Neurocomputing 541 (2023) 126270

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Ensemble of 6 DoF Pose estimation from state-of-the-art deep methods.

Ibon Merino [a,b,*], Jon Azpiazu [a], Anthony Remazeilles [a], Basilio Sierra [b]

[a] TECNALIA, Basque Research and Technology Alliance (BRTA), Mikeletegi Pasealekua 7, 20009 Donostia-San Sebastián, Spain
[b] Robotics and Autonomous Systems Group, Universidad del País Vasco/Euskal Herriko Unibertsitatea, San Sebastián 20009, Spain

ARTICLE INFO

Article history:
Received 2 November 2022
Revised 31 March 2023
Accepted 22 April 2023
Available online 25 April 2023
Communicated by Zidong Wang

Keywords:
Deep learning
Pose estimation
Ensemble
Stacked generalization

ABSTRACT

Deep learning methods have revolutionized computer vision since the appearance of AlexNet in 2012. Nevertheless, 6 degrees of freedom pose estimation is still a difficult task to perform precisely. Therefore, we propose 2 ensemble techniques to refine poses from different deep learning 6DoF pose estimation models. The first technique, merge ensemble, combines the outputs of the base models geometrically. In the second, stacked generalization, a machine learning model is trained using the outputs of the base models and outputs the refined pose. The merge method improves the performance of the base models on LMO and YCB-V datasets and performs better on the pose estimation task than the stacking strategy.

1. Introduction

Vision-based object pose estimation is a task with applications in different fields, such as robotics or augmented reality. Many techniques approach this task by estimating the 2D or 3D bounding box that corresponds to the pose of the object [1]. In contrast, this paper focuses on estimating the 6 degrees of freedom (6 DoF) pose of object also known as 6D localization, that is the 3 DoF translation and 3 DoF rotation that would align the object with a target object reference model. Fig. 1 shows an example of a 6 DoF pose estimation task. 6DoF pose estimation is usually represented in three diferent ways: Bounding Boxes, Axes and model overlapping. The Fig. 1 shows the 6DoF poses of the objects overlapping the models of the objects on the image.

The leaderboards of the main challenges presented in the Benchmark for 6D Object Pose estimation [2] (BOP challenge, one of the most well-known 6DoF estimation benchmarks) were led by classical non-deep learning methods based on Point Pair Features (PPF) [3–6] until 2019. Those methods were depth only methods that rely on geometrical features of pointclouds. Methods based on convolutional neural networks (CNNs) [7–10] were limited by the insufficient number of real training images, since annotation of 6DoF poses is expensive, and the large domain gap between real test images and the commonly used synthetically generated training images. On BOP 2020, 350 K physics-based rendered training images were provided to the participants in order to deal with these problems. Since that moment, deep learning methods have finally caught up and surpassed PPF-based methods. In relation with the input data used, on BOP 2020 almost all deep learning methods were trained on RGB images only. However, more recently, deep RGB-D approaches have obtained similar results to deep RGB only methods [11,12].

The main contributions of this paper are summarized as follows:

- An exhaustive analysis of existing RGB/RGB-D deep learning methods for 6DoF pose estimation.
- Two ensemble methods (merge and stacking) to combine state-of-the-art models in order to obtain more homogenous detection performance across different datasets.

In the experiments performed we compare how each base model and ensemble strategy perform for each metric, and the results show that the metrics obtained on the ensemble strategies are more homogenous than the ones obtained with the base models (or level-0). The rest of the paper is organized as follows: in Section 2 we present related works; in Section 3 we introduce the proposed ensemble method; in Section 5 we explain the conducted experiments; and, finally, we summarize the paper and the conclusions in Section 6.

* Corresponding author at: TECNALIA, Basque Research and Technology Alliance (BRTA), Mikeletegi Pasealekua 7, 20009 Donostia-San Sebastián, Spain.
E-mail addresses: ibon.merino@tecnalia.com (I. Merino), jon.azpiazu@tecnalia.com (J. Azpiazu), anthony.remazeilles@tecnalia.com (A. Remazeilles), b.sierra@ehu.eus (B. Sierra).

https://doi.org/10.1016/j.neucom.2023.126270
0925-2312/

Fig. 1. 6DoF pose estimation of Linemod objects from BOP challenge.

2. Background

For some time now, computer vision has been used to develop technologies to recognize objects. For industrial objects, many authors have designed specific methods to deal with this paradigm [13–15]. Nevertheless, generic vision techniques have also been used for recognition of industrial objects. The first approaches were based on feature matching or classification techniques: object features are detected, described and matched with known features from target objects.

Typically, features were hand-crafted to produce descriptors with convenient robustness or invariance to various conditions in order to make the matching between images and object models possible. SIFT [16] features have been one of the most popular hand-crafted features for a long time. More recently, deep neural network based methods became popular, thanks to the development of more powerful GPUs and larger datasets that enable the training of big convolutional neural network models.

Methods that **directly regress the rotation and translation** of objects are very common. The method proposed in [17] regresses 6DoF object poses using only synthetic images. Another pose estimator that directly regresses the rotation and translation is presented in [18]. This method calculates the 3D translation predicting the distance from object to camera and the rotation is estimated regressing to a quaternion representation. In [19], authors propose an end-to-end trained model that extends Mask R-CNN [20], the canonical 2D object detection network, to regress 3D translation and rotation. [21] is a dense fusion network that extracts pixel-wise dense feature embeddings from RGB-D images to predict the pose of the objects. However, this group of methods is not very scalable to regress poses of datasets with a high number of different classes.

Other approaches rely on **two-stage approaches** that first detect keypoints and later estimate 2D-3D or 3D-3D correspondences of keypoints using Perspective-n-Point (PnP) algorithms to estimate the poses. One of those methods estimates semantic keypoints and fits a deformable shape model to the 2D detections [22]. BB8 [23] predicts a 2D projection of the corners of the 3D bounding boxes as keypoints. [24] introduces a similar approach that detects the 2D projection of the 3D bounding box corners using a single shot convolutional neural network. [25] proposed a category-agnostic keypoint representation by combining what they call StarMap heatmap and their corresponding features as 3D locations in the canonical viewpoint. [26] combines global features for object segmentation and local features for coordinate regression followed by a RANSAC to optimize the object pose. [27] introduces a pixel-wise voting network (PVNet) to regress pixel-wise vectors pointing to the keypoints and uses those vectors to vote for the location of keypoints. This voting method has inspired many later methods. [28] predicts newly introduced features called NOCS that, combined with the depth map, can jointly estimate the 6D pose. Two-stage approaches do not train the entire pipeline at once. The keypoint prediction is trained while the correspondence matching to 3D and 6D pose estimation is obtained a posteiori.

Learning from **pointclouds** provides many benefits to robotics. PointNet++ [29] introduce a hierarchical neural network that applies PointNet [30] recursively. [11] estimates the 6Dof pose by detection 3D keypoints from pointclouds and RGB images and searching for correspondences similar to two-stage approaches. [12] is an improvement from the previous method that adds a bidirectional fusion layer to combine RBG information and geometric information. Those methods use the geometrical features that can be extracted from pointclouds that 2D images can not provide.

Even if the continuous development of new deep learning architectures (such as, MFDNet [31], ARHPE [32], EDMF [33], EHPE [34] or NGDNet [35]) and techniques led to significant performance improvement on many datasets, there is still a lack of a general method that works for every situation or dataset. Stacking and ensemble methods try to deal with this problem. They try to extract the ability of each individual method to obtain a more abstract method [36]. There are two critical design elements for this kind of models: the type of generalizer that is suitable to derive the higher-level model and what kind of attributes should be used as inputs [37].

Another challenge to deal with is all the hyperparameters that base learners and higher level models have to tune. To optimize this hyperparameter search, authors in [38] proposed a heuristic search-based stacking of classifiers. Authors from [39] also make

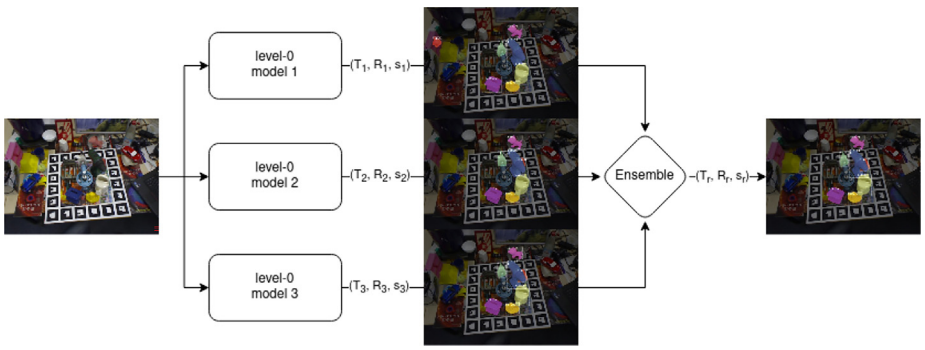


Fig. 2. Illustration of the ensemble approach, in which the output of level-0 models outcome are gathered to deduce a unique refined pose.

Table 1
Ensemble strategies and methods.

Strategy	Method
Merge	Simple
	Weighted
	Clustering
Stacking	SVR
	Decision tree
	Linear regression
	KNN
	MLP

use of meta-heuristic algorithms to configure the stacking ensemble.

Insect behavior based optimization algorithms are also a good solution to search the best configurations in a high range of possible hyperparameters [40–43]. Many more authors have used ensemble or stacking methods to improve the generalization of base learners [44,45]. Some of them even consider stacking a kind of super learning method [46].

Stacking and ensemble methods have also been used for computer vision systems. In [47], authors designed a support vector machine training method based on stacked generalization for image classification. Stacked generalization also improves accuracy for scene analysis and object recognition [48].

