pose estimation this actually makes sense – for example, [37, 67, 10] are initializing the translation estimate from a bounding box of an object. When the object is heavily occluded, the initialization can differ a lot. The same problem is present in our evaluation of CNOS/SAM-generated bounding boxes. Occluded objects will decrease the AP and AR metrics, even though they have been correctly detected from the perspective of modal evaluation. To illustrate the difference, we evaluated the CNOS predictions using both types of bounding boxes and show the results in Fig. 3.9. On average, AP and AR metrics drop by approximately 3.5 percentage points when switching to amodal type. This is a minor drop compared to other problems of CNOS, and cannot really be corrected without significant change in the architecture – any method relying only on some kind of visual segmentation, such as SAM, will have the same problem. However, note that the YCB-Video is a relatively "easy" dataset containing only a very small number of images with heavily occluded objects. On harder datasets with more occlusions, this can still make a significant difference.

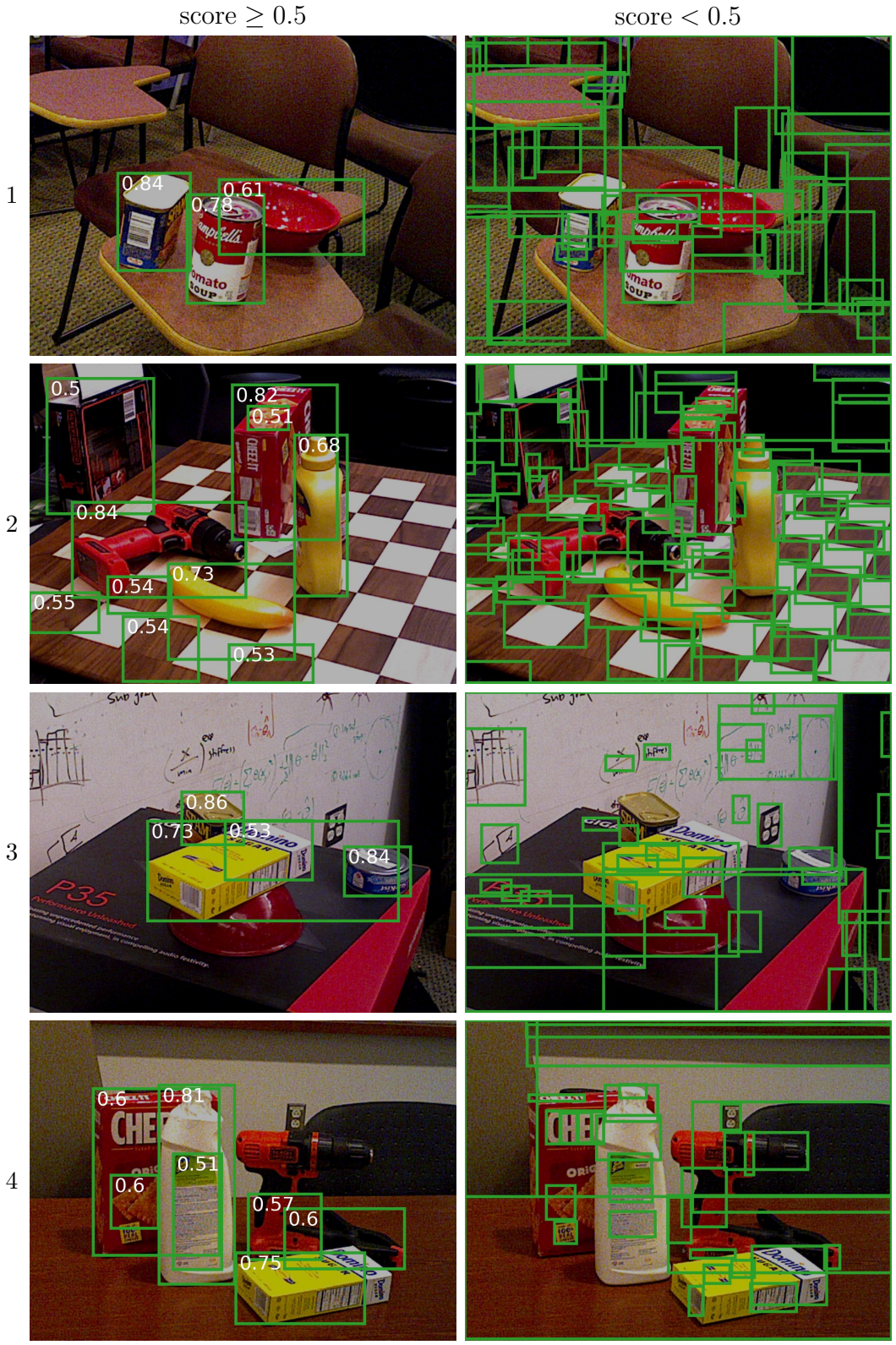

Figure 3.7: **Examples of CNOS detections with assigned confidence scores.** We show examples of object detections on images from the YCB-Video [25] dataset. Detections with scores of at least 0.5 are shown on the left, and detections with scores below 0.5 are shown on the right. Given the score threshold of 0.5, CNOS detected almost perfectly all objects in the first example, however, it has false positives and false negatives in the other 3 examples, showing the problems of using DINOv2 [13] feature similarity for confidence scores.

