

The challenge of simultaneous object detection and pose estimation: a comparative study

Daniel Oñoro-Rubio, Roberto J. López-Sastre, Carolina Redondo-Cabrera and Pedro Gil-Jiménez

GRAM, University of Alcalá, Alcalá de Henares, 28805, Spain

Abstract

Detecting objects and estimating their pose remains as one of the major challenges of the computer vision research community. There exists a compromise between localizing the objects and estimating their viewpoints. The detector ideally needs to be view-invariant, while the pose estimation process should be able to generalize towards the category-level. This work is an exploration of using deep learning models for solving both problems simultaneously. For doing so, we propose three novel deep learning architectures, which are able to perform a joint detection and pose estimation, where we gradually decouple the two tasks. We also investigate whether the pose estimation problem should be solved as a classification or regression problem, being this still an open question in the computer vision community. We detail a comparative analysis of all our solutions and the methods that currently define the state of the art for this problem. We use PASCAL3D+ and ObjectNet3D datasets to present the thorough experimental evaluation and main results. With the proposed models we achieve the state-of-the-art performance in both datasets.

Keywords: Pose estimation, viewpoint estimation, object detection, deep learning, convolutional neural network

1. Introduction

Over the last decades, the category-level object detection problem has drawn considerable attention. As a result, much progress has been realized, leaded mainly by international challenges and benchmarking datasets, such as the PASCAL VOC Challenges [1] or the ImageNet dataset [2]. Nevertheless, researchers soon identified the importance of not only localizing the objects, but also estimating their poses or viewpoints, *e.g.* [3, 4, 5, 6]. This new capability results fundamental to enable a true interaction with the world and its objects. For instance, a robot which merely knows the location of a cup but that cannot find its handle, will not be able to grasp it. In the end, the robotic solution needs to know a viewpoint estimation of the object to facilitate the inference of the visual affordance for the object. Also, in the augmented reality field, to localize and estimate the viewpoint of the objects, is a crucial feature in order to project a realistic hologram, for instance.

Technically, given an image, these models can localize the objects, predicting their associated bounding boxes, and are also able to estimate the relative pose of the object instances in the scene with respect to the camera. Figure 1 shows an example, where the viewpoint of the object is encoded using just the azimuth angle. In the image, the target objects are the sofa and the bicycle. Their locations are depicted by their bounding boxes (in green), and their azimuth angles are represented by the blue arrow inside the yellow circle.

The computer vision community rapidly detected the necessity of providing the appropriate annotated datasets, in order to

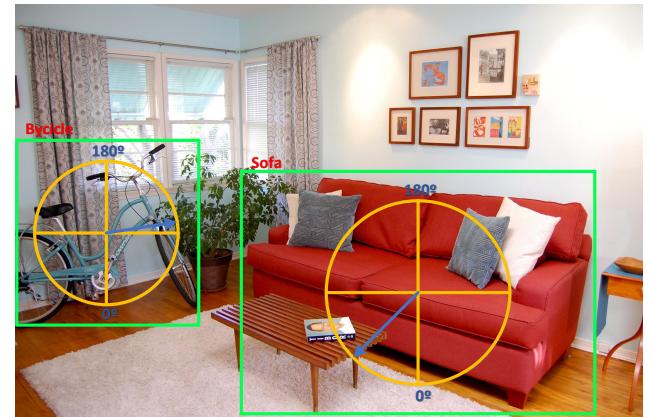


Figure 1: Object category detection and pose estimation example. In the image, the sofa and the bicycle are localized by the green bounding boxes. The blue arrow inside the yellow circles shows the azimuth angles of the objects, which is a form of viewpoint annotation.

experimentally validate the object detection and pose estimations approaches. To date, several datasets have been released. Some examples are: 3D Object categories [4], EPFL Multi-view car [7], ICARO [8], PASCAL3D+ [9] or ObjectNet3D [10].

Thanks to these datasets, multiple models have been experimentally evaluated. It is particularly interesting to observe how all the published approaches can be classified in two groups. In the first one, we find those models that decouple both problems (*e.g.* [11, 12, 13]), making first a location of the object, to later estimate its pose. In the second group we identify the approaches that solve both tasks simultaneously (*e.g.* [9, 14, 15]), because they understand that to carry out a correct location re-

Email address: daniel.onoro@edu.uah.es,
robertoj.lopez@uah.es, carolina.redondoc@edu.uah.es and
pedro.gil@uah.es (Daniel Oñoro-Rubio, Roberto J. López-Sastre, Carolina Redondo-Cabrera and Pedro Gil-Jiménez)

quires a good estimation of the pose, and vice versa.

But the discrepancies do not end here. Unlike the problem of object detection, where the metric for the experimental evaluation is clear, being this the mean Average Precision (mAP) defined in the PASCAL VOC Challenge, for the problem of object detection and pose estimation, multiple metrics have been adopted. This is motivated by the fact that not all the models understand the viewpoint estimation problem in the same way. Some solutions, *i.e.* the discrete approaches, consider that this is a classification problem, when others, *i.e.* the continuous models, understand the pose as a continuous variable, whose estimation must be approached by solving a regression problem.

This article is an attempt to provide a comparative study where these issues can be addressed. The main contributions of this work are as follows:

- We introduce three novel deep learning architectures for the problem of simultaneous object detection and pose estimation. Our models seek to perform a joint detection and pose estimation, trained fully end-to-end. We start with a model that fully integrates the tasks of object localization and object pose estimation. Then, we present two architectures that gradually decouple both tasks, proposing a final deep network where the integration is minimal. All our solutions are detailed in Sections 3.1 and 3.2.
- All our architectures have been carefully designed to be able to treat the pose estimation problem from a continuous or from a discrete perspective. We simply need to change the loss functions used during learning. This is detailed in Section 3.3. Therefore, in our experiments, we carefully compare the performance of these two families of methods, reporting results using four different loss functions. Therefore, this paper aims to shed some light on which perspective is more appropriate, keeping the network architecture fixed.
- Thanks to the proposed models, we are able to offer an experimental evaluation (see Section 4) designed to carefully analyze how coupled the detection and pose estimation tasks are, being this our final contribution. We also bring a detailed comparison with all the solutions that establish the state-of-the-art for the problem of object category detection and pose estimation. We carefully analyze all the models using two publicly available datasets: PASCAL3D+ [9] and ObjectNet3D [10].

2. Related Work

Object category detection and viewpoint estimation is a growing research field. Several are the methods that have contributed to improve the state of the art. Like we just have said, we can organize in two groups all the approaches in the literature.

In the first one, we find those models that understand that these two tasks, *i.e.* object localization and pose estimation,

must be solved separately [11, 12, 13]. The second group consists of the models where the detection and the viewpoint estimation are fully coupled [9, 14, 15, 16].

Within these two groups, one must note that while some models solve the pose estimation as a classification problem, *i.e.* the discrete approaches [15, 17], others treat the viewpoint estimation as a regression problem, *i.e.* the continuous solutions [12, 16, 18].

In this paper, we introduce three novel deep learning architectures for the problem of *joint* object detection and pose estimation. They all are extensions for the excellent Faster R-CNN object detection model [19]. We have designed them to gradually decouple the object localization and pose estimation tasks. Our models significantly differ from previous deep learning based approaches for the same tasks. For instance, if we consider the work of Tulsiani *et al.* [17], we observe that their solution is based on a detector (using the R-CNN [20]), followed by a pose classification network, fully decoupling both tasks. On the contrary, all our architectures are trained fully end-to-end, performing a joint detection and viewpoint estimation. Moreover, the deep architectures implemented are different. Massa *et al.* [15] also propose a *joint* model. However, their approach is completely different. They base their design on the Fast R-CNN detector [21]. Technically, they modify the Fast R-CNN output to provide the detections based on an accumulative sum of scores that is provided by the pose classification for each object category. In a different manner, our solutions are based on the Faster R-CNN, which is a distinct architecture. Moreover, in our work we explore not only a modification of the output of the networks, but multiple architecture designs where we can gradually separate the branches of the network dedicated to the object localization and the viewpoint estimation tasks.

Finally, this paper offers a detailed comparative study of solutions for the joint object *detection* and pose estimation problem. The study included in [22] focus on the different problem of object *classification* and pose estimation, *i.e.* they do not consider the object localization task.

3. Simultaneous detection and pose estimation models

In the following section, we formulate the learning problem for a joint detection and pose estimation. Then, we detail the proposed architectures, named: single-path, specific-path, and specific-network (Figure 2 shows an overview of all our designs). Technically, they all are extensions for the Faster R-CNN approach [19]. Finally, we provide a detailed analysis of the loss functions used in our experimental evaluation.

