Viewpoint refinement and estimation with adapted synthetic data

Pau Panareda Busto[a,b], Juergen Gall[b]

[a] *Airbus Group Innovations, TX4-ID*
Munich, Germany
[b] *University of Bonn, Institute of Computer Science III*
Bonn, Germany

Abstract

Estimating the viewpoint of objects in images is an important task for scene understanding. The viewpoint estimation accuracy, however, depends highly on the amount of training data and the quality of the annotation. While humans excel at labelling images with coarse viewpoint annotations like front, back, left or right, the process becomes tedious and the quality of the annotations decreases when finer viewpoint discretisations are required. To solve this problem, we propose a refinement of coarse viewpoint annotations, which are provided by humans, with synthetic data automatically generated from 3D models. To compensate between the difference between synthetic and real images, we introduce a domain adaptation approach that aligns the domain of the synthesized images with the domain of the real images. Experiments show that the proposed approach significantly improves viewpoint estimation on several state-of-the-art datasets.

Keywords: Domain adaptation, Pose estimation, Synthetic data

1. Introduction

In order to estimate the viewpoint of objects in images precisely, an accurate annotation of the training data is required. Humans, however, perform poorly for estimating the viewpoint of an object accurately as illustrated in Figure 1. Instead of annotating real images, synthetic data can be generated

Email addresses: pau.panareda-busto@airbus.com (Pau Panareda Busto), gall@iai.uni-bonn.de (Juergen Gall)

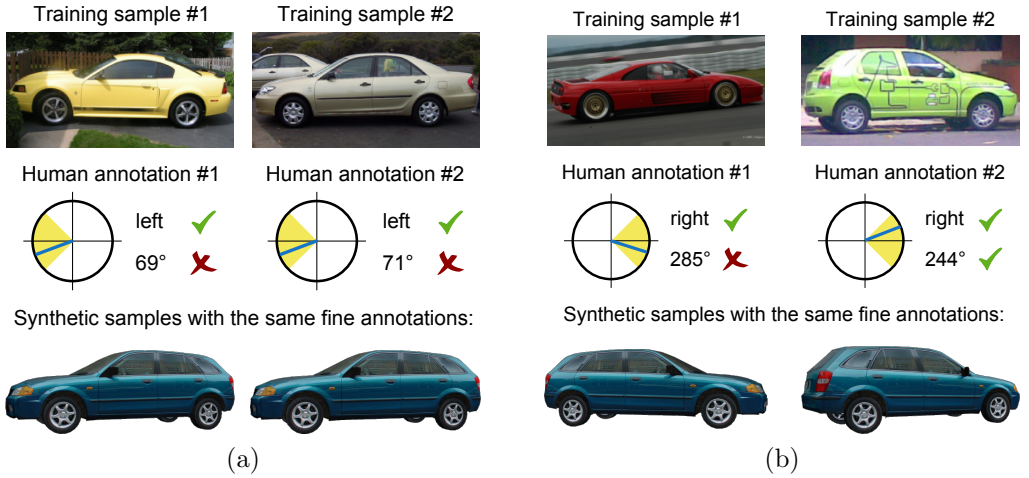


Figure 1: Faulty annotations of fine viewpoints are introduced in human-annotated training datasets. While coarse labels like left or right are correct, the viewpoint annotations in degrees are not precise (a) and sometimes inconsistent (b). Samples and fine annotations are taken from the Pascal3D+ dataset [7].

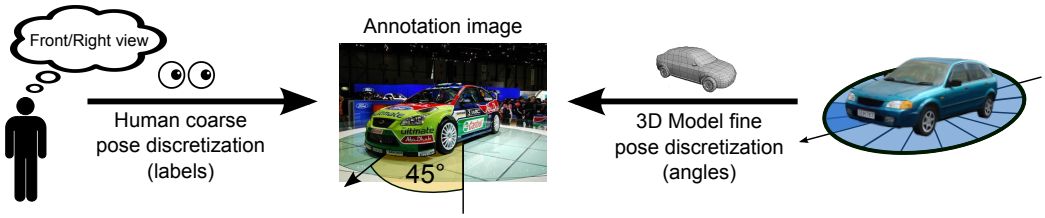
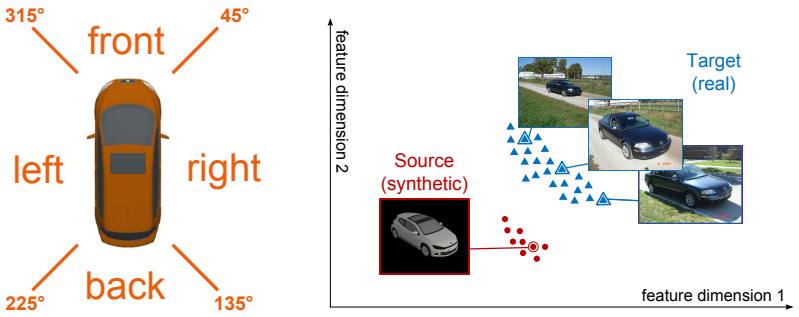


Figure 2: Humans are perfect for annotating coarse viewpoints of objects in real images, but fail to estimate pose accurately at a fine level. 3D graphic models can be used to synthesize data at very accurate fine angles, but it is time-consuming to model all appearance variations present in real images. We therefore propose to leverage the abilities of humans of estimating coarse viewpoints and the pose accuracy of synthetic data.

using 3D models [1, 2, 3, 4, 5, 6]. While synthetic data provides accurate viewpoints, it either lacks the realism of real images or it is very expensive to generate. In particular, collecting a large variation of textured 3D shapes and combining them with coherent background scenes and illumination conditions is time-consuming.

We address this issue by leveraging human annotators and synthetic data, as depicted in Figure 2, to avoid manual annotation by humans of fine viewpoints, which is time-consuming and erroneous, and to avoid the synthesis



(a) Coarse views (b) Illustration of features space

Figure 3: (a) The four views available for real images. (b) Synthetic and real images with the same annotated viewpoint lie in different domains within the feature space.

of a realistic dataset that captures the variations of real images, which is time and memory consuming. To this end, we ask humans to annotate only four coarse views, sketched in Figure 3a, and introduce an approach that refines the labels using synthetic data. Since synthetic data and real images belong to different domains as illustrated in Figure 3b, a domain adaptation approach is used for the refinement. General domain adaptation approaches like [8, 9], however, are not sufficient for label refinement since they fail to distinguish viewpoint rotations by 180 degrees. We therefore present a task-specific approach that takes advantage of the coarse labels of the real training samples.

A preliminary version of this work appeared in [10]. While the approach in [10] was limited to cars, we extend the method to other categories and provide a thorough experimental evaluation. We also evaluate our approach with state-of-the-art features extracted from convolutional neural networks (CNN) [11] and study the effect of truncated and occluded object instances. The evaluation, which is performed on five datasets for viewpoint estimation, reveals that our approach outperforms state-of-the-art domain adaptation methods.

2. Related Work

2.1. Viewpoint estimation

Methods for viewpoint estimation are often based on popular object class detectors [12, 13, 14, 15] and learn a discrete set of pose classifiers. In [16, 17, 18, 19], annotations from 2D images are enhanced with 3D metadata

to formulate 3D geometric models. On the contrary, [20] learns a mixture-of-templates that inherently captures the characteristics of projected views and [21] refines the hypothesis of 16 viewpoint detectors from 2D images with additional view specific Naïve Bayes classifiers. More recently, CNNs for object classification [11] have been retrained using 2D pose annotations in order to provide viewpoint probabilities as output channels coupled with the object class probability [22, 23]. In the study pursued in [24], simple frameworks that extract features from 2D bounding boxes with powerful encoders provided the same or even better viewpoint accuracies than state-of-the-art methods based on complex 3D models.

In contrast to classification approaches, regression approaches [25, 26] do not require a discretisation of the viewpoints. In [27], the viewpoint regression is integrated into a joint discriminative continuous parametrised model. However, [23] showed how discretised models with a high number of discrete poses, i.e., from 16 fine viewpoints onwards, started to get better accuracies compared to regression methods. For further studies, we refer to [28] for joint object detection with pose estimation through regression approaches and [29] for an analysis with regard to deep learning approaches.

