International Journal of Semantic Computing

PrimitivePose: Generic Model and Representation for 3D Bounding Box Prediction of Unseen Objects*

Andreas Kriegler†

Vision, Automation and Control, AIT Austrian Institute of Technology
Giefinggasse 6, Vienna, 1210, Austria
Visual Computing and Human-Centered Technology, TU Wien
Favoritenstraße 9-11, Vienna, 1040, Austria
andreas.kriegler@ait.ac.at, andreas.kriegler@tuwien.ac.at
http://orcid.org/0000-0002-5653-5181

Csaba Beleznai

Vision, Automation and Control, AIT Austrian Institute of Technology
Giefinggasse 6, Vienna, 1210, Austria
csaba.beleznai@ait.ac.at
http://https://orcid.org/0000-0003-1880-2979

Margrit Gelautz

Visual Computing and Human-Centered Technology, TU Wien
Favoritenstraße 9-11, Vienna, 1040, Austria
margrit.gelautz@tuwien.ac.at
http://orcid.org/0000-0002-9476-0865

Markus Murschitz

Vision, Automation and Control, AIT Austrian Institute of Technology
Giefinggasse 4, Vienna, 1210, Austria
markus.murschitz@ait.ac.at
http://orcid.org/0000-0002-5199-4602

Kai Göbel

Vision, Automation and Control, AIT Austrian Institute of Technology
Giefinggasse 6, Vienna, 1210, Austria
kai.goebel@ait.ac.at

Received 29 October 2022
Revised 7 March 2023
Accepted 18 April 2023
Published 9 August 2023

*Electronic version of an article published as: International Journal of Semantic Computing, Vol. 17, No.3, 2023, 387-410, DOI: https://doi.org/10.1142/S1793351X23620027, © copyright World Scientific Publishing Company, https://www.worldscientific.com/worldscinet/ijsc
†Corresponding author.

A considerable amount of research is concerned with the challenging task of estimating 3D pose and size for multi-object indoor scene configurations. Many existing models rely on a priori known object models, such as 3D CAD models and are therefore limited to a predefined set of object categories. This closed-set constraint limits the range of applications for robots interacting in dynamic environments where previously unseen objects may appear. This paper addresses this problem with a highly generic 3D bounding box detection method that relies entirely on geometric cues obtained from depth data percepts. While the generation of synthetic data, e.g. synthetic depth maps, is commonly used for this task, the well-known synth-to-real gap often emerges, which prohibits transition of models trained solely on synthetic data to the real world. To ameliorate this problem, we use stereo depth computation on synthetic data to obtain pseudo-realistic disparity maps. We then propose an intermediate representation, namely disparity-scaled surface normal images, which encodes geometry and at the same time preserves depth/scale information unlike the commonly used standard surface normals. In a series of experiments, we demonstrate the usefulness of our approach, detecting everyday objects on a captured dataset of tabletop scenes, and compare it to the popular PoseCNN model. We quantitatively show that standard surface normals are less adequate for challenging 3D detection tasks by comparing predictions from the model trained on disparity alone, surface normals and disparity-scaled surface normals. Additionally, in an ablation study we investigate the minimal number of training samples required for such a learning task. Lastly, we make the tool used for 3D object annotation publicly available at: `https://preview.tinyurl.com/3ycn8v5k`. A video showcasing our results can be found at: `https://preview.tinyurl.com/dzdzabek`.

Keywords: 3D bounding box prediction, unseen objects, surface normals, synthetic data, geometric primitives, object pose annotation

1. Introduction

The perception of previously unseen objects is a requirement to enable robots to operate successfully in real-world environments exhibiting vast diversity. To acquire the information necessary for interaction with these objects, approaches from 3D object recognition and object pose estimation are used. Nevertheless, many of the existing works, e.g. [4–9], require 3D models or other annotated data and as such are limited to a closed set of object classes (close-set constraint), i.e. the models are only capable of recognizing objects from a small, specific set of classes and fail at detecting novel objects from previously unseen classes. In recent works, attempts have been made to lift the closed-set constraint for 2D object detection [10] or 3D proposal generation [11] as well as object segmentation [12]. *Objectness*, or a measure for the probability that an object exists in a given region of interest [13], is an important concept in this regard. Where 2D objectness often relies on learned representations of spatial groupings (bounding box or segmentation estimates) [14], 3D objectness requires more geometry-oriented reasoning, where dimension, orientation and distance from the observer become key parameters to estimate. Therefore, for 3D objectness, typically more complex input data representations are required, e.g. RGB-D or LiDAR point clouds [15–17], segmentation masks [18] or entire CAD models [4, 19–21]. In this work, we want to learn 3D objectness and estimate oriented 3D bounding boxes of unseen objects in a class-agnostic manner, i.e. without a predefined set of object classes or CAD models.

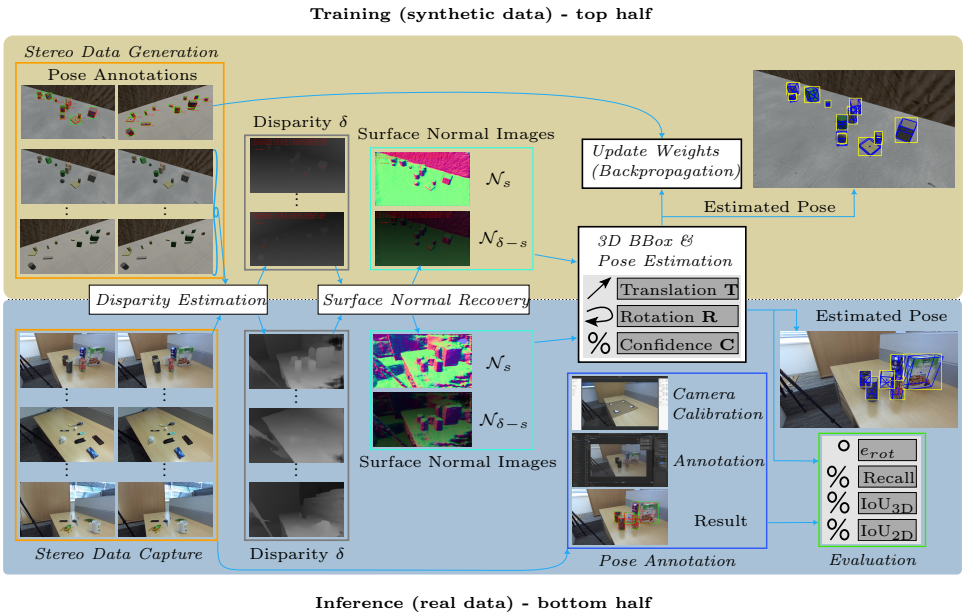


Fig. 1: The overall pipeline of our approach during training (top) and inference (bottom). **Training:** The Blender rendering engine [1] is used to create a vast data set of synthetic stereo image pairs featuring 3D geometric primitives in various configurations and view-point variations. Ground-truth location and pose annotations of all objects are exported. A stereo-matcher [2] is used to obtain dense disparity maps. We modify a keypoint based 3D detection model [3] and use the disparity maps δ, recovered SN images $\mathcal{N}_s$ or disparity-scaled SNs $\mathcal{N}_{\delta-s}$ for training. **Inference:** A data set of tabletop stereo-images featuring unseen and difficult objects was captured for evaluation. We use a novel annotation tool to obtain ground-truth 6DoF pose, location and size for all objects and evaluate the trained model for 3D bounding box prediction.

As is common in machine-learning applications, for learning-based models to perform well, large-scale high-quality data sets are required, which are very labor-intensive and challenging to obtain, especially for object-centric tasks. Using synthetic data to this end is becoming more popular [9, 22–24], but it has its own set of challenges. Most notably the data quality gap emerges when compared to real images, since image formation via real RGB cameras is governed by complex physical phenomena and it is difficult to recreate the variation and noise in RGB space of real cameras [25], requiring advanced synthesis/rendering solutions. Depth maps from stereo matching also hold the cues required for learning 3D objectness and can be fully synthesized, but a similar data quality gap to real depth becomes apparent.

On the other hand, empirical evidence [26, 27] shows that simulating stereo RGB images but then performing stereo matching for disparity estimation on the synthetic images delivers semi-synthetic depth maps similar to those estimated from real stereo images, reducing the sim-to-real gap. We take a similar approach in this work, leveraging a rendering engine to generate synthetic stereo image pairs for a subsequent disparity computation.

The type of representation used for the 3D information is an important component to facilitate the learning process. The surface normal (SN) space can be used to this end [28], where SN images are a 2D 3-channel representation of surface curvature in 3D space. They are highly scale/distance invariant - since the local orientation of a given object surface element is relatively independent from the viewing distance. This makes the learning task easier, but unfortunately any depth or metric scale information is lost when transitioning from disparity to surface normals as surface normals are spatial derivatives (slopes) of the depth data. Although surface normals alone enable pose estimation of objects, learning the object dimensions is hindered by this loss of global depth information and sense of scale. Therefore we propose to combine the SN images with disparities, scaling each normal vector by the disparity value at that pixel, which leads to a map well representing shapes (surface curvature) and also conveys information on the metric scale (distance).