3.1. Learning model for simultaneous detection and pose estimation

Our goal is to learn a strong visual representation that allows the models to: localize the objects, classify them and estimate their viewpoint with respect to the camera. Furthermore, we consider an *in the wild* setting where multiple objects of a

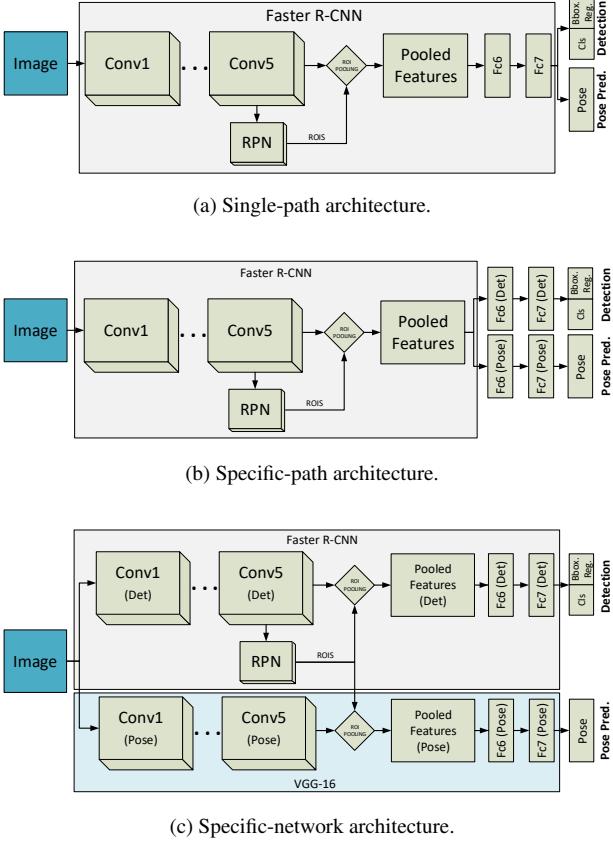


Figure 2: Proposed deep learning architectures for simultaneous object detection and pose estimation.

variety of categories appear in real-world scenarios, with a considerable variability on the background, and where occlusions and truncations are the rule rather than the exception.

Therefore, the supervised learning process starts from a training set $S = \{(x_i, t_i)\}_{i=1}^N$, where N is the number of training samples. For each sample i in the dataset, $x_i \in X$ represents the input image, and $t_i \in T$, with $t_i = (y_i, \beta_i, \phi_i)$, encodes the annotations for the three tasks to solve: classification (y_i), object localization (β_i) and pose estimation (ϕ_i). $y_i \in Y$ with $Y = [1, 2, \dots, C, C + 1]$ describes the object class, being C the total number of object categories. Category $C + 1$ is used to consider a generic background class. $\beta_i \in \mathbb{R}^4$ represents the bounding box localization of a particular object within image x_i . Finally, $\phi_i \in \mathbb{R}^3$ encodes the 3D viewpoint annotation for a particular object with respect to the camera position as a tuple of azimuth, elevation and zenith angles.

We propose to learn a convolutional neural network (CNN) [23] for simultaneous object detection and pose estimation. Technically, these CNNs are a combination of three main features which let the model achieve a sort of invariance with respect to imaging conditions: local receptive fields, shared convolutional weights, and spatial pooling. Each unit in a layer receives inputs from a set of units located in a small neighborhood of the previous layer. In the forward pass of a CNN, each output feature is computed by the convolution of the input feature from the previous layer. Therefore, these deep networks can be thought

of as the composition of a number of convolutional structure functions, which transform the input image to feature maps that are used to solve the target tasks.

For the particular problem of simultaneous object detection and viewpoint estimation, our CNN prediction \hat{t} should be expressed as follows,

$$\hat{t}_{\theta, W} = F_W \circ \mathbf{z}_\theta(x_i). \quad (1)$$

$\mathbf{z}_\theta : X \rightarrow \mathbb{R}^D$ represents the D -dimensional feature mapping that the network performs to the input images. Technically, it consists in the transformation of the input image x_i into a feature that is used to feed the output layers of our models. We encode in θ the trainable weights of the deep architecture that allow the network to perform the mapping. In our solutions, the weights in θ define the hidden layers that are shared by all the tasks that the deep network needs to solve.

F_W corresponds to the set of functions of the output layers. They take as input the deep feature map $\mathbf{z}_\theta(x_i)$. For the problem considered in this paper, our set of functions must address three different tasks: classification (y), object localization (β) and viewpoint estimation (ϕ). Therefore, $F_W = (f_{W^y}^y, f_{W^\beta}^\beta, f_{W^\phi}^\phi)$. $f_{W^y}^y$ with weights W^y produces the prediction for the object category, i.e. \hat{t}^y . $f_{W^\beta}^\beta$ predicts the object location \hat{t}^β . Finally, $f_{W^\phi}^\phi$ is in charge of the prediction of the viewpoint \hat{t}^ϕ .

According to the prediction model detailed in Equation 1, we define the following objective function to learn our multi-task neural network:

$$\arg \min_{\theta, W} \mathcal{L}(\theta, W, S), \quad (2)$$

where the loss function follows the equation,

$$\mathcal{L}(\theta, W, S) = \lambda_1 \mathcal{L}_y(\theta, W^y, S) + \lambda_2 \mathcal{L}_\beta(\theta, W^\beta, S) + \lambda_3 \mathcal{L}_\phi(\theta, W^\phi, S). \quad (3)$$

λ_i for $i \in (1, 2, 3)$ corresponds to the scalar value that controls the importance of a particular loss during training. For the classification loss \mathcal{L}_y we use a categorical cross-entropy function. A simple Euclidean loss is used for the object localization task loss \mathcal{L}_β . Finally, for the pose estimation loss \mathcal{L}_ϕ multiple options are considered. We detail them in Section 3.3.

3.2. The proposed architectures

3.2.1. Single-path architecture

Our first deep network design is the *single-path* architecture. It offers a natural extension of the Faster R-CNN model for the problem of simultaneous object detection and pose estimation. Technically, we simply add an extra output layer in order to predict the viewpoint of the object.

To understand the extension proposed, we proceed with a description of the original Faster R-CNN pipeline. As it is shown in Figure 2a, the Faster R-CNN consists of three stages. The first stage is performed by the convolutional layers. An input image passes through the convolutional network, to be transformed into a deep feature map. The second stage is represented by the Region Proposal Network (RPN), which serves as

an “attention” mechanism during learning. Technically, it is a fully convolutional (sub)network, which takes an image feature map as input, and outputs a set of rectangular object proposals, with their corresponding objectness scores. To go into details, this RPN takes the feature map obtained from the last convolutional layer (*e.g.* convolution 5 in a VGG16-based architecture), and adds a new convolutional layer which is in charge of learning to generate regions of interest (ROIs). In the third stage, these ROIs are used for pooling those features that are passed to the last two fully-connected (FC) layers. Finally, the responses coming from the last FC layer are used by the model: 1) to classify the ROIs into background or object; and 2) to perform a final bounding box regression for a fine-grained localization of the object. In Figure 2a we represent these two tasks with the blocks named as “Cls” (for classification) and “Bbox. Reg.” (for the bounding box regression). Technically, the “Cls” module is implemented with a softmax layer, and the “Bbox. Reg.” layer is a linear regressor for the four coordinates that define a bounding box.

In order to evaluate the capability of the Faster R-CNN for the task of pose estimation, guaranteeing a minimal intervention in the model architecture, we propose the *single-path* extension. It consists in the incorporation of an additional output layer (see box “Pose” in Figure 2a), connected to the last FC layer as well. The objective of this layer is to cast a prediction for the viewpoint, and to measure the loss for this task, propagating the appropriate gradients to the rest of the network during learning.

For training this *single-path* model, we solve the objective loss function of Equation 2. We give the same weight to each task, *i.e.* $\lambda_1 = \lambda_2 = \lambda_3 = 1$. Note that at this point, we do not specify whether the viewpoint estimation will be considered as a classification or regression problem. In this sense, different loss functions will be considered and evaluated in the experiments, in order to attain a high level of understanding of the simultaneous detection and pose estimation problem.

3.2.2. Specific-path architecture

The *specific-path* is our second approach. Our objective with this architecture is to explore the consequences of a slightly separation of the pose estimation task from the object class detection, learning *specific deep features* for each task.

As it is shown in Figure 2b, the extension we propose for this second approach consists in adding two independent FC layers, which are directly connected to the pose estimation layer. Note that we do not change the rest of the architecture, *i.e.* both the initial convolutional layers and the RPN module are shared. The pooled features are used to feed the two groups of FC layers that form two types of features: one for the object detection task, and the other for the viewpoint estimation. Therefore, during training, each network FC path learns its specific features based on its gradients, while the rest of layers learn a shared representation.

The model is learned solving the objective function shown in Equation 2. For the detection path, $\lambda_1 = \lambda_2 = 1$, and $\lambda_3 = 0$. For the pose path we solve the Equation 2 getting rid of

the object classification and bounding box regression losses, *i.e.* $\lambda_1 = \lambda_2 = 0$ and $\lambda_3 = 1$.

3.2.3. Specific-network architecture

With our third architecture, named *specific-network*, we attempt to separate as much as possible the detection and pose estimation tasks within the same architecture. The key idea of this design is to provide a model with two networks that can be fully specialized in their respective tasks, while they are learned simultaneously and end-to-end.

Consequently, as it is shown in Figure 2c, we design a model made of two independent networks: the *detection network* and the *pose network*. The *detection network* is in charge of fully performing the object localization task, as in the original design of the Faster R-CNN.

The *pose network* must focus on the viewpoint estimation task, without any influence of the detection objective. Therefore, this network has now its own initial convolutional layers. To align the detection and pose estimation, the *pose network* receives the ROIs generated by the RPN module of the *detection network*. Technically, an input image is forwarded simultaneously into both convolutional networks. The second stage of the Faster R-CNN, *i.e.* the generation of ROIs by the RPN, occurs in the detection network only. These ROIs are shared with the *pose network*. Finally, each network pools its own features from the generated ROIs, feeds its FC layers with these features, and produces its corresponding outputs.