2.2. Synthetic Data

The use of synthetic images from rendered models and scenes as training data started to gain attention in the context of pedestrian detection. While [6] only uses synthetic data generated from a popular game engine, [5] combines real with synthetic data from highly accurate 3D reconstructed humans. Both methods, however, do not consider the 3D information and collect only 2D images with automatically annotated bounding boxes.

Previously, the 3D spatial information of graphics models was already addressed in several works to estimate the viewpoint of object instances, as well as its localisation [1, 16, 18, 30, 31, 32]. These algorithms are computationally expensive, since the object geometry is used to learn the spatial 3D relations of parts or features. In contrast to these works, we use the rendered models to synthesize training images with accurate viewpoint annotations. Instead of rendering 3D data, synthetic data can also be generated by defining a parametric model for synthesizing geometric shapes from a particular object class, used in both recognition and reconstruction, as proposed by [19].

Recently, [33, 34] tested the impact of synthetic data in CNNs by training millions of synthesized images from 3D models. Thus, the main challenge becomes the generation of extremely large amounts of data with as much

intra-class variation as possible, e.g., viewpoint and shape, to avoid overfitting.

3D models have also been used to annotate datasets [7, 35] by manually superposing them on top of 2D object instances. While the 3D models support humans and improve the accuracy of the annotation, the annotation process with 3D models is very slow and still prone to annotation errors.

2.3. Domain Adaptation

Domain adaptation addresses the problem when the training and test data are at least partially from different domains. To this end, either a transformation of the domains is estimated before the training of a classifier [8, 36, 37] or the so-called source domain is used to regularize the learning of a classifier on the target domain [38, 39]. A popular choice in this context are support vector machines [9, 40, 41, 42, 43]. The approaches that estimate the transformations without a classifier like the geodesic flow kernel [8] learn mappings from the source and target domain into a joint, low-dimensional space. This can be done in an unsupervised manner where the target domain is unlabelled, or in a supervised or semi-supervised setting where the data from the target domain contains a few labelled samples. While these methods assume that the source and target domains are known, [44] minimise the distance between latent domains, rearranging clusters of the annotated classes based on feature similarities. In contrast to these works, we use domain adaptation in a weakly supervised setting where only coarse labels are available for the training images of the target domain.

During the last years, domain adaptation methods focused on the optimization process for the domain alignment, where additional constraints for the optimization have been proposed [9, 41, 42, 43, 45]. For instance, orthogonality constraints have been suggested for the transformation matrix [37, 39], as well as relaxation techniques to make the optimization solvable [8, 43]. Other approaches, on the contrary, excel by its speed and simplicity. [46] computes a subspace alignment between domains in closed form and [47] aligns the covariance matrix of the source data with whitening and recolouring, which is applied to synthetic data in [2].

Deep convolutional networks also had a dramatic impact in the field of domain adaptation. DeCAF [48] demonstrated how features extracted from CNNs outperform by a large margin classification accuracies of commonly used features after adaptation, e.g., Bag of Words or HOG features. While the standard adaptation techniques estimate the alignment after extracting

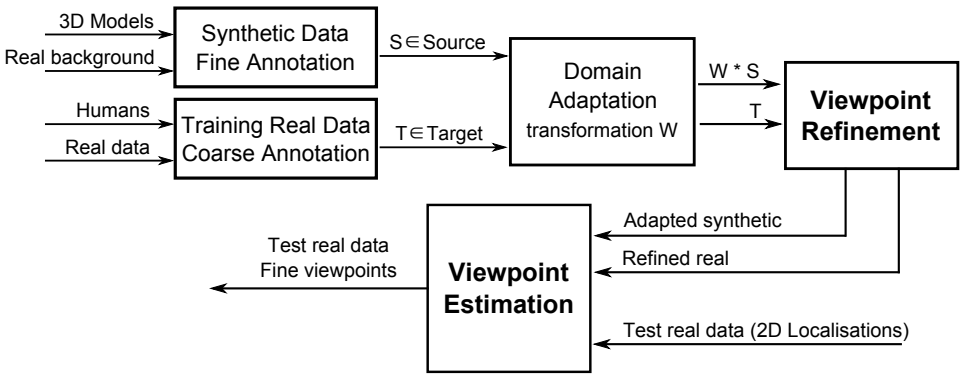


Figure 4: Proposed pipeline for viewpoint refinement and estimation of real data.

the features, several papers opted for training deep networks by combining source and target datasets with specific architectures and loss functions that jointly minimised the classification regressor and the distance between domains [49, 50, 51, 52].

3. Adapted synthetic data for viewpoint refinement and estimation

In this section we describe the automatic process of refining coarse annotations of real data into fine viewpoints using adapted synthetic data. As depicted in Figure 4, we initially request humans to coarsely annotate viewpoints of given 2D bounding boxes. Additionally, we also generate synthetic data with fine viewpoint annotations. This process is discussed in Section 3.1. Then, we adapt the synthetic data towards the real data, explained in Section 3.2, and assign fine viewpoints to the real data, further detailed in Section 3.3. We evaluate our approach for viewpoint refinement and viewpoint estimation. For viewpoint refinement, the coarse viewpoint is given and the goal is to estimate the fine viewpoint. For viewpoint estimation, the refined real and adapted synthetic data is used to train an estimator for fine-grained viewpoint estimation. The estimator is then evaluated on unseen test instances.

3.1. Generation of synthetic data from 3D models

In order to produce thousands of synthetic images, we first download free available 3D graphics models from the Internet. We then render the models, centred in the screen coordinate system, with 8 different light sources evenly spread around the object. Based on a Phong reflection model [53],

we emphasise the usage of diffuse lighting to highlight shape variations and deformations, reducing the impact of ambient illuminations and specular reflections. The resulting rendered virtual classes used in the experiments are shown in Figure 5a. The scene is completed with a real background image taken from [54] placed behind the rendered object.

Finally, the generation process reduces to a parametrised camera displacement with azimuth θ, elevation ϕ and object distance r. Although this configuration allows to move along the whole view-sphere, we simplify the fine viewpoint annotations to the Y-axis rotation, being the azimuth angle the most dominant factor to recognise viewpoint differences in feature space, as well as the most relevant plane in viewpoint estimation tasks [27]. Figure 5b shows some examples of synthetic images. While the process of synthesizing images does not require much effort, it does not generate realistic images since the unknown 3D geometry and light conditions of the background are not taken into account.

3.2. Domain Adaptation of synthetic data

Since synthetic data and real images belong to different domains, as illustrated in Figure 3b, we adapt the domain of the synthetic data to the real data. Our approach clusters the source (synthetic) and target (real) domains, and establishes correspondences between the clusters. The correspondences are then used to learn a mapping from the source domain to the target domain. The viewpoint annotations of the real images are then refined with viewpoint classifiers trained on the transformed synthetic data.

The learning of the mapping from the source to the target domain is discussed in Section 3.2.1 and the establishment of correspondences between clusters of both domains is discussed in Section 3.2.2.

3.2.1. Alignment from synthetic to real domain

To map the source data to the target domain, we have to learn a mapping from $\mathcal{S} \in \mathbb{R}^D$ to $\mathcal{T} \in \mathbb{R}^D$, where D denotes the dimensionality of the features. For label refinement, the dimensionality of the source and the target domain is the same. We consider a linear transformation, which is represented by a matrix $W \in \mathbb{R}^{D \times D}$, i.e., , $t = Ws$.

Let $S = \{s_1, ..., s_M\}$ and $T = \{t_1, ..., t_N\}$ denote the training samples of the source and target domains, respectively. M and N are the total amount of samples of each domain and we can assume that $M \geq N$, since we can always generate more synthetic data than annotated real images. We first

(a) 3D models for the 11 object classes used for the Pascal3D+ dataset [7].

(b) Synthesised images with different azimuth, elevation and distance configurations.