To address these problems, we extend our previous work [29] and present a stereo-based method for estimation of oriented 3D bounding boxes for unknown objects that does not require CAD models. The method based on an anchorless encoder-decoder (CenterNet [3]) and trained without any real image, can estimate the pose of unknown real objects scattered on a ground plane in an end-to-end manner in form of oriented 3D bounding boxes (see Figure 1). PrimitivePose, our proposed framework named in reference to the geometric primitives used for training, is evaluated for 3D bounding box prediction on two data sets: tabletop scenarios and more general environments. For the tabletop images, we collected stereo views of difficult objects from multiple cameras and manually annotated 6DoF object poses. For eight other environments, we employ PrimitivePose on the STIOS data set [30]. In both cases we compare against the popular PoseCNN method [8]. To pave the way for an adoption of our detection framework in a real-world robotics system, where inference time and computational footprint are limited, we evaluate two different backbones with different accuracy vs. speed trade-offs. We also provide evaluations and discussions which demonstrate the usefulness of our proposed notion of disparity-scaled surface normal images. Finally, since one could generate synthetic training data ad infinitum, we provide an ablation study regarding the necessary quantity of training data. In summary, the contributions of this work are:

- A novel model for 3D object detection of unseen objects called PrimitivePose trained without real images and compared against the state-of-the-art on a novel data set of tabletop images.
- Proposition of the notion of disparity-scaled surface normal images as an

input representation, simultaneously conveying information on depth/scale and object shape.We empirically verify their efficacy as a representation for learning 3D object recognition.
- A toolkit for 6DoF pose annotation of objects made publicly available.
- Determination of the sufficient quantity of training data for a 3D detection task when quasi-infinite synthetic training data is available.

2. Related Work

We split related works into three parts. First, we present works which use synthetic data for object-centric learning tasks. We then discuss different approaches to represent object geometries and surfaces via surface normals. We conclude by comparing with existing methods for class-agnostic object detection and pose estimation.

2.1. *Synthetic data for object recognition*

Object detection and segmentation in the robotic context can benefit from synthetic data generation. A synthetic camera and randomly dropped object models were used in [22] for fast generation of synthetic depth training data for 3D object segmentation. Similarly, in [24], a vast synthetic data set was successfully used for unseen object segmentation in tabletop environments. Object meshes and a simulated stereo sensor were used in [23] for generating synthetic stereo images to learn parallel-jaw grasping models. [9] relies fully on synthetic data for training a deep pose estimation network and shows that the synth-to-real gap can be bridged, but they only evaluate their method on known objects. The authors of [31] used BlenderProc4BOP, a derivative of the Blender [1] rendering engine, for generating training images for the BOP 2020 challenge. Challenge participants were provided with 350k training images, but no explanation or motivation for this number was given. In this work, we want to combine synthetic input generated using Blender with stereo matching - a process that yields depth data comparable to real-image based stereo depth [26]. SimNet [27] similarly exploits synthetic training data and stereo information for 3D oriented bounding box estimation of small objects but only uses a low-resolution disparity image estimate as representation while we use full-resolution disparity as well as SN and disparity-scaled SN images. The authors of [27] also did not disclose the amount of training images used to train their model, while we actively try to estimate a lower bound on the number of training samples required for our models to converge.

2.2. *Object representation and surface normals*

The approximation of object structure via 3D geometric primitives is a popular idea in computer graphics literature. Fitting primitives to CAD objects in LiDAR point clouds is a common task with its own benchmark [32]. In [33], man-made objects are modeled with geometric primitives at different abstraction levels. In [34], the

authors use a superquadric object representation, a generic geometric shape model commonly used in computer graphics, for pose estimation of primitive-shaped objects. Poses of strictly cylindrical objects were estimated in [26]. For approximation of object geometries, an adequate representation of structure is important. To this end, a SN description is one possibility, used already in classical approaches [35]. In end-to-end SN estimation methods [36, 37], the normals themselves are seen as the final model output. Other works use surface normals as intermediate representation for object pose estimation [28] [38], hand pose estimation [39], pose estimation of transparent objects via RGB-D data [40] or 3D model retrieval from a CAD model library [41]. We argue for the preservation of depth information when using surface normals. Additionally none of these methods are class-agnostic.

2.3. *Class-agnostic object pose estimation*

A number of class-agnostic pose estimation methods have recently been proposed to circumvent the problem of limited object classes [18–20,42–45]. [43] requires multiple views of the same object on a turntable, [46] also needs a number of views of the object in various orientations and [47] requires a RGB video scan for object pose estimation. In [18], a detection and tracking framework of novel objects is proposed requiring an expensive segmentation step limiting real-time capabilities. [44] and [19] predict generic 3D corner points of the 3D bounding box for class-agnostic pose estimation but [44] only reports performance on seen classes and [19] requires CAD models of new objects at test time. Similarly, [20, 42, 45] all use 3D models to adapt to objects unseen during training. The authors of CenterSnap [48] propose a single-shot pipeline for 3D reconstruction and 6D pose estimation, treating object instances as spatial centers inspired by CenterNet [3]. To learn shape-codes for each object, [48] uses an auto-encoder trained on 3D shapes from a set of CAD models. In contrast to above works, we do not need 3D shape information during training or testing. We introduce an "objectness" prior with a diverse set of 3D geometric primitives and exploit synthetic data to generate a vast data set of stereo image pairs of geometric primitives with annotation labels. Our method is not limited to any set of classes, requires a stereo depth image but no LiDAR point clouds, object meshes, CAD models or multiple viewpoints.

3. Methods

We first give a description of the synthetic data generation pipeline, then provide an explanation of the pose annotation tool; for both of these, we leverage the Blender [1] software program. Then, we formally describe the calculation of surface normals given a disparity map and our proposed change of scaling these vectors with disparity. Finally, we describe the learning scheme used for 3D bounding box estimation.

3.1. *Generating synthetic data for 3D object recognition*

This is the first usage of the Blender [1] software in our pipeline. For generating synthetic training images, we arbitrarily spawn a collection of 3D compact mesh objects including cuboids, cylinders and spheres to cover the geometry of many common objects. To be clear, we are not interested in classifying objects or assigning any semantic meaning, we use these primitives so that PrimitivePose can focus on object geometry instead of appearance. The number of objects, their size, location and rotation are randomized uniformly for every configuration. A virtual stereo camera rig, parameterized as a real camera (ZED 2[a]), looks at the scene. The camera's distance to the scene, view target and elevation angle are randomized. We use the EEVEE engine from Blender to render pairs of stereo images, while extracting ground truth object parameters, such as pose and size as well as object occlusion calculated via ray-tracing. The whole process takes around 0.5 seconds per image pair, depending on renderer settings and scene complexity. The entire training set includes around $108k$ raw disparity maps, surface normal images and disparity-scaled surface normal images, all derived from the generated stereo image pairs. All randomized parameters are sampled within a certain range, so that number of objects, object size and camera distance roughly correspond to a typical tabletop scene where a robot could be tasked with detecting and manipulating objects on the table.

3.2. *Object pose annotation*

Our pose annotation tool works for uncalibrated monocular cameras assuming the size of objects is known. It builds on a classic algorithm [49] for extrinsic camera calibration, embedded in a software to obtain a Blender-readable file. If camera calibration is known otherwise, this step can be skipped - actual annotation then takes place in Blender. The entire toolkit was tested on both Windows and Ubuntu operating systems.

3.2.1. *Requirements & setup*

Every piece of software required for the annotation toolkit is freely available, specifically required are:

- A single or multiple stationary cameras providing RGB imagery.
- A calibration pattern, most simply a square.
- The fSpy software [50] or another method to obtain a calibrated camera model in Blender-readable file format.
- The Blender software [1] and fSpy plugin [51].

Blender comes with its own distribution of python and installing packages into it

[a] https://www.stereolabs.com/zed-2/

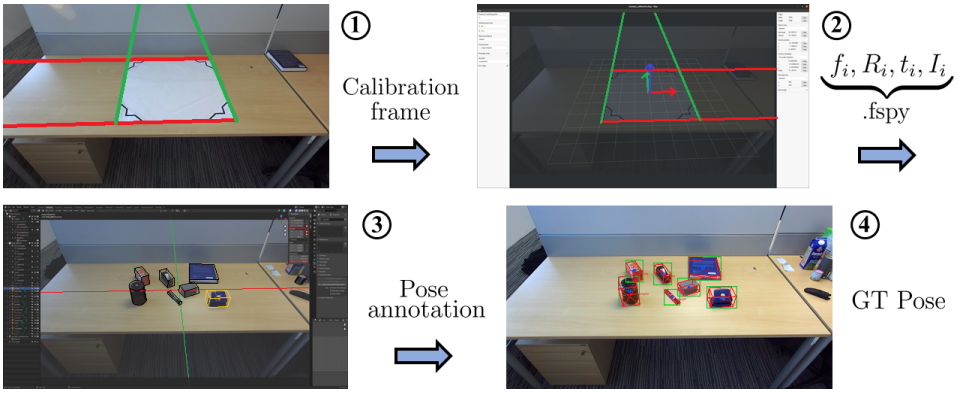


Fig. 2: 6DoF pose annotation for arbitrary objects with known size is possible for uncalibrated monocular cameras using our toolkit. After capturing a calibration frame (step ①) the pose of the camera is estimated and a virtual camera model obtained (step ②). The Blender [1] computer graphics software is then used for pose annotation of objects (step ③) and results are exported and visualized using our scripts (step ④). Best viewed in color and zoomed in.

is not entirely straight-forward. We have provided scripts that install these dependencies automatically, as well as scripts to launch Blender from the command-line or an integrated development environment. For more details regarding software setup, see the documentation in the repository.