Overall, we have an architecture with two specialized networks, that are synchronized to solve the object detection and pose estimation tasks in a single pass.

For learning this model we follow the same procedure as for the *specific-path*. We train our *detection network* to solve the Equation 2 where $\lambda_3 = 0$ and $\lambda_1 = \lambda_2 = 1$. The *pose network* is solved just for the pose problem, hence, $\lambda_1 = \lambda_2 = 0$ and $\lambda_3 = 1$. The main difference with respect the *specific-path* model is that there are no shared features, so each network is fully specialized to solve its corresponding task.

3.2.4. Why have we chosen these designs?

All our architectures are extensions of the Faster R-CNN approach [19]. Originally, the Faster R-CNN architecture was proposed to address the problem of object detection only. This model has systematically prevailing on all the detection benchmarks (*e.g.* PASCAL VOC [1], COCO [24] and ILSVRC detection [2]), where leading results are obtained by Faster R-CNN based models, albeit with deeper features (*e.g.* using deep residual networks [25]). So, following a simple performance criterion, we believe that the Faster R-CNN with its excellent results is a good choice.

Our second criterion for the selection of this Faster R-CNN architecture is related with the main objective of our research: propose and evaluate solutions for the problem of *simultaneous* object detection and viewpoint estimation. Note that we neither address the problem of pose estimation in a classification setup in isolation (*e.g.* [22], where the object localization problem is not considered), nor decouple the object detection and pose estimation tasks (*e.g.* [17]). Our models seek to perform a *joint*

detection and pose estimation, trained fully end-to-end, and the Faster R-CNN architecture is an ideal candidate to extend. All our solutions perform a direct pooling of regions of interests in the images from the internal RPN of the Faster R-CNN. This way, we do not need to use any external process to hypothesize bounding boxes (*e.g.* Selective Search [26]), hence performing a truly end-to-end simultaneous object detection and pose estimation model, where the weights of the fully convolutional RPN learn to predict object bounds and objectness scores at each position, to maximize not only the object detection accuracy, but also the viewpoint estimation performance.

Finally, we want to discuss our main arguments for the concrete extensions proposed in our architectures. Traditionally, the computer vision community working on the problem of pose estimation for object categories has been divided into two groups. Those that understand that the tasks of localizing objects and estimating their poses are decoupled tasks (*e.g.* [17, 12, 18, 13]), and those that advocate for jointly solving both tasks (*e.g.* [15, 27, 16, 14]). The architectures proposed in this paper move from a *fully* integration of both tasks, *i.e.* in the *single-path*, towards the *specific-network* model, where the integration is minimal. In this way, we can design an experimental evaluation to thoroughly analyze how coupled the detection and pose estimation tasks are. Moreover, all our experiments are carried on publicly available dataset which have been designed for the problem of detection and viewpoint estimation, therefore a direct comparison with previous methods that define the state of the art is also possible.

3.3. Loss functions for pose estimation

Unlike the well-defined object detection task, the viewpoint estimation problem has been traditionally considered from two different perspectives: the continuous and the discrete. Most methods in the literature adopt the discrete formulation. That is, they understand the pose estimation as a classification problem, relying on a coarse quantization of the poses for their multi-view object detectors (*e.g.* [11, 14, 27]). Only a few approaches consider that the pose estimation of categories is ultimately a continuous problem, *i.e.* a regression problem (*e.g.* [16, 18, 28]). In this paper, all our architectures are evaluated considering these two perspectives for the viewpoint estimation.

When we want our models to consider discrete outputs for the pose estimation (the “Pose” layer in Figure 2), we integrate the following categorical cross-entropy loss function in Equation 2:

$$\mathcal{L}_\phi(\theta, W^\phi, S) = -\frac{1}{N} \sum_{i=1}^N \log \left(\sigma_{l_i^\phi} (f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i)) \right), \quad (4)$$

where N is the number of samples, and $\sigma_{l_i^\phi}$ is the softmax function for the label l_i^ϕ .

When the pose estimation is considered from the continuous perspective, multiple adequate regression loss functions can be integrated. For all them, it is fundamental to deal with the circularity of the viewpoint. Therefore, we first represent the

orientation angles as points on a unit circle by the following transformation, $\mathbf{p}(\alpha) = (\sin(\alpha), \cos(\alpha))$, $\mathbf{p}(\alpha) \in \mathbb{R}^2$.

Probably, the simplest way to train the pose regressor is by using an Euclidean loss, as follows:

$$\mathcal{L}_\phi(\theta, W^\phi, S) = \frac{1}{2N} \sum_{i=1}^N \left(\mathbf{p}(l_i^\phi) - \mathbf{p}(f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i)) \right)^2. \quad (5)$$

A popular alternative to the Euclidean loss, is the Huber loss function,

$$\mathcal{L}_\phi(\theta, W^\phi, S) = \frac{1}{N} \sum_{i=1}^N \begin{cases} \frac{1}{2} \left(\mathbf{p}(l_i^\phi) - \mathbf{p}(f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i)) \right)^2 & \text{if } \left| \mathbf{p}(l_i^\phi) - \mathbf{p}(f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i)) \right| \leq \delta, \\ \delta \left| \mathbf{p}(l_i^\phi) - \mathbf{p}(f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i)) \right| - \frac{1}{2} \delta^2 & \text{otherwise} \end{cases}. \quad (6)$$

The advantage of this loss is that it tends to be more robust to outliers than the Euclidean loss.

Finally, we propose to also use the continuous cyclic cosine cost function, which is widely used in the natural language processing literature [29]. It is defined as follows,

$$\mathcal{L}_\phi(\theta, W^\phi, S) = \frac{1}{N} \sum_{i=1}^N \left(1 - \frac{\mathbf{p}(l_i^\phi) \mathbf{p}(f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i))}{\|\mathbf{p}(l_i^\phi)\| \|\mathbf{p}(f_{W^\phi}^\phi \circ \mathbf{z}_\theta(x_i))\|} \right). \quad (7)$$

4. Experiments

4.1. Implementation details

To perform our experiments, we have implemented all our models and loss functions using the deep learning framework Caffe [30]. The optimization is done by using the Stochastic Gradient Descent algorithm, with: a momentum of 0.9; a weight decay of 0.0005; and a learning rate of 0.001. The learning rate of the output layer for the pose estimation has been multiplied by a factor of 0.01, so as to guarantee that the network properly converges. We publicly release all our implementations¹.

We follow the standard procedure of the Faster R-CNN [19] for training the models in an end-to-end fashion. This way, for each training iteration, just one image is taken and passed through the first set of convolutions. In a second step, a collection of 128 region proposals is generated. These regions are used to build the batch to feed the last set of FC layers. This batch contains 32 samples of foreground samples and 96 samples of background.

For the experimental evaluation, we use two publicly available datasets, which have been especially designed for the evaluation of object detection and pose estimation models: PASCAL3D+ [9] and ObjectNet3D [10]. We strictly follow the experimental setup described in these datasets. In the following sections, more details are provided, as well as a thorough analysis of the results and main conclusions obtained.

4.2. Results in the PASCAL3D+ dataset

PASCAL3D+ [9] dataset is one of the largest and most challenging datasets for the problem of object detection and pose

¹The link to download all the models and software to reproduce the results will be inserted once the paper gets accepted.



Figure 3: Some images of the PASCAL3D+ dataset.

estimation. Technically, it consists of: 1) the images and annotations of the 12 rigid object categories provided with the PASCAL VOC 2012 dataset [1]; and 2) an additional set of 22,394 images taken from the ImageNet [2] dataset, for the same 12 categories. On average, it has more than 3000 instances per object category. The test set has 5823 images directly inherited from the PASCAL VOC 2012 test subset. Figure 3 shows some examples of images. One can clearly observe that the images provided contain objects “in the wild”. The standard PASCAL VOC annotation for all the objects (*i.e.* category label and bounding box), has been extended to provide a precise 3D pose. This has been done performing a manual alignment of the objects in the images with 3D CAD models. This way, azimuth, elevation and distance from the camera pose in 3D are provided for each object.

For our analysis, we follow the official experimental setup of the PASCAL3D+ [9]. The evaluation metric for the object detection and pose estimation is the Average Viewpoint Precision (AVP). This AVP is similar to the Average Precision (AP) for object detection. To compute the AVP, every output of the detector is considered to be correct if and only if the bounding box overlap with the ground truth annotation is larger than 50% *and* the viewpoint estimation for the azimuth angle is correct. When we consider a discrete space for the viewpoint, the viewpoint estimation is correct if it coincides with the ground truth azimuth label. On the contrary, if the viewpoint belongs to a continuous space, then, two viewpoint labels are correct if the distance between them is smaller than a fixed threshold of $\frac{2\pi}{v}$, where v is the number of views.

4.2.1. Network initialization analysis

One of the most common practices in deep learning consists in initializing a deep network architecture with the weights of a model pre-trained in a big dataset, such as ImageNet [2], and then start a fine tuning process for a specific task, typically using a different dataset.

For our problem of joint object detection and pose estimation, we also follow this popular recipe. In a nutshell, we fine tune our networks in the PASCAL3D+ dataset, using for the initialization of the weights two pre-trained models: the original VGG16 model [31] trained for the ImageNet dataset, and

the Faster R-CNN model [19] using only the training set of the PASCAL VOC 2012 dataset. Note that the validation set of the original PASCAL VOC 2012 is now the test set proposed in the PASCAL3D+, therefore, we do not allow the Faster R-CNN to be pre-trained on it. For the rest of model weights that are not covered by the pre-trained models, we basically follow a standard random initialization.