Figure 5: 3D graphics models for different object classes are rendered in front of real background images from [54] in order to automatically generate thousands of synthetic images with different accurate viewpoint annotations.

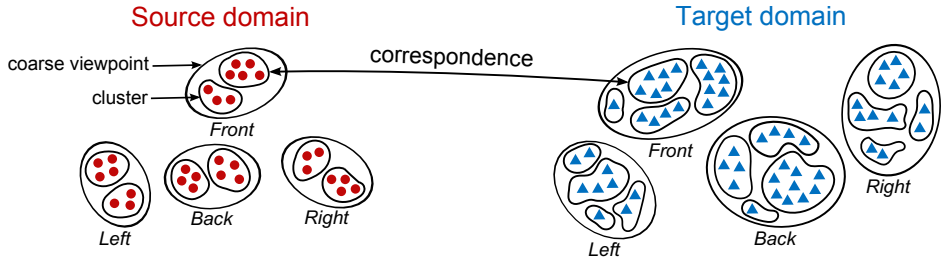


Figure 6: Each cluster in the target domain is assigned to a source cluster that belongs to the same coarse viewpoint. In this example, for an 8-view refinement: $V_i = 2$ and $K_i = 4$.

assume that for a subset of the target elements t_k we have already established a corresponding element in the source domain. The establishment of the correspondences $C = \{c_1, ..., c_K\}$ with (s_{c_k}, t_k) and $K \leq N$ will be explained in Section 3.2.2.

Given the correspondences, W can be learned by minimizing the objective

$$f(W) = \frac{1}{2} \sum_{k=1}^{K} ||W s_{c_k} - t_k||_2^2, \tag{1}$$

which can be expressed in matrix form:

$$f(W) = \frac{1}{2} ||W P_S - P_T||_F^2. \tag{2}$$

The matrices P_S and $P_T \in \mathbb{R}^{DxK}$ represent all assignments between source and target elements, where the columns denote the actual correspondences. We optimise the objective by non-linear optimisation. To this end, the derivatives of (2) are calculated by

$$\frac{\partial f(W)}{\partial W} = W(P_S P_S^T) - P_T P_S^T. \tag{3}$$

In our implementation, we use the local gradient-based optimization method of moving asymptotes [55], which is part of the NLOPT package [56].

3.2.2. Source-Target Correspondences

In order to minimize (1), we first have to establish correspondences between the source and the target data. To this end, we cluster the data in both domains. For the synthetic data, we use the known fine-grained poses

where each pose can be associated with one of the four coarse viewpoints $i = \{\text{front}, \text{back}, \text{left}, \text{right}\}$, i.e., $V = \sum_i V_i$, where V is the number of fine viewpoints for refinement. For the target domain, we only have the coarse viewpoints and therefore cluster the N_i training samples of one coarse viewpoint further by K-Means, where the number of clusters for each coarse viewpoint is given by K_i, i.e., $K = \sum_i K_i$. and $V_i \leq K_i \leq N_i$. If $K_i = N_i$ clustering is not performed since each target instance is considered as one cluster. If $K_i = V_i$, the number of clusters is equal to the number of fine viewpoints. For the clustering, we represent each image by a HOG or CNN feature vector and append the aspect ratio of the bounding box surrounding the object.

As illustrated in Figure 6, we establish correspondences between the clusters in the source and target domains, separately for each coarse viewpoint. To this end, we represent each cluster by its centroid. The sets of centroids are denoted by $\hat{S}^i = \{\hat{s}^i_1, ..., \hat{s}^i_{V_i}\}$ and $\hat{T}^i = \{\hat{t}^i_1, ..., \hat{t}^i_{K_i}\}$. The correspondences are then established by solving a bipartite matching problem:

$$\begin{aligned} &\underset{e_{vk}}{\operatorname{argmin}} \sum_{v=1}^{V_i} \sum_{k=1}^{K_i} e_{vk} \left\| \hat{s}^i_v - \hat{t}^i_k \right\|_2^2 \\ &\text{subject to} \quad \sum_v e_{vk} = 1 \quad \forall k \ , \quad \sum_k e_{vk} = a_v \quad \forall v \quad \text{and} \quad e_{vk} \in \{0, 1\} \quad \forall v, k \ . \end{aligned} \tag{4}$$

It assigns to each cluster in the target domain a unique cluster in the source domain. Since there can be more clusters in the target domain than in the source domain, each source is associated to $a_v = K_i/V_i$ target clusters. If K_i is not a multiple of V_i, i.e., $aV_i < K_i < (a+1)V_i$, we set $a_v = a + 1$ for the first $K_i - aV_i$ source clusters and $a_v = a$ otherwise. We use the Hungarian algorithm [57] to solve the problem and for any cluster pair with $e_{vk} = 1$, we obtain a correspondence c. The correspondences from all coarse views are then used to estimate the transformation W in (1).

3.3. Viewpoint Refinement and Estimation

The last step in our pipeline is the viewpoint refinement of the real training images. This is seen as a classification problem where we train on the transformed synthetic samples a linear SVM for each of the fine viewpoints $v = \{1, ..., V\}$. Then, we apply the linear SVMs corresponding to the coarse

viewpoint i of the real image and assign the fine pose with the highest scoring function:

$$f(x,i) = \underset{v=\{1,\ldots,V_i\}}{\operatorname{argmax}} \; w_v^T x + b_v, \tag{5}$$

where w_v and b_v are the weights and bias of the linear SVM for the fine viewpoint v.

For pose estimation on real test images, we also use linear SVMs in a one-vs-all classification procedure. For each fine viewpoint, we train a linear SVM using the real training images with refined pose labels and the synthetic training images, which have been transformed by domain adaptation, together.

4. Experiments

We evaluate our algorithm on two car and three multi-object datasets with fine annotated poses. From the former group, the *EPFL* [21] dataset contains sequences of 20 cars as they rotate by 360°, where one image is taken every 3-4°. These fine-grained poses allow us to test the refinement at higher levels of viewpoint discretisation. We take the first 10 car sequences as training (1179 images) and the last 10 as test data (1120 images). These cars are in a fixed location. Therefore, we also evaluate our method on the more realistic *KITTI* [54] benchmark, where images are recorded while driving along streets and roads. Due to the lack of bounding box annotations in the test data, we perform a 2-fold cross validation on the fully visible cars of the training set, containing 7481 images with 17463 cars, 7811 of those which are non-occluded. From the latter, the *3D Obj. Categorization* [58] dataset provides 10 image sets of cars and bikes in 8 different angles (every 45 degrees), permitting a refinement from 4 to 8 fine viewpoints. There are 2 elevations and 3 distances for each view, giving 48 images per object. We take 7 sets for training and 3 for testing. We also evaluate the method on the *Pascal3D+* [7], which contains occlusions and truncated object instances of several classes. The main part of this dataset enriches the PASCAL VOC 2012 [59] categories with 3D annotations for 11 rigid objects[1]: *aeroplane*, *bike*, *boat*, *bus*, *car*, *chair*, *dining table*, *motorbike*, *sofa*, *train* and *tv monitor*. The dataset has been further increased by images from the ImageNet

[1]In the standard protocol of [7], the class "bottle" is discarded due to its lack of viewpoint reference.

dataset [60], which are also augmented with 3D annotations for the same rigid objects. Therefore, we opt for evaluating both subsets separately, denoted in our experiments as *Pascal3D* and *ImageNet3D*, respectively, using their validation sets as test data.

The setup for the experiments is as follows. At first, we automatically generate synthetic data of textured 3D models for each object class. Following the evaluation protocol of [10], we take 15 car models for all experiments that only involve cars. For the multi-object evaluations, we make use of 10 models per class, thus decreasing the number of cars in order to keep an even quantity among all classes. The attached background images, randomly taken from the KITTI dataset [54], point towards the car's driving direction, allowing for synthetic vehicle placements, e.g., bike, bus, car and motorbike classes, in the centre of the image.