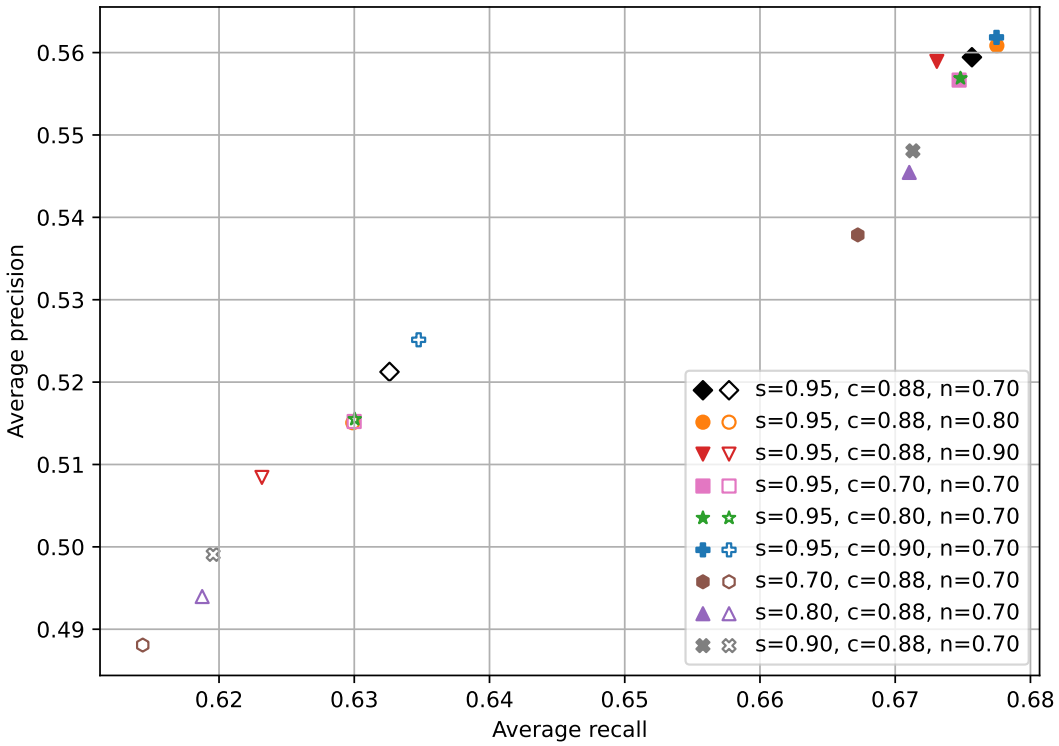

Figure 3.8: **Difference in CNOS evaluation results with and without considering the object identities.** On average, the AP and AR drops by 5 percentage points in our experiments when considering predicted object IDs assignments (non-filled markers) compared to evaluation ignoring them (filled markers). The difference is similar for various settings of stability, confidence and NMS thresholds.

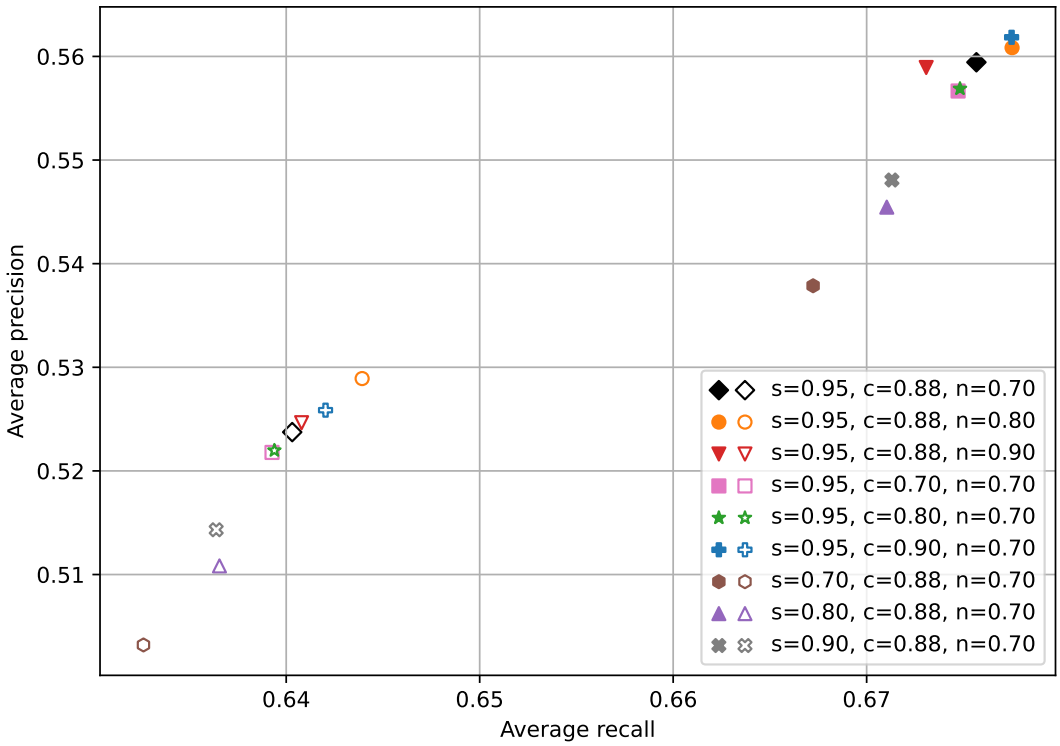

Figure 3.9: Difference in CNOS evaluation results when comparing with modal or amodal ground-truth bounding boxes, *i.e.* considering only the visible object parts or the whole object with occluded areas for the bounding box. Both AP and AR drops by about 3.5 percentage points when changed to evaluation with amodal bounding boxes. This is due to the nature of SAM [12], as it predicts masks only for visible parts of the objects.

4. Conclusion

In our work, we looked at problems of methods for 6D pose estimation and detection in uncontrolled environments. In the first part of our work, we investigated FocalPose [10], a method for joint 6D pose and camera focal length estimation. We identified the points where the method could be improved and introduced a new improved method (called FocalPose++ [14]). Our contributions for this first part can be summarized as follows:

- We re-derived the non-linear 6D pose and focal length update rule, and suggested a new version of the update rule. We showed that FocalPose method with our new update rule performs better, and that the original update rule is, in fact, an approximation of the new one. The benefits of our update rule lies in the fact that the translation update also reflects the focal length changes between the update iterations.