Here we simply want to explore what initialization procedure is the best option. Therefore, for this preliminary experiment, we just use our first architecture, the *Single-path*. The pose estimation is considered as a classification problem, using 360 discrete bins, and we employ the cross-entropy loss defined in Eq. 4.

Init. strategy	mAP	mAVP 4	mAVP 8	mAVP 16	mAVP 24
ImageNet	49.5	37.6	32.0	24.6	20.2
PASCAL VOC 2012	63.6	42.4	32.2	23.6	18.9

Table 1: Effect of the network initialization strategy in the PASCAL3D+ for the *Single-path* architecture.

Table 1 shows the main results using the described initialization strategies. In terms of object detection precision, *i.e.* mAP, the initialization of our model, using the PASCAL VOC 2012 datasets is the best option, by a considerable margin, with respect to the ImageNet based strategy. Interestingly, the mAP of our model (63.6) improves the state-of-the-art for the object detection task in the official PASCAL3D+ leaderboard², where the best mAP is of 62.5 reported in [15].

In terms of a joint object detection and pose estimation, we also report the mAVP for different sets of views (4, 8, 16 and 24). The ImageNet based initialization reports slightly better results only for the more fine grained setups of 16 and 24 views. When just 4 or 8 views are considered, the initialization process using the PASCAL VOC 2012 is the best option, considering its high detection precision. This first experiment also reveals that it seems to be a trade-off between how good the system is localizing objects and how accurate the pose predictions are. Overall, we conclude that the best initialization strategy is clearly the

²Official PASCAL3D+ leaderboard is available at <http://cvgl.stanford.edu/projects/pascal3d.html>

Losses	mAP	mAVP 4	mAVP 8	mAVP 16	mAVP 24
Discrete (Eq. 4)	63.6	42.4	32.2	23.6	18.9
Euclidean (Eq. 5)	64.3	47.9	34.7	23.2	17.6
Huber (Eq. 6)	64.5	46.1	31.5	20.2	15.2
Cyclic Cosine (Eq. 7)	55.6	42.1	32.2	22.5	17.5

Table 2: Loss function analysis for the PASCAL3D+ dataset. Object detection and viewpoint estimation performances are reported.

one based on the PASCAL VOC 2012 dataset. Therefore, for the rest of experiments, we follow this initialization strategy.

4.2.2. Discrete vs. Continuous approaches analysis

As we have discussed in Section 3.3, the pose estimation problem can be treated following either a discrete approach, *i.e.* as a classification problem, or a continuous approximation, *i.e.* as a regression problem. One of the main objectives of our study is to shed light on this discussion.

We have carefully designed all our architectures, so they all can consider a discrete and a continuous approximation to the pose estimation problem. We simply have to *change* the Pose estimation layer, and its associated loss function. Up to four different loss functions are analyzed in these experiments, one for the discrete case and three for the continuous approach.

When the discrete scenario is considered, we follow the cross-entropy loss function in Equation 4. Technically, our architectures consider 360 different classes for the azimuth angle. For each category in the dataset (except for the background), we learn a specific pose estimator, therefore, we need to define a softmax function with a length of $360 \times C$ elements, where C is the number of classes. During learning, we have opted to *mask* the softmax layer, propagating only the error for the elements that correspond to the pose of the foreground class.

For the continuous pose estimation problem, our networks learn to directly perform the regression of the two values corresponding to the conversion to polar coordinates the azimuth angle. We design our deep models to learn a particular regressor for each object category. And again, during learning, only the regressor that corresponds to the associated class label of the sample in the training batch, is allowed to propagate errors. Following this continuous setup, we analyze the three different loss functions introduced in Section 3.3: the Euclidean loss (Eq. 5), the Huber loss (Eq. 6), and the Cyclic cosine loss (Eq. 7).

Table 2 shows the main results, when the different loss functions are used. Discrete, Euclidean and Huber losses exhibit a very similar detection performance (mAP). Only when the Cyclic Cosine loss is used, a substantial drop of the detection performance is reported. The reason we find to explain this fact is that during training, the Cyclic Cosine loss can eventually produce larger gradients than the detection loss. This issue causes that the learning process tends to focus more “attention” on the pose estimation task, obtaining a deep model with a worse object localization accuracy. A simple adjustment of the λ values in Eq. 3 did not properly work in our experiments. Another possibility could be to perform a power normalization of the gradients produced by the different losses at the same level

of the network. However, we did not explore this option. Instead, we opted for applying the clipping gradient strategy [32], with a threshold value of 5.

If we analyze now the mAVP, where both object detection and viewpoint estimation accuracies are considered, we can observe that, in general, the best performance is reported when the Euclidean loss based model is used. Moreover, within the group of continuous viewpoint estimation models, the Euclidean is the clear winner. Therefore, for the rest of the paper, when a continuous viewpoint model is learned, we use the Euclidean loss. Interestingly, the continuous approach wins the discrete model only when 4 and 8 set of views are considered. For 16 and 24 views, the discrete model retrieves a slightly better performance. In our experiments, we have noted that the continuous pose estimation approaches tend to offer smooth predictions that are concentrated around the most frequent viewpoint of the training set. However, the discrete approach, with a Softmax loss, does not suffer that much from this pose annotation bias.

Figure 4 shows a detailed comparison of the performance between a discrete and a continuous approach for a pair of representative object categories: *car* and *bus*. *Car* is the class with the largest amount of samples in the PASCAL3D+ dataset, *i.e.* 1004 instances of non-difficult objects. The annotated views for cars are distributed quite homogeneously across all the poses, although they are slightly biased towards the frontal and rear views. Category *bus* provides only 301 samples, and the pose is clearly concentrated in the frontal view.

For the category *Car*, Figures 4a and 4b show that the performance of both models (continuous and discrete) are comparable. The continuous pose model tends to get confused with nearby views, while the discrete approach reports more errors with opposite viewpoints. The scenario changes when one inspects the results for the *Bus* object category. Figures 4c and 4d show that the performance of the continuous model is slightly worse than the one of the discrete model. Like we detail above, the continuous model tends to concentrate its predictions around the pose annotated bias (*i.e.* the frontal). Observe the bar diagram in 4c, where most of the Rear views are assigned to Frontal views.

We want to conclude this analysis, adding an additional dimension to the discussion: the influence (in the performance) of the evaluation metric used. The problem of *simultaneous* detection and pose estimation has not been associated with either a clear experimental evaluation process or an evaluation metric. Obviously, part of the problem is that discrete and continuous approaches, being of a different nature, have been evaluated in different ways. As a result, multiple evaluation metrics have been proposed, *e.g.* Pose Estimation Average Precision (PEAP) [5], Average Orientation Similarity (AOS) [33] and AVP [9]. We refer the reader to [34], where an extensive analysis of the different evaluation metrics is presented.

We have compared the performance of the AVP and the AOS metrics. Our experiments reveal that the AVP metric tends to favor discrete approaches, while the AOS metric favors the continuous models. For instance, for the category *bus*, Figure 5 shows the precision-recall curves when the different metrics are used. When the AVP metric is used, the discrete approach (Dis-

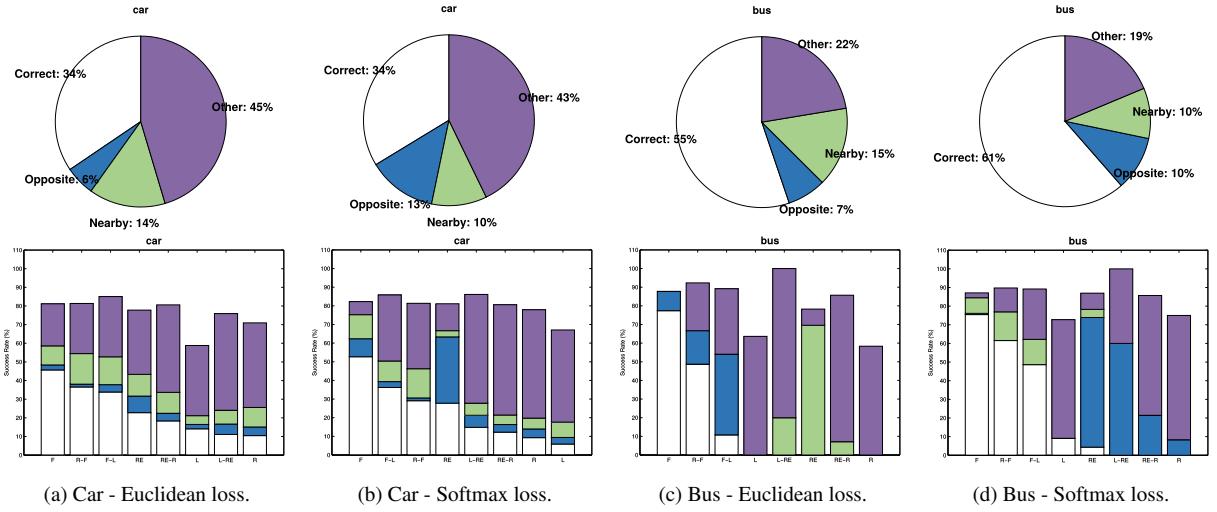


Figure 4: Viewpoint estimation performance detailed analysis. A comparison between continuous (with Euclidean loss) and discrete (with a Softmax loss) models for categories *Car* and *Bus*. (a) and (b) contain the results for the *car* category, while (c) and (d) show the results for the *bus* class. First row include pie charts showing the general performance of the models, where it is reported the percentage of: correct detections, confusions with opposite viewpoints, confusions with nearby poses, and the rest of errors (Other). Second row shows a detailed analysis, of the same type of errors, considering 8 set of viewpoints (F: Frontal, R-F: Right-Frontal, F-L: Frontal-Left, RE: Rear, RE-R: Rear-Right, L: Left, L-RE: Left-Rear and R: Right).