3.2.2. *The annotation process*

The process takes place in four steps (see Figure 2).

Camera setup and data capture ①: A pattern showing two sets of two parallel lines, the sets being mutually orthogonal, has to be placed and a side length specified. In practice this can be done using a square of known side length placed somewhere visible in the scene, or one can use the edges of a table. With one frame captured for calibration purposes and the camera fixed in place, the remaining images can be collected. The functionality to use more than one camera is provided.

Camera calibration ②: Calibration takes place in the fSpy software. It is recommended to use two vanishing points for higher accuracy, and required if the camera intrinsic parameters are not known. Changes to the orientation of the coordinate frame, the principal point and other helpful settings can all be made within the software. Estimated parameters are displayed on the right side and saved with the image into a single file.

6D Object pose annotation ③: With the previously created file loaded in

Blender, annotation can commence. Depending on the requirements, objects can be modeled with only cuboid meshes, geometric primitives, recreated in Blender manually, or existing 3D models can be loaded. For our evaluation, objects with shape similar to solids of revolution, i.e. objects that can be created by rotating a planar curve around one axis of revolution, were modeled with a cylinder while all other objects were modeled with a cuboid. Depending on the task and captured images, it is very likely that not all of the objects of interest are also present in all of the images. In this case it is possible to hide the object meshes - no annotations are then exported for these objects. Annotation consists of dragging the Blender objects in 3D space to align with their counterparts on the 2D image. Since camera pose is known, this amounts to 6D object pose annotation in reference to the camera coordinate system.

Export of annotations (step ④): With all objects aligned, the provided script *get_pose_anno.py* calculates and exports pose annotations for all visible objects in the scene. Camera-object pose relations, 3D locations of objects, object size and occlusion metrics, specifically the percentage of surface points occluded by another object as well as the percentage of total image pixels an object occupies, are exported. The former occlusion metric can be used to filter out highly occluded objects, while the latter is useful for filtering out either very small or highly occluded objects. For the calculation of object occlusion, a ray-tracing algorithm sends rays from the camera focal point through every pixel of the virtual image plane and stores all intersections of the rays with object meshes. If there are multiple hits at a given pixel, the closest hit surface belongs to the visible object and every subsequent object is occluded at that pixel. The different components of the script are visually summarized in Figure 3.

Annotation takes around 3-5 minutes per image, depending on the annotator's familiarity with the software. Tthe toolkit is therefore well suited for annotation of test data sets with multiple hundred images. The tool also allows to have several cameras observing the same scene from different viewpoint. This helps create more accurate pose annotations, since 2D-3D alignments can be validated from multiple views, and yields an image pose annotation pair for each camera.

3.3. *From disparity to surface normals*

We use an off-the-shelf learning-based stereo matching model, AANet [40], to obtain disparity estimates from synthetic, generated image pairs. Although recent monocular approaches [52, 53] have delivered good results, stereo vision is still preferred for accurate depth estimation [54], since monocular depth estimation suffers from global consistency and can introduce geometric distortions [26]. We transform the disparity maps into a surface-normal representation as follows. Let δ_i be the i-th

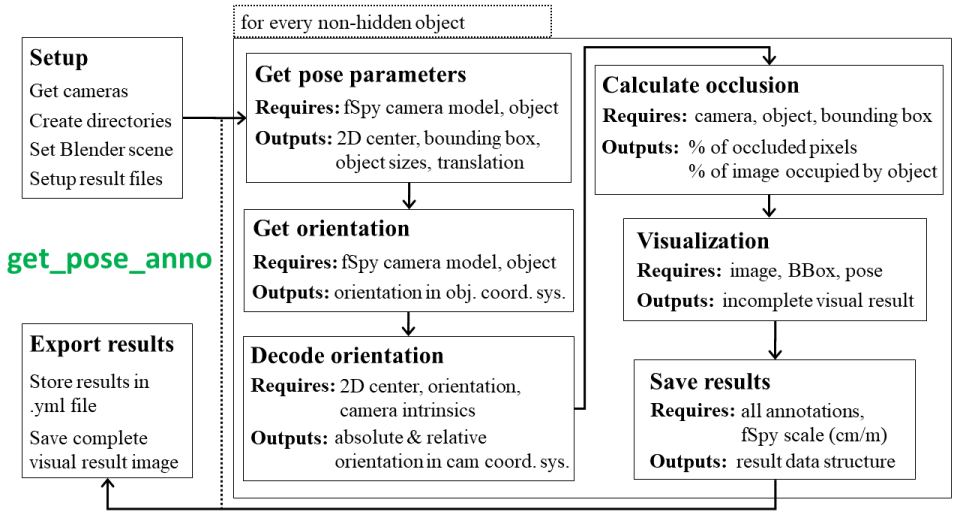


Fig. 3: Conceptual overview of the different components from the script *get_pose_anno.py* used to calculate and export 3D pose annotations for objects.

disparity image and d_i the resulting depth image:

$$d_i = \frac{f \cdot b}{\delta_i + \epsilon} \tag{1}$$

where f is the horizontal focal length in pixels, b is the baseline in millimeters and ϵ a very small constant. We compute the two-dimensional gradient images $\nabla_x, \nabla_y \in \mathbb{R}^{w \times h \times 1}$ in x and y direction via convolution of the image with Sobel kernels of size $k = 3$, using the OpenCV implementation[b], where $w = 896$ and $h = 512$ are the width and height of the image:

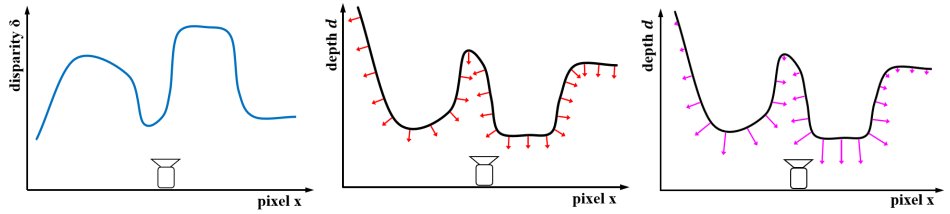
$$\nabla_x = Sobel(d_i, 1, 0, k) \qquad \nabla_y = Sobel(d_i, 0, 1, k). \tag{2}$$

The vector field of all surface normals $\mathcal{N}_i \in \mathbb{R}^{w \times h \times 3}$ to all supporting gradient planes can then be constructed and normalized to unit length via the Frobenius matrix norm:

$$\mathcal{N}_i = \frac{[\nabla_x, \nabla_y, \nabla_z]}{||[\nabla_x, \nabla_y, \nabla_z]||_F} \tag{3}$$

Here the three two-dimensional gradient images are stacked channel-wise. For estimates of the gradient map ∇_z see [37] – we have found in practice that making the simplifying assumption $\nabla_z = \mathbf{1}$ works similarly well, i.e. the models trained on the surface normals show similar convergence behavior and performance on the

[b] https://docs.opencv.org/3.4/d2/d2c/tutorial_sobel_derivatives.html



(a) Disparity δ (b) Surface normals $\mathcal{N}_s$ (c) δ-scaled SN $\mathcal{N}_{\delta-s}$

Fig. 4: In a simplified 2D scene with two objects, the object surfaces visible to the camera could result in disparity estimates as shown in (a). On depth $d \propto \delta^{-1}$ the surface normals along the contour are calculated. If only this 3D vector is used for object shape representation, information of distance and scale are lost, which motivates our usage of disparity-scaled surface normals $\mathcal{N}_{\delta-s}$. The 3D vector length is now scaled by the disparity value at that pixel.

validation set. For every pixel of $\mathcal{N}_i$, the three channels express the surface normal direction at that pixel. The normals are then scaled from $[-1, 1]$ to $[0, 1]$:

$$\mathcal{N}_i = 0.5 \cdot (\mathcal{N}_i + 1). \tag{4}$$

A standard surface normal image $\mathcal{N}_{s,i}$ is then obtained by scaling each of the three image channels, each encoding one dimension of the 3D surface normal vector, to $[0, 255]$.

$$\mathcal{N}_{s,i} = \mathcal{N}_{s,i} \cdot 255 \tag{5}$$

For the disparity-scaled surface normals $\mathcal{N}_{\delta-s,i}$, first a scaling factor s is obtained for $[0, 255]$ range via

$$s = \frac{255.0}{max\_disp} \tag{6}$$

where the maximum disparity value max_disp is 192 for AANet [40]. All three channels of the original disparity maps are then scaled with this factor

$$\delta_i = [\delta_{i,1} \cdot s, \delta_{i,2} \cdot s, \delta_{i,3} \cdot s] \tag{7}$$

and stacked for a three-channel disparity vector field. To obtain the final disparity-scaled surface normals $\mathcal{N}_{\delta-s,i}$, the original unit normals $\mathcal{N}_i$ are scaled with the vector field δ_i:

$$\mathcal{N}_{\delta-s,i} = \mathcal{N}_i \cdot \delta_i. \tag{8}$$

This representation encodes the orientation of the 3D surface normal vector at every pixel (proportion of RGB values), while the length of the vector (magnitude of RGB values) encodes the disparity information. For a conceptual visualization see Figure 4.