- For the rendering of synthetic training data, we introduced two different distributions of object poses and focal lengths, which better matches the real training data compared to the uniform distribution used in [10]. We achieved significantly better results when training on synthetic data generated using our parametric distribution, validating our approach.

- We replaced previously used DINO instance classifier by ML-Decoder [103], an attention-based classification head that significantly improved the instance classification accuracy. However, we found that the performance effect on the final 6D pose estimation accuracy is relatively small. These findings are consistent with the results in [10], suggesting that FocalPose works reasonably well even with only a similar or approximate 3D model.

- We evaluated our improved FocalPose++ method with ground-truth bouding boxes and classes, providing an upper-bound on the model performance when changing the detector and instance classifier. Although FocalPose++ achieves state-of-the-art performance, replacing the detector and classifier can improve the method by up to an additional 15% in measured metrics.

In the second part, we analyzed CNOS, a method for novel object segmentation, commonly used as an object detector for the 6D pose estimation of novel objects. We performed several experiments measuring the performance of the two CNOS stages and identified their problems. Our main findings are summarized as follows:

- In the first CNOS stage, Segment Anything Model (SAM) achieves a good ratio between average recall (AR) and the average number of object proposals per image. On the YCBV-dataset, it achieves an average recall of almost 74% with the default CNOS parameter settings, *i.e.* it localizes the objects depicted in the image by one of the proposed bounding boxes in 74% of the cases. When carefully changing the parameters, the AR can increase to values above 80%, but the number of proposals also increases (approximately by a factor of 3.5), making the following second stage of CNOS more challenging.

- When comparing AP (average precision metric, *i.e.* area under precision-recall curve) between the first and the second stage of CNOS, the AP increases a lot because CNOS assigns a high confidence score only to detections similar to object templates, in contrast to SAM that tries to detect *all* objects. However, sometimes the assigned score is low even for correct detections, resulting in a decrease in AR (average recall) by around 7.5 percentage points from the first to the second CNOS stage. We also found that changing the SAM parameters to achieve better recall in the first stage often does not result in better recall of the entire CNOS pipeline.
- For the second stage of CNOS, we presented the scoring lower and upper bounds with the current recall achieved by SAM in the first stage. In particular, we presented the lower bound by assigning a random score to each predicted detection, and the upper bound by setting the score of each predicted detection to the IoU with the best matching ground truth detection. With the default CNOS parameter setting, the AP metric of CNOS is at 80% between the lower and upper bounds. We presented several qualitative examples of CNOS detections, showing that the scoring by DINOv2 feature similarity lacks the notion of *objectness*, *i.e.* whether a detection actually corresponds to an object or not. In addition, the scores for some correct and some incorrect detections are relatively close to each other. In practice, this makes it difficult to find a good confidence score threshold to filter out unconfident predictions. This suggests that the second stage could be improved by training a confidence score model in a discriminative manner, similar *e.g.* MegaPose coarse estimator [67].
- Finally, we have also evaluated the correctness of the CNOS-predicted object identities and evaluated how it affects the AP and AR metrics. We found that in our experiments, both AP and AR drop by approximately 5 percentage points compared to the evaluation that ignores the object identities, *i.e.* the evaluation that does not count in the object ID prediction errors. We also investigated how the performance changes when using amodal bounding boxes (*i.e.* including occluded parts of objects) instead of modal bounding boxes (covering only visible parts of the objects). The change in evaluation results from modal to amodal bounding boxes makes a smaller difference, about 3.5 percentage points drop in AP and AR.

Future work

6D pose estimation is still an evolving area. It offers various research directions and has a great potential in the case of uncontrolled environments. In our work, we have shown examples of methods that can estimate the 6D pose without camera calibration or that can work with novel objects. Leveraging the potential of these two tasks, one possible direction is to combine them, for example, to combine FocalPose [10] and MegaPose [67] into a single method for the 6D pose estimation of novel objects without camera calibration. Having such a method would allow us to do 6D pose estimation in the wild on YouTube videos where camera calibration is unknown. Potential applications include learning and planning robotic manipulation skills from online instructional videos or images lacking

metadata. Several works have successfully used these approaches for robotic imitation of object interactions. For example, [115] reconstructs the hand-object trajectory for robotic simulation of the interaction using reinforcement learning, while [116] also estimates the motion of the human holding the observed object. Video demonstrations can also be used for task and motion planning, while utilizing the 6D pose estimates to guide the robot [6]. Learning from such sources also has an important advantage – the possible scale of the collected datasets. The importance of scale can be seen in natural language processing, where language models such as GPT [117] are trained using large-scale data sets, leading to much better generalization and language understanding. A similar trend can be seen in computer vision. Such large-scale datasets are typically obtained by data scraping (*e.g.* HowTo100M [118] searches and downloads YouTube videos), while in some other cases (such as egocentric videos in the Ego4D [119] dataset) the videos are manually collected by humans. Similar approaches have also been used for image datasets [120, 121, 12]. Alternatively, synthetic data can be used for training, as shown in both FocalPose [10] and MegaPose [67]. This is very convenient for training the 6D pose estimation methods, as obtaining precise 6D pose annotations from real images on a large scale is complicated. However, to be able to generalize to novel objects, a large variety of 3D models has to be used as well. Fortunately, this is also possible thanks to the recent release of the Objaverse-XL [122] dataset, which contains over 10 million 3D meshes. Training a model on photorealistically rendered images of Objaverse objects could lead to a truly generalizable method working in a zero-shot setting on novel objects.