The synthetic images are obtained in two configurations to evaluate the impact of different viewpoint granularities. For refinements $V \leq 36$, we rotate the θ angle of the camera every 10 degrees in clockwise order, for a total of 36 fine viewpoints. For finer refinements, i.e., $V > 36$, we synthesise every 1 degree ending up with 360 fine viewpoints. Besides, we capture 4 levels of elevation, $\phi = \{0, 15, 30, 45\}$, and 3 distances, $r = \{1.75, 2.25, 2.75\}$ in virtual world coordinates. Due to the special case of aeroplane instances in the air, we consider for this specific class views below the horizontal plane assigning $\phi = \{\text{-}30, \text{-}15, 0, 15, 30\}$. The pose labels are then quantised to their closest angle of the V fine poses. The first viewpoint $v = 1$ lies at $\theta = 0$ in all quantisation levels. Some examples of the synthesised data are illustrated in Figure 5b.

Our first evaluation, in Section 4.1, measures the accuracy of our viewpoint refinement, extracting the bounding boxes of the real training images and converting the given viewpoints into the four coarse views, that is: $front = (315°, ..., 45°)$, $right = [45°, ..., 135°]$, $back = (135°, ..., 225°)$ and $left = [225°, ..., 315°]$. Then, in Section 4.2, we evaluate the viewpoint estimation of the real test images having as training the adapted synthetic data and the refined real data. We use the given bounding boxes if the images are not already cropped. Neither coarse nor fine viewpoints are used for the test images.

Several widely used feature descriptors are evaluated to measure the performance of the method in different feature spaces. For HOG features [13], we rescale the bounding boxes to 128×128 pixels and extract descriptors with 8 bins (31 channels/bin), as in [10]. Additionally, we extend it with

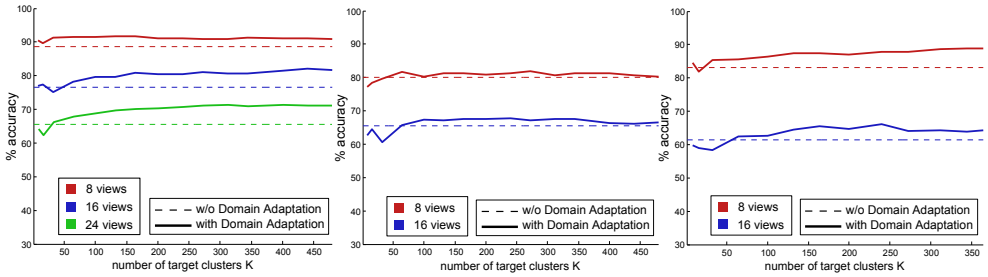
state-of-the-art CNN features from the AlexNet model [11], extracting the feature maps from the last fully connected layer (CNN-fc7) and the last convolutional layer (CNN-pool5), with 4096 and 9216 dimensions from the standard 227×227 patch input, respectively. The reported accuracy values of both layers come from re-trained models using the 36-viewpoint synthetic dataset and modifying the output layer with 36 classification channels. In the experiments with datasets that do not contain occlusions [58, 21, 54], the annotated instances are rescaled preserving the aspect ratio. For the evaluations that include occluded objects [7], the annotations are warped in order to reduce the influence of overlapping objects and truncated borders.

4.1. Viewpoint Refinement

We first evaluate the accuracy of our approach for pose refinement on the real training images. To this end, we use the coarse labels of the real training images and refine the viewpoints as described in Section 3.3. We then evaluate the accuracy of the refined labels on the real training images in conjunction with the transformed synthetic samples after the domain adaptation process. For the initial parameter evaluation of our technique, we stick to extracted HOG features of car models. Then, we test the performance of our viewpoint refinement for all descriptors and classes.

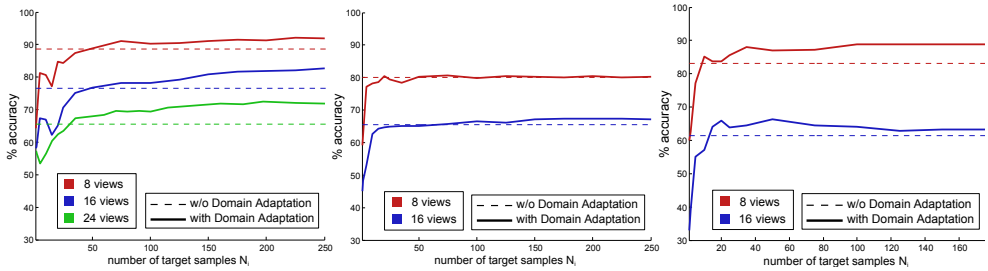
Impact of number of target clusters. As described in Section 3.2.2, we cluster each coarse view by K-Means. We therefore evaluate the impact of the number of target clusters K on the viewpoint refinement. The results for the different datasets and V refined viewpoints used for evaluation are shown in Figure 7. As baseline, we use linear SVMs trained on the synthetic data without domain adaptation. The accuracy tends to stabilize when the number of clusters is sufficiently large. The finer the viewpoints are the more clusters are also needed.

Impact of number of target samples. Although annotating real images by coarse viewpoints is easy to do, it also takes time. We therefore evaluate the impact of the number of coarsely labelled target samples N. To avoid any clustering artefacts, we set $K_i = N_i$, i.e., each target sample itself is a cluster. We also keep the numbers of the real images N_i for each of the four viewpoints equal while increasing N. The results in Figure 8 show that already 50-75 annotated samples per coarse view give a boost in performance compared to the baseline. This means that very little time is actually required for the annotation task.



(a) EPFL dataset (b) KITTI dataset (c) Pascal3D dataset

Figure 7: Impact of the number of target clusters K for viewpoint refinement.



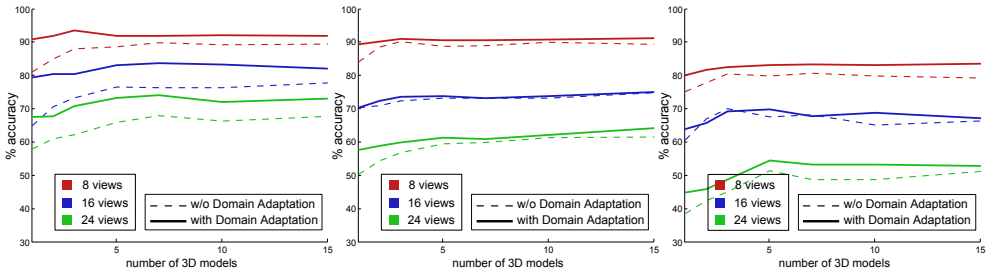
(a) EPFL dataset (b) KITTI dataset (c) Pascal3D dataset

Figure 8: Impact of the number of target samples N_i per coarse view for the refinement.

Impact of number of 3D models. We also evaluate the impact of the amount of 3D models used to generate synthetic data. Figure 9 shows how the accuracy tends to stabilise with already 7-8 models. Generally, the behaviour is comparable when using 10 or 15 models in the experiments.

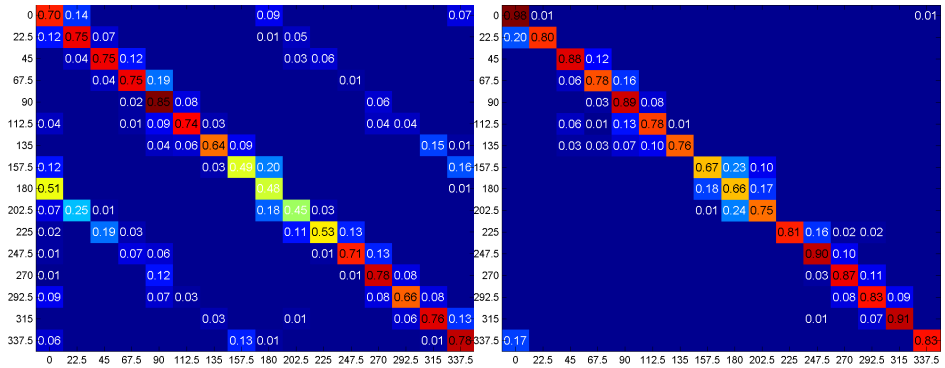
Weak supervision. If the target samples are not annotated by the four coarse views, we can still perform unsupervised domain adaptation. In this case, we observe a substantial amount of wrong viewpoint estimates by 180 degrees as shown by the confusion matrix in Figure 10a. In contrast, we resolve these errors by using the coarse viewpoints of the real images as weak supervision as shown in Figure 10b. This shows that using coarse annotations of real images, which are inexpensive to annotate, significantly increases the viewpoint refinement accuracy.