3. Proposed method

The ability of neural networks to retrieve abstract information from data enables them to attain competitive performances in many fields and contexts. Nevertheless, not all models are effective against every task or dataset. Indeed, the optimal selection of the

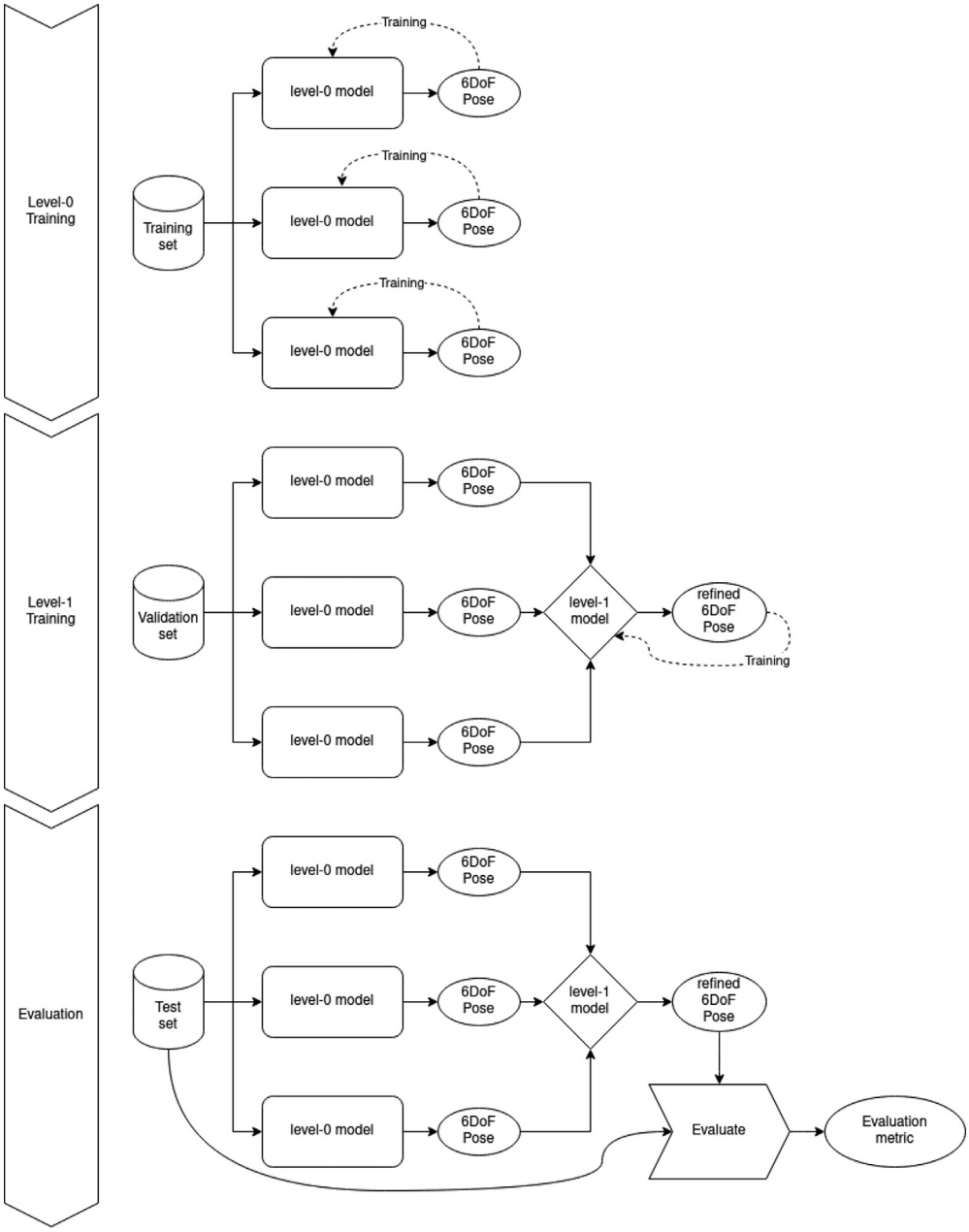


Fig. 3. Stacking poses of 3 level-0 models.

model type and structure, and the comprehension of the reasons that explain the results [49] is one of the main challenges that deep learning can still not solve.

Since looking for the correct deep learning approach is tedious, we propose a method that learns when a given algorithm is more appropriate than another one. Our method is an ensemble model that enables the fusion of the performance of different models. We have defined two different strategies: merging and stacking. Both strategies take the output of different models and refine the obtained poses using the knowledge obtained from each model. The output of each model is composed of a 3D translation T, a 3D rotation R, related to the pose of the object with respect to the camera, and a score s that is the confidence that each model has for that prediction. Given n different models, T_i, R_i and s_i represent the translation, the rotation and the score outcome of the i-th method, where $i \in \{1..n\}$. We defined the refined translation as T_r, the refined rotation as R_r and the refined score as s_r, the outcome of our fusion approach.

Fig. 2 shows the general idea of the ensemble method. Both ensemble approaches, merge and stacking, rely on a set of deep learning models (named from now on as level-0 models) that provide the initial predictions of the pose of the object.

3.1. Level-0 models

We have used as level-0 algorithms three state-of-the-art 6DoF pose estimation deep models: PVN3D [11], FFB6D [12] and Cosypose [50]. These three methods have led the BOP challenge 2020 leaderboard in different datasets.

PVN3D [11] is a novel method for 6DoF object pose estimation from a single RGBD image. This method is based on a deep Hough voting network that takes as input RGBD images and detects 3D keypoints as is performed in RGB-based 6DoF estimation. Once the 3D keypoints are detected, these are used to estimate the parameters of the 6D pose using a least-squares fitting. The method has outperformed state-of-the-art methods on several

Fig. 4. Real test images from LMO.

Fig. 5. Real test images from YCB-V.

Fig. 6. Real test images from T-LESS.

(a) Image from LM PBR training set.

(b) Image from YCB-V PBR training set.

(c) Image from T-LESS PBR training set.

Fig. 7. PBR training images.

Table 2
Level-0 results.

Dataset		PVN3D	FFB6D	COSYPOSE
LMO PBR	AR_{VSD}	0.4740	0.4450	**0.4805**
	AR_{MSSD}	**0.6523**	0.6309	0.6062
	AR_{MSPD}	0.7105	0.6704	**0.8122**
	AR	0.6123	0.5821	**0.6330**
YCB-V PBR	AR_{VSD}	**0.5888**	0.3387	0.5166
	AR_{MSSD}	**0.6955**	0.4093	0.5538
	AR_{MSPD}	0.5649	0.2758	**0.6527**
	AR	**0.6164**	0.3413	0.5744
YCB-V S + R	AR_{VSD}	0.7291	**0.7855**	0.7731
	AR_{MSSD}	0.8541	**0.8791**	0.8421
	AR_{MSPD}	0.7441	0.7856	**0.8502**
	AR	0.7757	0.8167	**0.8218**
T-LESS PBR	AR_{VSD}	0.1789	0.1291	**0.5714**
	AR_{MSSD}	0.2067	0.1268	**0.5892**
	AR_{MSPD}	0.2039	0.1165	**0.7605**
	AR	0.1965	0.1241	**0.6404**
T-LESS S + R	AR_{VSD}	0.2255	0.1985	**0.6691**
	AR_{MSSD}	0.2826	0.2396	**0.6946**
	AR_{MSPD}	0.2666	0.2291	**0.8212**
	AR	0.2582	0.2224	**0.7283**

benchmarks such as Linemod [51] and YCB-V [52]. PVN3D trains its architecture using the multi-task loss (Eq. 4) jointly training keypoints detection module (Eq. 1), semantic segmentation module (Eq. 2) and and objects center voting module (Eq. 3).

$$L_{keypoints} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}||of_i^j - of_i^{j*}||\mathbb{I}(p_i \in I), \tag{1}$$

where of_i^j is the translation offset, of_i^{j*} is the ground truth translation offset, M is the total number of selected target keypoints, N is the total number of seeds and $\mathbb{I}$ is an indicating function that equates to 1 only when point p_i belongs to instance I, and 0 otherwise.

$$L_{semantic} = -\alpha(1-q_i)^{\gamma} log(q_i), \quad \text{where} \quad q_i = c_i \cdot l_i \tag{2}$$

with α the α-balance parameter, γ the focusing parameter, c_i the predicted confidence for the i_{th} point belongs to each class and l_i the one-hot representation of ground true class label.

$$L_{center} = \frac{1}{N}\sum_{i=1}^{N}||\Delta x_i - \Delta x_i^*||\mathbb{I}(p_i \in I), \tag{3}$$

where N denotes the total number of seed points on the object surface and Δx_i^* is the ground truth translation offset from seed p_i to the instance center.

$$L_{multi-task} = \lambda_1 L_{keypoints} + \lambda_2 L_{semantic} + \lambda_3 L_{center}, \tag{4}$$

where λ_1, λ_2 and λ_3 are the weights for each task. In our case, we use 2.0 for λ_2 and 1.0 for λ_1 and λ_3.

FFB6D [12] or Full Flow Bidirectional Fusion Network for 6D Pose Estimation is a deep learning network that combines appearance information from RGB images and geometric information from depth images during the representation learning stage (encoding and decoding). Instead of learning each feature independently and fusing them at the end like DenseFusion [21] or PVN3D [11], the fusion is applied to each encoding and decoding layer giving complementary information from 3D to 2D and the other way round. After the representation learning stage, both features are concatenated and 6D pose parameters are estimated as in PVN3D [11]. FFB6D uses the same multi-task loss that was presented in PVN3D. The main difference between FFB6D and PVN3D is the architecture of the feature extraction module. PVN3D extracts geometric features and color features and afterwards fuses them, while FFB6D fuses the geometric and color information on each layer of the network.