In the area of novel object detection and segmentation, several potential directions can also be identified. One direction is to improve current two-stage methods, such as CNOS [11]. Although the Segment Anything Model used in the first stage of CNOS works really well, we have seen cases where it was unable to retrieve some objects, *e.g.* the drill from the YCB-Video dataset. In general, SAM also segments areas that do not correspond to any "object" from our point of view. If changing the SAM parameters is not enough, would it be possible to fix these problems by re-training it to distinguish between "things" and "stuff" [123]? Alternatively, this could be done in the second stage, *i.e.* changing the scoring to include also the notion of objectness. Another possible direction is to eliminate template rendering and use the 3D model directly to detect objects. This could be done using OpenShape [124], a recent model that extracts embeddings from 3D models while aligning them with the CLIP [113, 125] embedding space. Therefore, it is possible to directly compare a 3D object model with an image segment without the need to render the object templates. Also, the combination of SAM and OpenShape raises another interesting question: Would it be possible to create a single-stage detector of unseen objects? SAM has shown that it is possible to use CLIP-embedded text as a prompt to their mask encoder. Thus, OpenShape could be used to obtain CLIP embeddings of the 3D models, and SAM would be prompted with these embeddings. Although the idea is simple, several questions remain unanswered, *e.g.* how to deal with multiple instances of one object in an image.

Finally, having fast and robust object detection and 6D pose estimation methods for novel objects that can be scaled to a large object database, Objaverse-XL [122] could be used to model 3D scenes seen in real images when the 3D model

is not available at all, *i.e.* we do not have the model even at inference time. This could be done either by simply retrieving similar objects from Objaverse, or *e.g.* via 3D object reconstruction with diffusion models while using Objaverse as a grounding mechanism.

Acknowledgements

This work was partly supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140), and by the European Union's Horizon Europe projects euROBIN (No. 101070596) and AGIMUS (No. 101070165). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

Bibliography

[1] D G Lowe. Object recognition from local scale-invariant features. In *CVPR*, volume 2, pages 1150–1157, September 1999.

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

[3] Lawrence G Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.

[4] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[5] Pengyuan Wang, Fabian Manhardt, Luca Minciullo, Lorenzo Garattoni, Sven Meier, Nassir Navab, and Benjamin Busam. Demograsp: Few-shot learning for robotic grasping with human demonstration. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5733–5740. IEEE, 2021.

[6] Kateryna Zorina, David Kovar, Florent Lamiraux, Nicolas Mansard, Justin Carpentier, Josef Sivic, and Vladimir Petrik. Multi-contact task and motion planning guided by video demonstration. In *ICRA 2023-International Conference on Robotics and Automation*, 2023.

[7] Mederic Fourmy, Vojtech Priban, Jan Kristof Behrens, Nicolas Mansard, Josef Sivic, and Vladimir Petrik. Visually guided model predictive robot control via 6d object pose localization and tracking, 2023.

[8] Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Transactions on graphics (TOG)*, 33(4):1–12, 2014.

[9] Chung-Yi Weng, Brian Curless, and Ira Kemelmacher-Shlizerman. Photo wake-up: 3d character animation from a single photo. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5908–5917, 2019.

[10] G. Ponimatkin, Y. Labbe, B. Russell, M. Aubry, and J. Sivic. Focal length and object pose estimation via render and compare. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[11] Van Nguyen Nguyen, Thibault Groueix, Georgy Ponimatkin, Vincent Lepetit, and Tomas Hodan. Cnos: A strong baseline for cad-based novel object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2134–2140, 2023.

[12] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023.

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

[14] M. Cífka, G. Ponimatkin, Y. Labbe, B. Russell, M. Aubry, V. Petrik, and J. Sivic. Focalpose++: Focal length and object pose estimation via render and compare, 2023. `https://arxiv.org/abs/2312.02985`.

[15] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[16] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate O(n) solution to the PnP problem. *IJCV (International Journal of Computer Vision)*, 81:155–166, 2009.

[17] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. Spie, 1992.

[18] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.

[19] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, pages 404–417. Springer, 2006.

[20] A Collet and S S Srinivasa. Efficient multi-view object recognition and full pose estimation. In *ICRA*, pages 2050–2055, May 2010.

[21] 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.

[22] S Hinterstoisser, S Holzer, C Cagniart, S Ilic, K Konolige, N Navab, and V Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *ICCV*, pages 858–865, November 2011.

[23] 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 *CVPR*, pages 3828–3836, 2017.

[24] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *CoRL*, 2018.

[25] 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.

[26] 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 *ICCV*, pages 1521–1529, 2017.

[27] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6D object pose prediction. In *CVPR*, pages 292–301, 2018.

[28] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-DoF object pose from semantic keypoints. In *ICRA*, pages 2011–2018. IEEE, 2017.

[29] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6D object pose estimation. In *CVPR*, pages 3385–3394, 2019.

[30] Chen Song, Jiaru Song, and Qixing Huang. HybridPose: 6D object pose estimation under hybrid representations. In *CVPR*, pages 431–440, 2020.

[31] Kiru Park, Timothy Patten, and Markus Vincze. Pix2Pose: Pixel-wise coordinate regression of objects for 6D pose estimation. In *ICCV*, pages 7668–7677, 2019.

[32] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *CVPR*, pages 4561–4570, 2019.

[33] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. DPOD: 6D pose object detector and refiner. In *CVPR*, pages 1941–1950, 2019.

[34] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep iterative matching for 6D pose estimation. In *ECCV*, pages 683–698, 2018.

[35] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6D pose refinement in RGB. In *ECCV*, pages 800–815, 2018.

[36] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Generalized feedback loop for joint hand-object pose estimation. *TPAMI (IEEE Transactions on Pattern Analysis and Machine Intelligence)*, 2019.

[37] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *ECCV*, 2020.

[38] Olivier Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. MIT press, 1993.

[39] Alex M Andrew. Multiple view geometry in computer vision. *Kybernetes*, 2001.

[40] Gaku Nakano. A versatile approach for solving pnp, pnpf, and pnpfr problems. In *ECCV*, pages 338–352. Springer, 2016.

[41] Adrian Penate-Sanchez, Juan Andrade-Cetto, and Francesc Moreno-Noguer. Exhaustive linearization for robust camera pose and focal length estimation. *TPAMI (IEEE Transactions on Pattern Analysis and Machine Intelligence)*, 35(10):2387–2400, 2013.

