Abstract

Understanding the 3D world without supervision is currently a major challenge in computer vision as the annotations required to supervise deep networks for tasks in this domain are expensive to obtain on a large scale. In this paper, we address the problem of unsupervised viewpoint estimation. We formulate this as a self-supervised learning task, where image reconstruction provides the supervision needed to predict the camera viewpoint. Specifically, we make use of pairs of images of the same object at training time, from unknown viewpoints, to self-supervise training by combining the viewpoint information from one image with the appearance information from the other. We demonstrate that using a perspective spatial transformer allows efficient viewpoint learning, outperforming existing unsupervised approaches on synthetic data, and obtains competitive results on the challenging PASCAL3D+ dataset.

1. Introduction

Object viewpoint estimation is one of the key components required to enable autonomous systems to understand the 3D world. Earlier methods [20, 21, 4] were successfully demonstrated to work in controlled environments. While more recent work, benefiting from modern learnable representations [55, 17, 32], have been shown to help other vision tasks such as object detection and 3D reconstruction [31] and have been deployed in various applications [46, 16, 33].

In this paper, we focus on recovering the 3D pose of an object relative to the camera viewing it from a single image. This requires an accurate understanding of the 3D structure of the objects depicted in an image, and is can be complicated by challenges such as object symmetry. While large-scale image datasets annotated with semantic supervision (e.g. [11, 34]) have been a key enabler for modern deep networks, obtaining viewpoint annotations can be extremely laborious, expensive, and error-prone. For instance, the annotation procedure for the commonly used object viewpoint benchmark, PASCAL3D+ [59], required annotators to manually select an appropriate 3D CAD model from a pool of models and click on the 2D locations for a set of predefined landmarks for each object instance. Similarly, creating 3D face pose estimation datasets such as [63] involves a time consuming step of fitting morphable 3D face models to 2D face images. As a result, creating large-scale viewpoint datasets for a diverse set of objects, especially when 3D CAD models are unavailable, is challenging using existing annotation pipelines.

To overcome this problem, we propose a self-supervised object viewpoint estimation method, ViewNet, that learns from an unlabeled (i.e. without ground truth pose) collection of image pairs from a given object category (see Figure 1). Our method exploits multi-view consistency and does not require any manual viewpoint annotations. Our work is inspired by the analysis by synthesis and conditional generation paradigms [61, 54, 35, 37] where we learn to disentangle viewpoint and 3D appearance of objects and reconstruct images based on these disentangled factors. To this end, we leverage the information contained in image pairs of the same object with different viewpoints. Such pairs may be generated synthetically or obtained from videos. Given such an image pair, our method extracts 3D appearance from the first image and viewpoint from the second one, and reconstructs the second one based on this factorization. At test time, our method can be used to predict the viewpoint relative to the camera of objects from single im-
images. Unlike previous work [56, 23], we are able to leverage supervision directly from pixel values, allowing for more efficient supervision, as well as enabling the generation of images from new viewpoints.

Our main contributions are: (i) A new conditional generation approach for estimating viewpoint from single images via self-supervision. Our model encodes strong geometric consistency, enabling it to accurately generate novel views that can be used to refine its own predictions. (ii) A detailed evaluation of our method on ShapeNet and PASCAL3D+, where we outperform related approaches. (iii) We highlight the limitations of current evaluation procedures by showing that a large portion of object instances in certain categories are captured from similar viewpoints rendering their evaluation biased. We also show that the calibration step commonly used to align estimated unsupervised viewpoints with the ground-truth introduces additional biases.

2. Related work

Supervised pose estimation. Effective pose estimation from images has many real-world applications, e.g. in robotics or autonomous vehicles, and thus has been extensively studied. While early works performed pose estimation by matching low-level image descriptors [20, 21, 4], more recent approaches employ deep networks for predicting 3D bounding boxes [45, 53, 17], or classifying viewpoints directly [55, 28]. Keypoint prediction is a closely related task, and multitask setups where both are jointly learned have been shown to be successful [55, 62]. Alternatively, one can be used to learn the other, as one can recover pose by aligning keypoints [43], or discover them by enforcing a pose-aware sparse representation of objects [52].