Cosypose [50] is a multi-view multi-object 6D pose estimator. In the paper, a single-view single-object estimation method to generate 6D hypothesis and a method for matching the predicted hypothesis to generate a single consistent scene are introduced. With the generated scene, global refinement is applied across different objects and views. They outperform current state-of-the-art results for single-view and multi-view. The single-view 6D pose estimation of Cosypose is carried out following the focal loss adapted in [53] and handle symmetries as in [28]. The object candidate matching first removes the object candidates that are not consistent across views using a RANSAC procedure. Moreover, for the scene refinement they introduce a reconstruction loss (Eq. 5) that operates at a level of objects.

$$L(T_{P_n}, T_{C_a}|T_{C_a}O_{a,\alpha}) = \min_{S\in S(l)}\frac{1}{|\chi_l|}\sum_{x\in\chi_l}||\pi_a(T_{C_a}O_{a,\alpha}Sx) - \pi_a\left(T_{C_a}^{-1}T_{P_n}x\right)||, \tag{5}$$

where $||\cdot||$ is a truncated L2 loss, and for the rest of the parameters check the original paper [50].

The scores of PVN3D and FFB6D are the confidence of the point level instance segmentation of each object. The scores of Cosypose, instead, are the confidence of the initial 2D predictions of the objects.

All three models have their advantages and disadvantages, none of them outperforms the others on every dataset (this can be seen in the BOP Challenge leaderboard). Fig. 2 includes an example of the predictions of the 3 methods. In this example level-0 model 1 is PVN3D, model 2 is FFB6D and model 3 Cosypose. Predictions of the 3 level-0 models are similar except from the ape (red colored) which PVN3D can not locate correctly. A combination of the outputs refines the poses obtaining better results.

For simplicity, our methods deal with the SiVo (Single instance Various objects) task instead of the ViVo (Various instances Various objects) task. We detect a single instance of multiple objects.

4. Ensemble

In order to obtain the refined pose, we define two different ensemble strategies: merge and stacking. Merge methods are defined by mixing the individual results of the level-0 methods

mathematically or algorithmically. Stacking methods, instead, make use of statistical methods or machine learning methods to refine the level-0 poses.

Table 1 summarizes all the proposed strategies and methods.

4.1. Ensemble-Merge

We have defined three merge methods that take the output poses of the level-0 models and calculate a refined pose. The three defined methods are: simple merge, weighted merge and clustered merge.

4.1.1. Simple merge

The first approach is a simple merge method that outputs the mean of the input poses. Eq. 6 and Eq. 7 explains how to obtain the resulting translation and rotation, respectively. Computing a rotation mean is not trivial and can be obtained several ways. In this paper, we use the chordal L2 mean which provides the rotation that minimizes the square of the difference between rotation matrices [54].

$$T_r = \frac{1}{n}\sum_{i=1}^{n} T_i \tag{6}$$

$$R_r = \text{argmin}_{R \in SO(3)} \sum_{i=1}^{n} ||R_i - R||^2 \tag{7}$$

where R is a rotation that belongs to the three-dimensional space of rotations $SO(3)$.

4.1.2. Weighted merge

Simple merge considers each of the models equally even if a model itself outputs a pose with a low confidence. Therefore, this second approach takes into account the scores each model gives

Table 3
Merge ensemble results.

Dataset		Simple (S)	Weighted (W)	Simple clustering (SC)	Weighted clustering (WC)
LMO PBR	AR_{VSD}	0.4776	0.4820	0.4807	**0.4822**
	AR_{MSSD}	0.6473	0.6411	0.6496	0.6509
	AR_{MSPD}	0.7049	0.7338	0.7416	0.7434
	AR	0.6099	0.6189	0.6240	0.6255
YCB-V PBR	AR_{VSD}	0.4877	0.5277	0.5451	0.5502
	AR_{MSSD}	0.5565	0.6003	0.6150	0.6235
	AR_{MSPD}	0.4370	0.5228	0.5312	0.5403
	AR	0.4938	0.5503	0.5638	0.5713
YCB-V S + R	AR_{VSD}	0.7930	0.7748	**0.7958**	0.7946
	AR_{MSSD}	0.8733	0.8528	0.8733	**0.8816**
	AR_{MSPD}	0.8033	0.8473	0.8132	0.8294
	AR	0.8233	0.8249	0.8275	**0.8352**
T-LESS PBR	AR_{VSD}	0.1741	0.2730	0.2732	0.2883
	AR_{MSSD}	0.1998	0.2887	0.2921	0.3073
	AR_{MSPD}	0.1897	0.3177	0.3232	0.3394
	AR	0.1879	0.2931	0.2962	0.3117
T-LESS S + R	AR_{VSD}	0.2410	0.3578	0.3275	0.3521
	AR_{MSSD}	0.2922	0.3950	0.3699	0.3912
	AR_{MSPD}	0.2822	0.4154	0.3802	0.4055
	AR	0.2718	0.3894	0.3592	0.3830

Table 4
Results of stacking.

Dataset		Stacking Ridge (R)	Stacking SVR (SVR)	Stacking tree (T)	Stacking Linear (L)	Stacking KNN (KNN)	Stacking MLP (MLP)
LMO PBR	AR_{VSD}	0.3165 (s)	0.3146	0.1991	0.3165 (s)	0.3195	0.4408
	AR_{MSSD}	0.4375 (s)	0.5833	0.4707	0.4376 (s)	0.5789	0.6176
	AR_{MSPD}	0.6354 (s)	0.6298	0.5451	0.6355 (s)	0.6363	0.6812
	AR	0.4631 (s)	0.5092	0.4050	0.4632 (s)	0.5115	0.5799
YCB-V PBR	AR_{VSD}	0.4040 (s)	0.4479 (s)	0.2994 (s)	0.4040 (s)	0.3928	0.4462 (s)
	AR_{MSSD}	0.4896 (s)	0.5773 (s)	0.4487 (s)	0.4896 (s)	0.5117	0.5462 (s)
	AR_{MSPD}	0.3793 (s)	0.4268 (s)	0.2615 (s)	0.3792 (s)	0.3623	0.4175 (s)
	AR	0.4243 (s)	0.4840 (s)	0.3365 (s)	0.4243 (s)	0.4223	0.4700 (s)
YCB-V S + R	AR_{VSD}	0.7233 (s)	0.6024	0.3548	0.7233 (s)	0.564	0.6676
	AR_{MSSD}	0.8113 (s)	0.7967	0.6208	0.8113 (s)	0.7777	0.7979
	AR_{MSPD}	0.7218 (s)	0.6574 (s)	0.4023	0.7216 (s)	0.6446	0.6938 (s)
	AR	0.7521 (s)	0.6835 (s)	0.4593	0.7521 (s)	0.6621	0.7159
T-LESS PBR	AR_{VSD}	0.0441 (s)	0.1431 (s)	0.0513	0.0441 (s)	0.0714	0.0421
	AR_{MSSD}	0.0364 (s)	0.1301 (s)	0.0470	0.0364 (s)	0.0600 (s)	0.0535
	AR_{MSPD}	0.0521 (s)	0.1372 (s)	0.0560	0.0521 (s)	0.0685 (s)	0.0656
	AR	0.0442 (s)	0.1368 (s)	0.0514	0.0442 (s)	0.0664 (s)	0.0537
T-LESS S + R	AR_{VSD}	0.0589 (s)	0.2216 (s)	0.0747	0.0590 (s)	0.1079	0.0736
	AR_{MSSD}	0.0694 (s)	0.2500	0.0996	0.0694 (s)	0.1182	0.1198
	AR_{MSPD}	0.0789 (s)	0.2433	0.1023	0.0789 (s)	0.1208	0.1133
	AR	0.0691 (s)	0.2370 (s)	0.0922	0.0691 (s)	0.1156	0.1022

to its estimation. These scores are usually describing how confident the models are about the object detected. This score does not provide any information about how precise the pose is, but the confidence of the class of the detection.

Each of the poses is weighted accordingly to the obtained score over the total score. The weight of each model is calculated following Eq. 8 and the refined translation and rotation are obtained respectively with Eqs. 9 and 10.

$$w_i = \frac{1}{(1-s)^2 + \varepsilon} \tag{8}$$

$$T_r = \sum_{i=1}^{n} w_i \cdot T_i \tag{9}$$

Table 5
Comparative of all level-0 and ensemble methods. ∗: Cosypose, △: PVN3D, ☆:FFB6D

Dataset	Metric	Level-0	Merge	Stacking	CDPNv2	Pix2Pose
LMO PBR	VSD	0.480∗	**0.482 (WC)**	0.440 (MLP)	0.469	0.473
	MSSD	0.652△	0.650 (WC)	0.618 (MLP)	0.689	0.631
	MSPD	**0.812**∗	0.743 (WC)	0.681 (MLP)	0.731	0.659
	AR	**0.633**∗	0.626 (WC)	0.580 (MLP)	0.630	0.588
YCB-V PBR	VSD	0.589△	0.550 (WC)	0.448 (SVR-s)	0.511	-
	MSSD	0.696△	0.624 (WC)	0.577 (SVR-s)	0.603	-
	MSPD	**0.653**∗	0.540 (WC)	0.426 (SVR-s)	0.483	-
	AR	**0.616**∗	0.571 (WC)	0.484 (SVR-s)	0.532	-
YCB-V S + R	VSD	0.786☆	**0.796 (SC)**	0.723 (R-s/L-s)	0.590	0.766
	MSSD	0.879☆	**0.882 (WC)**	0.811 (R-s/L-s)	0.701	0.817
	MSPD	**0.850**∗	0.847 (W)	0.722 (R-s)	0.565	0.758
	AR	0.822∗	**0.835 (WC)**	0.752 (R-s/L-s)	0.619	0.780
T-LESS PBR	VSD	**0.571**∗	0.288 (WC)	0.143 (SVR-s)	0.368	-
	MSSD	**0.589**∗	0.307 (WC)	0.130 (SVR-s)	0.449	-
	MSPD	**0.761**∗	0.339 (WC)	0.137 (SVR-s)	0.488	-
	AR	**0.640**∗	0.312 (WC)	0.137 (SVR-s)	0.435	-
T-LESS S + R	VSD	**0.669**∗	0.358 (W)	0.222 (SVR-s)	0.385	0.438
	MSSD	**0.695**∗	0.395 (W)	0.250 (SVR)	0.489	0.548
	MSPD	**0.821**∗	0.415 (W)	0.243 (SVR)	0.516	0.549
	AR	**0.728**∗	0.389 (W)	0.237 (SVR-s)	0.464	0.512