[42] Yinqiang Zheng, Shigeki Sugimoto, Imari Sato, and Masatoshi Okutomi. A general and simple method for camera pose and focal length determination. In *CVPR*, pages 430–437, 2014.

[43] Roger Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344, 1987.

[44] Markéta Dubská, Adam Herout, Roman Juránek, and Jakub Sochor. Fully automatic roadside camera calibration for traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1162–1171, 2014.

[45] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[46] David A. Forsyth and Jean Ponce. *Computer Vision - A Modern Approach, Second Edition.* Pitman, 2012.

[47] Qian Chen, Haiyuan Wu, and Toshikazu Wada. Camera calibration with two arbitrary coplanar circles. In *ECCV*, pages 521–532. Springer, 2004.

[48] Yaming Wang, Xiao Tan, Yi Yang, Xiao Liu, Errui Ding, Feng Zhou, and Larry S Davis. 3D pose estimation for fine-grained object categories. In *ECCVW*, pages 0–0, 2018.

[49] Alexander Grabner, Peter M Roth, and Vincent Lepetit. Gp2c: Geometric projection parameter consensus for joint 3d pose and focal length estimation in the wild. In *CVPR*, pages 2222–2231, 2019.

[50] Yaohang Han, Huijun Di, Hanfeng Zheng, Jianyong Qi, and Jianwei Gong. Gcvnet: Geometry constrained voting network to estimate 3D pose for fine-grained object categories. In *PRCV*, pages 180–192. Springer, 2020.

[51] Mathieu Aubry, Daniel Maturana, Alexei A. Efros, Bryan Russell, and Josef Sivic. Seeing 3D chairs: Exemplar part-based 2D-3D alignment using a large dataset of CAD models. In *CVPR*, 2014.

[52] Yang Xiao, Xuchong Qiu, Pierre-Alain Langlois, Mathieu Aubry, and Renaud Marlet. Pose from shape: Deep pose estimation for arbitrary 3D objects. In *BMVC*, 2019.

[53] Yang Xiao, Yuming Du, and Renaud Marlet. PoseContrast: Class-agnostic object viewpoint estimation in the wild with pose-aware contrastive learning. In *3DV*, 2021.

[54] 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 *CVPR*, 2022.

[55] 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 *CVPR*, 2020.

[56] Yang Xiao and Renaud Marlet. Few-shot object detection and viewpoint estimation for objects in the wild. In *European Conference on Computer Vision (ECCV)*, 2020.

[57] Yunzhi Lin, Jonathan Tremblay, Stephen Tyree, Patricio A. Vela, and Stan Birchfield. Single-stage keypoint-based category-level object pose estimation from an RGB image. In *ICRA*, 2022.

[58] 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 *CVPR*, 2019.

[59] Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. In *ECCV*, 2020.

[60] Fu Li, Ivan Shugurov, Benjamin Busam, Minglong Li, Shaowu Yang, and Slobodan Ilic. Polarmesh: A star-convex 3d shape approximation for object pose estimation. *IEEE Robotics and Automation Letters*, 7(2):4416–4423, 2022.

[61] Fabian Manhardt, Gu Wang, Benjamin Busam, Manuel Nickel, Sven Meier, Luca Minciullo, Xiangyang Ji, and Nassir Navab. CPS++: Improving class-level 6D pose and shape estimation from monocular images with self-supervised learning. *arXiv preprint arXiv:2003.05848*, 2020.

[62] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. KPAM: Keypoint affordances for category-level robotic manipulation. In *ISRR*, 2019.

[63] Bowen Wen, Wenzhao Lian, Kostas Bekris, and Stefan Schaal. Catgrasp: Learning category-level task-relevant grasping in clutter from simulation. *arXiv preprint arXiv:2109.09163*, 2021.

[64] Ivan Shugurov, Fu Li, Benjamin Busam, and Slobodan Ilic. OSOP: A multi-stage one shot object pose estimation framework. In *CVPR*, 2022.

[65] Brian Okorn, Qiao Gu, Martial Hebert, and David Held. ZePHyR: Zero-shot pose hypothesis rating. In *ICRA*, 2021.

[66] Jianqiu Chen, Mingshan Sun, Tianpeng Bao, Rui Zhao, Liwei Wu, and Zhenyu He. Zeropose: Cad-model-based zero-shot pose estimation, 2023.

[67] 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 *CoRL*, 2022.

[68] Evin Pınar Örnek, Yann Labbé, Bugra Tekin, Lingni Ma, Cem Keskin, Christian Forster, and Tomas Hodan. Foundpose: Unseen object pose estimation with foundation features, 2023.

[69] Van Nguyen Nguyen, Thibault Groueix, Mathieu Salzmann, and Vincent Lepetit. Gigapose: Fast and robust novel object pose estimation via one correspondence, 2023.

[70] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[71] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[72] Hedi Harzallah, Frédéric Jurie, and Cordelia Schmid. Combining efficient object localization and image classification. In *2009 IEEE 12th international conference on computer vision*, pages 237–244. IEEE, 2009.

[73] Koen EA Van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. Segmentation as selective search for object recognition. In *2011 international conference on computer vision*, pages 1879–1886. IEEE, 2011.

[74] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[75] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[76] 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, 2015.

[77] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *CVPR*, pages 2961–2969, 2017.

[78] 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.

[79] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[80] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[81] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[82] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, and Furu Wei. Image as a foreign language: Beit pretraining for all vision and vision-language tasks, 2022.

[83] Ayush Jaiswal, Yue Wu, Pradeep Natarajan, and Premkumar Natarajan. Class-agnostic object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 919–928, 2021.

[84] Dahun Kim, Tsung-Yi Lin, Anelia Angelova, In So Kweon, and Weicheng Kuo. Learning open-world object proposals without learning to classify. *IEEE Robotics and Automation Letters*, 7(2):5453–5460, 2022.