In practice, this results in brightness changes of the (color-encoded) surface normal image which are inverse proportional to the depth at each pixel (see middle part of Figure 1).

3.4. *3D Object pose parameterization*

Lastly we require an adequate and learnable representation of object pose and size, with respect to the camera position and orientation. We make the simplifying assumption that all objects are either lying flat on the table surface or are standing upright. In other words, the only rotation of an object that we are concerned with is the one about its gravity vector or the yaw angle ψ_{obj}, with pitch and roll angles $\theta_{obj} = \phi_{obj} = 0$. An object then only has four degrees of freedom, which helps us avoid problems of symmetry common in full SO(3) group transformations. Since we do care about the pose in reference to the camera, the camera's pitch θ_{cam} and yaw ψ_{cam} angles are relevant, meaning we need to predict three parameters of translation x, y, z and two relative angles $\psi_{obj}^{cam}, \theta_{obj}^{cam}$, all in reference of the camera coordinate system.

Still, there are other potential symmetries that we have to handle. First, we define the frontal face of an object to be the vertical side directly facing the camera if all rotations are zero and camera and object are in line. Regardless of primitive, we then clamp $\psi_{obj}^{cam} \in [\frac{-\pi}{2}, \frac{\pi}{2}]$ and $\theta_{obj}^{cam} \in [0, \frac{\pi}{2}]$ to avoid ambiguities. Where applicable, due to the rotational symmetry of cylinders, we set $\psi_{obj}^{cam} = 0$, both in the training data and evaluation data. For cuboids, we need to create constraints taking into account the different side lengths. We define the following canonic pose: With zero rotations and object and camera in line - i.e. $\theta_{obj} = \theta_{cam} = 0$, $\psi_{obj} = \psi_{cam} = 0$, $y = 0$, $x > 0$ and $z < 0$ - the cuboid's bottom edge parallel to the horizon is set to be y and defined to be the shorter side. The bottom edge normal to y and "vanishing" towards the horizon is then edge x and the z-axis follows right-hand coordinate frame convention. This means that, if the bottom horizon-parallel edge were to be longer, it is actually the xz-side facing the camera and the cuboid is rotated with $\psi_{obj} \approx \pm 90°$. With these constraints, most ambiguities are appropriately resolved (except for upright square prisms, which we leave for future work). This parameterization allows us to use simple Euler angles for representing the pose of objects relative to the camera pose, the objective becomes learnable and the model is incentivized to learn object size and rotations in tandem.

4. Experiments

In sections 4 and 5, we aim to answer the following questions with experiments: (1) How well does PrimitivePose estimate oriented 3D bounding boxes around objects on tabletops from a stereo observation? (2) Can we lower the inference time with a lower-memory network while maintaining good accuracy? (3) Which intermediate data representation (disparity, surface normals, disparity-scaled surface

normals) is best suited for this task? (4) Does PrimitivePose transfer to more complex environments and objects? (5) Can a optimal number of training samples be found for such a task?

To answer the first three questions we report results on our captured data set of tabletop images in subsection 5.1, for question four we use the STIOS [30] data set (subsection 5.2) and for question five our synthetic validation set (subsection 5.3).

4.1. *Experimental setup*

Using a center-point detection architecture we optimize multiple learning tasks to directly regress bounding box parameters around any objects in an image. PrimitivePose, a modified version of CenterNet [3], learns to estimate 3D oriented bounding boxes using 17 parameters in total grouped according to their loss functions: $\mathcal{L}_{hm}$ - 3 parameters for the heatmap-based 2D object center, $\mathcal{L}_{ang}$ - 4 parameters for the relative object-camera rotations ($sin(\cdot)$ and $cos(\cdot)$ for the two angles $\psi_{obj}^{cam}, \theta_{obj}^{cam}$), $\mathcal{L}_{dim}$ - 3 parameters for object dimensions, $\mathcal{L}_{dep}$ - 1 parameter for the depth, i.e. the distance between object and camera center along the view-axis, and $\mathcal{L}_{box}$ - 4 parameters for the 2D bounding box to enable 2D IoU correspondence matching, although this is not a necessity for 3D object detection itself. One could learn the $3D$ translation vector directly but we have found in practice that learning the backprojected $2D$ center of an object and the distance or depth yields better results. With known intrinsic camera parameters the transformation is a simple linear operation. We modify CenterNet [3] by adding additional heads to the network architecture, enabling us to learn pose, location and size parameters in addition to the standard 2D objectives.

Since short inference time is desirable for the PrimitivePose framework to enable real-world adoption, and our previous version [29] used the computationally expensive DLA [55] backbone architecture, we exchanged DLA for the HarDNet [56] backbone, which decreases inference time by roughly 40%. This necessitated a few changes regarding the hyperparameters of the model. While originally with DLA we used exclusively L1 loss, weighting each term equally, we now use L1 loss for all but $\mathcal{L}_{hm}$ where we use FocalLoss [57] instead. Additionally, loss terms $\mathcal{L}_{hm}$ and $\mathcal{L}_{box}$ are weighted with $w_{hm} = 10.0$ and $w_{box} = 8.0$ respectively. DLA-models were trained for 23 epochs with a learning rate of $0.5e-3$ decayed by a factor of ten at epoch 20, whereas HarDNet-models were trained for 16 epochs, with a learning rate of $1.0e-4$ decayed by a factor of ten at epochs seven and eleven. Changes to the learning rate and decay were made following preliminary experiments and evaluations, whereas the number of epochs was reduced since both models converge very quickly (epochs five to ten) and training the DLA-variants for 23 epochs was largely unnecessary in the first place; this holds true regardless of the intermediate representation $(\delta, \mathcal{N}_s, \mathcal{N}_{\delta-s})$. We use a detection threshold of 0.2 in all experiments.

4.2. *Dataset*

The tabletop test set consists of roughly 200 images from multiple camera viewpoints showing a generic wooden table with various objects scattered on it. These objects include compact cuboid-like objects such as books, objects with an axis of rotational symmetry such as markers and cans, objects that are cylindrical but have an additional identifying feature such as handles on cups, as well as challenging non-compact objects like a pair of scissors and plastic toy bugs with outreaching appendages. No fiducial markers are required, and all objects are entirely novel to the system at inference. Only the geometry of some of the objects is related to the 3D primitives seen during training. From the pool of roughly 25 objects, we created three evaluation subsets: only using large objects, only using small objects, or a mixture of the two, where we made the distinction using the volume of the 3D bounding box with a cut-off at the median.

4.3. *Metrics*

Since we do not use any 3D models at any stage, we cannot adopt the popularly used ADD metric [58] that compares model points. Instead, an object pose, or oriented 3D bounding box, is considered correctly predicted if the scalar rotational error $e_{rot} \in [0, 180]$ [59]

$$e_{rot} = \arccos\left((Trace(\hat{\mathbf{R}}\bar{\mathbf{R}}^{-1}) - 1)/2\right) \times \frac{180}{\pi} \tag{9}$$

is below a certain threshold $\theta \in \{2, 5, 10, 15, 25, 40\}^\circ$. Ground truth and predicted rotation matrices $\bar{\mathbf{R}}$ and $\hat{\mathbf{R}}$ are created using the general extrinsic rotation matrices from the three ground-truth and predicted Euler angles.
Furthermore, we report 3D IoU between predicted and true 3D bounding boxes, using the method from Objectron [60].

4.4. *Correspondence matching*

After obtaining bounding box predictions from inference, we still have to solve the correspondence problem: matching ground truth and predicted bounding boxes. It should be noted at this point that, due to imperfect camera calibration, the ground truth distance from camera to object center following manual annotation can be off by a few centimeters, which could have significant impact on calculated 3D IoU considering many of our objects are flat or small. Therefore, to report 3D detection results (e_{rot}), we use 2D IoU or the Jaccard index [61] for a more stable object matching. To report 3D IoU, we use the 3D IoU itself to find matches, but allow a tolerance of $d_{tol} \in \{0, 4, 8, 12, 16, 20\}\%$ in depth. To do so, we uniformly sample 3D bounding box candidates along the vector connecting camera and object centers. Finally, we use the occlusion measure calculated automatically during annotation to filter all objects with occlusion greater than 90% for the evaluation.

4.5. *PoseCNN comparison*

We compare pose predictions from PrimitivePose against PoseCNN [8]. PoseCNN was originally trained on 21 YCB objects [62] which look, at least geometrically, similar to ours, which justifies our comparison against the method. We apply PoseCNN on the left stereo image resized to $640x480$, using the default detection threshold of 0.2 and changing the intrinsic camera matrix for model reconstruction accordingly. PoseCNN predicts 3D pose in the form of oriented 3D YCB object models, which enables a comparison in terms of pose prediction recall. Unfortunately, since the actual size of the YCB objects or the 3D bounding box encompassing them, is not publicly reported, we cannot make the transition from 3D object models to bounding boxes for evaluating 3D IoU. Lastly, it is not possible to retrain PoseCNN on the same synthetic data PrimitivePose was trained on, since PoseCNN requires 3D YCB object models already during training.

