Viewpoint Estimation—Insights & Model

Gilad Divon and Ayellet Tal

Technion – Israel Institute of Technology

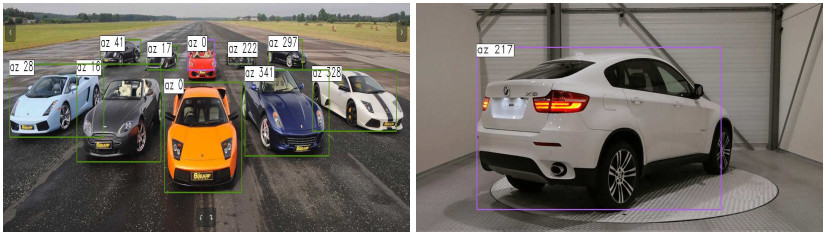


Fig. 1: **Viewpoint estimation.** Given an image containing objects from known categories, our model estimates the viewpoints (azimuth) of the objects.

Abstract. This paper addresses the problem of viewpoint estimation of an object in a given image. It presents five key insights and a CNN that is based on them. The network's major properties are as follows. (i) The architecture jointly solves detection, classification, and viewpoint estimation. (ii) New types of data are added and trained on. (iii) A novel loss function, which takes into account both the geometry of the problem and the new types of data, is propose. Our network allows a substantial boost in performance: from 36.1% gained by SOTA algorithms to 45.9%.

1 Introduction

Object category viewpoint estimation refers to the task of determining the viewpoints of objects in a given image, where the objects belong to known categories, as illustrated in Figure.1. This problem is an important component in our attempt to understand the 3D world around us and is therefore a long-term challenge in computer vision [1,2,3,4], having numerous application [5,6]. The difficulty in solving the problem stems from the fact that a single image, which is a projection from 3D, does not yield sufficient information to determine the viewpoint. Moreover, this problem suffers from scarcity of images with accurate viewpoint annotation, due not only to the high cost of manual annotation, but mostly to the imprecision of humans when estimating viewpoints.

Convolutional Neural Networks were recently applied to viewpoint estimation [7,8,9], leading to large improvements of state-of-the-art results on PASCAL3D+. Two major approaches were pursued. The first is a regression approach, which handles the continuous values of viewpoints naturally [10,11,8].

This approach manages to represent the periodic characteristic of the viewpoint and is invertible. However, as discussed in [7], the limitation of regression for viewpoint estimation is that it cannot represent well the ambiguities that exist between different viewpoints of objects that have symmetries or near symmetries.

The second approach is to treat viewpoint estimation as a classification problem [7,9]. In this case, viewpoints are transformed into a discrete space, where each viewpoint (angle) is represented as a single class (bin). The network predicts the probability of an object to be in each of these classes. This approach is shown to outperform regression, to be more robust, and to handle ambiguities better. Nevertheless, its downside is that similar viewpoints are located in different bins and therefore, the bin order becomes insignificant. This means that when the network errs, there is no advantage to small errors (nearby viewpoints) over large errors, as should be the case.

We follow the second approach. We present five key insights, some of which were discussed before: (i) Rather than separating the tasks of object detection, object classification, and viewpoint estimation, these should be integrated into a unified framework. (ii) As one of the major issues of this problem is the lack of labeled real images, novel ways to augment the data should be developed. (iii) The loss should reflect the geometry of the problem. (iv) Since viewpoints, unlike object classes, are related to one another, integrating over viewpoint predictions should outperform the selection of the strongest activation. (v) CNNs for viewpoint estimation improve as CNNs for object classification/detection do.

Based on these observations, we propose a network that improves the state-of-the-art results by 9.8%, from 36.1% to 45.9%, on PASCAL3D+ [12]. We touch each of the three components of any learning system: architecture, data, and loss. In particular, our architecture unifies object detection, object classification, and viewpoint estimation and is built on top of Faster R-CNN. Furthermore, in addition to real and synthetic images, we also use flipped images and videos, in a semi-supervised manner. This not only augments the data for training, but also lets us refine our loss. Finally, we define a new loss function that reflects both the geometry of the problem and the new types of training data.