Recent work has proposed modeling the topology of the viewpoint space by quantifying the uncertainty with a von Mises distribution [44], learning 2D image embeddings that are equivariant to 3D pose [13], employing a spherical exponential mapping at the regression output [32], or introducing cylindrical convolutions [27]. However, all of these approaches are supervised and requires pose annotations from datasets such as PASCAL3D+ [59] or LINEMOD [21]. The first of which was manually annotated, and the second was created in a controlled lab setup where poses were collected with each image. Alternatively, coarse viewpoint estimation can be obtained without manual annotations using structure from motion algorithms on videos [48, 40]. Ground truth pose annotations are challenging to acquire, and recent benchmarks still require human intervention in order to set the coordinate system for each instance and to correct automatic pose errors [1].

3D-aware representations. A parallel line of work learns representations that are aware of the underlying 3D structure of objects from images. Earlier works employ auto-encoders to disentangle pose and object appearance, with [58] or without [30] pose supervision. More recent works extend this disentangled pose learning from in-plane rotations to full 3D poses by crafting models that reason with spherical representations [10, 12], apply 3D rotations on embeddings to reconstruct images from a different viewpoint [47], use denoising auto-encoders to better extract viewpoint information [51], or by generalizing variational auto-encoders to spherical functions [14]. First proposed for 2D feature maps, spatial transformers [24] provide a way to apply in-plane transformations to any representation using spatial resampling and were later extended to 3D convolutions [61]. These sampling operations can be used to represent complete 3D scenes from multiple views [50]. In a related analysis by synthesis approach, [8] also learn pose representations via an appearance based reconstruction loss. At inference time, they iteratively optimize for the viewpoint that minimizes the appearance loss between the synthesized view and the input image. However, apart from a few unrealistically simplified experiments, all of these methods require 3D annotations in order to learn meaningful embeddings. Unlike in-plane rotations, which are simple enough to learn in an unsupervised way, 3D rotations can cause drastic appearance changes that are often too complex for networks to learn without pose annotations [35].

Viewpoint-conditioned generation. An increasingly popular way of learning interpretable representations is by using a generation process conditioned on the relevant information. The two main ways of building such representation rely either on encode-decoder approaches, using image pairs where semantics are shared [57, 25, 35], or on adversarial models to generate new samples in a controllable way [9, 38]. Both techniques have been shown to estimate viewpoints without labels [54, 23, 37]. Encode-decoder approaches are closely related to the field of unsupervised 3D reconstruction [19, 39, 41, 42]. In contrast to [54, 23] that do both 3D reconstruction and pose estimation, we propose a simpler fully self-supervised approach that is able to leverage appearance matching as supervision, allowing for novel view synthesis that can be used to further refine predictions.

SSV [37] uses an adversarial model to generate objects with random rotations while learning to regress viewpoint at the same time. In contrast, our proposed method ensures geometric consistency during the image generation process, allowing for more robust viewpoint estimation. Furthermore, GAN training can be unstable [36, 3], an issue often reflected in the auxiliary objectives required to guide training. In contrast, our method operates via image reconstruction alone, and can easily generate images from novel viewpoints. Several non-adversarial generative approaches have also been proposed [35, 8] that reconstruct specific object instances in order to leverage pixel-level supervision. However, unlike our approach, these methods require at least a partially labeled training set.
3. Method

Given a collection of unlabeled images \( T \), at training time we aim to learn a function \( f^v : \mathcal{I} \to \mathcal{V} \) that can map from image space \( \mathcal{I} \) to pose space \( \mathcal{V} \). At test time, we can then apply this function to a single image \( I \), containing an object of interest, in order to estimate its 3D viewpoint \( v \) relative to the camera. 3D viewpoints can be represented in different ways, including the Euler angles (azimuth, elevation and tilt), or with a rotation matrix \( R \in SO_3 \), and we use both representations interchangeably.