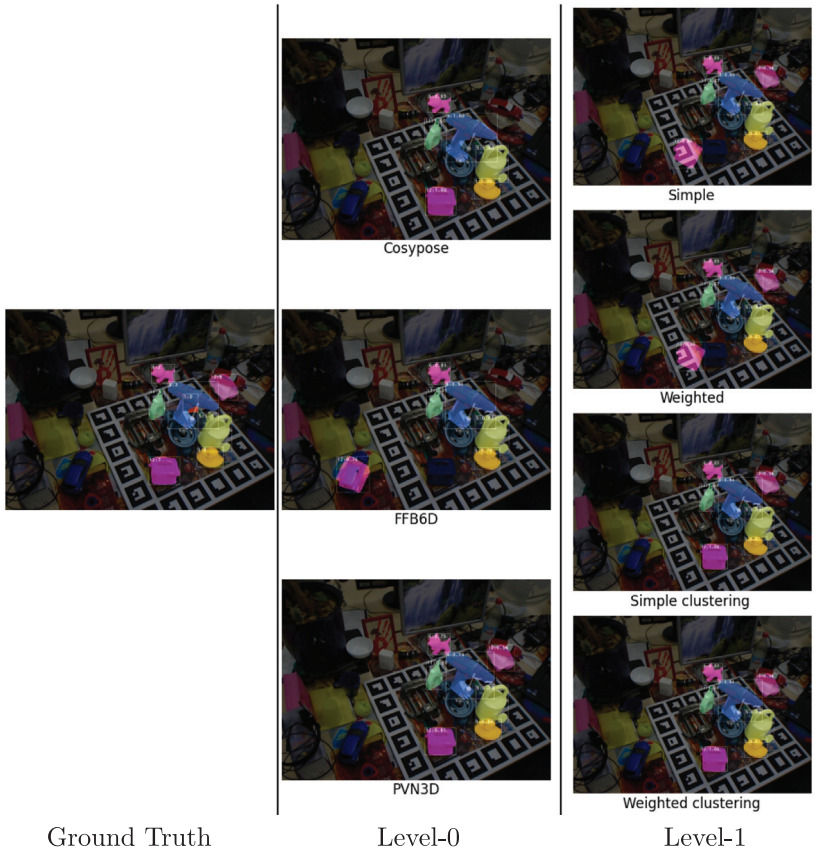


Fig. 8. Qualitative comparative of merge methods on a LMO dataset image.

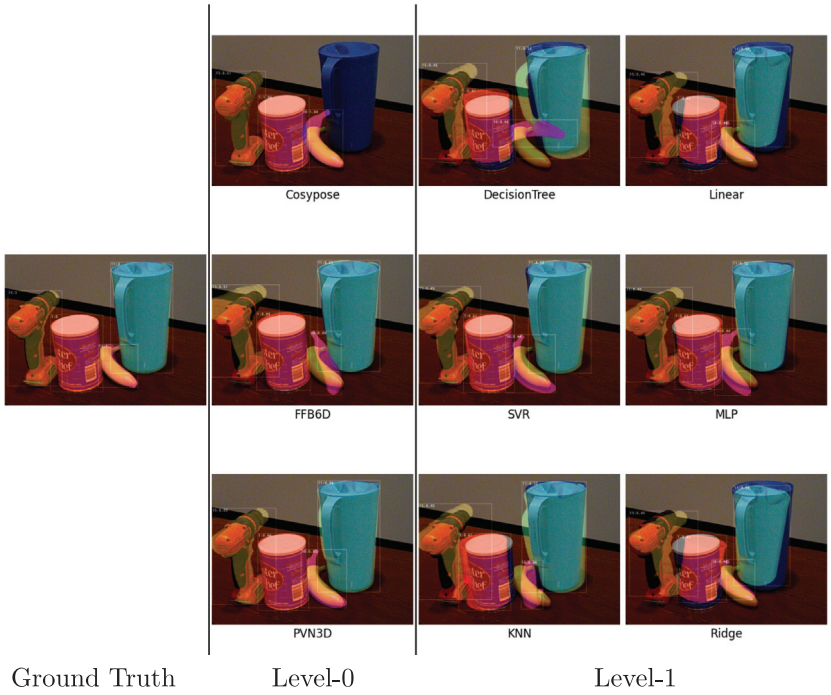


Fig. 9. Qualitative comparative of stacking methods on a YCB-V dataset image.

Fig. A.10. Qualitative comparative of merge methods on LMO dataset image 8 from scene 2.

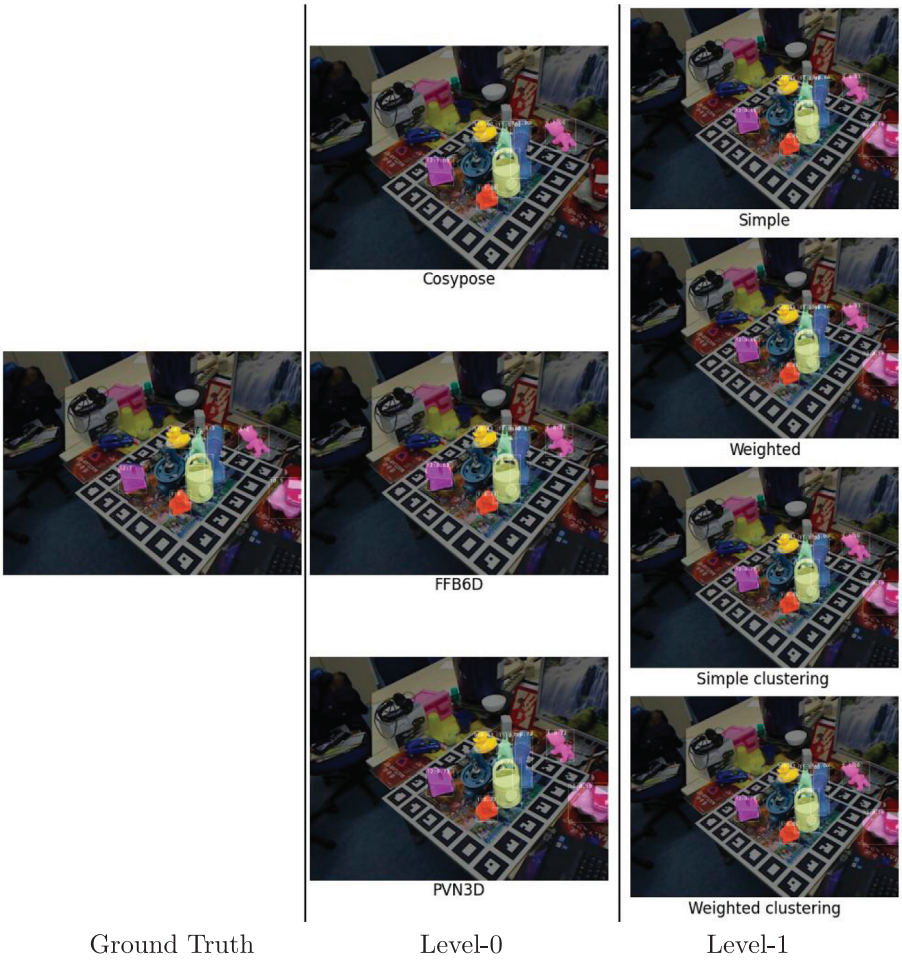


Fig. A.11. Qualitative comparative of merge methods on LMO dataset image 110 from scene 2.

$$R_r = \text{argmin}_{R \in SO(3)} \sum_{i=1}^{n} w_i ||R_i - R||^2 \quad (10)$$

where ε is very small positive number close to zero used to prevent the division by zero.

4.1.3. Clustering merge

A limitation of the previous approach is that if a model makes a bad estimation, the resulting pose may be badly influenced, even if the remaining methods were quite precise. That is, the resulting pose may be worse than the best of the initial poses.

Therefore, we designed a method based on clustering the candidate poses. To reduce the impact of incorrect poses, we focus the processing on the most similar poses, which is deduced by the cluster gathering more poses. For an equal number of poses, the cluster with the highest mean confidence is chosen. Then, to obtain the refined pose, a simple merge or a weighted merge is applied only on the poses belonging to the selected cluster. Algorithm 1 explains how the clusters are created and how the resulting pose is obtained. The distance thresholds have been selected based on previous experiments. The distance function is the Euclidean distance between the two given poses. Only translation is considered, rotation is not used to calculate the distance.

Algorithm 1: Clustering merge method

```
function CLUSTERING(poses)
    n_items ← len(poses)
    if n_items = 1 then
        result ← poses[0]
    else
        biggest_cluster ← []
        for i ∈ (0, len(poses)) do
            cluster ← [poses[i]]
            for j ∈ (i, len(poses)) do
                if distance(poses[i], poses[j]) < threshold then
                    cluster.append(pose[j])
                end if
            end for
            if len(biggest_cluster < cluster) then
                biggest_cluster ← cluster
            else if len(biggest_cluster == cluster)
                if score(biggest_cluster) < score(cluster) then
                    biggest_cluster ← cluster
                end if
            end if
```

(*continued on next page*)

(*continued*)

Algorithm 1: Clustering merge method

```
    end for
    result ← merge(biggest_cluster)   ▷simple merge or weighted merge.
  end if
  return result
end function
```

4.2. Ensemble-Stacking

Stacking or stacked generalization is an ensemble method that combines different machine learning models using another machine learning algorithm [36,37,46]. The base models or level-0 models are trained using the training data. Then, the validation data is predicted using the trained models, generating a new dataset where the independent variables are the outputs of each model. Finally, the combiner or level-1 model is trained using the validation data. Fig. 3 shows the full pipeline of training and evaluating this model.