Accuracy of the viewpoint refinement. We finally compare the refinement accuracy of our method with different popular domain adaptation tech-



(a) EPFL dataset (b) ImageNet3D dataset (c) Pascal3D dataset

Figure 9: Impact of the number of 3D car models for viewpoint refinement.



(a) Unlabelled target samples (b) 4 viewpoint labels in target samples

Figure 10: Confusion matrix for EPFL dataset in a 16-viewpoint refinement. (a) Without supervision rotations by 180 degrees are sometimes confused. (b) When weak supervision from the four coarse viewpoint labels is used, these confusions are resolved.

niques [8, 46, 47]. For the refinement after domain adaptation, we use linear SVMs as described in Section 3.3. As baseline, we use the linear SVMs trained on the synthetic data without domain adaptation. The geodesic flow kernel (GFK) [8] is an unsupervised domain adaptation method that maps both domains to a common subspace in a Grassmannian manifold. The approach can also be used for supervised domain adaptation, but it did not improve the results in our experiments. We therefore report the results for the unsupervised approach for each coarse viewpoint. The same applies to the sub-space alignment technique (SA) [46], that maps both domains to a common subspace using the d largest eigenvectors. In both cases, the number of chosen sub-dimensions d is kept as large as possible to avoid a significant

loss in accuracy. Lastly, we also test the current state-of-the-art adaptation method named CORAL [47]. Without any dimensionality reduction, it decorrelates the source samples by whitening and re-colours them by the covariance matrix of the target data.

For our method, we report the refinement accuracy for four different clustering settings. For the first three, we set V equal to the number of views for fine-grained viewpoint estimation as in the previous experiments. We report numbers for $K = V$, $K = 100$ and $K = N$. For the first two settings, we report the mean accuracy and its standard deviation over 10 runs since K-Means depends on the random initialization. In the last setting, each target sample is a cluster.

We first report the results only for the fully visible object exemplars and compare HOG, CNN-fc7 and CNN-pool5 features in Table 1. The accuracies of both kinds of CNN features outperform the results of the HOG features, especially when using finer refinements. While CNN-pool5 achieves the best overall result, it is outperformed by CNN-fc7 for very fine viewpoints $V \geq 72$. While $K = N$ performs best in almost all cases, $K = 100$ and $K = V$ achieves the highest accuracy in a few cases. Overall, $K = N$ with CNN-pool5 features performs best on all three datasets.

We also evaluated the accuracy when V is also set to the number of synthetic samples M, i.e., each synthetic image is a cluster. In this case, the accuracy drops significantly for all datasets and feature descriptors. This shows that the synthetic data needs to be quantized according to the fine-grained views.

Table 1 also compares our approach to other domain adaptation methods [8, 46, 47]. In nearly all setting and feature combinations, our method outperforms the generic domain adaptation methods. On KITTI with 16 views and CNN-pool5 features, our approach achieves an accuracy of 70% compared to 50% obtained by [8, 46, 47].

In contrast to the datasets [58, 21, 54], the datasets Pascal3D and ImageNet3D contain many occluded and truncated objects. The results for these two datasets are reported in Figure 2. We report the accuracies for both CNN features using $K = N$ and compare it to the baseline without domain adaptation. Except for the 8 view refinement on ImageNet3D, our approach outperforms the baseline by around 4%. The CNN-pool5 features achieve the highest accuracy as it was previously observed on the other two datasets.

	3DObjCat [58]		EPFL [21]							KITTI [54]	
	HOG										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	97.62	98.81	88.65	76.63	66.11	58.59	33.95	14.92	9.17	80.04	65.61
GFK [8]	97.62	98.81	88.92	79.64	66.38	57.67	34.20	15.31	8.60	80.07	65.42
SA [46]	97.62	98.81	88.88	77.66	66.11	58.59	33.87	14.92	9.08	80.04	65.61
CORAL [47]	94.94	98.21	91.13	76.98	67.42	59.51	35.87	15.77	9.21	79.80	61.18
V=views, K=V	98.57 (0.53)	98.21 (0.51)	90.41 (1.65)	77.32 (2.01)	66.67 (2.11)	58.75 (1.89)	33.29 (1.55)	17.08 (1.33)	8.87 (1.29)	77.15 (1.30)	64.56 (1.67)
V=views, K=100	99.11 (0.36)	**99.80 (0.09)**	91.57 (0.47)	79.62 (0.65)	70.16 (0.70)	59.46 (1.01)	33.99 (0.80)	17.55 (0.46)	9.66 (0.31)	**80.32 (1.41)**	**67.37 (1.47)**
V=views, K=N	**99.70**	99.40	**92.00**	**81.82**	**71.85**	**64.99**	**39.59**	**17.69**	**9.85**	78.78	67.05
V=M, K=N	92.86	98.02	85.69	76.69	67.02	63.33	31.55	12.77	6.20	75.70	62.92
	AlexNet CNN-fc7										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	93.75	99.21	89.87	77.98	71.50	64.81	43.57	20.01	10.16	**74.60**	54.57
GFK [8]	94.35	98.81	89.90	76.63	70.56	63.21	43.27	19.29	9.74	69.61	53.75
SA [46]	93.75	99.21	89.87	77.98	71.40	64.81	43.70	20.39	10.34	**74.60**	54.57
CORAL [47]	90.18	99.40	89.43	72.01	62.43	51.77	29.05	10.55	6.12	68.67	42.43
V=views, K=V	95.54 (0.37)	99.40 (0.20)	79.82 (2.20)	67.77 (2.02)	73.89 (1.75)	61.68 (1.60)	38.88 (1.37)	**23.57 (1.39)**	**12.47 (0.98)**	59.70 (1.97)	43.95 (2.02)
V=views, K=100	95.24 (0.40)	99.21 (0.22)	**92.01 (0.70)**	79.53 (0.91)	75.32 (0.93)	65.35 (1.08)	38.85 (0.96)	19.80 (0.81)	11.10 (0.66)	68.93 (2.10)	**57.13 (2.42)**
V=views, K=N	**97.02**	**99.80**	87.92	**83.61**	**77.34**	**67.44**	**46.10**	19.71	10.75	70.91	57.09
V=M, K=N	93.75	97.22	85.81	74.10	65.18	60.85	36.50	17.51	7.47	69.28	47.11
	AlexNet CNN-pool5										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	98.21	98.81	93.26	76.87	72.52	62.14	38.11	18.59	7.70	75.63	49.04
GFK [8]	97.62	98.81	93.07	76.96	72.05	62.27	36.57	18.16	8.28	74.69	49.24
SA [46]	98.21	98.81	93.26	76.87	72.52	62.14	37.91	18.53	7.70	75.63	49.10
CORAL [47]	87.20	96.63	77.33	60.65	53.15	40.52	22.01	8.59	4.10	74.86	49.74
V=views, K=V	97.86 (0.67)	99.60 (0.10)	79.60 (1.37)	72.58 (1.66)	67.24 (1.25)	51.93 (1.31)	38.62 (1.00)	19.19 (0.99)	8.17 (1.01)	69.14 (1.09)	52.36 (1.38)
V=views, K=100	99.10 (0.12)	97.62 (0.31)	93.59 (0.61)	80.29 (0.72)	75.40 (0.77)	63.39 (0.55)	37.26 (0.38)	18.61 (0.27)	8.01 (0.17)	79.08 (1.12)	68.59 (1.32)
V=views, K=N	**100.0**	**100.0**	**95.65**	**84.77**	**77.63**	**69.05**	**43.56**	**19.68**	**8.76**	**80.83**	**70.07**
V=M, K=N	96.13	99.21	86.68	76.75	68.67	67.28	35.91	16.71	5.81	74.05	47.15

Table 1: Accuracy of the coarse-to-fine viewpoint refinement for different domain adaptation techniques. For the methods with K-Means clustering, the mean and standard deviation (brackets) over 10 runs are provided.