Thus, this paper makes two major contributions. First, it presents insights that should be the basis of viewpoint estimation algorithms (Section 2). Second, it introduces a network (Section 3) that achieves SOTA results (Section 4). Our network is based on three additional contributions: a loss function that uniquely suits pose estimation, a novel integration concept, which takes into account the surroundings of the object, and new ways of data augmentation.

2 Our Insights in a nutshell

We start our study with short descriptions of five insights we make on viewpoint estimation. In the next section, we introduce an algorithm that is based on these insights and generates state-of-the-art results.

1. *Rather than separating the tasks of object detection, object classification, and viewpoint estimation, these should be integrated into a unified network.* In [7], an

off-the-shelf R-CNN [13] was used. Given the detection results, a network was designed to estimate the viewpoint. In [8] classification and viewpoint estimation were solved jointly, while relying on bounding box suggestions from Deep Mask [14]/Fast R-CNN [15]. We propose a different architecture that combines the three tasks and show that training the network jointly is beneficial. This insight is in accordance with similar observations made in other domains [16,17,18].

2. *As one of the major issues of viewpoint estimation is the lack of labeled real images, novel ways to augment the data are necessary.* In [7,8] it was proposed to use both real data and images of CAD models, for which backgrounds were randomly synthesized. We propose to add two new types of training data, which not only increase the volume of data, but also benefit learning. First, we horizontally flip the real images. Since the orientation of these images is known, yet no new information regarding detection and classification is added, they are used within a new loss function to focus on viewpoint estimation. Second, we use unlabeled videos of objects for which, though we do not know the exact orientation, we do know that subsequent frames should be associated with nearby viewpoints. This constraint is utilized to gain better viewpoint predictions. Finally, as a minor modification, rather than randomly choosing backgrounds for the synthetic images, we choose backgrounds that suit the objects, e.g. backgrounds of the ocean should be added to boats, but not to airplanes.

3. *The loss should reflect the geometry of the problem, since viewpoint estimation is essentially a geometric problem, having geometric constraints.* In [7], the loss considers the geometry by giving larger weights to bins of close viewpoints. In [8], it was found that this was not really helpful and viewpoint estimation was solved purely as a classification problem. We show that geometric constraints are very helpful. Indeed, our loss function considers (1) the relations between the geometries of triplets of images, (2) the constraints posed by the flipped images, and (3) the constraints posed by subsequent frames within videos

4. *Integration of the results is helpful.* Previous works chose as the final result the bin that contains the viewpoint having the strongest activation. Instead, we integrate over all the viewpoints within a bin and choose as the final result the bin that maximizes this integral. Interestingly, this idea has an effect that is similar to that of denoising and it is responsible for a major improvement in performance.

5. *As object classification/detection CNNs improve, so do CNNs for viewpoint estimation.* In [7] AlexNet [19] was used as the base network, whereas in [9,8] VGG [20] was used. We use ResNet [21], not only because of its better performance in classification, but also due to its skip-connections concept. These connections enable the flow of information between non-adjacent layers and by doing so, preserve spatial information from different scales. This idea is similar to the multi-scale approach of [9], which was shown to benefit viewpoint estimation.

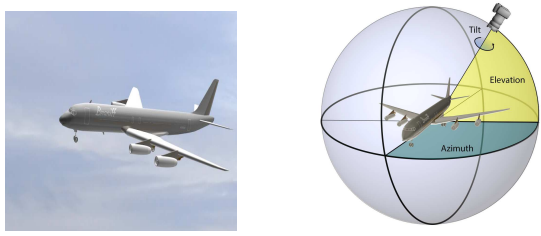
A concise view on the contribution of the insights: Table 1 summarizes the influence of each insight on the performance of viewpoint estimation. Our results are compared to those of [9,8,7]. The total gain of our algorithm is 9.8% compared to [8]. Section 4 will analyze these results in depth.

Method	Score (mAVP24)
[7]:AlexNet/R-CNN-Geometry-synthetic+real	19.8
[9]: VGG/R-CNN-classification-real	31.1
[8]: VGG/Fast R-CNN-classification-synthetic+real	36.1
Ours: Insights 1,5 - Architecture	40.6
Ours: Insights: 1,4,5 - Integration	43.2
Ours: Insights: 1,3,4,5 - Loss	44.4
Ours: Insights: 1,2,3,4,5 - Data	**45.9**

Table 1: **Contribution of the insights.** This table summarizes the influence of our insights on the performance. The total gain is 9.8% compared to [8].