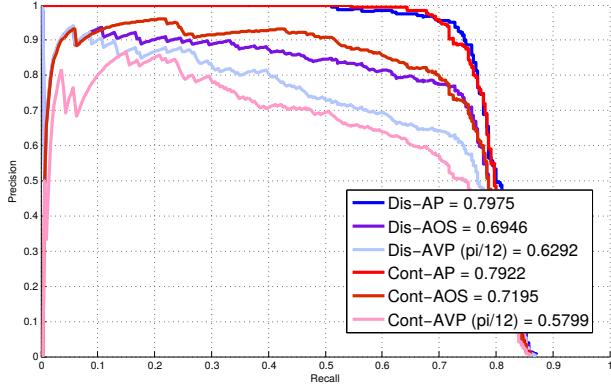


Figure 5: Detection and pose estimation performance for the *bus* category. A comparison based on evaluation metrics AOS and AVP, for both continuous (red tonalities) and discrete (blue tonalities) approaches.

AVP) obtains a higher average precision, compared to the one reported for the continuous model (Cont-APV). On the other hand, when the AOS metric is followed, the average precision is slightly superior for the continuous model, *i.e.* Dis-AOS < Cont-AOS.

In any case, taking into account the observations made in [34], we would like to remark that the AOS metric is not an adequate measurement of the object detection and pose estimation problem. In [34], the authors show that this metric is dominated mainly by the detection performance, masquerading the pose estimation precision. Therefore, for the rest of our study, we choose to use an evaluation procedure based only on the AVP metric.

Overall, based on these results, we conclude that continuous viewpoint estimation models tend to accumulate errors at nearby poses, while discrete pose estimation approaches errors

are more likely to occur in opposite views. Objectively, errors with close poses are not as important as errors associated with opposite poses. We believe that the continuous models could result more attractive for the problem we are dealing with. However, if the amount of training data is not large enough, and is not well balanced in terms of pose annotations, a discrete estimation model, *i.e.* based on a classifier, is the best option. This is the normal situation in all datasets, and also in the PASCAL3D+. Therefore, *for the rest of our study*, we opt for a discrete model.

4.2.3. Independent vs Joint object detection and pose estimation

A quick reading of the scientific literature reveals two main models for tackling the problem of detecting and estimating the pose of object categories. On the one hand we find those who decouple both tasks. The detector is trained and executed separately to locate objects in the images. Subsequently, the pose estimator is responsible for associating a pose to the detected object. On the other hand, we have the models that are trained to solve both tasks together. In this section, we analyze the performance of these two families of works. To do so, we offer a detailed comparison of the proposed architectures in Section 3.2, with existing state-of-the-art models that belong to one family or another.

We need to start this experimental evaluation making the following observations with respect to the three architecture proposed in this paper. Technically, our 3 network designs present a clear evolution in terms of the degree of coupling of the tasks of detection and pose estimation. Our *Single-path* approach clearly belongs to the *joint* family. Note that in this architecture, all the features of the network are shared for both tasks. With the *Specific-path* architecture we advance one step

forward in the decoupling degree. It is a *hybrid* system, where the convolutional layer features are shared, while the FC layers are split into two paths: one for the object localization and one for the pose estimation. Finally we propose the *Specific-network*. Although it should be considered as an architecture belonging to the group of *independent*, we cannot forget that it actually proposes a new paradigm, where both networks, specialized in different tasks, can be trained end-to-end. Note that although the networks learn their characteristics in a decoupled way, the ROIs produced by the network in charge of the location are shared with the network for the estimation of the pose, which somehow conditions their learning. This end-to-end methodology clearly differs from the rest of state-of-the-art *independent* models (*e.g.* [11]).

Table 3 shows the results for all of our architectures in the PASCAL3D+ dataset. Overall, our two *independent* models report a better performance than the *Single-path* architecture. For the specific case of 4 set of views, the best performance is given by the *Specific-path* model, which achieves the best AVP for 6 of 11 categories. For the rest of set of views (8, 16 and 24), the best performance is consistently achieved by our *Specific-network* architecture. The obtained results show that the *independent* approaches perform better than *joint* approaches. In Figure 6, we show some qualitative examples produced by our *Specific-network* architecture.

We now compare our best model, *i.e.* the *Specific-network*, with the state-of-the-art models in Table 4. First of all, our *Specific-network* reports the best object detection results: see last column in Table 4.

Depending on the the number set of views used for the evaluation in the PASCAL3D+ we can identify different winners, even from different families of methods. For instance, *joint* models retrieve the best results, in terms of mAVP, for 4, 8 and 24 views. For 16 views, it is the *independent* model in [11] the one reporting the best performance.

Regarding all the results in Table 4 we can conclude that the *independent* approaches exhibit a better accuracy over most of the *joint* models.

Note that the state-of-the-art for 24 view sets if achieved by the Craft-CNN [15], which uses synthetic CAD models during learning. This is also the case for the RenderCNN [27]. The rest of models, including ours, do not use any extra data in form of CAD models. Note that the *Specific-network* systematically reports a better performance than the RenderCNN, for instance. The Single-Shot approach [35] is the clear winner for 4 and 8 set of views, and the VP&KP [11] wins for 16 set of views. In all these scenarios, our *Specific-network* reports a higher detection accuracy than the winner model. This aspect is relevant, because the metric used tends to favor detectors with a lower localization precision. We refer the reader to the study in [34] for more details. In other words, the more detections that are retrieved by a model, the greater the likelihood that the objects for which pose estimations have to be assigned are objects that, being more difficult to detect, appear occluded or truncated, or that are too small, aspects which naturally complicate a correct estimation of the viewpoint.

Every model comes with its own detector: VP&KP uses the

R-CNN [37], Craft-CNN uses the Fast R-CNN [21], and we follow the Faster R-CNN architecture [19]. How can we evaluate the actual influence of the detector in the viewpoint estimation performance? In order to shed some light on this issue, we have decided to perform an additional experiment. We have taken the code of the VP&KP model provided by the authors. This model defines an *independent* type architecture, where two completely decoupled and different deep networks are used: one for detection, and one for the pose estimation. We start using our *Specific-path* model which has the best detection performance, and we run it over the training images. We then collect these detections on the training data to enrich the ground truth data. Note that we only collect those detections whose overlap with the original ground truth is grater than 70%. This is equivalent to the jittering technique applied in the original paper but taking into account the bounding box distribution of the detector. With this *extended* training data, we proceed to train the original pose estimator in [11]. For the test images, we recover our detections, and apply the described pose estimator on them. We call this pipeline: Improved VP&KP (Imp-VP&KP). Technically, the detector of the original VP&KP has been improved, using the Faster R-CNN now.

As we can see in Table 5, our Improved VP&KP systematically reports better results than the original work. Moreover, in Figure 7 we present a comparison between the Imp-VP&KP and the results of the Craft-CNN [15] for 24 views. We can observe how by simply updating the object detector, the model of [11] can easily get the same performance as the Craft-CNN [15].

4.2.4. The side effect of the pose estimation in the joint system

The systems that address the object detection and pose estimation problems simultaneously, in principle, have multiple benefits, compared with the models that decouple both tasks. They are clearly more efficient, in terms of computational cost. Note that during training, for instance, both task are learned simultaneously. Moreover, for a test image, the localization of the object, and the estimation of its pose is obtained at the same time, not needing to process the images with a complex pipeline consisting of a detector followed by a viewpoint estimator. In a joint system, most of the operations are shared between tasks.

In spite of these advantages, our experiments reveal that there is a trade-off between doing the object localization accurately and casting a precise estimation for the viewpoint. Ideally, a good detector should be invariant to the different poses of an object, *e.g.* it should correctly localize frontal and rear views of cars. This would push the detection models to learn representations that are not adequate to discriminate between the different poses, being this what a good pose estimator should learn.

In Table 6 we report some results that can help us to understand the mentioned trade-off. The first two rows show the results reported by Massa *et al.* [15]. They show a comparison between their joint and independent approaches. Their independent solution clearly obtains a better performance for the object detection than the joint model, but also one can observe how the pose estimation precision, in terms of mAVP, decreases.