4.2.1. Level-1 models

Level-1 methods combine the results that level-0 models have output. The following level-1 methods have been used in the proposed method:

- SVR [55] or support vector regression is based on the support vector machine concept [56]. This method assumes that the input vectors are non-linearly mapped to a high-dimension feature space. Nevertheless, the SVR computational complexity does not depend on the dimensionality of the input space, and it has excellent generalization capability with high prediction accuracy.
- Decision Trees [57] are formed by a collection of rules that are modelled according to input variables values. Variables are selected to get the best split to differentiate the values of the dependent variable. The rule splits the tree into two nodes and the process is repeated for each child node recursively. The method stops when no more gain can be made or by some pre-set stopping rules.
- The linear Regression [58] is based on the representation of a linear equation. The coefficients are learned from the training data using statistical properties(e.g., means, standard deviations, correlations or/and covariance), using Ordinary Least Squares to minimize the squared residuals or using the Gradient Descent method, for instance. We have used 2 linear regressions: Ordinary least squares Linear Regression and Ridge Linear Regression (Linear least squares with l2 regularization).
- KNN Regressor or K-Nearest Neighbors regressor [59,60] is a non-parametric model that uses the feature similarity to predict values. It uses the Euclidean, Manhattan or Minkowski distance as the KNN classification model. The target is predicted by local interpolation of the targets associated to the nearest neighbors in the training set.

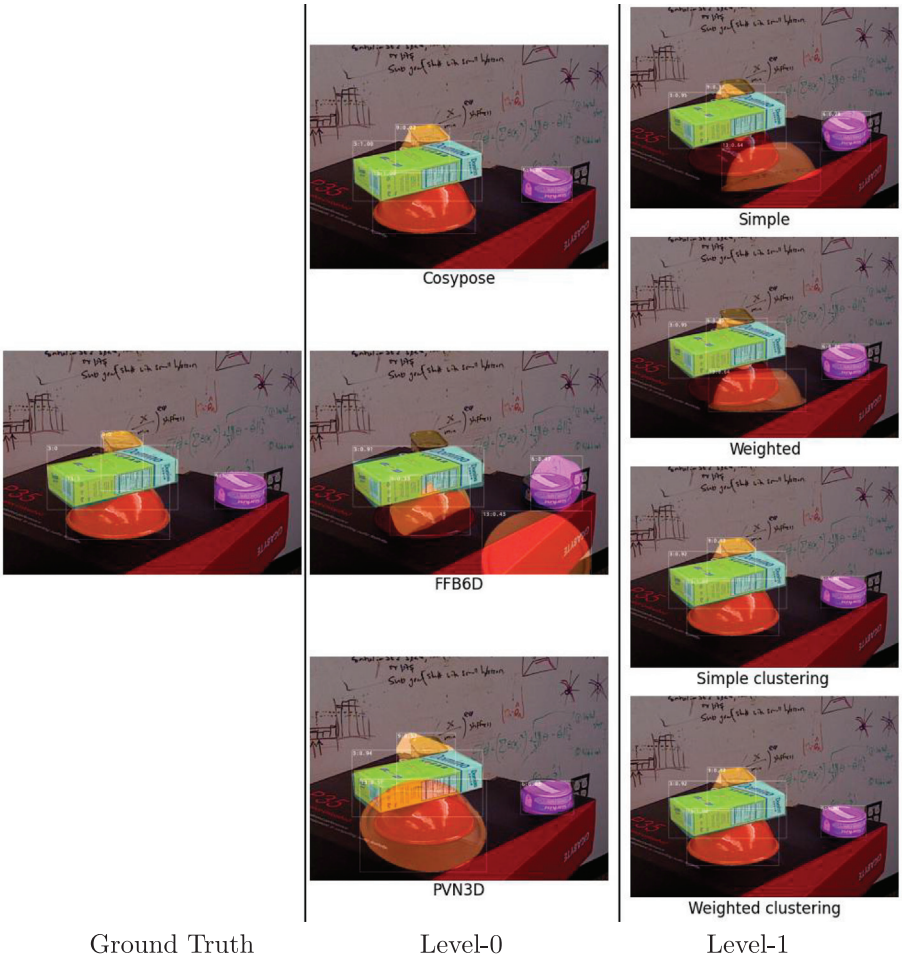


Fig. A.12. Qualitative comparative of merge methods on YCBV dataset image 1130 from scene 49.

- The MultiLayer Perceptron or MLP [61] is a type of feedforward Artificial Neural Network. MLP is usually a neural network containing a single hidden layer. We use Mean Squared Error (MSE) as loss function.

5. Experiments

We evaluate our method on a subset of the BOP challenge [2,62] benchmark. This objective of this challenge is to set a common benchmark to compare different state-of-the-art 6 DoF object pose estimation methods. The challenge defines an evaluation methodology and datasets to use. We compare the results obtained with the state-of-the-art methods (level-0) and the ensemble methods.

5.1. Datasets

BOP challenge includes 7 core datasets. We use a subset of those datasets including Linemod-Occlusion [51,63], T-LESS [64] and YCB-Video [52]. This subset is selected for simplicity and because it is relevant for our use case. Each dataset is composed of 50 k PBR (Physically-based renderer) synthetic images generated using BlenderProc for the BOP challenge for training, another training set with real and synthetic images that each of the datasets provides and test images. BOP challenge is evaluated using a subset from the test images (BOP test images). Figs. 4–7 show images of the LMO dataset, YCB-V dataset, T-LESS dataset and PBR images from all datasets, respectively.

5.1.1. Linemod

Linemod-Occlusion (LM-O) [51,63] is a dataset first proposed by [51] and later redefined by [63]. The initial dataset consists of 15 texture-less objects on a cluttered desk. The later redefinition takes only into account the scene '000002' with extra ground-truth information. This configuration is more challenging and only uses the 13 objects with proper 3D models (2 were omitted since proper 3D models were missing) and correct ground-truth poses (Some ground truth poses from the original dataset were incorrect). In the BOP challenge, the LM-O has available the 50 K PBR training images generated with BlenderProc (provided by BOP challenge), 400 k PBR training images from Microsoft Research (MSR) [65], synthetic training images of isolated objects generated in [63] and test images.

5.1.2. YCB-Video

YCB-Video [52,18] is a 6D pose estimation dataset composed of 21 objects from the YCB dataset [52] observed in 92 videos with 133827 frames. 80 videos are used for training and 12 videos for testing. In the BOP challenge, the YCB-V dataset has the 50 k PBR training images generated with BlenderProc (provided by BOP challenge), original synthetic training images, real training images and test images.

5.1.3. T-LESS

T-LESS [64] is an industry-relevant texture-less object dataset. The objects are often similar in shape and some objects are parts of others. Three different synchronized sensors have been used to

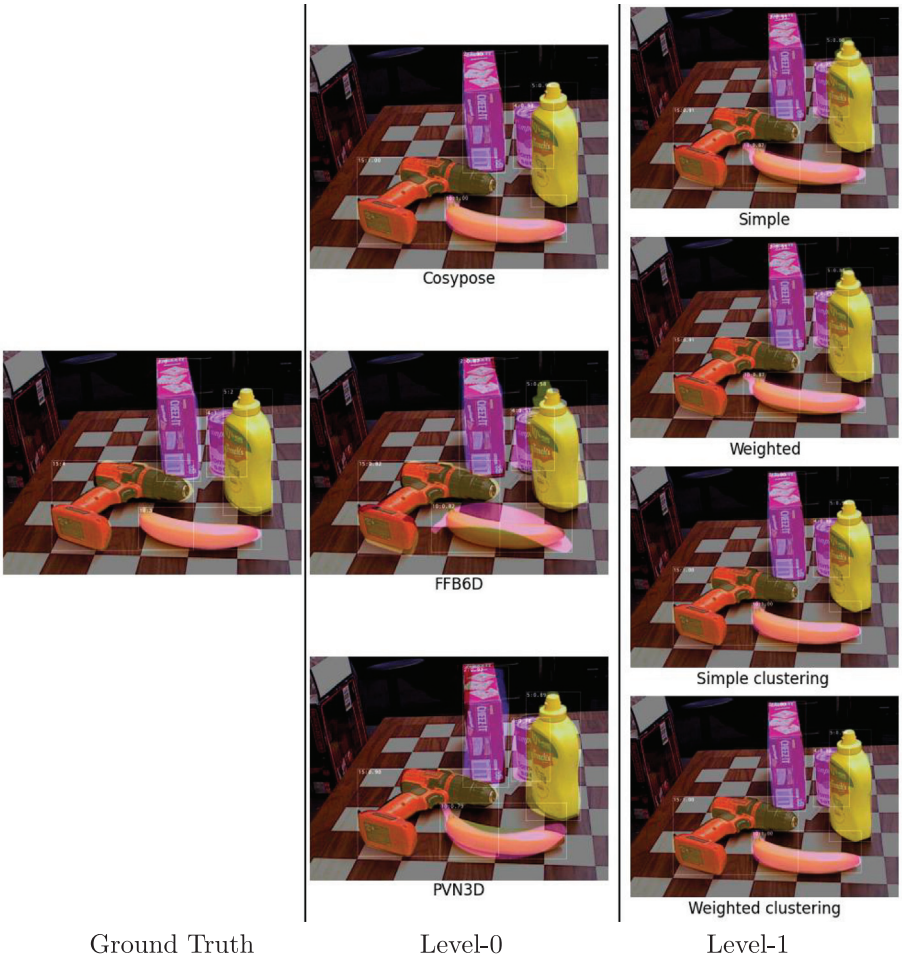


Fig. A.13. Qualitative comparative of merge methods on YCBV dataset image 1052 from scene 50.

capture the real training and test images. Real training images depict individual objects against a black background. In the BOP challenge, the T-LESS dataset has the 50 k PBR training images generated with BlenderProc (provided by BOP challenge), synthetic training images of isolated objects, real training images of isolated objects and test images.

5.2. Evaluation metrics

The BOP challenge task is 6D localization of Varying number of instances of a Varying number of objects in a single RGB-D image (ViVo task). In the ViVo task, each image can contain 0 or n instances of 1 or j objects, where j is always less or equal to J, the number of objects of the corresponding dataset. In our case, we simplify the problem, and we focus on the 6D localization of Single instance of Varying number of objects in a single RGB-D image (SiVo task). In the SiSo task, each image can contain 0 or 1 instances of 1 or j objects.