5. Results

5.1. *Tabletop results*

The qualitative results in Figure 5 show that PrimitivePose predicts accurate 3D bounding boxes for most objects. Not much difference exists between the two backbones, it seems the faster HarDNet backbone in some cases even delivers more accurate results (see for example the predictions in the top left corner). PoseCNN struggles in detecting most of the objects, most likely since PoseCNN relies heavily on RGB information and while YCB and our objects share geometric features, texture and colors are different.

Observing all visual predictions, problems with the plastic toy bugs, where predicted bounding boxes are consistently too small, and the coffee/tea cups, where the rotation indicated by the handle is rarely inferred correctly, become apparent in Figure 6. Neither of these failure cases are surprising, since non-compact objects or objects with cavities were not among the objects rendered for the training data in the former case and the model was trained to predict bounding boxes around cylindrical objects to face the camera directly for the latter case. No significant difference between intermediate representations or model backbones are noticeable.

Table 1 and Table 2 provide detailed quantitative detection results for backbones DLA and HarDNet, respectively. Table 1 also shows quantitative results for PoseCNN, confirming the poor results observed visually. Looking at Table 1 a few things are noticeable: if only the rotational error e_{rot} is taken into account (upper half), both raw disparity maps δ and disparity-scaled surface normals $\mathcal{N}_{\delta-s}$ produce good results, specifically for the common error threshold of $e_{rot} <^{\circ}$ predictions range from 40% to 60% . Perhaps surprising to see is that predictions are consistently more accurate for smaller than for larger objects. One possible explanation is that the larger objects often have 2 sides of roughly equal length (square prisms), leading to a possible 90° flipping ambiguity. PoseCNN gives poor results, most

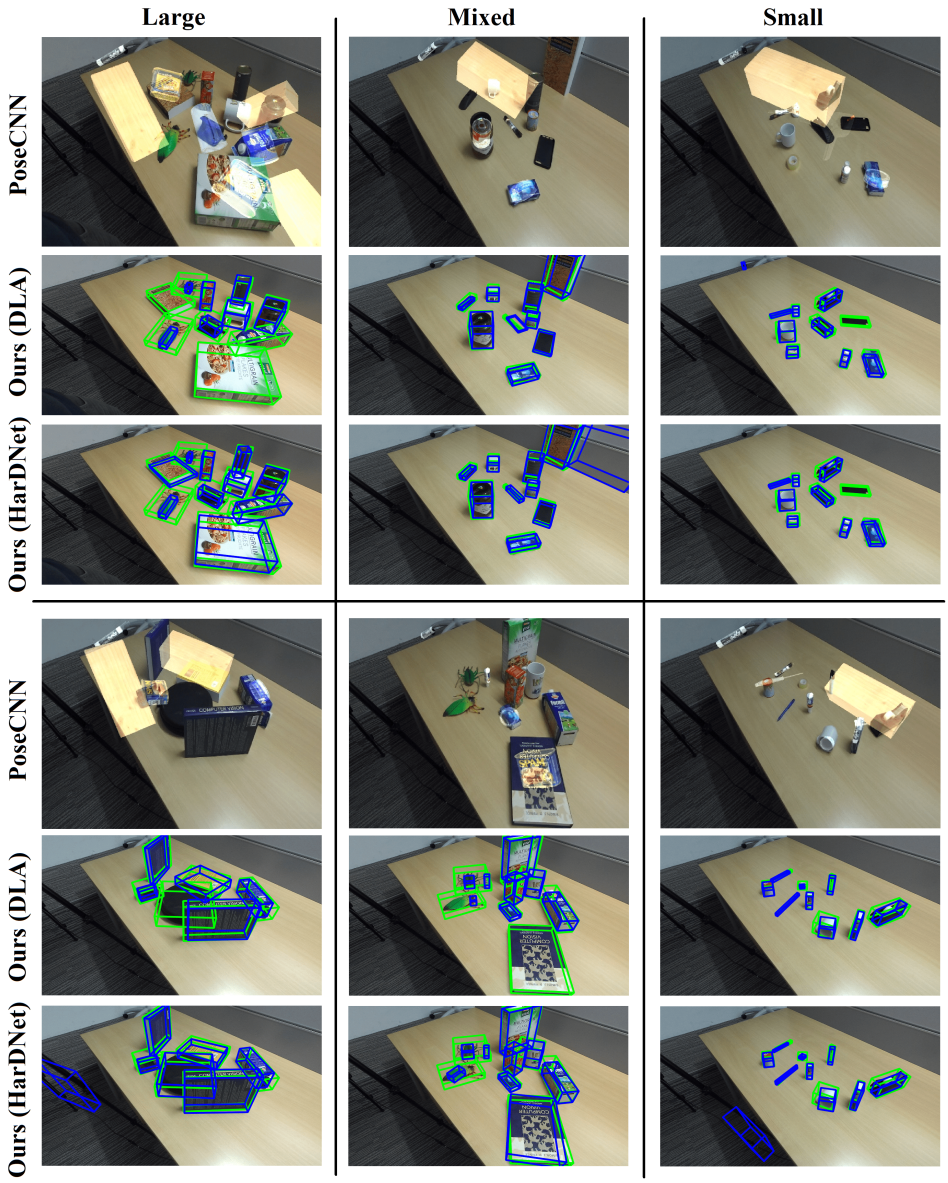


Fig. 5: Comparing predictions from PoseCNN [8] (rows one and four) and PrimitivePose with the DLA [55] (rows two and five) and HarDNet [56] backbone (rows three and six) on a set of tabletop images, splitting objects into three bins according to their size (3D bounding box volume). Ground truth bounding boxes are green, predictions blue.

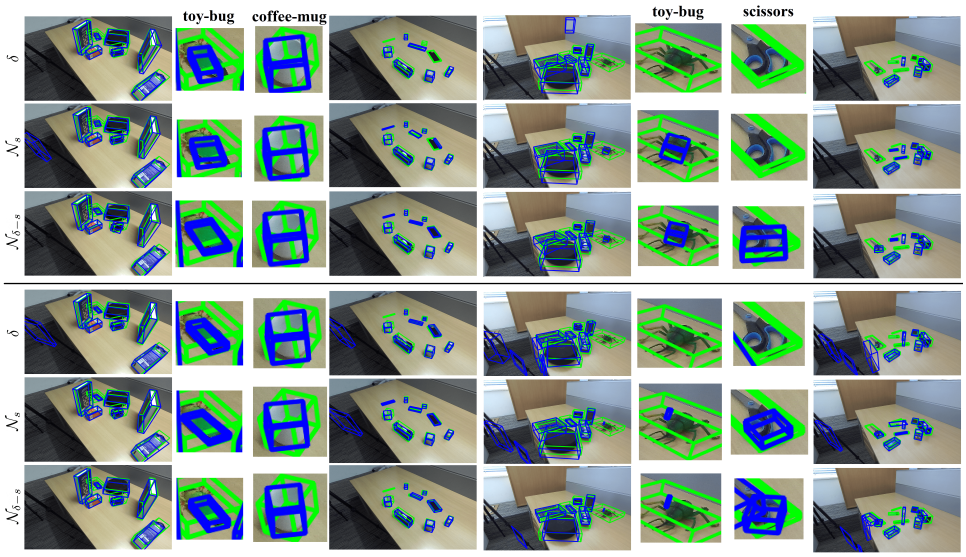


Fig. 6: Failure cases: Objects that PrimitivePose struggles with (DLA backbone on the top, HarDNet backbone on the bottom). Errors happen most commonly for non-compact objects (toy-bugs, scissors) or due to rotational symmetry (mug). Ground truth bounding boxes are green, predictions blue.

likely due to appearance changes of the objects compared to YCB, highlighting the importance of geometric cues. For 3D IoU seen in the lower half of the table, the loss of depth information with the standard surface normals $\mathcal{N}_s$ leads to expected inferior results. It is still interesting to see that raw disparity alone seems to be the best suited data representation, across object sizes and thresholds. Regarding the comparison with PoseCNN it should be noted that the actual size of the YCB objects or the 3D bounding box encompassing them is not publicly reported, we cannot make the transition from 3D object models to bounding boxes for evaluating 3D IoU.

The merit of the disparity-scaled surface-normal representation $\mathcal{N}_{\delta-s}$ becomes obvious from Table 2 where it achieves top results across error thresholds and object sizes.

Table 3 lastly summarized these findings, averaging across all thresholds and tolerances. PrimitivePose with the HarDNet backbone trained on $\mathcal{N}_{\delta-s}$ achieves the best results across the board. This is particularly noticeable for 3D IoU. Analyzing the estimated parameters in more detail revealed that PrimitivePose + HarDNet is better at capturing the correct correspondence between object size and distance. To "nudge" the optimization process in the correct direction for learning the desired function, the surface-normals appear to be only useful if depth information is also preserved.

Table 1: Pose prediction recall (e_{rot}, upper half) and 3D IoU (IoU_{3D}, lower half) for PrimitivePose with the DLA backbone: The models were trained on representations raw disparity δ, surface normals $\mathcal{N}_s$ or disparity-scaled surface normals $\mathcal{N}_{\delta-s}$ and applied on our tabletop test set. The upper half additionally gives results using the model from [8]. We split our test set equally according to the object-sizes (volume of the 3D bounding box) into images showing only small objects, only large objects or a mixture of the two. **Bold** numbers emphasize the best representation for a given error threshold and object size.