As ground-truth viewpoints of the objects in \( T \) are challenging to acquire, we formulate our problem as a self-supervised task that uses principles from conditional generation and synthesis by analysis. To this end, we propose to factorize the viewpoint and appearance of objects via two functions \( f^v \) and \( f^a \). Given an image \( I \), \( f^a \) outputs an appearance feature \( a \) for the object contained in it. The decoder \( f^d \), can reconstruct the image \( I \) given the pose of the object \( v \) and its appearance \( a \). \( f^v \), \( f^a \), and \( f^d \) are instantiated as neural networks parameterized by \( \theta^v \), \( \theta^a \) and \( \theta^d \) respectively. Clearly such a factorization is not guaranteed without some constraints on \( f^v \) and \( f^a \). To overcome this ambiguity, we use image pairs of rigid objects at training time that differ by their viewpoint. Such pairs can be extracted from video sequences, generated by perturbing still images, or rendered from 3D CAD models. Hence we assume that the set of unlabeled images \( T \) can be described as \( N \) image pairs \( T = \{(I_i, I'_i)\}_{i=1}^N \) where each pair contains images of the same object instance from two different viewpoints \( \{v_i, v'_i\} \), where the actual viewpoint information, relative or absolute, is not available. Given an image pair \( (I_i, I'_i) \), we propose to extract pose features \( v \) from \( I_i \) and appearance features \( a' \) from \( I'_i \), and use them to reconstruct \( I_i \). An overview of our model is shown in Figure 2. Our learning task consists of solving the following objective:

\[
\min_{\theta^v, \theta^a, \theta^d} \sum_{(I, I') \in T} \|f^d(f^a(I'), f^v(I)) - I\|. 
\] (1)

3.1. Pose estimation network \( f^v \)

We design the pose estimation network to output a point on the 3D unit sphere \( i.e. f^v(I) = v \in S^2 \), and uniquely map each point on the sphere to a viewpoint. To this end, we apply an orthogonalization operation to \( f^v \)'s output with the following steps. First we define an arbitrary vector \( u \in S^2 \) that represents the upwards direction, then we apply two successive cross products, \( w = v \times u \) and \( u' = w \times v \), and normalize the results to obtain orthogonal vectors. Finally we define the rotation matrix \( R \) as \( [v, w, u'] \). This matrix is then used to rotate the object representation during the generative stage, described later. This approach uses an arbitrarily chosen upwards direction, meaning we assume images do not contain in-plane rotations. However, in the more general case, \( u \) can be learned jointly with \( v \), effectively describing the full range of 3D rotations.

The main pitfall of unsupervised viewpoint estimation is the collapse of predictions caused by symmetries. Current approaches work well on simple objects \( e.g. \) a cube with each face colored differently. However, real world objects tend to have at least one, if not multiple, symmetric viewpoint pairs. We say that two viewpoints \( v, v' \) form a symmetric pair, \( v \sim v' \), if the image produced by observing the object from \( v \) is close to that produced from \( v' \). For instance, in most cars, \( (a, e, t) \sim (a + \pi, e, t) \) form a symmetric pair for any azimuth \( a \), elevation \( e \), and camera tilt \( t \). As a result of this, unsupervised methods based on reconstruction often equate those two viewpoints, leading to a collapse of the predictions. Different workarounds have been proposed to mitigate this, such as using adversarial losses to enforce a prior on the pose distribution [54], using multiple prediction heads [54, 23], or enforcing some symmetric consistency in the predictions using a flipped version of the image [37]. The main drawback of this last approach is that it is only valid for a left-right planar symmetry, and would likely fail in the aforementioned car example. To overcome this problem, we use multiple prediction heads for our pose estimator, resulting in multiple hypotheses for \( v \). Each head can learn to specialize in a subset of the viewpoints, and in the case of a symmetric pair \( v \sim v' \), both can simultaneously be predicted by two different heads.

In practice, each predictor head \( f^v_{m} \) outputs a viewpoint prediction, and the one associated with the lowest reconstruction error is chosen as the prediction at training time:

\[
v^* = f^v_{m^*}(I) \quad \text{s.t.} \quad m^* = \arg\min_{m \in M} \|f^d(f^a(I'), f^v_m(I)) - I\|. 
\] (2)

where \( f^v_m \) denotes \( m \)-th viewpoint predictor and \( M \) is the number of heads. Gradients will only be propagated through \( m^* \), ensuring that symmetric pairs get separated.