To evaluate the level-0 models and all the ensemble methods, we have followed the evaluation methodology that the BOP challenge follows. Three pose-error functions are used: Visible Surface Discrepancy (VSD) [2,66], Maximum Symmetry-Aware Surface Distance (MSSD) [67] and Maximum Symmetry-Aware Projection Distance (MSPD) [62]. An estimated pose is considered correct if the value for each specific pose-error function (VSD, MSSD or MSPD) is less than a given threshold. The recall is the ratio between the correctly predicted poses over the annotated poses. Given different settings of the threshold and misalignment tolerances, the average recall of a specific pose-error function e (AR_e) is defined as the average of all the recalls for the different configurations on that pose-error function. The Average Recall (AR) is the average of the three specific pose-error function Average Recalls ($AR_{VSD}, AR_{MSSD}, AR_{MSPD}$). Each dataset and method is evaluated by the Average Recall (AR).

5.2.1. VSD

VSD only considers visible parts of the objects. Therefore, VSD treats poses that can not be distinct from one another as equivalent. Eq. 11 shows how to calculate the VSD error.

$$e_{VSD}\left(\widehat{D},\overline{D},\widehat{V},\overline{V},\tau\right) = avg_{p\in\widehat{V}\cup\overline{V}}\begin{cases} 0 & \text{if} p\in\widehat{V}\cap\overline{V}\wedge|\widehat{D}(p)-\overline{D}(p)|<\tau \\ 1 & \text{otherwise}\end{cases} \tag{11}$$

where $\widehat{D}$ and $\overline{D}$ are distance maps from rendering the model in the estimated pose $\widehat{P}$ and ground-truth pose $\overline{P}$, respectively; and $\widehat{V}$ and $\overline{V}$ the visibility masks. AR_{VSD} is calculated from the misalignment tolerance ranging from 5% to 50% of the object diameter with a step of 5%, and the threshold from 0.05 to 0.5 with a step of 0.05.

5.2.2. MSSD

The maximum distance is relevant for robotic manipulation as it indicates if there is a chance of a successful grasp. Compared to ADD/ADI [66,51], this metric behaves more independently of

Fig. A.14. Qualitative comparative of stacking methods on LMO dataset image 47 from scene 2.

the sampling of the mesh. Eq. 12 shows how to calculate MSSD error.

$$e_{MSSD}\left(\hat{\mathbf{P}}, \bar{\mathbf{P}}, S_M, V_M\right) = min_{\mathbf{s}\in S_M} max_{\mathbf{x}\in V_M} \|\hat{\mathbf{P}}\mathbf{x} - \bar{\mathbf{P}}\mathbf{S}\mathbf{x}\|_2 \quad (12)$$

where $\hat{P}$ is the estimated pose, $\bar{P}$ is the ground-truth pose, S_M is a set of global symmetry transformations and V_M is a set of mesh vertices of the model. AR_{MSSD} is calculated from the threshold ranging from 5% to 50% of the object diameter with a step of 5%.

5.2.3. MSPD

This measure is relevant for augmented reality applications and suitable for evaluation of RGB-only methods. Eq. 13 shows how to calculate MSPD error.

$$e_{MSPD}\left(\hat{\mathbf{P}}, \bar{\mathbf{P}}, S_M, V_M\right) = min_{\mathbf{s}\in S_M} max_{\mathbf{x}\in V_M} \|\text{proj}\left(\hat{\mathbf{P}}\mathbf{x}\right) - \text{proj}(\bar{\mathbf{P}}\mathbf{S}\mathbf{x})\|_2 \quad (13)$$

where *proj* is the 2D projection operation. AR_{MSPD} is calculated from the threshold ranging from $5r$ to $50r$ with a step of $5r$, where $r = w/640$ and w is the width of the image in pixels.

6. Results

We have defined 3 merging methods: simple, weighted and clustering; and a stacking method. First we are presenting the scores of level-0 models to compare them with all the ensemble methods defined.

Table 2 shows the results obtained for LMO PBR, YCB-V PBR, YCB-V S + R, T-LESS PBR and T-LESS S + R. S + R (SYN + REAL) is composed of the synthetic pbr images and the real images. For each dataset and metric, the highest score is set to bold (one per row). If we focus on the Average Recall(*AR*), Cosypose is the best level-0 approach on every dataset except for YCB-V PBR. The three models have similar *AR* on LMO PBR and YCB-V S + R. For YCB-V PBR, PVN3D and Cosypose behave similarly while FFB6D has lower *AR*. Results on T-LESS are significantly different: Cosypose outperforms FFB6D and PVN3D. This is because PVN3D and FFB6D work better for datasets like LMO and YCB-V since their authors designed these methods for those datasets, and the implementation is not prepared for ViVo task as Cosypose is. Indeed, the T-LESS dataset has more than one instance of the same object on some images. This leads to poor performance in many examples.

If we focus on the average recall of each metric, AR_{VSD} behaves similarly to *AR*. Even though, for this metric Cosypose is not the best for every dataset (PVN3D wins on YCB-V PBR, and FFB6D wins on YCB-V S + R).

On AR_{MSSD}, Cosypose is not the best method on any dataset, but, on AR_{MSPD}, Cosypose is the best on every dataset. On the one hand, MSSD is designed to verify if a pose has a chance of a successful grasp. On the other hand, MSPD does not evaluate the alignment along the optical axis (*Z* axis) and is designed for evaluating RGB-only methods. Therefore, MSSD is higher for PVN3D and FFB6D

Fig. A.15. Qualitative comparative of stacking methods on LMO dataset image 387 from scene 2.

since both methods use pointclouds instead of only RGB images. And for the same reason, Cosypose has higher MSPD because it is more suitable for working with RGB only input. Cosypose only uses depth if some pointcloud level refinement is performed, such as ICP.

The first strategy we are evaluating is the merge ensemble. Table 3 shows the results of the different merging methods for each dataset. The scores are underlined if, for that dataset and metric, the merge method has obtained a higher value than the corresponding highest level-0 score, and bold scores are the highest from the underlined scores for each row. On LMO PBR, results are close to Cosypose for all metrics, but Weighted clustering has the highest AR_{VSD}. On YCB-V PBR, PVN3D wins on AR_{VSD}, AR_{MSSD} and AR and none of the merge methods achieves better results. Nevertheless, weighted clustering has the highest AR_{VSD}, AR_{MSSD} and AR for YCB-V S + R. Merge methods can not deal with the T-LESS dataset since two out of three level-0 methods have very poor predictions. Even though, merge methods improve the scores of PVN3D and FFB6D.

The second strategy is stacking. Table 4 shows results of the stacking in which metrics with (s) denote that including the scores of the level-0 models improves the metric of the level-1. In this case, none of the level-1 models of the stacking improves the metrics obtained with the level-0 models. Moreover, there is not a level-1 model that stands out from the others. For each dataset, a different level-1 model obtains the highest metric.

Table 5 shows a comparative of the different level-0, all ensemble methods and two state-of-the-art methods leading the BOP challenge (CDPNv2 [7] and Pix2Pose [8]). As expected, some level-0 work better for some datasets and the leaderboard is very heterogeneous. On merge methods, the weighted clustering is by far the best approach, winning on 14 out of 20 rows. On stacking, instead, there is not a clear winner and a different level-1 works better for each dataset. Nevertheless, almost on every metric on the LMO and YCB-V datasets merge methods are better than the state-of-the-art methods.

Even if our initial hypothesis was that stacking strategy would lead to a better performance than merge strategy, since level-1 models could learn to distinguish which methods work better for each object and combine them in order to minimize the errors of the refined predictions, that is not the reality. This could be due to various reasons which may include that bad predictions of level-0 models lead to a bad level-1 model performance, or level-1 models were not able to learn a pattern. Nevertheless, if we focus on qualitative results (Figs. 8 and 9) we can see that the obtained results are valid. Fig. 8 shows a qualitative comparison of the merge method on the LMO dataset. Simple and weighted merge have some errors on the cat object but both clustering methods refine the prediction and obtain a better result. For example, on Fig. 9, even if each individual prediction of the level-0 models is not perfect, the predictions that are not present in some predictions is obtained from the others and refined. SVR and MLP outper-

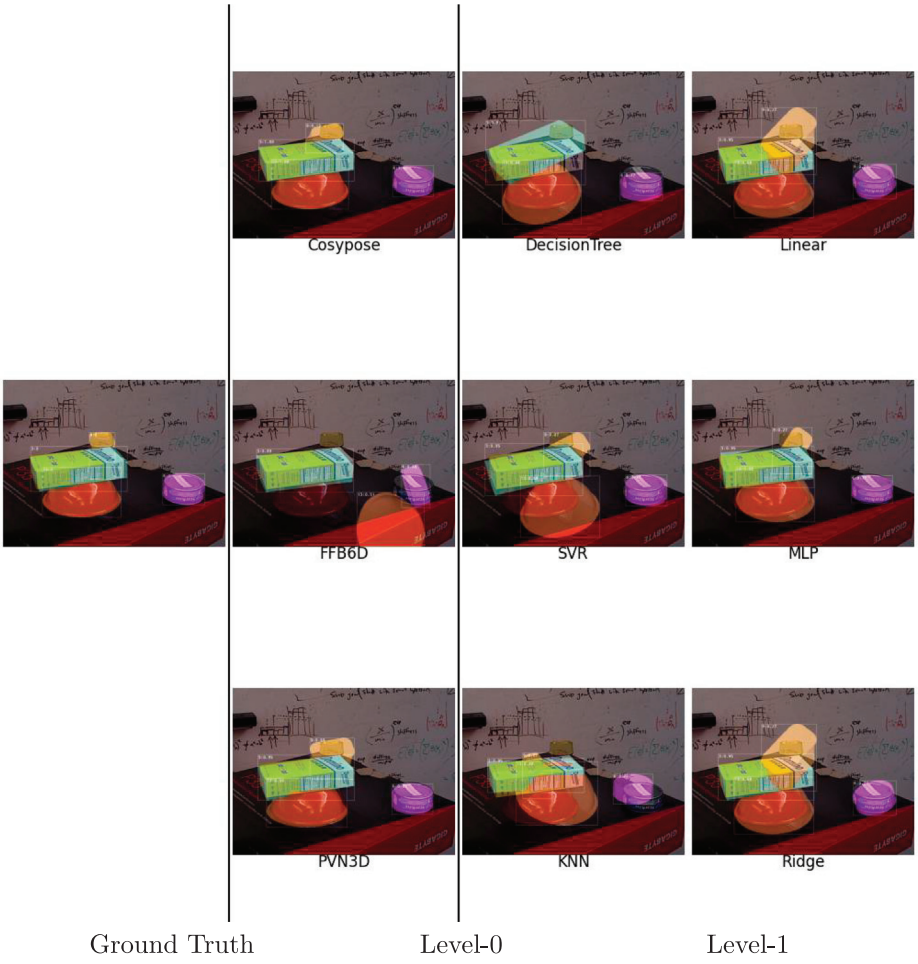


Fig. A.16. Qualitative comparative of stacking methods on YCBV dataset image 1557 from scene 49.