Methods	Aero	Bike	Boat	Bus	Car	Chair	Table	MBike	Sofa	Train	Monitor	Avg.
AP Object Detection												
Single-path	78.1	74.3	47.2	79.7	70.2	28.2	53.0	76.0	52.0	79.5	60.8	63.6
Specific-path	78.5	73.1	49.3	79.2	70.3	32.3	52.7	78.0	58.0	77.9	64.6	64.9
Specific-network	77.8	74.2	47.9	78.7	70.3	30.7	52.9	78.1	56.5	77.7	62.7	64.3
AVP 4 Views - Joint Object Detection and Pose Estimation												
Single-path	52.4	41.7	18.6	66.2	45.3	14.2	26.1	44.7	40.4	63.7	52.9	42.4
Specific-path	56.7	54.7	24.1	66.2	50.2	17.3	30.1	55.7	44.0	61.6	60.4	47.4
Specific-network	58.4	57.0	23.2	66.3	53.3	16.9	27.9	60.9	41.5	60.1	52.6	47.1
AVP 8 Views - Joint Object Detection and Pose Estimation												
Single-path	42.9	28.9	11.1	52.7	38.8	10.5	18.1	32.0	28.3	50.2	40.5	32.2
Specific-path	47.2	38.3	16.3	47.2	43.0	12.8	25.5	47.5	33.2	53.4	43.5	37.1
Specific-network	51.3	43.2	14.4	54.6	46.1	13.3	21.8	48.4	33.8	49.4	41.7	38.2
AVP 16 Views - Joint Object Detection and Pose Estimation												
Single-path	22.8	19.5	7.8	54.4	31.8	6.8	14.0	20.5	15.6	42.9	23.6	23.6
Specific-path	33.4	25.9	10.1	51.3	32.7	8.0	20.1	23.8	25.9	38.0	32.5	27.4
Specific-network	36.7	30.5	11.7	57.4	39.7	8.9	21.8	29.6	25.5	38.0	31.9	30.2
AVP 24 Views - Joint Object Detection and Pose Estimation												
Single-path	18.1	15.3	4.4	44.8	27.2	5.2	11.8	13.7	14.0	36.9	16.9	18.9
Specific-path	26.0	18.3	7.7	40.6	29.3	5.2	15.9	18.4	20.3	36.7	24.4	22.1
Specific-network	22.9	21.8	8.8	45.0	33.2	7.0	18.2	20.8	16.9	33.4	21.8	22.7

Table 3: Object detection and pose estimation results in the PASCAL3D+ dataset. Comparison between all our architectures. In gray color we show our *joint* solution, *i.e.* the *Single-path* architecture. The remaining architectures (*Specific-path* and *Specific-network*) can be classified in the group of *independent* approaches.

It is also interesting to observe, in the last rows of Table 6, how this trade-off between object detection and pose estimation performances also affects the model of Poirson *et al.* [35]. We can see that when they try to train their Single-Shot joint model to be more discriminative in terms of poses, *i.e.* increasing the number of sets of views from 4 to 24, the object detection accuracy tends to decrease.

If we now analyze the performance reported by our solutions, from the *Single-path* to the *Specific-network*, we note that the detection performance slightly increases for our *independent* models, but we are able to also report a better performance in terms of pose estimation. We explain this fact with the type of deep architectures we have proposed. Both the *Specific-path* and the *Specific-network* can not be categorized as truly independent models: we do not completely decouple the tasks of object localization and pose estimation. Ours is an exercise or relaxing the amount of shared information between these tasks, which defines a training process able to enforce the networks to learn representations that are adequate for both tasks.

4.3. Results in the ObjectNet3D dataset

In this work, we also perform a detailed experimental evaluation of our models in the large scale dataset for 3D object recognition ObjectNet3D [10]. It consists of 100 categories, 90.127 images and more than 200.000 annotated objects. This dataset has been carefully designed for the evaluation of the problems of object detection, classification, and pose estimation. Similarly to the PASCAL3D+, the object pose annotation is the result of the manual alignment of a 3D CAD model with the target object. Figure 8 shows some examples of this dataset.

Like we describe in Section 4.1, we strictly follow the experimental setup detailed in [10]. Only the training data is used to learn the models. We then report our results using the validation and test sets. For the evaluation metric, Xiang *et al.* [10]

propose a generalization of the AVP. They basically extend the AVP to consider the prediction of the three angles provided in the annotation: azimuth, elevation and in-plane rotation. Technically, the solutions must provide an estimation for these three variables. Then, the corresponding predicted rotation matrix \hat{R} is constructed. The difference between the ground truth pose R , and the prediction encoded in \hat{R} is computed using a geodesic distance as follows:

$$d(R, \hat{R}) = \frac{1}{\sqrt{2}} \|\log(\hat{R}^T R)\|. \quad (8)$$

According to Xiang *et al.* in [10], for the AVP, an estimation is considered to be correct if $d(R, \hat{R}) < \frac{\pi}{6}$.

With respect to the technical implementation of our models, note that they cast a prediction for the three pose angles (azimuth, elevation and in-plane rotation) simultaneously. We repeat the same initialization procedure, using a pre-trained model on the ImageNet dataset. Again, we use the Stochastic Gradient Descent optimizer, with a momentum of 0.9, and the weight decay is set to 0.0005. This time we fix to 1 all the specific learning rates for each layer. The training is performed in an end-to-end fashion following the Faster R-CNN procedure [19].

4.3.1. Discrete vs. Continuous approaches analysis

In our experiments with the PASCAL3D+ dataset, one of the main conclusions obtained has been that the discrete pose estimation models, based on classifiers, give better results than continuous pose estimation models. When the number of training samples is not large enough, and the pose annotations are not well balanced, a discrete estimation model is generally the best option. Now, with the novel ObjectNet3D dataset, which provides more viewpoint annotations for more object categories, we have the opportunity to explore whether we can obtain a better performance for the continuous approaches.



Figure 6: Qualitative results produced by the *Specific-network* in the PASCAL3D+ dataset. In green, we depict the ground truth annotations, while in red we show the results produced by our model. Rectangles correspond to the bounding boxes, while the arrows depict annotated orientations of the objects.

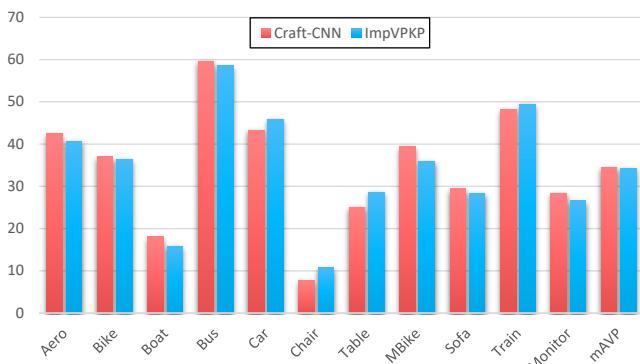


Figure 7: Comparison of Craft-CNN and the Imp-VP&KP experiment for 24 views.

We follow the same procedure described in Section 4.2.2 for our previous Discrete vs. Continuous approaches analysis. We use our *Single-path* model, which is trained for a continuous pose estimation task, solving a regression problem using the Euclidean and Huber losses. When the discrete pose estimation problem is tackled, we simply learn a classifier employing the Softmax loss.

Table 7 reports the obtained results of our *Single-path* architecture, trained on the training set, and evaluated over the validation set. In our experiments, we observe a similar performance among all the models, but this time the continuous pose estimation architectures exhibit a small advantage, like we have previously suggested. Therefore, for the rest of the experiments in this dataset, we use the continuous viewpoint architecture, employing the Huber loss.

4.3.2. Comparison of our architectures

In this section, we propose to analyze the performance of all our architectures, *i.e.* the Single-path, the Specific-path and the Specific-network, in this novel dataset. We only use the training set for learning the models, and the evaluation is carried in the validation set. Figure 9 shows that for this dataset, all our models report a very similar performance. Note how the AP reported for the object localization task is almost identical for the three networks, while for the pose estimation the Specific-path exhibits a slightly superior AVP. In any case, we conclude that for this dataset, there is no clear winner within our models. Therefore, now that the amount of training data in the ObjectNet3D dataset has increased considerably, it seems that there are no major differences between treating the problem of locating objects and estimating their pose jointly or separately.

4.3.3. A comparison with the State-of-the-art

In this section, we provide a comparison with the state-of-the-art models reported by Xiang *et al.* [10]. For the joint 2D detection and continuous 3D pose estimation task, they propose a modification of the Fast R-CNN [21] model, using two different base architectures: the VGG-16 [31] and the AlexNet [38]. Technically, they add a viewpoint regression FC branch just after the FC7 layer. Their network is trained to jointly solve three tasks: classification, bounding box regression and viewpoint regression. The FC layer for viewpoint regression is of size 3×101 , *i.e.*, for each class, it predicts the three angles of azimuth, elevation and in-plane rotation. The smoothed L1 loss is used for viewpoint regression.