At test time, ViewNet only requires the pose prediction network \( f^v \), and does not need \( f^a \) or \( f^d \) in order to make a prediction. To achieve this, we jointly train a selection head, which is tasked with picking the best prediction for each input image given the range of options. The task is to minimize the cross-entropy between the selection prediction and a one-hot distribution representing \( m^* \), computed via Eqn. 2. Although \( m^* \) is not guaranteed to be the prediction closest to ground truth pose, we observe it is enough to differentiate between symmetric viewpoint pairs. Compared with [23], this allows us to efficiently maintain multiple hypotheses at test time, which translates to more robust predictions, and we do not require complex solutions such as reinforcement learning as in [54].
I

an affine transformation to the features parameterized by
ally encoded by AdaIN layers (see Figure 2), which apply
mains constant. Its purpose is to encode the average object
z
process at different scales.

in GANs \[5, 38, 37\] due to its ability to adapt the generation
tailly developed for style transfer, this approach is popular
use of adaptive instance normalization (AdaIN) \[22\]. Ini-
use an object-conditional generation process which makes
3.3. Decoder network \(f^d\)

As they are applied uniformly over each feature channels, it
is impossible for them to alter local information of the fea-
tures. Additionally, a standard encoder-decoder architecture
would only use \(a\) as input of the decoder, meaning fine de-
tails of the object can be lost during the complex decoding
process. By comparison, applying transformations across
different layers means they can influence all levels of the
reconstruction, resulting in more faithful reconstructions.

3.2. Appearance encoding network \(f^a\)

The appearance \(f^a(I') = a' \in \mathbb{R}^n\) of the object rep-
resented in the input image is also learned with a con-
volutional network. In a standard encoder-decoder architec-
ture, \(a'\) would be used as an input to \(f^d\) to produce a re-
construction. However, this offers no guarantee that the
viewpoint \(v'\) and appearance \(a'\) embeddings are correctly
factorized. In particular, information about \(v'\) could be
encoded in \(a'\). This means that a change in \(v'\) could
induce changes in the appearance of the reconstruction. In
extreme cases, the network could even ignore \(v\) and recon-
struct \(I\) by memorizing the \((I, I')\) pairs. To mitigate this, we
use an object-conditional generation process which makes
use of adaptive instance normalization (AdaIN) \[22\]. Ini-
ially developed for style transfer, this approach is popular
in GANs \[5, 38, 37\] due to its ability to adapt the generation
process at different scales.

Our generation pipeline works by refining a random
static code \(z \in \mathbb{R}^m\) through the decoder network to the
final rendering stage. \(z\) is randomly picked from a normal
distribution at the beginning of the training process, and re-
 mains constant. Its purpose is to encode the average object
in a canonical pose. The appearance of the object is gradu-
ally encoded by AdaIN layers (see Figure 2), which apply
an affine transformation to the features parameterized by \(a\).

As they are applied uniformly over each feature channels, it
is impossible for them to alter local information of the fea-
tures. Additionally, a standard encoder-decoder architecture
would only use \(a\) as input of the decoder, meaning fine de-
tails of the object can be lost during the complex decoding
process. By comparison, applying transformations across
different layers means they can influence all levels of the
reconstruction, resulting in more faithful reconstructions.

3.3. Decoder network \(f^d\)

In order to ensure an accurate viewpoint prediction, we
aim to strictly enforce geometric consistency during the
generation process. To this end, \(f^d\) is modeled using 3D
convolutional layers, and uses a 3D spatial transformer
with perspective for image rendering, similar to those used
in \[61\], and is combined with a pseudo-ray tracing opera-
tion inspired by \[56\]. Placing the rotation at the final stage
of the network, as close as possible to the reconstruction
loss, ensures that gradients are efficiently propagated to \(f^v\).
The absence of parametric transformations between \(f^v\) and
the target image guarantees that viewpoint errors cannot be
compensated for by convolutional layers, as can happen in
GAN-based models.

Our rendering module consists of three main steps and
is related to those used in \[61, 56, 54, 23\], however our
pipeline also makes use of texture information. Specifically,
the steps involve: (i) Rotation. Given a 3D volume \(V\), a spa-
tial transformer can be used to rotate it using a rotation ma-
trix \(R\). The new volume is obtained by resampling the data
along the rotated axis. (ii) Perspective. Similarly, perspec-
tive can be simulated with a spatial transformer. The single
point perspective of a pinhole camera will have the effect of
decreasing the apparent size of objects proportional to dis-
tance. We can therefore resample the volume by dilating
close points and contracting distant ones. (iii) Projection-
based ray-tracing. Finally, the volume is projected to a two-
dimensional image plane. As parts of the objects will be
subject to self-occlusion, we use a pseudo ray-tracing op-
eration to compute which voxels will appear in the output
image, ensuring geometric consistency.