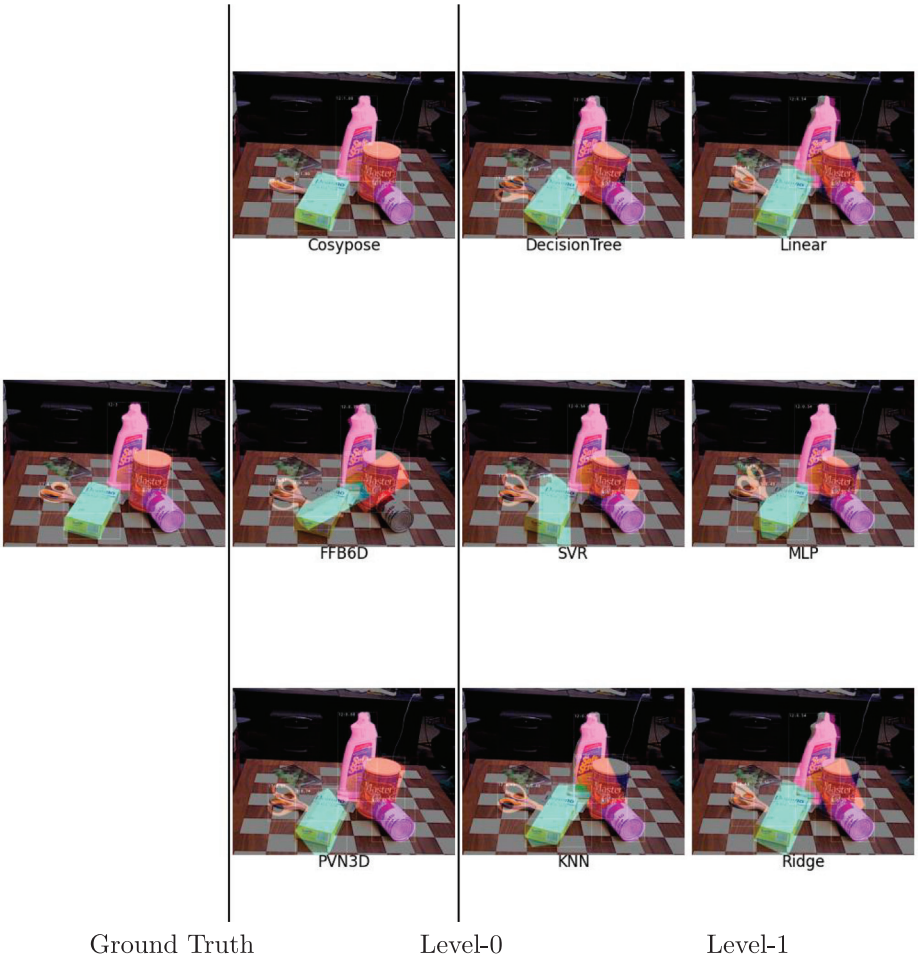


Fig. A.17. Qualitative comparative of stacking methods on YCBV dataset image 672 from scene 51.

form other level-1 models on situations like this. More examples of qualitative results can be found in the Appendix A.

7. Conclusions

To deal with 6DoF pose estimation task, we have designed some ensemble strategies. We have proposed 2 ensemble strategies for deep learning techniques: merge and stacking. On the one hand, merge strategy is based on the average of the predictions. This enables a geometrical refinement of the poses and, including the clustering technique, excludes the poses that do not represent the same instance. On the other hand, stacking trains a machine learning model (level-1) that inputs the outputs of the base models (level-0) to refine the estimated poses. Within merge strategies, the best method is weighted clustering which achieves 0.8352 AR, 0.7946 AR_{VSD}, 0.8816 AR_{MSSD} on YCB-V dataset using synthetic and real images, outperforming results obtained with FFB6d, PVN3D and Cosypose. Weighted clustering also improves AR_{VSD} on LMO. Stacking methods, instead, did not improve none of the results obtained with the level-0 models. The obtained results are for every dataset and level-1 model worse than the results obtained with level-0 models.

As stated before, qualitative results show that predictions are good enough for many applications. For example, robotic applications, where a robot must pick objects and little rotation discrepancies are not crucial, could obtain good results with the ensemble methods.

Future work will include using different and more level-0 models, discarding level-0 models that are not prepared for VIVO task and validating the obtained results in a real robotic application whose objective is to correctly grasp the maximum number of objects.

Data availability

Data will be made available on request.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Ibon Merino reports financial support was provided by Basque Government.

Acknowledgements

This paper has been supported by the project PROFLOW under the Basque program ELKARTEK, grant agreement No. KK-2022/00024.

Appendix A. More examples of voting and stacking

Figs. A.10 and A.11 and Figs. A.12 and A.13 show more examples of the comparatives of merge methods used on LMO and YCBV, respectively; and Figs. A.14 and A.15 and Figs. A.16 and A.17 show more examples of comparative of stacking methods for the LMO and YCBV datasets, respectively.

References

[1] G. Marullo, L. Tanzi, P. Piazzolla, E. Vezzetti, 6d object position estimation from 2d images: a literature review, Multimedia Tools and Applications (2022) 1–39.

[2] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. Glent Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, C. Rother, BOP: Benchmark for 6D object pose estimation, European Conference on Computer Vision (ECCV).

[3] B. Drost, M. Ulrich, N. Navab, S. Ilic, Model globally, match locally: Efficient and robust 3d object recognition, in: 2010 IEEE computer society conference on computer vision and pattern recognition, IEEE, 2010, pp. 998–1005.

[4] C. Raposo, J.P. Barreto, Using 2 point+ normal sets for fast registration of point clouds with small overlap, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2017, pp. 5652–5658.

[5] J. Vidal, C.-Y. Lin, X. Lladó, R. Martí, A method for 6d pose estimation of free-form rigid objects using point pair features on range data, Sensors 18 (8) (2018) 2678.

[6] P. Rodrigues, M. Antunes, C. Raposo, P. Marques, F. Fonseca, J.P. Barreto, Deep segmentation leverages geometric pose estimation in computer-aided total knee arthroplasty, Healthcare technology letters 6 (6) (2019) 226–230.

[7] Z. Li, G. Wang, X. Ji, Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 7678–7687.

[8] K. Park, T. Patten, M. Vincze, Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 7668–7677.

[9] S. Zakharov, I. Shugurov, S. Ilic, Dpod: 6d pose object detector and refiner, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 1941–1950.

[10] M. Sundermeyer, Z.-C. Marton, M. Durner, R. Triebel, Augmented autoencoders: Implicit 3d orientation learning for 6d object detection, International Journal of Computer Vision 128 (3) (2020) 714–729.

[11] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, J. Sun, Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 11632–11641.

[12] Y. He, H. Huang, H. Fan, Q. Chen, J. Sun, Ffb6d: A full flow bidirectional fusion network for 6d pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 3003–3013.

[13] Perkins, A model-based vision system for industrial parts, IEEE Transactions on Computers C-27 (2) (1978) 126–143.

[14] Tsuji Yachida, A versatile machine vision system for complex industrial parts, IEEE Transactions on Computers C-26 (9) (1977) 882–894.

[15] D. Kragic, M. Vincze, Vision for robotics, Now Publishers Inc, 2009.

[16] D.G. Lowe, Object recognition from local scale-invariant features, in: Proceedings of the seventh IEEE international conference on computer vision, Vol. 2, IEEE, 1999, pp. 1150–1157.

[17] J. Rambach, C. Deng, A. Pagani, D. Stricker, Learning 6dof object poses from synthetic single channel images, in: 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), IEEE, 2018, pp. 164–169.

[18] Y. Xiang, T. Schmidt, V. Narayanan, D. Fox, Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes, arXiv preprint arXiv:1711.00199.

[19] W. Zou, D. Wu, S. Tian, C. Xiang, X. Li, L. Zhang, End-to-end 6dof pose estimation from monocular rgb images, IEEE Transactions on Consumer Electronics 67 (1) (2021) 87–96.

[20] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.

[21] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, S. Savarese, Densefusion: 6d object pose estimation by iterative dense fusion, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 3343–3352.

[22] G. Pavlakos, X. Zhou, A. Chan, K.G. Derpanis, K. Daniilidis, 6-dof object pose from semantic keypoints, in: 2017 IEEE international conference on robotics and automation (ICRA), IEEE, 2017, pp. 2011–2018.

[23] M. Rad, V. Lepetit, Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 3828–3836.

[24] B. Tekin, S.N. Sinha, P. Fua, Real-time seamless single shot 6d object pose prediction, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 292–301.

[25] X. Zhou, A. Karpur, L. Luo, Q. Huang, Starmap for category-agnostic keypoint and viewpoint estimation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 318–334.