Object size	Large				Mixed				Small			
Representation	δ	$\mathcal{N}_s$	$\mathcal{N}_{\delta-s}$	[8]	δ	$\mathcal{N}_s$	$\mathcal{N}_{\delta-s}$	[8]	δ	$\mathcal{N}_s$	$\mathcal{N}_{\delta-s}$	[8]
$e_{rot} < 2°$	**27.5**	19.1	22.0	0.0	**22.5**	14.6	19.3	0.0	**47.7**	29.1	29.7	0.0
$e_{rot} < 5°$	**38.6**	33.6	37.2	0.0	32.8	26.4	**34.5**	1.2	**55.6**	40.0	43.4	0.0
$e_{rot} < 10°$	**45.5**	42.3	45.3	2.5	39.2	35.7	**42.1**	1.6	**60.3**	45.0	50.4	0.8
$e_{rot} < 15°$	49.9	47.5	**50.7**	3.7	43.6	40.8	**46.9**	2.3	**62.7**	48.0	54.6	1.1
$e_{rot} < 25°$	53.8	51.7	**55.8**	4.4	47.1	44.9	**50.8**	3.7	**65.0**	51.2	57.6	4.1
$e_{rot} < 40°$	57.1	55.5	**59.8**	7.4	49.9	48.2	**54.2**	9.3	**67.4**	54.0	60.5	10.6
IoU_{3D} $d_{tol} = 0\%$	**15.3**	6.8	12.4	–	**12.2**	1.8	7.0	–	**5.0**	0.2	2.4	–
IoU_{3D} $d_{tol} = 4\%$	**20.0**	9.1	15.9	–	**16.7**	3.0	10.2	–	**8.6**	0.6	4.0	–
IoU_{3D} $d_{tol} = 8\%$	**24.2**	11.7	19.2	–	**20.5**	4.7	13.4	–	**12.4**	1.5	6.4	–
IoU_{3D} $d_{tol} = 12\%$	**27.3**	14.4	21.9	–	**23.0**	6.6	16.3	–	**15.5**	3.0	8.7	–
IoU_{3D} $d_{tol} = 16\%$	**29.5**	16.8	24.0	–	**24.7**	8.5	18.6	–	**17.6**	4.7	10.6	–
IoU_{3D} $d_{tol} = 20\%$	**31.0**	18.9	25.7	–	**25.9**	10.2	20.3	–	**19.0**	6.1	12.3	–

Table 2: Pose prediction recall (e_{rot}, upper half) and 3D IoU (IoU_{3D}, lower half) for PrimitivePose with the HarDNet backbone, analogously to Table 1.

Object size	Large			Mixed			Small		
Representation	δ	$\mathcal{N}_s$	$\mathcal{N}_{\delta-s}$	δ	$\mathcal{N}_s$	$\mathcal{N}_{\delta-s}$	δ	$\mathcal{N}_s$	$\mathcal{N}_{\delta-s}$
$e_{rot} < 2°$	22.3	19.4	**25.4**	24.5	**28.5**	24.2	33.9	28.4	**38.7**
$e_{rot} < 5°$	34.6	36.8	**39.7**	35.4	32.8	**37.8**	44.4	41.8	**47.2**
$e_{rot} < 10°$	45.6	48.5	**50.1**	44.2	42.0	**45.3**	**58.6**	48.6	51.5
$e_{rot} < 15°$	52.3	55.0	**56.0**	49.0	47.3	**49.9**	**54.5**	52.8	54.4
$e_{rot} < 25°$	57.2	59.5	**60.3**	52.1	50.8	**52.7**	**57.3**	55.6	56.8
$e_{rot} < 40°$	61.1	63.1	**63.8**	54.7	53.5	**55.2**	**59.7**	58.1	59.2
IoU_{3D} $d_{tol} = 0\%$	20.3	18.5	**21.4**	13.2	9.2	**14.4**	3.9	2.2	**4.7**
IoU_{3D} $d_{tol} = 4\%$	26.5	24.1	**27.9**	18.6	13.0	**19.7**	7.9	4.7	**9.7**
IoU_{3D} $d_{tol} = 8\%$	30.6	27.9	**32.3**	22.4	16.0	**23.6**	11.6	7.6	**13.8**
IoU_{3D} $d_{tol} = 12\%$	33.2	30.5	**35.0**	24.7	18.6	**26.2**	14.2	10.1	**16.8**
IoU_{3D} $d_{tol} = 16\%$	34.9	32.3	**36.8**	26.3	20.5	**27.9**	16.0	12.0	**18.8**
IoU_{3D} $d_{tol} = 20\%$	36.1	33.6	**38.0**	27.4	21.8	**29.1**	17.3	13.4	**20.2**

5.2. *Application on the STIOS dataset*

To showcase the potential of our framework for application in other environments, we evaluate on the STIOS [30] data set. It was chosen because: (1) it provides stereo images, which is a necessity for our method; (2) all objects in it are YCB

Table 3: Condensed 3D detection results for different representations, backbones and across the three subsets. Models trained with our proposed representation via disparity-scaled surface normals $\mathcal{N}_{\delta-s}$ achieve the best results across the board.

Representation		δ		$\mathcal{N}_s$		$\mathcal{N}_{\delta-s}$	
Backbone		DLA	HarDNet	DLA	HarDNet	DLA	HarDNet
Recall	Large	45.4	45.5	41.6	47.1	45.1	**49.2**
	Mixed	39.2	43.3	35.1	41.2	41.3	**44.2**
	Small	46.3	50.1	44.6	47.6	49.4	**51.3**
	Average	45.2	46.3	40.4	45.0	45.3	**48.1**
IoU_{3D}	Large	24.6	30.3	13.0	27.8	19.9	**31.9**
	Mixed	20.5	22.1	5.8	16.5	14.3	**23.5**
	Small	13.0	11.8	2.7	8.3	7.4	**14.0**
	Average	18.6	20.7	6.4	16.8	13.9	**22.5**

objects and were seen by PoseCNN; (3) many objects can be approximated with a 3D primitive enabling application of our model; (4) objects lie flat on the surface and are not stacked on top of each other. STIOS contains recordings from a stereo ZEDCam in eight different environments. Roughly 25 images exist for each of the eight environments - images taken from different camera poses and/or showing mixed configurations of objects. Ground truth segmentation masks, 2D bounding boxes derived from the masks and object class labels are provided but no pose annotations, which is why we only provide qualitative prediction results. As Figure 7 shows, predictions are better in uncluttered environments that resemble tabletops, like the wooden-table, white-table or office-carpet. In many STIOS images, the objects of interest are fairly close to the camera, which resulted in the stereo matching algorithm [2] reaching maximum disparity (which cannot be increased by the user) and the disparity maps becoming corrupted. Of course, the camera used in STIOS is also not exactly the same, which has an impact on 2D to 3D reprojection predictions, for example the object centroid. Even so, a reasonable amount of objects were recognized and sensible 3D bounding boxes predicted, as confirmed by Figure 7.

5.3. *Training set size ablation study*

With the emergence of tools to synthetically generate data, no clear limit exists for the necessary size of the training set, yet it is reasonable to assume that after a certain point a model would no longer benefit from additional samples, or only marginally, hitting a point of saturation. In an ablation study we try to find this point, while also considering the three different possible representations disparity δ, surface normals $\mathcal{N}_s$ and disparity-scaled surface normals $\mathcal{N}_{\delta-s}$, investigating their efficacy of pushing the learning model towards convergence while using as little training data as necessary. Six different sizes of mutually exclusive training sets

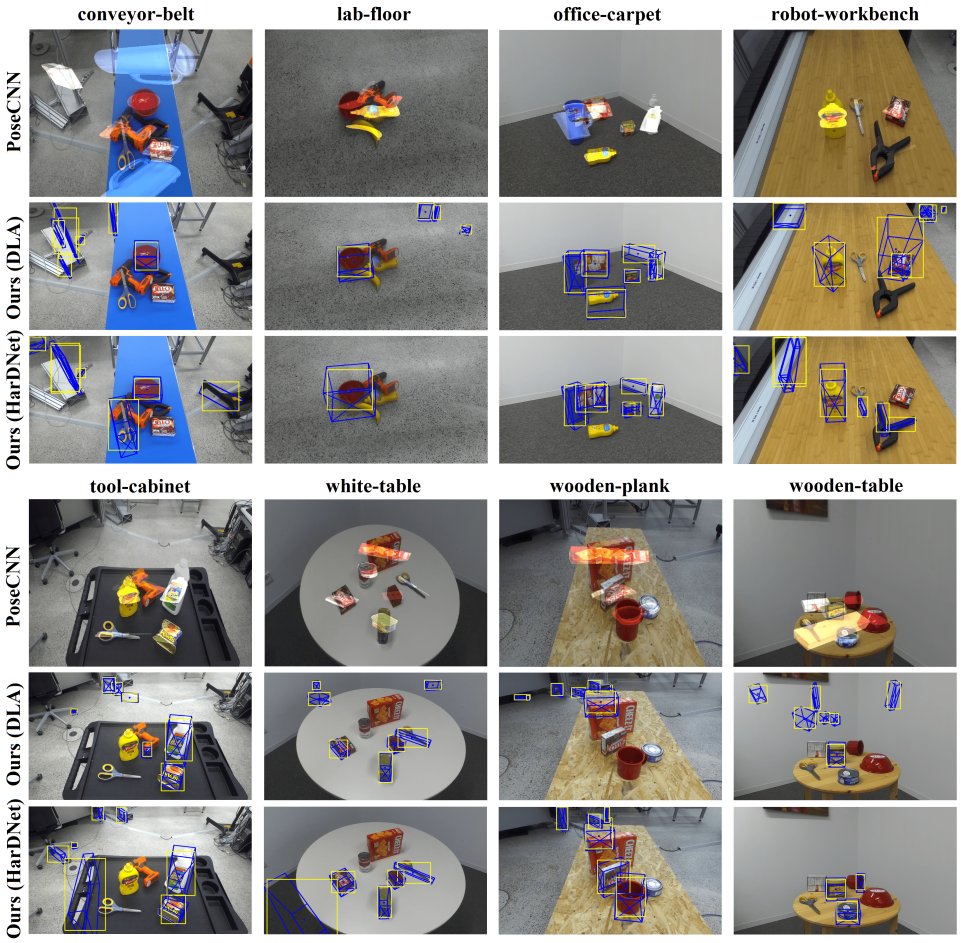


Fig. 7: 3D bounding box predictions on the STIOS [30] data set. Our model correctly detects many objects and gives reasonable orientation and size estimates even though it has never seen any of the objects. Blue boxes: 3D pose estimates, yellow rectangles: 2D object bounding boxes.