[85] Muhammad Maaz, Hanoona Rasheed, Salman Khan, Fahad Shahbaz Khan, Rao Muhammad Anwer, and Ming-Hsuan Yang. Class-agnostic object detection with multi-modal transformer. In *European Conference on Computer Vision*, pages 512–531. Springer, 2022.

[86] Adam Bielski and Paolo Favaro. Move: Unsupervised movable object segmentation and detection. *Advances in Neural Information Processing Systems*, 35:33371–33386, 2022.

[87] Haiwen Huang, Andreas Geiger, and Dan Zhang. Good: Exploring geometric cues for detecting objects in an open world. *arXiv preprint arXiv:2212.11720*, 2022.

[88] Yuming Du, Wen Guo, Yang Xiao, and Vincent Lepetit. 1st place solution for the uvo challenge on image-based open-world segmentation 2021. *arXiv preprint arXiv:2110.10239*, 2021.

[89] Yuming Du, Yang Xiao, and Vincent Lepetit. Learning to better segment objects from unseen classes with unlabeled videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3375–3384, 2021.

[90] Yuyang Zhao, Zhun Zhong, Nicu Sebe, and Gim Hee Lee. Novel class discovery in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2022.

[91] Subhankar Roy, Mingxuan Liu, Zhun Zhong, Nicu Sebe, and Elisa Ricci. Class-incremental novel class discovery. In *European Conference on Computer Vision*, pages 317–333. Springer, 2022.

[92] Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. Unseen object instance segmentation for robotic environments, 2021.

[93] Yu Xiang, Christopher Xie, Arsalan Mousavian, and Dieter Fox. Learning rgb-d feature embeddings for unseen object instance segmentation, 2021.

[94] Maximilian Durner, Wout Boerdijk, Martin Sundermeyer, Werner Friedl, Zoltan-Csaba Marton, and Rudolph Triebel. Unknown object segmentation from stereo images, 2021.

[95] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all, 2023.

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

[97] Xu Zhao, Wenchao Ding, Yongqi An, Yinglong Du, Tao Yu, Min Li, Ming Tang, and Jinqiao Wang. Fast segment anything. *arXiv preprint arXiv:2306.12156*, 2023.

[98] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[99] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[100] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3D: Dataset and methods for single-image 3D shape modeling. In *CVPR*, 2018.

[101] Christopher Bingham. An antipodally symmetric distribution on the sphere. *The Annals of Statistics*, 2(6):1201–1225, 1974.

[102] Igor Gilitschenski, Roshni Sahoo, Wilko Schwarting, Alexander Amini, Sertac Karaman, and Daniela Rus. Deep orientation uncertainty learning based on a bingham loss. In *International Conference on Learning Representations*, 2020.

[103] Tal Ridnik, Gilad Sharir, Avi Ben-Cohen, Emanuel Ben-Baruch, and Asaf Noy. Ml-decoder: Scalable and versatile classification head, 2021.

[104] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[105] Tal Ridnik, Hussam Lawen, Asaf Noy, and Itamar Friedman. Tresnet: High performance gpu-dedicated architecture, 2020.

[106] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects, 2023.

[107] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Zero-shot category-level object pose estimation, 2022.

[108] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Yolo by ultralytics, 2023. `https://github.com/ultralytics/ultralytics`.

[109] Martin Sundermeyer, Tomáš Hodaň, Yann Labbé, 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 (CVPR) Workshops*, pages 2785–2794, June 2023.

[110] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[111] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021.

[112] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains, 2020.

[113] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[114] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59:167–181, 2004.

[115] Austin Patel, Andrew Wang, Ilija Radosavovic, and Jitendra Malik. Learning to imitate object interactions from internet videos. *arXiv preprint arXiv:2211.13225*, 2022.

[116] Kateryna Zorina, Justin Carpentier, Josef Sivic, and Vladimír Petrík. Learning to manipulate tools by aligning simulation to video demonstration. *IEEE Robotics and Automation Letters*, 7(1):438–445, 2021.

[117] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[118] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video

embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2630–2640, 2019.

[119] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[120] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models, 2022.

[121] Alexander Cong Li, Ellis Langham Brown, Alexei A Efros, and Deepak Pathak. Internet explorer: Targeted representation learning on the open web. In *International Conference on Machine Learning*, pages 19385–19406. PMLR, 2023.

[122] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects, 2023.

[123] Edward H Adelson. On seeing stuff: the perception of materials by humans and machines. In *Human vision and electronic imaging VI*, volume 4299, pages 1–12. SPIE, 2001.

[124] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. *arXiv preprint arXiv:2305.10764*, 2023.

[125] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning, 2022.

List of Figures

List of Tables

List of Abbreviations

CNN	convolutional neural network
R-CNN	region-based convolutional neural network
ViT	Vision Transformer
TP	true positive
TN	true negative
FP	false positive
FN	false negative
GT	ground-truth
IoU	intersection over union
AR	average recall
AP	average precision
AUC	area under curve
NMS	non-maximum suppression