4.2. Viewpoint Estimation

We finally evaluate the accuracy of the pose estimation on the real test images. To this end, we train the viewpoint estimator described in Section 3.3 on the synthetic data (*syn*), the real training data (*real*) with refined viewpoint labels or on both datasets (*joint*). For the refinement, we use our approach with $K = N$ (*with DA*) and compare it to the refinement without domain adaptation (*w/o DA*). We report the results for the datasets with non-occluded object instances in Table 3, where we also compare the accuracy of the pose estimator when the fine ground-truth viewpoint annotations of the real training images (*gt*) are used for training. This serves as

		PASCAL3D										
		AlexNet CNN-fc7										
views		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**68.38**	**65.22**	55.87	66.66	**65.23**	61.71	**49.79**	67.31	**64.22**	55.68	**70.28**	62.75
	V=views, K=N	66.00	63.43	**56.57**	**77.94**	64.77	**62.49**	46.10	**69.17**	62.13	**57.38**	69.83	**63.26**
16	w/o DA	45.62	33.35	30.98	52.37	45.70	39.06	30.37	46.39	33.04	37.72	**38.83**	39.40
	V=views, K=N	**47.18**	**41.03**	**32.92**	**60.29**	**46.99**	**41.48**	**30.42**	**50.70**	**43.36**	**42.15**	36.96	**43.04**
24	w/o DA	28.24	30.26	22.83	35.14	33.93	27.30	25.47	37.88	**24.67**	27.57	**31.76**	29.55
	V=views, K=N	**32.49**	**31.58**	**23.41**	**40.32**	**34.56**	**28.81**	**25.98**	**42.10**	23.11	**32.13**	24.55	**30.82**
		AlexNet CNN-pool5										
		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	68.80	67.71	58.06	64.73	**69.21**	62.62	49.84	69.73	59.64	50.61	55.96	61.53
	V=views, K=N	**71.10**	**72.75**	**59.77**	**68.81**	68.34	**63.91**	**63.45**	**72.36**	**68.13**	**54.74**	**73.29**	**66.97**
16	w/o DA	47.35	41.10	**32.61**	65.10	47.46	39.45	**37.23**	50.63	37.43	33.77	36.55	42.61
	V=views, K=N	**48.90**	**48.33**	32.05	**66.58**	**49.25**	**44.09**	33.09	**52.60**	**42.63**	**35.27**	**41.47**	**44.93**
24	w/o DA	30.89	30.21	25.31	41.78	33.99	29.34	**32.43**	37.09	**28.74**	28.69	25.12	31.24
	V=views, K=N	**32.57**	**33.55**	**27.74**	**44.12**	**35.55**	**31.48**	31.64	**42.36**	26.97	**29.86**	**27.90**	**33.07**

		ImageNet3D										
		AlexNet CNN-fc7										
views		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**70.80**	**74.43**	**59.63**	**81.45**	**89.02**	**77.78**	**55.07**	74.86	**76.06**	**66.49**	**84.11**	**73.61**
	V=views, K=N	69.46	73.79	59.55	73.68	87.03	75.26	52.19	**76.22**	71.70	61.33	80.45	70.97
16	w/o DA	**51.28**	48.76	34.07	51.23	69.86	61.12	46.18	51.34	**60.38**	**39.60**	36.08	50.00
	V=views, K=N	51.25	**55.35**	**35.50**	**52.61**	**72.38**	**66.43**	**49.84**	**56.59**	57.94	36.69	**54.10**	**53.52**
24	w/o DA	40.66	36.17	23.11	39.53	59.68	50.04	29.35	36.34	49.40	**21.17**	28.83	37.66
	V=views, K=N	**45.28**	**41.25**	**24.51**	**46.28**	**62.93**	**52.08**	**31.90**	**40.31**	**50.24**	20.94	**35.73**	**41.04**
		AlexNet CNN-pool5										
		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**71.50**	77.30	**58.99**	86.19	91.62	**79.43**	**60.41**	74.66	**81.59**	56.71	**70.77**	**73.56**
	V=views, K=N	70.79	**78.71**	56.79	**86.38**	**92.30**	78.59	57.57	**76.63**	74.31	**64.80**	68.96	73.25
16	w/o DA	51.16	54.08	36.05	64.10	**73.45**	67.62	39.06	54.25	**53.97**	37.82	51.35	52.98
	V=views, K=N	**53.14**	**57.93**	**36.44**	**69.09**	73.35	**67.89**	**49.92**	**55.51**	53.75	**38.31**	**52.77**	**55.28**
24	w/o DA	39.59	37.92	**22.75**	52.52	60.01	56.40	26.00	33.75	52.88	22.09	36.07	40.00
	V=views, K=N	**41.88**	**48.44**	22.65	**55.57**	**64.45**	**58.59**	**32.41**	**37.15**	**54.30**	**31.54**	**37.30**	**44.03**

Table 2: Accuracy of the coarse-to-fine viewpoint refinement for the Pascal3D+ and ImageNet3D datasets that contain occlusions and truncated object instances.

an upper bound of the accuracy in comparison to the setting with only weak supervision.

When comparing the results of the domain adaptation for the synthetic, real or both training sets with the results without domain adaptation, we observe that the domain adaptation improves the viewpoint estimation for all scenarios, with the exception of the CNN-fc7 features for EPFL and KITTI with 8 viewpoint refinement.

Using refined real target images (*with DA real*) for training is in most cases sufficient. The adapted synthesized training data, however, performs better for fine-grained viewpoints $V \geq 72$ since the real images do not necessary provide enough samples for each viewpoint. Combining the real and

		3DObjCat [58]		EPFL [21]							KITTI [54]	
		HOG										
		8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*99.31*	*99.07*	*80.06*	*73.57*	*64.15*	*55.65*	*36.10*	*12.77*	*1.44*	*82.23*	*77.89*
w/o DA	syn	75.69	93.06	65.98	60.92	46.42	36.37	22.62	8.55	3.61	58.69	47.25
	real	**99.31**	**99.54**	76.04	65.46	50.65	43.67	23.08	2.65	0.39	74.43	55.69
	joint	88.89	99.07	72.52	63.81	51.71	44.10	23.02	8.55	4.38	72.75	54.30
with DA	syn	90.97	93.98	74.62	67.01	55.20	44.84	**25.96**	9.61	4.86	64.28	54.07
	real	**99.31**	**99.54**	**78.37**	69.04	54.67	**47.60**	23.40	3.96	0.64	**74.46**	56.28
	joint	93.06	99.07	75.73	**71.93**	**56.91**	**47.60**	24.59	**9.70**	**5.64**	73.23	**59.04**
		AlexNet CNN-fc7										
		8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*93.75*	*97.69*	*67.65*	*59.75*	*53.25*	*42.45*	*25.44*	*10.45*	*1.89*	*80.31*	*76.51*
w/o DA	syn	72.92	91.67	62.08	55.35	48.32	40.75	24.16	9.33	4.25	49.31	35.45
	real	84.72	**97.69**	65.77	57.91	48.62	40.61	17.09	2.34	0.41	**67.44**	43.02
	joint	77.78	94.44	**68.55**	59.05	51.54	43.86	24.79	10.02	4.80	61.41	43.52
with DA	syn	75.69	92.13	64.91	59.65	**54.58**	**44.77**	**26.20**	**9.06**	4.34	41.87	37.64
	real	**86.81**	**97.69**	64.51	59.61	52.20	41.64	22.95	3.04	0.60	64.30	**50.14**
	joint	79.86	96.76	67.61	**62.83**	53.93	43.79	24.10	8.82	**5.33**	61.77	49.95
		AlexNet CNN-pool5										
		8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*100.0*	*99.07*	*80.35*	*71.97*	*66.92*	*54.53*	*35.48*	*12.15*	*1.43*	*87.00*	*86.88*
w/o DA	syn	86.81	90.74	72.10	62.86	55.90	44.49	25.88	11.11	5.10	49.24	34.18
	real	**100.0**	98.61	79.49	66.06	59.65	45.62	24.66	2.35	0.00	75.48	48.86
	joint	97.22	98.61	79.40	67.55	61.35	49.35	26.18	10.69	5.78	73.16	47.66
with DA	syn	96.53	97.69	77.03	70.81	62.69	50.78	27.80	10.20	**6.69**	66.01	34.49
	real	**100.0**	**99.07**	79.83	**73.54**	62.95	51.38	**27.84**	9.46	0.31	**80.51**	**66.57**
	joint	97.22	98.61	**79.85**	72.80	**65.13**	**52.10**	27.67	**11.94**	6.65	80.34	64.52

Table 3: Pose estimation accuracy on unlabelled test data using real training data, synthetic data or both training sets. All datasets contain non-occluded object instances.

synthetic data for training (*with DA joint*) works very well for any viewpoint discretisation and is therefore recommended in practice.