($1k$, $5k$, $10k$, $25k$, $60k$, $108k$) were used. The metric we use for evaluation is simply minimal achieved validation loss while training the model, running a validation epoch after every training epoch, where we put aside 10% of the entire data set for validation. Figure 8 shows our findings with the DLA backbone on the left and HarDNet on the right.

The absolute value of the loss should not be considered, since the loss functions are different for the two backbones. We observe that the convergence behavior for the HarDNet variant on the right appears to be more stable, and that saturation appears to be reached around $100k$ training samples. PrimitivePose with DLA

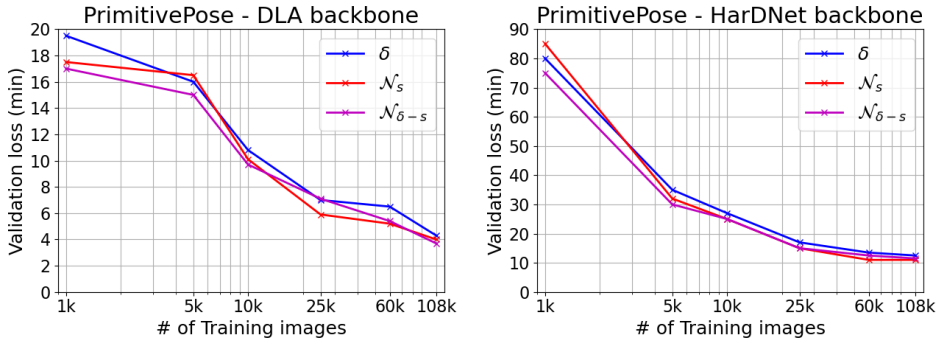


Fig. 8: Convergence behaviour of the PrimitivePose model with backbones DLA (left) and HarDNet (right) and three different intermediate data representations (disparity δ, surface normals $\mathcal{N}_s$ and disparity-scaled surface normals $\mathcal{N}_{\delta-s}$) over the number of training samples.

does not seem to have reached saturation yet and shows a somewhat more erratic behavior.

Regarding the representations, with few training samples available, the strong structure inherent to the disparity-scaled surface normals $\mathcal{N}_{\delta-s}$ seem to be beneficial for model training, an effect that diminishes with more samples available. It is also noteworthy, that models trained on the standard normals $\mathcal{N}_s$ achieved the worst performance on our test set (see Table 3 middle block), despite reaching a loss very similar to models trained with the other representations, hinting at the importance of not only the neural network architecture and hyperparameters but also intermediate data representation.

6. Conclusion

In this paper we presented an approach to predict generic oriented 3D object bounding boxes for previously unseen objects from stereo data. Our models were trained on representations derived from stereo depth computed exclusively from synthetic stereo image pairs and yield a good generalization on real images. We analyzed three intermediate representations - raw disparity maps, surface normal images and a novel representation via disparity-scaled surface normal images - and showed the importance of preserving depth information. We evaluated our models on a set of self-recorded real images showing difficult tabletop scenes with arbitrary, unseen objects and drew comparisons with the popular PoseCNN method. We improved inference time and prediction accuracy from previous work by exchanging the model backbone. We further applied our method PrimitivePose on a second data set and show promising 3D detection results even though our model has never seen any of the objects and only relies on generic 3D geometric cues. Finally, an ablation study demonstrates that, $60k$ to $100k$ training samples more than suffice to achieve

good model convergence for this task. We believe our approach is useful towards an increasingly open-ended object recognition task in a robotic context. Alternatively, it could also be used for object-centric event detection in video streams, in combinations with frameworks such as [63].

7. Limitations and Future Work

A few assumptions and limitations were made for PrimitivePose in this work: Detected objects can be reasonably approximated by a single 3D geometric primitive, requiring object compactness. Thin and/or deformable objects are also problematic due to the difficulty in finding coherent surface normals. Objects are also expected to lie flat on the ground plane (table), which can avoid problems regarding symmetry of pose under SO(3) transformations. Although randomized within a range, there is a certain bias regarding scene parameters, i.e. camera distance from the scene, object sizes etc. If objects are close to one another and similarly oriented, they might get merged since the system has no RGB information and cannot distinguish between them in such edge cases. By relaxing some of these constraints, the suitability of our approach for more complex scenes including, for example, overlapping and rotated objects could be shown in the future.

References

[1] BlenderOnlineCommunity, Blender - a 3d modelling and rendering package (2018).
[2] H. Xu and J. Zhang, Aanet: Adaptive aggregation network for efficient stereo matching, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Seattle, 2020), pp. 1959–1968.
[3] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang and Q. Tian, Centernet: Keypoint triplets for object detection, in *International Conference on Computer Vision (ICCV)* (Seoul, 2019), pp. 6569–6578.
[4] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song and L. J. Guibas, Normalized object coordinate space for category-level 6d object pose and size estimation, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Long Beach, 2019), pp. 2637–2646.
[5] G. Brazil and X. Liu, M3d-rpn: Monocular 3d region proposal network for object detection, in *International Conference on Computer Vision (ICCV)* (Seoul, 2019), pp. 9286–9295.
[6] H. Bradler, A. Kretz and R. Mester, Urban traffic surveillance (uts): A fully probabilistic 3d tracking approach based on 2d detections, in *Intelligent Vehicles Symposium (IV)* (nagoya, 2021), pp. 1198–1205.
[7] Z. Qin, J. Wang and Y. Lu, Monogrnet: A general framework for monocular 3d object detection, *Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **44** 5170–5184 (2022).
[8] Y. Xiang, T. Schmidt, V. Narayanan and D. Fox, Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes, in *Robotics: Science and Systems XIV (RSS)* **abs/1711.0**, (Robotics: Science and Systems Foundation, 2018).
[9] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox and S. Birchfield, Deep object pose estimation for semantic robotic grasping of household objects, in *Conference on Robot Learning (CoRL)* (Zürich, 2018), pp. 306–316.

[10] A. Jaiswal, Y. Wu, P. Natarajan and P. Natarajan, Class-agnostic object detection, in *Winter Conference on Applications of Computer Vision (WACV)* (Waikoloa, 2021), pp. 919–928.

[11] X. Chen, K. Kundu, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler and R. Urtasun, 3d object proposals for accurate object class detection, in *Neural Information Processing Systems (NIPS)* (Montreal, 2015), pp. 424–432.

[12] M. Durner, W. Boerdijk, M. Sundermeyer, W. Friedl, Z. C. Marton and R. Triebel, Unknown object segmentation from stereo images, in *IEEE International Conference on Intelligent Robots and Systems* (Prague, 2021), pp. 4823–4830.

[13] S. Liang, B. Wu, Y. Fan, X. Wei and X. Cao, Parallel rectangle flip attack: A query-based black-box attack against object detection, in *International Conference on Computer Vision (ICCV)* (Montreal, 2021), pp. 7697–7707.

[14] B. Xiong, S. D. Jain and K. Grauman, Pixel objectness: Learning to segment generic objects automatically in images and videos, *Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **41** 2677–2692 (8 2019).

[15] Y. Shi, J. Huang, X. Xu, Y. Zhang and K. Xu, Stablepose: Learning 6d object poses from geometrically stable patches, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Nashville, 2021), pp. 15217–15226.

[16] D. M. de Oliveira, C. C. B. Viturino and A. G. S. Conceicao, 6d grasping based on lateral curvatures and geometric primitives, in *Latin American Robotics Symposium (LARS)* (Natal, 2021), pp. 138–143.

[17] K. Yang and X. Chen, Unsupervised learning for cuboid shape abstraction via joint segmentation from point clouds, *Transactions on Graphics* **40** 1–11 (2021).

[18] B. Wen and K. Bekris, Bundletrack: 6d pose tracking for novel objects without instance or category-level 3d models, in *Conference on Intelligent Robots and Systems (IROS)* (Prague, 2021), pp. 8067–8074.