For each entry in the 3D volume \(V\), the first three chan-
nels \(C\) represent the RGB channels of an image, while the
fourth one \(Q\) is an occupancy map, containing information
about the shape of the object. The value of each cell is in-
terpreted as the probability of the object occupying the cor-
responding spatial location. To compute the projection, we
have to estimate where each light ray is likely to stop. Since
we already accounted for the perspective, all our rays are
parallel, leaving only the depth of each stopping point to be

Figure 2: Overview of ViewNet. \(f^v\) is the viewpoint prediction network. At training time, \(f^a\) encodes the object appearance
embedding from image \(I'\) which is decoded by \(f^d\) into a 3D representation and transformed by the estimated viewpoint into
an image in the same pose as \(I\) using the projection module. This reconstruction, which can be segmented, is then compared
to \(I\) to guide training. Yellow blocks indicate learned parameters, while green ones are fixed or analytical modules.
computed. Compared with [56], we do not have to compute a path for each light ray i.e. it is embedded in the shape of the tensor. This means we can compute all lights paths simultaneously using efficient parallel operations, in a manner similar to the orthographic projection used in [15]. The probability of the light ray, at pixel coordinates $i, j$, stopping at depth $k$ is given by

$$Q'_{i,j,k} = Q_{i,j,k} \prod_{l=0}^{k-1} (1 - Q_{i,j,l}),$$  

(3)

with the convention that an empty product is equal to 1. The first term represents the probability of the voxel at coordinates $(i, j, k)$ being occupied, and the second one is the probability of all the previous ones being not visible. Hence, the final pixel value at coordinate $i, j$ is

$$\hat{I}_{i,j} = \sum_{k=1}^{n} \left[ C_{i,j,k} \times Q_{i,j,k} \times \prod_{l=0}^{k-1} (1 - Q_{i,j,l}) \right].$$  

(4)

This is similar to the formulation in [56, 54], although in our case, the ray-tracing is parallelized and used to sample RGB values, rather than computing depth or ray termination.

A failure case of our approach consists of ViewNet using the volume $V$ as a canvas and “painting” the object in different poses on the sides. More generally, this results in errors in the predicted shape of the object, since we do not know which pixels belong to it. To address this, instead of trying to directly estimate occupancy $Q$, we learn $Q'$ such that $Q = S + Q'$ where $S$ is a three dimensional Gaussian distribution centered on $V$. $Q'$ can be interpreted as a residual that deforms a shape prior $S$ so that it matches the shape of the observed object. $S$ encodes a prior for the shape and position of the object, while discouraging the network from using voxels that are far away from the center of the volume.

### 3.4. Cycle consistency supervision

Using appearance supervision, as opposed to only object silhouettes as in [61, 54, 23], enables ViewNet to also represent appearance information. This has two key advantages. First, our method can generate images of objects from novel views. Second, we can use these novel views to regularize our model during training by enforcing consistency between a generated image and its known viewpoint.

Given a randomly sampled viewpoint $\tilde{v} \sim U(V)$, we can render a novel image $\tilde{I} = f^d(\tilde{v}, a')$ using appearance information in $a'$ extracted from image $I'$. By feeding this to $f^a$, we can compute the distance between the sampled viewpoint $\tilde{v}$ and its estimated viewpoint $f^a(\tilde{I})$, i.e. $L_{cycle} = ||f^a(\tilde{I}) - \tilde{v}||$ and backpropagate this error to the viewpoint estimator. Assuming the reconstructions are of sufficient quality, this allows us to generalize beyond the potentially limited set of poses that are present in the training set, and these newly generated samples help regularize the viewpoint estimation network.

### 4. Experiments

Here we present 3D pose estimation results on both synthetic and real image datasets.

#### 4.1. Implementation details

ViewNet consists of three sub-networks: $f^v$, $f^a$, and $f^d$. Both $f^v$ and $f^a$ contain seven convolutional layers interleaved with batch normalization and ReLU activation functions respectively. $f^v$ takes a $64 \times 64$ RGB image $I$ as input and outputs $M = 3$ viewpoint hypotheses. $f^a$ encodes a second RGB image $I'$, depicting the same object instance captured from another viewpoint, and outputs a 256 dimensional appearance vector. The input to $f^d$ is a 1024 dimensional fixed canonical code vector. The canonical code is passed through seven 3D transposed convolutions, each followed by a ReLU, and the feature maps are further conditioned on the output of $f^a$ via adaptive instance normalization (AdaIN) layers. The resulting 3D feature map is projected to an image based on the predicted pose and used to compute the reconstruction error w.r.t. $I$. We use a perceptual loss [26], as it provides more informative gradients compared with standard pixel-level reconstruction losses. The supplementary material provides additional details.