Table 4 reports the accuracies for the Pascal3D+ and ImageNet3D datasets using CNN-pool5 features. On these datasets the adapted synthesized training data performs already better than the real data for $V \geq 16$ fine viewpoints. As before, combining the refined real data and the adapted synthesized data for training performs well for any viewpoint discretisation $V = 8, 16, 24$. It is interesting to note that our weakly supervised approach (*with DA joint*) even outperforms the fully supervised approach (*gt*) due to the training data augmentation by the adapted synthetic images.

5. Conclusions

In this work, we have presented an approach for weakly supervised domain adaptation for the task of viewpoint estimation. It uses synthetic data to

			PASCAL3D											
			AlexNet CNN-pool5											
			aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	*gt*	*49.49*	*43.79*	*25.51*	*34.16*	*44.71*	*42.97*	*22.19*	*51.68*	*26.56*	*23.86*	*21.99*	*35.17*
		syn	41.80	44.95	**25.25**	33.81	38.46	40.71	13.83	46.96	50.17	17.42	23.44	34.25
	with DA	syn	43.07	**51.32**	25.04	35.64	42.90	**41.67**	10.46	45.81	52.34	20.39	30.60	36.29
		real	42.01	49.51	22.23	34.01	44.69	41.59	**14.17**	**53.06**	26.36	16.09	30.07	33.07
		joint	**46.20**	51.27	24.93	**38.34**	**47.03**	41.18	12.96	52.23	**56.15**	**22.07**	**34.04**	**38.76**
16	w/o DA	*gt*	*33.30*	*25.51*	*13.46*	*24.12*	*30.27*	*27.45*	*10.15*	*27.79*	*11.43*	*18.23*	*13.46*	*21.38*
		syn	27.96	27.93	11.33	27.75	26.92	28.41	10.00	**31.03**	42.69	**16.86**	19.95	24.62
	with DA	syn	31.12	**29.07**	12.35	30.05	31.32	28.97	13.38	29.25	**47.62**	14.34	**24.96**	**26.58**
		real	29.20	22.00	13.46	18.31	33.15	27.23	9.06	27.46	21.19	15.33	23.06	19.04
		joint	**33.96**	27.22	**13.97**	**30.14**	**33.81**	**29.35**	**13.92**	29.47	43.24	15.88	21.19	26.55
24	w/o DA	*gt*	*23.79*	*17.58*	*9.72*	*18.23*	*24.40*	*18.37*	*7.97*	*20.03*	*9.37*	*13.78*	*14.07*	*17.03*
		syn	16.97	18.31	10.84	23.88	21.77	20.89	5.16	19.38	32.54	9.13	10.82	17.24
	with DA	syn	19.56	21.27	**10.86**	**26.85**	25.76	22.22	7.16	**24.81**	**35.53**	8.64	**15.52**	**19.83**
		real	18.45	15.82	7.71	11.19	25.40	18.91	6.85	15.89	11.33	9.18	12.17	13.90
		joint	**21.63**	**22.65**	9.39	26.84	**26.65**	**22.69**	**9.13**	20.35	32.51	**10.0**0	10.98	19.34
			ImageNet3D											
			AlexNet CNN-pool5											
			aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	*gt*	*59.76*	*66.99*	*49.25*	*65.53*	*84.91*	*58.27*	*37.04*	*67.63*	*37.59*	*24.59*	*35.86*	*53.40*
		syn	46.33	62.12	26.06	58.74	76.43	66.56	28.74	61.45	66.63	17.16	37.05	49.75
	with DA	syn	47.94	64.59	29.36	54.74	76.96	**72.06**	24.68	63.90	73.76	19.38	33.20	50.96
		real	50.58	64.95	**37.80**	64.04	**83.00**	61.33	29.99	66.91	60.03	20.40	**48.90**	53.45
		joint	**50.85**	**65.74**	36.81	**65.21**	82.83	69.67	**39.14**	**69.19**	**76.71**	**21.22**	39.12	**56.06**
16	w/o DA	*gt*	*40.62*	*43.29*	*35.12*	*42.16*	*71.11*	*35.92*	*24.23*	*40.88*	*24.61*	*14.76*	*17.55*	*35.48*
		syn	33.40	37.54	15.98	40.93	59.57	54.36	17.22	37.23	36.03	17.51	12.12	32.90
	with DA	syn	34.06	40.36	16.45	46.48	59.99	**57.91**	17.41	38.71	39.73	19.46	19.12	35.43
		real	33.96	**41.28**	20.05	40.12	**66.79**	41.64	15.67	38.75	22.41	11.57	14.85	31.55
		joint	**36.12**	40.59	**21.40**	**49.82**	66.57	54.02	**22.97**	**41.09**	**40.43**	**26.79**	**20.76**	**38.23**
24	w/o DA	*gt*	*30.16*	*35.86*	*26.25*	*45.33*	*63.20*	*28.37*	*19.99*	*30.03*	*18.08*	*13.38*	*17.14*	*29.80*
		syn	26.29	25.01	10.48	35.16	49.38	42.49	13.49	26.69	25.21	6.19	17.33	25.24
	with DA	syn	25.61	**34.56**	13.55	36.92	52.59	**50.04**	15.27	**30.44**	32.61	8.91	20.85	29.30
		real	25.41	29.64	12.98	19.54	56.96	31.64	13.19	25.33	18.54	8.61	10.82	22.96
		joint	**28.95**	32.54	**14.55**	**40.40**	**57.93**	45.33	**18.08**	29.64	**37.71**	**12.04**	**21.72**	**30.81**

Table 4: Pose estimation accuracy for the Pascal3D+ and ImageNet3D datasets that contain occlusions and truncated object instances.

refine the viewpoint annotations of the coarsely labelled training images. Using coarse viewpoint annotations of real images as weak supervision together with accurately annotated synthesized images is not only a very efficient approach to collect training data for fine-grained viewpoint estimation, it also allows to achieve an accuracy that goes beyond the abilities of human annotators. Our evaluation on five datasets for viewpoint estimation showed that our approach outperforms generic domain adaptation methods and even outperforms fully supervised methods in some cases.

References

[1] R. Mottaghi, Y. Xiang, S. Savarese, A coarse-to-fine model for 3D pose estimation and sub-category recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 418–426.

[2] B. Sun, K. Saenko, From virtual to reality: Fast adaptation of virtual object detectors to real domains, in: British Machine Vision Conference, 2014.

[3] D. Vázquez, A. López, D. Ponsa, J. Marín, Cool world: domain adaptation of virtual and real worlds for human detection using active learning, in: Advances in Neural Information Processing Systems, Workshop on Domain Adaptation: Theory and Applications, 2011.

[4] D. Vázquez, A. López, J. Marín, D. Ponsa, D. Gerónimo, Virtual and real world adaptation for pedestrian detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 36 (4) (2014) 797–809.

[5] L. Pishchulin, A. Jain, C. Wojek, M. Andriluka, T. Thormählen, B. Schiele, Learning people detection models from few training samples, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 1473–1480.