Table 8 shows the comparison with the state-of-the-art models, but now in the test set of the ObjectNet3D dataset. On the first two rows of the table, we include the results of the VGG-16 and the AlexNet based models reported in [9]. The last row shows the performance of our *Specific-path* model. First, note

Methods	Aero	Bike	Boat	Bus	Car	Chair	Table	MBike	Sofa	Train	Monitor	mAVP	mAP
AVP 4 Views - Joint Object Detection and Pose Estimation													
VDPM [9]	34.6	41.7	1.5	26.1	20.2	6.8	3.1	30.4	5.1	10.7	34.7	19.5	26.8
DPM-VOC+VP [14]	37.4	43.9	0.3	48.6	36.9	6.1	2.1	31.8	11.8	11.1	32.2	23.8	27.0
Craft-CNN [15]	-	-	-	-	-	-	-	-	-	-	-	-	-
Single-Shot [35]	64.6	62.1	26.8-	70.0	51.4	11.3	40.7	62.7	40.6	65.9	61.3	50.7	61.0
SubCNN [36]	61.4	60.4	21.1	63.0	48.7	23.8	17.4	60.7	47.8	55.9	62.3	47.5	60.7
RenderCNN [27]	50.0	50.5	15.1	57.1	41.8	15.7	18.6	50.8	28.4	46.1	58.2	39.7	56.9
VP&KP [11]	63.1	59.4	20.3	69.8	55.2	25.1	24.3	61.1	43.8	59.4	55.4	49.1	56.9
Specific-network	58.4	57.0	23.2	66.3	53.3	16.9	27.9	60.9	41.5	60.1	52.6	47.1	64.3
AVP 8 Views - Joint Object Detection and Pose Estimation													
VDPM [9]	23.4	36.5	1.0	35.5	23.5	5.8	3.6	25.1	12.5	10.9	27.4	18.7	29.9
DPM-VOC+VP [14]	28.6	40.3	0.2	38.0	36.6	9.4	2.6	32.0	11.0	9.8	28.6	21.5	28.3
Craft-CNN [15]	-	-	-	-	-	-	-	-	-	-	-	-	-
Single-Shot [35]	58.7	56.4	19.9	62.4	42.2	10.6	34.7	58.6	38.8	61.2	49.7	45.1	60.4
SubCNN [36]	48.8	36.3	16.4	39.8	37.2	19.1	13.2	37.0	32.1	44.4	26.9	31.9	60.7
RenderCNN [27]	44.5	41.1	10.1	48.0	36.6	13.7	15.1	39.9	26.8	39.1	46.5	32.9	56.9
VP&KP [11]	57.5	54.8	18.9	59.4	51.5	24.7	20.5	59.5	43.7	53.3	45.6	44.5	56.9
Specific-network	51.3	43.2	14.4	54.6	46.1	13.3	21.8	48.4	33.8	49.4	41.7	38.2	64.3
AVP 16 Views - Joint Object Detection and Pose Estimation													
VDPM [9]	15.4	18.4	0.5	46.9	18.1	6.0	2.2	16.1	10.0	22.1	16.3	15.6	30.0
DPM-VOC+VP [14]	15.9	22.9	0.3	49.0	29.6	6.1	2.3	16.7	7.1	20.2	19.9	17.3	28.3
Craft-CNN [15]	-	-	-	-	-	-	-	-	-	-	-	-	-
Single-Shot [35]	46.1	39.6	13.6	56.0	36.8	6.4	23.5	41.8	27.0	38.8	36.4	33.3	60.0
SubCNN [36]	28.0	23.7	10.0	50.8	31.4	14.3	9.4	23.4	19.5	30.7	27.8	24.5	60.7
RenderCNN [27]	27.5	25.8	6.5	45.8	29.7	8.5	12.0	31.4	17.7	29.7	31.4	24.2	56.9
VP&KP [11]	46.6	42.0	12.7	64.6	42.7	20.8	18.5	38.8	33.5	42.5	32.9	36.0	56.9
Specific-network	36.7	30.5	11.7	57.4	39.7	8.9	21.8	29.6	25.5	38.0	31.9	30.2	64.3
AVP 24 Views - Joint Object Detection and Pose Estimation													
VDPM [9]	8.0	14.3	0.3	39.2	13.7	4.4	3.6	10.1	8.2	20.0	11.2	12.1	29.5
DPM-VOC+VP [14]	9.7	16.7	2.2	42.1	24.6	4.2	2.1	10.5	4.1	20.7	12.9	13.6	27.1
Craft-CNN [15]	42.4	37.0	18.0	59.6	43.3	7.6	25.1	39.3	29.4	48.1	28.4	34.4	59.9
Single-Shot [35]	33.4	29.4	9.2	54.7	35.7	5.5	23.0	30.3	27.6	44.1	34.3	28.8	59.3
SubCNN [36]	20.7	16.4	7.9	34.6	24.6	9.4	7.6	19.9	20.0	32.7	18.2	19.3	60.7
RenderCNN [27]	21.5	22.0	4.1	38.6	25.5	7.4	11.0	24.4	15.0	28.0	19.8	19.8	56.9
VP&KP [11]	37.0	33.4	10.0	54.1	40.0	17.5	19.9	34.3	28.9	43.9	22.7	31.1	56.9
Specific-network	22.9	21.8	8.8	45.0	33.2	7.0	18.2	20.8	16.9	33.4	21.8	22.7	64.3

Table 4: Comparison with the state-of-the-art in the PASCAL3D+ dataset. In gray color we show the *joint* solutions.

that we are able to report a better detection performance than the AlexNet based solution in [9]. Second, although the VGG-16 based architecture of [9] reports the best detection results, if we simultaneously consider the object localization and viewpoint estimation accuracies, *i.e.* using the AVP metric, our model is the clear winner. This fact is particularly relevant, if we consider that the pose estimation is bounded by the detection performance, according to the evaluation metric used. Overall, this implies that our model is more accurate predicting poses.

In a detailed comparison of our solution with the VGG-16 based architecture used in [9], we find the following differences that also help to explain the results obtained. First, while our model is trained fully end-to-end, the approach in [10] consists in training the Region Proposal Network of [19] first, and then using these proposals to fine-tune their model for the object detection and pose estimation tasks. Therefore, their model is mainly trained to optimize the detection performance, which explains why our *Specific-path* reports a slightly lower mAP. Second, there are significant differences in how the pose estimation is performed. In [10], a regressor is trained to directly predict viewpoint values in degrees. We, instead, decompose each angle into two polar coordinates. This decomposition naturally takes into account the cyclic nature of viewpoint angles. This explains why our *Single-path* model reports a better per-

formance for the pose.

We finally show some qualitative results for the ObjectNet3D dataset in Figure 10.

5. Conclusion

In this work, we have presented a complete analysis of the state of the art for the problem of simultaneous object detection and pose estimation. We have designed an experimental validation, using the PASCAL3D+ and ObjectNet3D datasets, where we can evaluate the degree of coupling that exists among the tasks of object localization and viewpoint estimation. For doing so, we have introduced three deep learning architectures, which are able to perform a joint detection and pose estimation, where we gradually decouple these two tasks. With the proposed models we have achieved the state-of-the-art performance in both datasets. We have concluded that decoupling the detection from the viewpoint estimation task have benefits on the overall performance of the models.

Furthermore, we have extended the comparative analysis of all our approaches considering the pose estimation as a discrete or a continuous problem, according to the two families of work in the literature. In our experiments, we have analyzed the main factors that need to be considered during the system design and

Methods	Aero	Bike	Boat	Bus	Car	Chair	Table	MBike	Sofa	Train	Monitor	Avg.
AVP 4 Views - Joint Object Detection and Pose Estimation												
VP&KP [11]	63.1	59.4	20.3	69.8	55.2	25.1	24.3	61.1	43.8	59.4	55.4	49.1
Imp-VP&KP	70.8	66.2	37.9	75.5	61.6	17.7	39.5	68.9	49.6	67.0	62.8	56.1
AVP 8 Views - Joint Object Detection and Pose Estimation												
VP&KP [11]	57.5	54.8	18.9	59.4	51.5	24.7	20.5	59.5	43.7	53.3	45.6	44.5
Imp-VP&KP	63.9	61.4	29.0	63.3	56.2	15.8	32.8	65.3	42.0	60.6	53.6	49.4
AVP 16 Views - Joint Object Detection and Pose Estimation												
VP&KP [11]	46.6	42.0	12.7	64.6	42.7	20.8	18.5	38.8	33.5	42.5	32.9	36.0
Imp-VP&KP	51.2	43.2	20.4	68.9	47.3	17.7	30.1	40.8	36.5	44.7	38.9	39.6
AVP 24 Views - Joint Object Detection and Pose Estimation												
VP&KP [11]	37.0	33.4	10.0	54.1	40.0	17.5	19.9	34.3	28.9	43.9	22.7	31.1
Imp-VP&KP	40.7	36.4	15.8	58.5	45.8	10.7	28.5	35.9	28.3	49.5	26.6	34.3

Table 5: VP&KP [11] vs. Imp-VP&KP experiment.



Figure 8: ObjectNet3D image samples.

Method	mAP	mAVP
Craft-CNN (AlexNet) [15]		
Joint - 24 views	48.6	21.1
Independent - 24 views	51.6	20.5
Ours		
Joint - 24 views (Single-path)	63.6	18.9
Independent - 24 views (Specific-path)	64.9	22.1
Independent - 24 views (Specific-network)	64.3	22.7
Single-Shot [35]		
Joint - 4 views	61.0	50.7
Joint - 8 views	60.4	45.1
Joint - 16 views	60.0	33.3
Joint - 24 views	59.3	28.8

Table 6: Analysis of the trade-off between object detection and pose estimation performance.

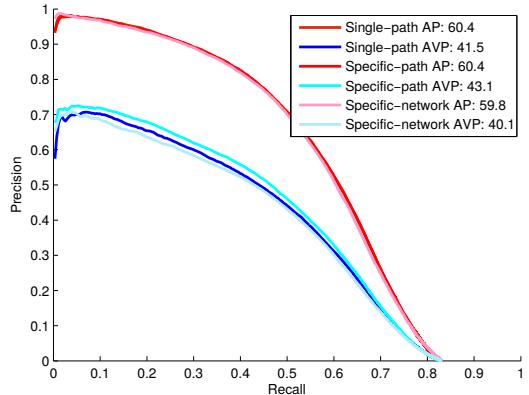


Figure 9: Object detection and pose estimation performance of our Single-path, Specific-path and Specific-network architectures in the ObjectNet3D dataset. Both AP and AVP metrics, with their associated precision-recall curves, are reported.