In all experiments, we set the minibatch size to 64 and use the Adam optimizer [29]. For each experiment, we train a separate model per category, and select the model with the best performances on a held out validation set, stopping the training if no improvement is observed for 30 epochs. As our method is unsupervised, all viewpoints are predicted up to a random rotation. In order to evaluate our model, we must align its predictions with the ground truth. The standard alignment technique, performed by [54, 23], involves computing the rotation that best aligns the predicted viewpoints with the ground truth. This is obtained from a small batch of validation images, using the orthogonal Procruste algorithm. An alternative alignment procedure, used in [37], learns the parameters of a more flexible affine transformation that best maps the predicted viewpoints and to the ground truth. This can shrink and/or expand the predicted viewpoint estimation compared to applying a single 3D rotation to translate them. We discuss potential issues with this approach in section 4.3. We report performances in standard viewpoint estimation measures: accuracy at 30° and median angular error.

#### 4.2. ShapeNet results

Following [54, 23], we evaluate ViewNet on the ShapeNet dataset [6], which contain 7.5k, 6.8k, and 4k 3D CAD models for cars, chairs, and planes respectively.
Figure 3: (a) - (c) Comparison of ground truth versus predicted azimuth on three ShapeNet categories. A perfect predictor would appear as a single diagonal line. (d) Candidate reconstructions for each head. The left image is the input of the pose estimator, and the three following images are the renderings for each head, ranked by increasing reconstruction error.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy (%) ↑</th>
<th>Median error (°) ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>airplane</td>
<td>car</td>
</tr>
<tr>
<td>MVC [54]</td>
<td>69</td>
<td>87</td>
</tr>
<tr>
<td>Pointcloud [23]</td>
<td>75</td>
<td>86</td>
</tr>
<tr>
<td>ViewNet</td>
<td>82</td>
<td>89</td>
</tr>
<tr>
<td>ViewNet + cycle</td>
<td>86</td>
<td>91</td>
</tr>
</tbody>
</table>

Table 1: ShapeNet results for unsupervised methods. Bold entries are the best performing models for each category.

Training, validation, and testing sets are created by splitting CAD models into (0.7, 0.1, 0.2) fractions respectively. To render image pairs, we randomly select viewpoints and light sources uniformly over the $[0^\circ, 360^\circ]$ azimuth range and $[-20^\circ, 40^\circ]$ elevation range. We report results for the standard setting where each CAD model is rendered from five random viewpoints at train and test time.

The results in Table 1 show that ViewNet outperforms existing unsupervised approaches, except for median error on cars. ViewNet learns to reconstruct textures in addition to shape, and this supervision is more informative compared to only binary masks, as we can leverage texture cues to efficiently disentangle symmetric viewpoints. For example, the car category shows a second line of predictions shifted by $180^\circ$. This corresponds to the $(a, e, t) \sim (a + \pi, e, t)$ symmetry mentioned in section 3.1. Other categories showcase different symmetry induced issues, with planes and chairs having a retrograde symmetry $(a, e, t) \sim (\pi - a, e, t)$. Samples for each proposed viewpoint are shown in Figure 3 (d). One can see that the global input shape is matched relatively well across the different predicted views. Interestingly, reconstruction error is not necessarily directly correlated with viewpoint error, as the second proposed viewpoint for the car has lower reconstruction error than the third, despite being rendered from a completely different viewpoint.