[19] G. Pitteri, S. Ilic and V. Lepetit, Cornet: Generic 3d corners for 6d pose estimation of new objects without retraining, in *International Conference on Computer Vision Workshops (ICCVW)* (Seoul, 2019), pp. 2807–2815.

[20] G. Pitteri, A. Bugeau, S. Ilic and V. Lepetit, 3d object detection and pose estimation of unseen objects in color images with local surface embeddings, in *Asian Conference on Computer Vision (ACCV)* (Kyoto, 2020), pp. 38–54.

[21] A. Kundu, Y. Li and J. M. Rehg, 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Salt Lake address, 2018), pp. 3559–3568.

[22] M. Danielczuk, M. Matl, S. Gupta, A. Li, A. Lee, J. Mahler and K. Goldberg, Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data, in *International Conference on Robotics and Automation (ICRA)* (Montreal, 2019), pp. 7283–7290.

[23] J. Drogemuller, C. X. Garcia, E. Gambaro, M. Suppa, J. Steil and M. A. Roa, Automatic generation of realistic training data for learning parallel-jaw grasping from synthetic stereo images, in *International Conference on Advanced Robotics (ICAR)* (Institute of Electrical and Electronics Engineers (IEEE), Ljubljana, 2021), pp. 730–737.

[24] C. Xie, Y. Xiang, A. Mousavian and D. Fox, Unseen object instance segmentation for robotic environments, *Transactions on Robotics* **37** 1343–1359 (2021).

[25] D. V. Vargas, B. Liao and T. Kanzaki, Perceptual deep neural networks: Adversarial robustness through input recreation, *Computing Research Repository (CoRR)* **abs/2009.01110** (2020).

[26] A. Kriegler, C. Beleznai and M. Gelautz, Evaluation of monocular and stereo depth

data for geometry-assisted learning of 3d pose, in *Proceedings of the OAGM Workshop 2021: Computer Vision and Pattern Analysis Across Domains* (St. Pölten, 2021), pp. 1–7.

[27] T. Kollar, M. Laskey, K. Stone, B. Thananjeyan and M. Tjersland, Simnet: Enabling robust unknown object manipulation from pure synthetic data via stereo, in *Conference on Robot Learning (CoRL)* (London, 2021), pp. 938–948.

[28] O. Kundu and S. Kumar, A novel geometry-based algorithm for robust grasping in extreme clutter environment; a novel geometry-based algorithm for robust grasping in extreme clutter environment, in *International Conference on Robot and Human Interactive Communication (RO-MAN)* (New Delhi, 2019), pp. 1–8.

[29] A. Kriegler, C. Beleznai, M. Murschitz, K. Göbel and M. Gelautz, Primitivepose: 3d bounding box prediction of unseen objects via synthetic geometric primitives, in *International Conference on Robotic Computing (IRC)* (Naples, 2022), pp. 190–197.

[30] M. Durner and W. Boerdijk, Stereo instances on surfaces (stios) (2021).

[31] T. Hodan, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother and J. Matas, Bop challenge 2020 on 6d object localization, in *European Conference on Computer Vision Workshops (ECCVW)* (Glasgow, 2020), pp. 577–594.

[32] C. Romanengo, A. Raffo, Y. Qie, N. Anwer and B. Falcidieno, Fit4CAD: A Point Cloud Benchmark for Fitting Simple Geometric Primitives in CAD Objects, *Computers & Graphics* **102** 133–143 (2022).

[33] H. Fang, Geometric modeling of man-made objects at different level of details (2019).

[34] R. Hachiuma and H. Saito, Pose estimation of primitive-shaped objects from a depth image using superquadric representation, *Applied Sciences* **10** (2020).

[35] R. T. Chin and C. R. Dyer, Model-based recognition in robot vision, *Computing Surveys* **18** 67–108 (1986).

[36] J. Zhang, J. J. Cao, H. R. Zhu, D. M. Yan and X. P. Liu, Geometry guided deep surface normal estimation, *CAD Computer Aided Design* **142** (2022).

[37] R. Fan, H. Wang, B. Xue, H. Huang, Y. Wang, M. Liu and I. Pitas, Three-filters-to-normal: An accurate and ultrafast surface normal estimator, *Robotics and Automation Letters* **6** 5405–5412 (2021).

[38] N. Nejatishahidin, P. Fayyazsanavi and J. Kosecka, Object pose estimation using mid-level visual representations, in *International Conference on Intelligent Robots and Systems (IROS)* (Kyoto, 2022), pp. 13105–13111.

[39] C. Wan, A. Yao and L. V. Gool, Hand pose estimation from local surface normals, in *European Conference of Computer Vision (ECCV)* **9907**, (Springer International Publishing, 2016), pp. 554–569.

[40] C. Xu, J. Chen, M. Yao, J. Zhou, L. Zhang and Y. Liu, 6dof pose estimation of transparent object from a single rgb-d image, *Sensors* **20** 1–19 (12 2020).

[41] A. Bansal, B. Russell and A. Gupta, Marr revisited: 2d-3d alignment via surface normal prediction, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Las Vegas, 2016), pp. 5965–5974.

[42] M. Dani, K. Narain and R. Hebbalaguppe, 3dposelite: A compact 3d pose estimation using node embeddings, in *Winter Conference on Applications of Computer Vision (WACV)* (Waikoloa, 2021), pp. 1877–1886.

[43] Y. Ge, J. Zhao and L. Itti, Pose augmentation: Class-agnostic object pose transformation for object recognition, in *European Conference on Computer Vision (ECCV)* (Glasgow, 2020), pp. 138–155.

[44] A. Grabner, P. Roth and V. Lepetit, Gp2c: Geometric projection parameter consensus for joint 3d pose and focal length estimation in the wild, in *International Conference on Computer Vision (ICCV)* (Seoul, 2019), pp. 2222–2231.

[45] Y. Xiao, X. Qiu, P. A. Langlois, M. Aubry and R. Marlet, Pose from shape: Deep pose estimation for arbitrary 3d objects, in *British Machine Vision Conference (BMVC)* (Cardiff, 2019), pp. 1–18.

[46] Y. He, Y. Wang, H. Fan, J. Sun and Q. Chen, Fs6d: Few-shot 6d pose estimation of novel objects, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (New Orleans, 2022), pp. 6814–6824.

[47] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang and X. Zhou, Onepose: One-shot object pose estimation without cad models, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (New Orleans, 2022), pp. 6825–6834.

[48] M. Z. Irshad, T. Kollar, M. Laskey, K. Stone and Z. Kira, Centersnap: Single-shot multi-object 3d shape reconstruction and categorical 6d pose and size estimation, in *International Conference on Robotics and Automation (ICRA)* (Philadelphia, 2022), pp. 10632–10640.

[49] E. Guillou, D. Méneveaux, E. Maisel and K. Bouatouch, Using vanishing points for camera calibration and coarse 3d reconstruction from a single image, *The Visual Computer* **16** 396–410 (2000).

[50] P. Gantelius, D. Herenu, T. Bredsdorff and L. Garcia-Tornel, fspy (2018).

[51] P. Gantelius, D. Herenu, T. Bredsdorff and SavMartin, fspy-blender (2018).

[52] C. Godard, O. M. Aodha, M. Firman and G. Brostow, Digging into self-supervised monocular depth estimation, in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (Seoul, 2019), pp. 3828–3838.

[53] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler and V. Koltun, Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer, *Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **3** 1623–1637 (2020).

[54] N. Smolyanskiy, A. Kamenev and S. Birchfield, On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach, in *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (Salt Lake address, 2018), pp. 1120–11208.

[55] F. Yu, D. Wang, E. Shelhamer and T. Darrell, Deep layer aggregation, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Salt Lake address, 2018), pp. 2403–2412.

[56] P. Chao, C.-Y. Kao, Y.-S. Ruan, C.-H. Huang and Y.-L. Lin, Hardnet: A low memory traffic network, in *International Conference on Computer Vision* (Seoul, 2019), pp. 3552–3561.

[57] T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, Focal loss for dense object detection, in *International Conference on Computer Vision (ICCV)* (Venice, 2017), pp. 2999–3007.

[58] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige and N. Navab, Model Based Training, Detection and Pose Estimation of Texture-less 3D Objects in Heavily Cluttered Scenes, in *Asian Conference on Computer Vision (ACCV)* (Daejeon, 2012), pp. 548–562.

[59] T. Hodăn, J. Matas and Štěpán Obdržálek, On evaluation of 6d object pose estimation, in *European Conference on Computer Vision Workshops (ECCVW)* (Amsterdam, 2016), pp. 609–619.

[60] A. Ahmadyan, L. Zhang, A. Ablavatski, J. Wei, M. Grundmann and G. Research, Objectron: A large scale dataset of object-centric videos in the wild with pose annotations, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (Nashville, 2021), pp. 7822–7831.

[61] P. Jaccard, The distribution of the flora in the alpine zone, *New Phytologist* **11** 37–50

(1912).

[62] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel and A. M. Dollar, The ycb object and model set: Towards common benchmarks for manipulation research, in *International Conference on Advanced Robotics (ICAR)* (Istanbul, 2015), pp. 510–517.

[63] F. Persia, F. Bettini and S. Helmer, An interactive framework for video surveillance event detection and modeling, in *Conference on Information and Knowledge Management* (Singapore, 2017), pp. 2515–2518.