[26] A. Nigam, A. Penate-Sanchez, L. Agapito, Detect globally, label locally: Learning accurate 6-dof object pose estimation by joint segmentation and coordinate regression, IEEE Robotics and Automation Letters 3 (4) (2018) 3960–3967.

[27] S. Peng, Y. Liu, Q. Huang, X. Zhou, H. Bao, Pvnet: Pixel-wise voting network for 6dof pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4561–4570.

[28] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, L.J. Guibas, Normalized object coordinate space for category-level 6d object pose and size estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2642–2651.

[29] C.R. Qi, L. Yi, H. Su, L.J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, Advances in neural information processing systems 30.

[30] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660.

[31] H. Liu, S. Fang, Z. Zhang, D. Li, K. Lin, J. Wang, Mfdnet: Collaborative poses perception and matrix fisher distribution for head pose estimation, IEEE Transactions on Multimedia 24 (2021) 2449–2460.

[32] H. Liu, T. Liu, Z. Zhang, A.K. Sangaiah, B. Yang, Y. Li, Arhpe: Asymmetric relation-aware representation learning for head pose estimation in industrial human–computer interaction, IEEE Transactions on Industrial Informatics 18 (10) (2022) 7107–7117.

[33] H. Liu, C. Zheng, D. Li, X. Shen, K. Lin, J. Wang, Z. Zhang, Z. Zhang, N.N. Xiong, Edmf: Efficient deep matrix factorization with review feature learning for industrial recommender system, IEEE Transactions on Industrial Informatics 18 (7) (2021) 4361–4371.

[34] H. Liu, T. Liu, Y. Chen, Z. Zhang, Y.-F. Li, Ehpe: Skeleton cues-based gaussian coordinate encoding for efficient human pose estimation, IEEE Transactions on Multimedia.

[35] T. Liu, J. Wang, B. Yang, X. Wang, Ngdnet: Nonuniform gaussian-label distribution learning for infrared head pose estimation and on-task behavior understanding in the classroom, Neurocomputing 436 (2021) 210–220.

[36] D.H. Wolpert, Stacked generalization, Neural networks 5 (2) (1992) 241–259.

[37] K.M. Ting, I.H. Witten, Issues in stacked generalization, Journal of artificial intelligence research 10 (1999) 271–289.

[38] A. Ledezma, R. Aler, D. Borrajo, Heuristic Search-Based Stacking of Classifiers, IGI Global, 2002.

[39] A. Gupta, A.R. Thakkar, Optimization of stacking ensemble configuration based on various metaheuristic algorithms, IEEE (2014) 444–451.

[40] Y. Chen, M.L. Wong, An ant colony optimization approach for stacking ensemble, in, Second World Congress on Nature and Biologically Inspired Computing (NaBIC), IEEE 2010 (2010) 146–151.

[41] Y. Chen, M.-L. Wong, Applying ant colony optimization in configuring stacking ensemble, IEEE (2012) 2111–2116.

[42] P. Shunmugapriya, S. Kanmani, Optimization of stacking ensemble configurations through artificial bee colony algorithm, Swarm and Evolutionary Computation 12 (2013) 24–32.

[43] J. Kozak, Ensemble methods, in: Decision Tree and Ensemble Learning Based on Ant Colony Optimization, Springer, 2019, pp. 107–118.

[44] M.P. Sesmero, A.I. Ledezma, A. Sanchis, Generating ensembles of heterogeneous classifiers using stacked generalization, Wiley interdisciplinary reviews: data mining and knowledge discovery 5 (1) (2015) 21–34.

[45] S. Nair, A. Gupta, R. Joshi, V. Chitre, Combining varied learners for binary classification using stacked generalization, arXiv preprint arXiv:2202.08910.

[46] A.I. Naimi, L.B. Balzer, Stacked generalization: an introduction to super learning, European journal of epidemiology 33 (5) (2018) 459–464.

[47] C.-F. Tsai, Training support vector machines based on stacked generalization for image classification, Neurocomputing 64 (2005) 497–503, trends in Neurocomputing: 12th European Symposium on Artificial Neural Networks 2004.

[48] L. Peppoloni, M. Satler, E. Luchetti, C.A. Avizzano, P. Tripicchio, Stacked generalization for scene analysis and object recognition, in: IEEE 18th International Conference on Intelligent Engineering Systems INES 2014, 2014, pp. 215–220.

[49] A. Voulodimos, N. Doulamis, A. Doulamis, E. Protopapadakis, Deep learning for computer vision: A brief review, Computational intelligence and neuroscience (2018).

[50] Y. Labbé, J. Carpentier, M. Aubry, J. Sivic, Cosypose: Consistent multi-view multi-object 6d pose estimation, in: European Conference on Computer Vision, Springer, 2020, pp. 574–591.

[51] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, N. Navab, Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes, in: Asian conference on computer vision, Springer, 2012, pp. 548–562.

[52] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, A.M. Dollar, The ycb object and model set: Towards common benchmarks for manipulation research, in: 2015 international conference on advanced robotics (ICAR), IEEE, 2015, pp. 510–517.

[53] A. Simonelli, S.R. Bulo, L. Porzi, M. López-Antequera, P. Kontschieder, Disentangling monocular 3d object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1991–1999.

[54] F.L. Markley, Y. Cheng, J.L. Crassidis, Y. Oshman, Averaging quaternions, Journal of Guidance, Control, and Dynamics 30 (4) (2007) 1193–1197.

[55] H. Drucker, C.J. Burges, L. Kaufman, A. Smola, V. Vapnik, Support vector regression machines, Advances in neural information processing systems 9.

[56] C. Cortes, V. Vapnik, Support-vector networks, Machine learning 20 (3) (1995) 273–297.

[57] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and regression trees, Routledge, 1984.

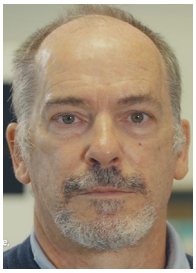
[58] D.C. Montgomery, E.A. Peck, G.G. Vining, Introduction to linear regression analysis, John Wiley & Sons, 2021.

[59] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE transactions on information theory 13 (1) (1967) 21–27.
[60] N.S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician 46 (3) (1992) 175–185.
[61] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural networks 2 (5) (1989) 359–366.
[62] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, J. Matas, BOP challenge 2020 on 6D object localization, European Conference on Computer Vision Workshops (ECCVW).
[63] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, C. Rother, Learning 6d object pose estimation using 3d object coordinates, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 536–551.
[64] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, X. Zabulis, T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects, IEEE Winter Conference on Applications of Computer Vision (WACV).
[65] T. Hodaň, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. Sinha, B. Guenter, Photorealistic image synthesis for object instance detection, IEEE International Conference on Image Processing (ICIP).
[66] T. Hodaň, J. Matas, Š. Obdržálek, On evaluation of 6d object pose estimation, in: European Conference on Computer Vision, Springer, 2016, pp. 606–619.
[67] B. Drost, M. Ulrich, P. Bergmann, P. Hartinger, C. Steger, Introducing mvtec itodd-a dataset for 3d object recognition in industry, in: Proceedings of the IEEE international conference on computer vision workshops, 2017, pp. 2200–2208.

Ibon Merino received his B.S. degree in Computer Science from the University of the Basque Country, Donostia-San Sebastian, Spain in 2017 and his M. Sc. in Computer engineering and intelligent systems from the same university in 2018. Currently, he is working towards the Ph.D. degree at Tecnalia and the University of the Basque Country. His research focuses in 3D Computer Vision applied for robotics and merging deep learning methods and non-deep learning methods.

Jon Azpiazu received his B.S degree in Computer Engineer from the University of the Basque Country - Euskal Herriko Unibertsitatea, UPV/EHU (2003) and Certificate of Advanced Studies in the "Computer Engineering: Communications, Control and Artificial Intelligence" programme from the same university in 2005. In 2005 he joined Fatronik (later Tecnalia) as a researcher, initially participating in projects related to Artificial Intelligence and Machine Learning; later his research focused on aspects of Artificial Vision, leading the line of Artificial Vision in the field of robotics. In 2013 he joined SINTEF, the largest Scandinavian research centre, as a scientific researcher. During this stage, he participates and leads several research projects related to robotics and machine vision, both applied research projects (mainly concepts of use of robotics for the oil & gas industry), as well as basic research projects such as a project on semantic perception of the environment for snake robots. In 2017 he rejoined Tecnalia in the robotics group, leading two European projects in this field.

Dr. Anthony Remazeilles obtained a MSc in Computer Science and Artificial Intelligence from the University of Rennes 1 in 2001, and a PhD in Computer science from the University of Rennes in 2004. From 2001 to 2006, his research concerned the vision-based control of robotics systems within large navigation spaces. From 2006 to 2008, he was a Post-Doctoral researcher at the Commissariat ? l'Energie Atomique (CEA-LIST), Fontenay aux Roses, France. He was involved in the development of a semi-autonomous mobile arm providing vision-based grasping assistance to disabled people within the framework of the ITEA ANSO project that received an ITEA Appreciation Award in 2008. He has contributed to more than 25 peer-reviewed articles (conferences and journals), and holds a patent on vision-based object grasping. In 2004, he received the Best Paper Award in Pattern Recognition in the French Congress on Pattern Recognition and Artificial Intelligence (RFIA). Since 2008, he has been working as Researcher and then Project Leader in the Health Division of Tecnalia, designing intelligent robotics systems for human-centred applications. Dr. Remazeilles has been actively contributing to various project sponsored by the European Commission, such as Florence, CogLaboration, STIFF-FLOP, SARAFun, ROSIN, Robotunion, Eurobench, TraceBot.



Prof. Basilio Sierra is Full Professor at the Computer Science and Artificial Intelligence Department in the University of the Basque Country. Prof. Sierra has published more than 50 papers in journals with impact factor and more than 100 conference papers. His areas of scientific interest focus on Robotics, Machine Learning, Data Analysis and Computer Vision, having made numerous contributions in this area both at the theoretical level and in the development of new algorithms and their application in solving real problems. He has supervised 20 doctoral theses, and participated in several research projects related with Artificial Intelligence.