Ablation study. In Table 2, we study the effect of each proposed component in our pipeline. First, we reduce the number of heads in the viewpoint estimators to one and observe a large overall drop in the viewpoint accuracy. For the car category, the single-head estimator cannot deal with the front/back symmetries, resulting in a large performance loss. Second, we modify ViewNet to reconstruct a binary segmentation mask similar to [54, 23], instead of the pixel values. Using segmentation masks as targets achieves results comparable with previous segmentation-based approaches in Table 1. This indicates that ViewNet can leverage texture information to achieve better predictions. Third, we remove our Gaussian shape prior and directly estimate the occupancy grid $Q$, instead of $Q'$, and observe that this does not have any significant affect on planes and cars, but causes a dramatic drop for chairs as the network tries to ‘paint’ the object on the faces of the volume. Next we evaluate the conditioning strategy by removing the AdaIN layers and feeding the output of $f^a$ in the first layer of $f^d$, similar to a traditional encoder-decoder pair. While this does not cause drastic performance issues, the reconstructions are less accurate, limiting the ability of this model to use them for self-training. Finally, we replace the analytic...
Table 2: Ablation study results. Here we compare different variants of ViewNet on ShapeNet.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (%)</th>
<th>Median error (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>airplane</td>
<td>car</td>
</tr>
<tr>
<td></td>
<td>airframe</td>
<td>car</td>
</tr>
<tr>
<td>ViewNet</td>
<td>82</td>
<td>89</td>
</tr>
<tr>
<td>Single-head</td>
<td>72</td>
<td>51</td>
</tr>
<tr>
<td>Segmentation Target</td>
<td>71</td>
<td>85</td>
</tr>
<tr>
<td>No Shape Prior</td>
<td>78</td>
<td>89</td>
</tr>
<tr>
<td>Encoder-Decoder</td>
<td>82</td>
<td>89</td>
</tr>
<tr>
<td>HoloGAN-Decoder</td>
<td>66</td>
<td>52</td>
</tr>
<tr>
<td>Constant</td>
<td>20</td>
<td>22</td>
</tr>
</tbody>
</table>

renderer with a learnable decoder using the deconvolutional architecture from [38]. In addition to causing the largest performance drop of all ablations, reconstructions from this model do not exhibit geometric consistency as the generated views do not smoothly change as the object rotates. Qualitative samples are provided in the supplementary materials.

4.3. PASCAL3D+ results

Next we evaluate ViewNet on the challenging real-world PASCAL3D+ [59] dataset. It contains real images from the PASCAL VOC and ImageNet datasets along with annotated viewpoint, including azimuth and elevation. As this dataset does not provide image pairs that contain the same object instance with varying viewpoints, we evaluate our models with 10-views per CAD model trained on ShapeNet. As PASCAL3D+ images have backgrounds, we synthetically add random background images from SUN397 [60] to our ShapeNet rendered views during training. These backgrounds are only added to the input training pairs to make ViewNet robust to backgrounds at test time. However, ViewNet is trained to reconstruct only the object, as it would require additional logic to reconstruct the background.

We report our results in Table 3. We observe that for some categories e.g., bottle, bus, sofa, train, and tv monitors, the ranges of viewpoints it contains are extremely restricted and concentrated around specific viewpoints. We reason that the viewpoint alignment procedure used for unsupervised methods is very effective in reaching strong performances on these classes. To test this hypothesis, we build a simple viewpoint predictor, a constant predictor, that outputs the average viewpoint from the validation set for each object category. This mimics the behavior of an untrained viewpoint estimator that has not learned anything useful and gets calibrated on validation data. We see that this method performs surprisingly well and even outperforms [17], a supervised approach on some categories. Even on non-trivial categories, the constant predictor performs surprisingly well, for instance, it obtains 43% accuracy on airplanes and 58% on chairs. By comparison, the same predictor on ShapeNet achieves a much lower performance (see Table 2), as the dataset was specifically crafted not to be biased.

To mitigate biases in the evaluation set, we propose a different evaluation strategy that consists of splitting the viewpoint space into discrete bins, and then averaging performance over each bin. Doing so prevents biased predictors from reaching near-perfect performance. Results under this scheme are presented in the supplementary material.

As an additional baseline, we reproduce the setup used in SSV [37] and fit a linear regressor to VGG16 [49] Conv5 features, and train it to regress the pose using the same small number of PASCAL3D+ images we use to align our predictions – see ‘VGG View’ results in Table 3.

We directly evaluate our ShapeNet trained ViewNet model on PASCAL3D+ images. We provide results for two alignment methods, the optimal rotation using orthogonal Procrustes, and the linear regression as used in SSV [37], which takes the predicted viewpoint and applies a linear regressor to modify its predictions. Depending on the category, two behaviors can be identified: either the two alignment procedures provide similar results (e.g., bike, bottle), or the linear regression approach significantly outperforms the optimal rotation. We observe that the second behavior is correlated with categories where PASCAL3D+ contains highly biased viewpoints, i.e., where most viewpoints are clustered around a single one. We theorize that the linear regression approach can artificially boost performance in those categories by collapsing viewpoint predictions towards the common view. This can be achieved by learning zero weights for the predicted viewing angles and encoding the average viewpoint as the bias term.