A. Attachments

A.1 FocalPose++ qualitative results

Figure A.1: **Qualitative results for Pix3D chairs.**

Figure A.2: **Qualitative results for Pix3D beds.**

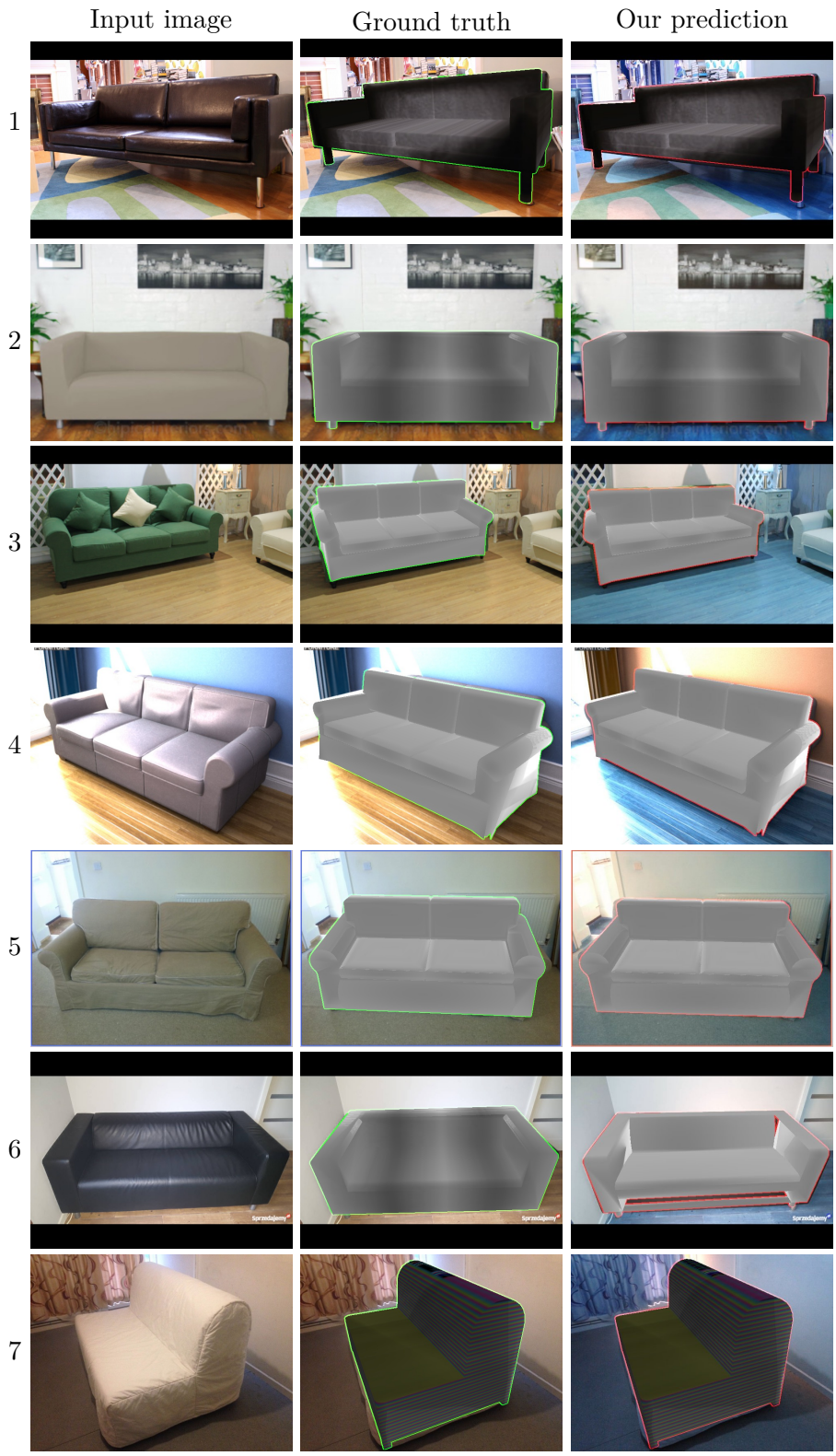

Figure A.3: **Qualitative results for Pix3D sofas.**

Figure A.4: **Qualitative results for Pix3D tables.**

Figure A.5: **Qualitative results for CompCars.**

Figure A.6: **Qualitative results for StanfordCars.**

Figure A.7: **Examples of failures in the Pix3D dataset.** We show typical failure modes of FocalPose++: symmetric objects (rows 1-2), local minima (rows 3-4) and misalignment due to the incorrect model (row 5-7).