[6] J. Marín, D. Vázquez, D. Gerónimo, A. López, Learning appearance in virtual scenarios for pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 137–144.

[7] 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, 2014, pp. 75–82.

[8] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2066–2073.

[9] J. Hoffman, E. Rodner, J. Donahue, K. Saenko, T. Darrell, Efficient learning of domain-invariant image representations, in: International Conference on Learning Representations, 2013.

[10] P. Busto, J. Liebelt, J. Gall, Adaptation of synthetic data for coarse-to-fine viewpoint refinement, in: British Machine Vision Conference, 2015.

[11] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[12] B. Leibe, A. Leonardis, B. Schiele, Combined object categorization and segmentation with an implicit shape model, in: IEEE European Conference on Computer Vision, Vol. 2, 2004, p. 7.

[13] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, 2005, pp. 886–893.

[14] P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (9) (2010) 1627–1645.

[15] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 580–587.

[16] J. Liebelt, C. Schmid, Multi-view object class detection with a 3D geometric model, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 1688–1695.

[17] S. Fidler, S. Dickinson, R. Urtasun, 3d object detection and viewpoint estimation with a deformable 3d cuboid model, in: Advances in Neural Information Processing Systems, 2012, pp. 611–619.

[18] B. Pepik, M. Stark, P. Gehler, B. Schiele, Teaching 3D geometry to deformable part models, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3362–3369.

[19] M. Hejrati, D. Ramanan, Analysis by synthesis: 3D object recognition by object reconstruction, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 2449–2456.

[20] C. Gu, X. Ren, Discriminative mixture-of-templates for viewpoint classification, in: IEEE European Conference on Computer Vision, Springer, 2010, pp. 408–421.

[21] M. Ozuysal, V. Lepetit, P. Fua, Pose estimation for category specific multiview object localization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 778–785.

[22] S. Tulsiani, J. Malik, Viewpoints and keypoints, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2015, pp. 1510–1519.

[23] B. Pepik, M. Stark, P. Gehler, T. Ritschel, B. Schiele, 3D object class detection in the wild, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 1–10.

[24] A. Ghodrati, M. Pedersoli, T. Tuytelaars, Is 2D information enough for viewpoint estimation?, in: British Machine Vision Conference, 2014.

[25] M. Torki, A. Elgammal, Regression from local features for viewpoint and pose estimation, in: IEEE International Conference on Computer Vision, IEEE, 2011, pp. 2603–2610.

[26] M. Fenzi, L. Leal-Taixe, B. Rosenhahn, J. Ostermann, Class generative models based on feature regression for pose estimation of object categories, in: IEEE Conference on Computer Vision and Pattern Recognition, 2013.

[27] K. He, L. Sigal, S. Sclaroff, Parameterizing object detectors in the continuous pose space, in: IEEE European Conference on Computer Vision, Springer, 2014, pp. 450–465.

[28] F. Massa, M. Aubry, R. Marlet, Convolutional neural networks for joint object detection and pose estimation: A comparative study, in: International Conference on Learning Representations, 2015.

[29] M. Elhoseiny, T. El-Gaaly, A. Bakry, A. Elgammal, A comparative analysis and study of multiview cnn models for joint object categorization and pose estimation, in: International Conference on Machine Learning, 2016, pp. 888–897.

[30] J. Schels, J. Liebelt, R. Lienhart, Learning an object class representation on a continuous viewsphere, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3170–3177.

[31] M. Stark, M. Goesele, B. Schiele, Back to the future: learning shape models from 3D CAD data, in: British Machine Vision Conference, Vol. 2, 2010, p. 5.

[32] M. Z. Zia, M. Stark, B. Schiele, K. Schindler, Detailed 3D representations for object recognition and modeling, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (11) (2013) 2608–2623.

[33] 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: IEEE International Conference on Computer Vision, 2015, pp. 2686–2694.

[34] X. Peng, B. Sun, K. Ali, K. Saenko, Learning deep object detectors from 3d models, in: IEEE International Conference on Computer Vision, 2015.

[35] K. Matzen, N. Snavely, Nyc3dcars: A dataset of 3D vehicles in geographic context, in: IEEE International Conference on Computer Vision, 2013, pp. 761–768.

[36] R. Gopalan, R. Li, R. Chellappa, Domain adaptation for object recognition: An unsupervised approach, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 999–1006.

[37] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, M. Salzmann, Unsupervised domain adaptation by domain invariant projection, in: IEEE International Conference on Computer Vision, 2013, pp. 769–776.

[38] S. J. Pan, I. W. Tsang, J. T. Kwok, Q. Yang, et al., Domain adaptation via transfer component analysis, IEEE Transactions on Neural Networks 22 (2) (2011) 199–210.

[39] I. Jhuo, D. Liu, D. Lee, S. Chang, Robust visual domain adaptation with low-rank reconstruction, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2168–2175.

[40] J. Yang, R. Yan, A. G. Hauptmann, Cross-domain video concept detection using adaptive svms, in: ACM International Conference on Multimedia, 2007, pp. 188–197.

[41] Y. Aytar, A. Zisserman, Tabula rasa: model transfer for object category detection, in: IEEE International Conference on Computer Vision, 2011, pp. 2252–2259.

[42] L. Duan, D. Xu, I. Tsang, J. Luo, Visual event recognition in videos by learning from web data, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (9) (2012) 1667–1680.

[43] Z. Xu, W. Li, L. Niu, D. Xu, Exploiting low-rank structure from latent domains for domain generalization, in: IEEE European Conference on Computer Vision, 2014, pp. 628–643.

[44] B. Gong, K. Grauman, F. Sha, Reshaping visual datasets for domain adaptation, in: Advances in Neural Information Processing Systems, 2013, pp. 1286–1294.

[45] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting visual category models to new domains, in: IEEE European Conference on Computer Vision, 2010, pp. 213–226.

[46] B. Fernando, A. Habrard, M. Sebban, T. Tuytelaars, Unsupervised visual domain adaptation using subspace alignment, in: IEEE International Conference on Computer Vision, IEEE, 2013, pp. 2960–2967.

[47] B. Sun, J. Feng, K. Saenko, Return of frustratingly easy domain adaptation, in: AAAI Conference on Artificial Intelligence, 2015.

[48] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: A deep convolutional activation feature for generic visual recognition, in: International Conference on Machine Learning, 2014, pp. 647–655.

[49] S. Chopra, S. Balakrishnan, R. Gopalan, DLID: Deep learning for domain adaptation by interpolating between domains, in: ICML workshop on challenges in representation learning, Vol. 2, 2013.

[50] Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation, in: International Conference on Machine Learning, 2015, pp. 1180–1189.
URL http://jmlr.org/proceedings/papers/v37/ganin15.pdf

[51] M. Ghifary, W. B. Kleijn, M. Zhang, Domain adaptive neural networks for object recognition, in: PRICAI 2014: Trends in Artificial Intelligence, Springer, 2014, pp. 898–904.

[52] E. Tzeng, J. Hoffman, T. Darrell, K. Saenko, Simultaneous deep transfer across domains and tasks, in: IEEE International Conference on Computer Vision, 2015, pp. 4068–4076.

[53] B. T. Phong, Illumination for computer generated pictures, Communications of the ACM 18 (6) (1975) 311–317.

[54] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the KITTI vision benchmark suite, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361.

[55] K. Svanberg, A class of globally convergent optimization methods based on conservative convex separable approximations, SIAM Journal on Optimization 12 (2) (2002) 555–573.

[56] S. G. Johnson, The NLopt nonlinear-optimization package (2007–2010). URL `http://ab-initio.mit.edu/nlopt`

[57] H. W. Kuhn, The Hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1-2) (1955) 83–97.

[58] S. Savarese, L. Fei-Fei, 3D generic object categorization, localization and pose estimation, in: IEEE International Conference on Computer Vision, 2007, pp. 1–8.

[59] M. Everingham, L. Van Gool, C. Williams, J. Winn, A. Zisserman, The Pascal visual object classes (VOC) challenge, International Journal of Computer Vision 88 (2) (2010) 303–338.

[60] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.