Similar to our ShapeNet experiments, we also evaluate the impact of training with our cycle based generated views. Depending on the categories, it often provides a small accuracy boost at the cost of higher median error. This median error increase could be due to the higher domain gap between generated views and real-world images.

4.4. Other dataset results

Up until this point, we have only trained ViewNet on the synthetic ShapeNet dataset and evaluated it on either synthetic or real data. Our method can also be trained on real-data that consists of image pairs of the same object which vary in their viewpoints. To this end, we use the recently proposed Objectron dataset [1], and the Freiburg cars dataset [48]. For Objectron we train on the chair category, as it is present in ShapeNet and contains sufficiently diverse high-quality images in contrast to the other categories where images are blurry or there are too few videos. While ViewNet does not require segmentation masks at test time, it does require segmented objects as the target for training.

Objectron. We first randomly sample ten frames per video, and obtain foreground masks using two different semantic segmentation methods: DeepLabV3 [7], trained on
Table 3: PASCAL3D+ results. Bold entries indicate the best performing models in each category. Entries followed by a star (*) use a linear regression alignment procedure, and those without use a single global rigid alignment.

COCO [34] ground-truth segmentation masks and a weakly supervised method [2], trained on Objectron frames using only image-level labels. We start from a model pretrained on ShapeNet to prevent overfitting on the relatively low amount of instances from Objectron. ViewNet without cycles obtains 91% and 89% accuracy with 8.8° and 10.1° median error on PASCAL3D+ cars for the supervised and weakly-supervised segmentation settings respectively. This is a significant improvement from the 83% accuracy obtained by using only the ShapeNet trained model.

Freiburg Cars. As the dataset only contains 48 videos, we use all frames, i.e. between 120 and 130 per instance. We also use segmentation masks obtained from a pre-trained supervised Mask R-CNN model [18]. Results are shown in Table 4. ViewNet obtains stronger results than the unsupervised approach of [40]. Adding our cycle loss does not improve performances as real cars exhibit specular reflections that ViewNet is unable to reproduce. As [40] does not report accuracy, we estimated it to be 50%, which correspond to a median error of 30°.

4.5. Limitations

ViewNet requires foreground masks at training time as the model is unable to extract background information from the appearance image. In experiments on real data, we use pre-trained segmentation models [7, 2, 18] to estimate these masks. However, it is important to note that the viewpoint estimator can be applied to unsegmented images at test time. Our method also relies on having image pairs during training in order to disentangle viewpoint and object appearance, which limits its real-world application to video datasets. Finally, we assume that object appearance is independent from the viewpoint, but this assumption is often violated by non-Lambertian surfaces, e.g. cars. We qualitatively analyze some failure cases of our method in the supplementary material.

5. Conclusion

We presented ViewNet, a self-supervised approach for learning object viewpoint estimation from image pairs. By ensuring geometric consistency during generation, we can accurately synthesize new views from objects and use them to refine our network predictions, outperforming current approaches on both synthetic and real datasets. Finally, we highlighted evaluation issues on the commonly used PASCAL3D+ dataset. We demonstrate that there are significant biases in the dataset and even simple baseline methods can perform well, suggesting a need for new benchmarks with more varied 3D poses.

Acknowledgments. OM is supported by Toyota Motor Europe, and HB by EPSRC Visual AI grant EP/T028572/1.
References


[27] Sunghun Joung, Seungryong Kim, Hanjae Kim, Minsu Kim, Ig-Jae Kim, Junghyun Cho, and Kwanghoon Sohn. Cylindrical convolutional networks for joint object detection and
[37] Siva Karthik Mustikovela, Varun Jampani, Shalini De Mello, Sifei Liu, Umar Iqbal, Carsten Rother, and Jan Kautz. Self-supervised viewpoint learning from image collections. In conference on Computer Vision and Pattern Recognition (CVPR), 2020. 1, 2, 3, 4, 5, 7, 8
on Computer Vision and Pattern Recognition (CVPR), pages 292–301, 2018. 2