A.2 CNOS qualitative results

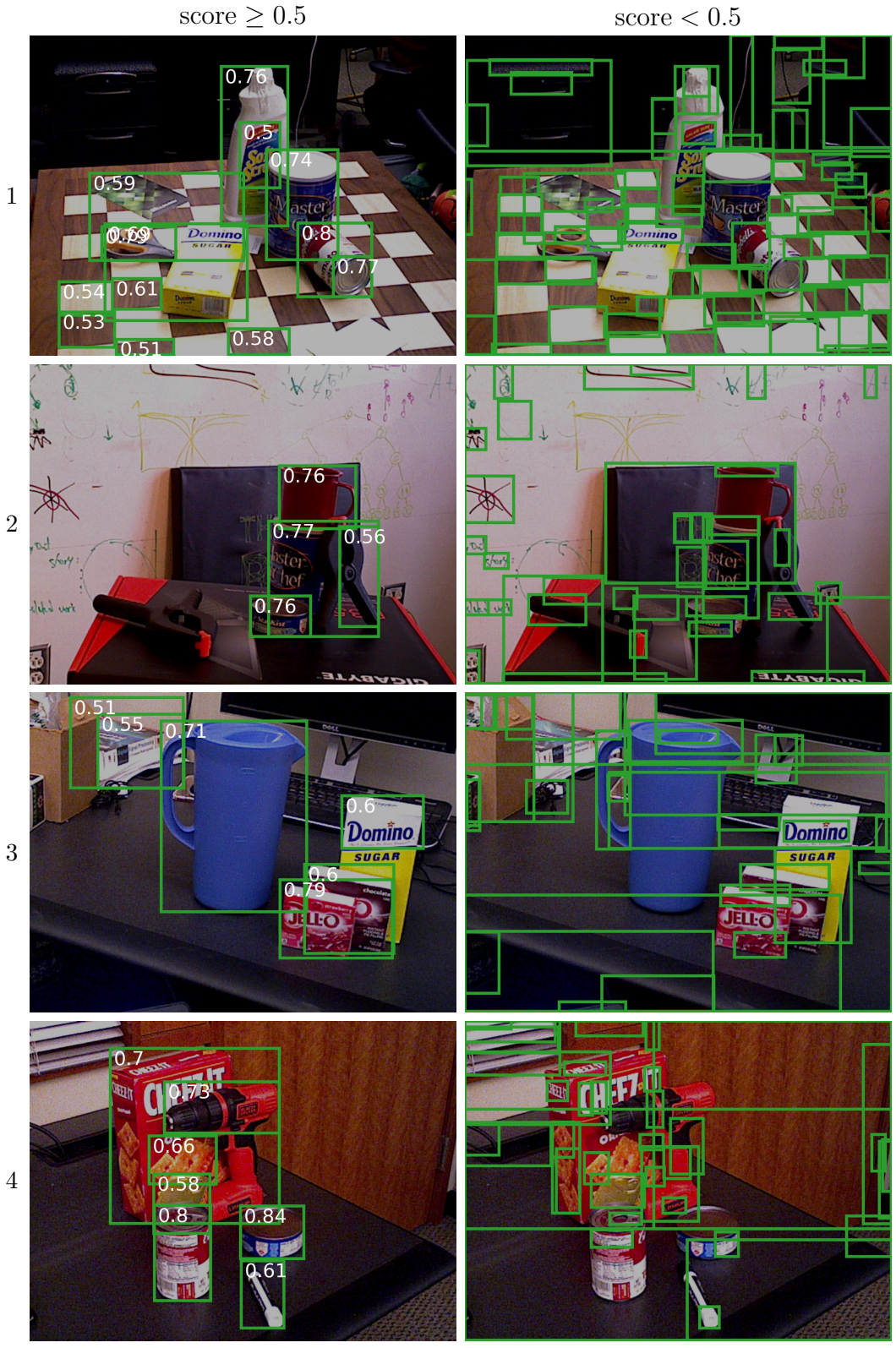

Figure A.8: **Example CNOS detections with assigned confidence scores.**

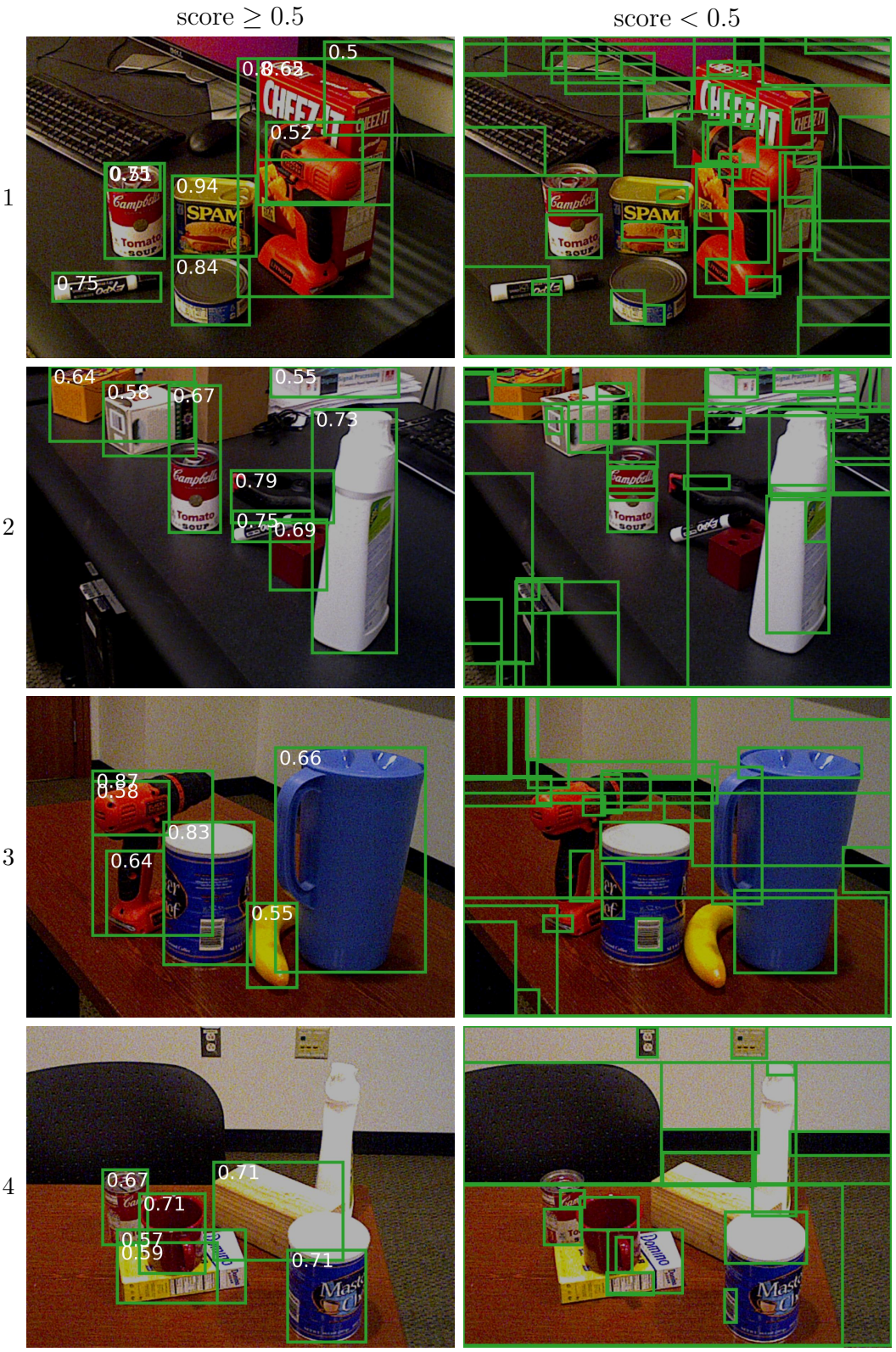

Figure A.9: **Example CNOS detections with assigned confidence scores.**