Losses	mAP	mAVP
Discrete (Eq. 4)	59.7	40.9
Euclidean (Eq. 5)	60.5	41.2
Huber (Eq. 6)	60.4	41.5

Table 7: Loss function analysis for the ObjectNet3D dataset. Object detection and viewpoint estimation performances are reported.

training. Despite the similar performance among the different approaches, we have observed a difference between the discrete and the continuous models. We conclude that the continuous approaches are more sensitive to the pose bias in the annotation than the discrete models, hence requiring bigger datasets.

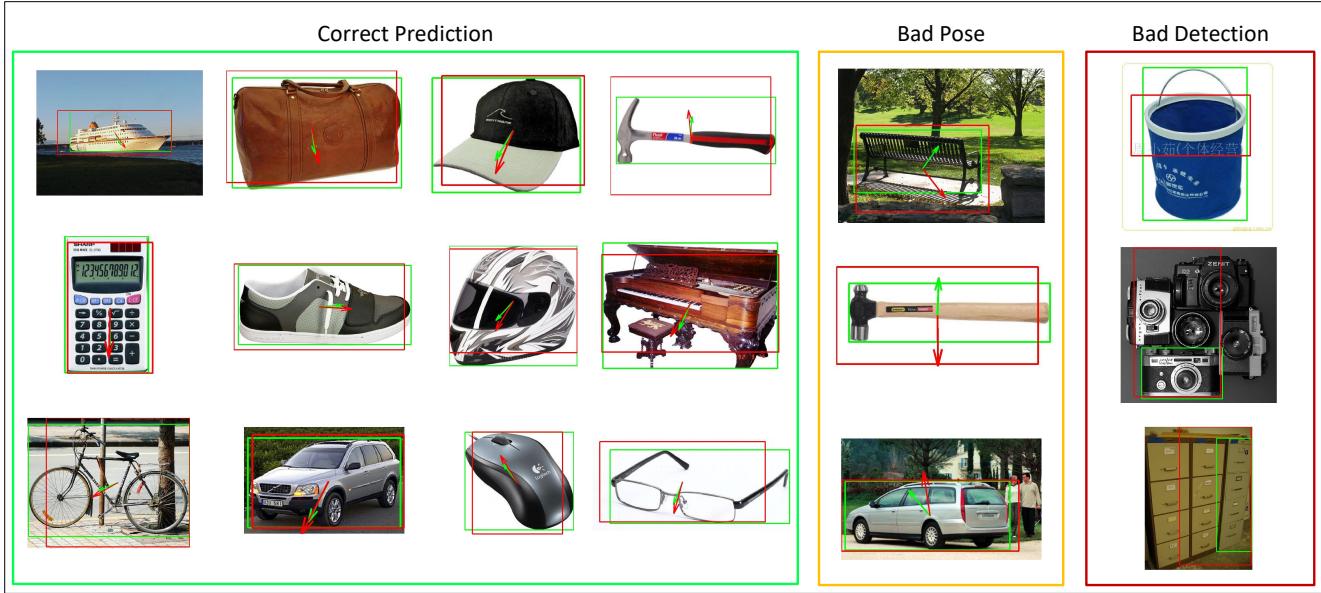


Figure 10: Qualitative results produced by the Specific-path in the ObjectNet3D. In green, we depict the ground truth annotations, while in red we show the results produced by our model. Rectangles correspond to the bounding boxes, while the arrows depict annotated orientations of the objects.

Method	mAP	mAVP
AlexNet [10]	54.2	35.4
VGG-16 [10]	67.5	42.6
Our	64.2	46.7

Table 8: Comparison with state-of-the-art models in the ObjectNet3D dataset.

Acknowledgments

This work is supported by project PREPEATE, with reference TEC2016-80326-R, of the Spanish Ministry of Economy, Industry and Competitiveness. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPU used for this research. Cloud computing resources were kindly provided through a Microsoft Azure for Research Award.

References

References

- [1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, International Journal of Computer Vision 88 (2010) 303–338.
- [2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: CVPR, 2009.
- [3] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, L. Van Gool, Towards multi-view object class detection, in: CVPR, 2006.
- [4] L. Savarese, S. and Fei-Fei, View synthesis for recognizing unseen poses of object classes, in: ECCV, 2008.
- [5] R. J. Lopez-Sastre, T. Tuytelaars, S. Savarese, Deformable part models revisited: A performance evaluation for object category pose estimation, in: ICCV 2011, 1st IEEE Workshop on Challenges and Opportunities in Robot Perception, 2011.
- [6] S. Yingze-Bao, M. Sun, S. Savarese, Toward coherent object detection and scene layout understanding, Image and Vision Computing 29 (2011) 569 – 579.
- [7] M. Ozuysal, V. Lepetit, P. Fua, Pose estimation for category specific multiview object localization, in: CVPR, 2009.
- [8] R. López-Sastre, C. Redondo-Cabrera, P. Gil-Jiménez, S. Maldonado-Bascón, ICARO: Image collection of annotated real-world objects, <http://agamenon.tsc.uah.es/Personales/rlopez/data/icaro>, 2010.
- [9] Y. Xiang, R. Mottaghi, S. Savarese, Beyond PASCAL: A benchmark for 3D object detection in the wild, in: IEEE Winter Conference on Applications of Computer Vision (WACV), 2014.
- [10] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, S. Savarese, ObjectNet3D: A large scale database for 3D object recognition, in: ECCV, 2016.
- [11] S. Tulsiani, J. Malik, Viewpoints and keypoints, in: CVPR, 2015.
- [12] D. Glasner, M. Galun, S. Alpert, R. Basri, G. Shakhnarovich, Viewpoint-aware object detection and continuous pose estimation, Image and Vision Computing 30 (2012) 923–933.
- [13] C. Redondo-Cabrera, R. Lopez-Sastre, Because better detections are still possible: Multi-aspect object detection with boosted hough forest, in: BMVC, 2015.
- [14] B. Pepik, M. Stark, P. Gehler, B. Schiele, Teaching 3D geometry to deformable part models, in: CVPR, 2012.
- [15] F. Massa, R. Marlet, M. Aubry, Crafting a multi-task CNN for viewpoint estimation, in: BMVC, 2016.
- [16] C. Redondo-Cabrera, R. Lopez-Sastre, T. Tuytelaars, All together now: Simultaneous object detection and continuous pose estimation using a hough forest with probabilistic locally enhanced voting, in: BMVC, 2014.
- [17] S. Tulsiani, J. Carreira, J. Malik, Pose induction for novel object categories, in: ICCV, 2015.
- [18] M. Fenzi, L. Leal-Taixé, B. Rosenhahn, J. Ostermann, Class generative models based on feature regression for pose estimation of object categories, in: CVPR, 2013.
- [19] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, in: NIPS, 2015.
- [20] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: CVPR, 2014.
- [21] R. Girshick, Fast R-CNN, in: ICCV, 2015.
- [22] M. Elhoseiny, T. El-Gaaly, A. Bakry, A. M. Elgammal, A comparative analysis and study of multiview cnn models for joint object categorization and pose estimation, in: ICML, 2016.
- [23] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, L. D. Jackel, Handwritten digit recognition with a back-propagation network, in: NIPS, 1990, pp. 396–404.
- [24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft COCO: Common objects in context, in: ECCV, 2014.

- [25] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: CVPR, 2016.
- [26] J. Uijlings, K. van de Sande, T. Gevers, A. Smeulders, Selective search for object recognition, International Journal of Computer Vision 104 (2013) 154–171.
- [27] H. Su, C. R. Qi, Y. Li, L. J. Guibas, Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3d model views, in: ICCV, 2015.
- [28] L. Beyer, A. Hermans, B. Leibe, Biternion nets: Continuous head pose regression from discrete training labels, in: Pattern Recognition, volume 9358 of *Lecture Notes in Computer Science*, Springer, 2015, pp. 157–168.
- [29] A. Singhal, Modern information retrieval: a brief overview, BULLETIN OF THE IEEE COMPUTER SOCIETY TECHNICAL COMMITTEE ON DATA ENGINEERING 24 (2001) 2001.
- [30] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional architecture for fast feature embedding, arXiv preprint arXiv:1408.5093 (2014).
- [31] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556 (2014).
- [32] R. Pascanu, T. Mikolov, Y. Bengio, On the difficulty of training recurrent neural networks, in: ICML, 2013.
- [33] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the KITTI vision benchmark suite, in: CVPR, 2012.
- [34] C. Redondo-Cabrera, R. J. Lopez-Sastre, Y. Xiang, T. Tuytelaars, S. Savarese, Pose estimation errors, the ultimate diagnosis, in: ECCV, 2016.
- [35] P. Poirson, P. Ammirato, C. Fu, W. Liu, J. Kosecka, A. C. Berg, Fast single shot detection and pose estimation, in: 3DV, 2016.
- [36] Y. Xiang, W. Choi, Y. Lin, S. Savarese, Subcategory-aware convolutional neural networks for object proposals and detection, in: IEEE Winter Conference on Applications of Computer Vision, 2017.
- [37] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: CVPR, 2014.
- [38] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: NIPS, 2012.