3 Model

Recall that we treat viewpoint estimation as a classification problem. Though a viewpoint is defined as a 3D vector, representing the camera orientation relative to the object (Figure 2), we focus on the azimuth; finding the other angles is equivalent. The set of possible viewpoints is discretized into 360 classes, where each class represents 1°. This section presents the different components of our suggested network, which realizes the insights described in the previous section.



(a) Input (b) Output

Fig. 2: **Problem definition**. Given an image containing an object (a), the goal is to estimate the camera orientation (Euler angles) relative to the object (b).

3.1 Architecture

Hereafter, we describe the implementation of Insights 1,4&5, focusing on the integration of classification, object detection and viewpoint estimation. Figure 3 sketches our general architecture. It is based on Faster R-CNN [16], which both detects and classifies. As a base network within Faster R-CNN, we use ResNet [21], which is shown to achieve better results for classification than VGG. Another advantage of ResNet is its skip connections. To understand their importance, recall that in contrast to our goal, classification networks are trained to

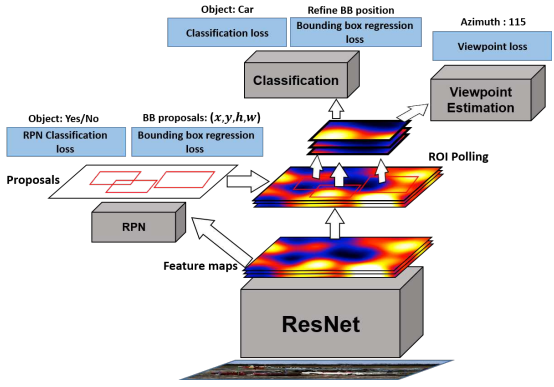


Fig. 3: **Network architecture.** Deep features are extracted by ResNet and passed to RPN to predict bounding boxes. After ROI pooling, they are passed both to the classification head and to the viewpoint estimation head. The output consists of a set of bounding boxes (x, y, h, w), and for each of them—the class of the object within the bounding box and its estimated viewpoint.

ignore viewpoints. Skip connection allow the data to flow directly, without being distorted by pooling, which is known to disregard the inner order of activations.

A viewpoint estimation head is added on top of Faster R-CNN. It is built similarly to the classification head, except for the size of the fully-connected layer, which is 4320 (the number of object classes * 360 angles).

The resulting feature map of ResNet is passed to all the model's components: to the *Region Proposal Network (RPN)* of Faster R-CNN, which predict bounding boxes, to the classification component, and to the viewpoint estimation head. The bounding box proposals are used to define the pooling regions that are input both to the classification head and to the viewpoint estimation head. The latter outputs for each bounding box a vector, in which every entry represents a viewpoint prediction, assuming that the object in the bounding box belongs to a certain class, e.g. entries 0-359 are the predictions for boats, 360-719 for bicycles etc. The relevant section of this vector is chosen as the output once the object class is predicted by the classification head. The final output of the system is a set of bounding boxes (x, y, h, w), and for each of them—the class of the object in the bounding box and object's viewpoint for this class—integrating the results of the classification head and the viewpoint estimation head.

Implementation details: Within this general framework, three issues should be addressed. First, though viewpoint estimation is defined as a classification problem, we cannot simply use the classification head of Faster R-CNN as is for the viewpoint estimation task. This is so since the periodical pooling layers within the network are invariant to the location of the activation in the feature map. This is undesirable when evaluating an object's viewpoint, since different viewpoints have the same representation after pooling that uses Max or Average.

To solve this problem, while still accounting for the importance of the pooling layers, we replace only the last pooling layer of the viewpoint estimation head with a fully connected layer (of size 1024). This preserves the spatial information, as different weights are assigned to different locations in the feature map.

Second, in the original Faster R-CNN, the bounding box proposals are passed to a non-maximum suppression function in order to reduce the overlapping bounding box suggestions. Bounding boxes whose *Intersection over Union (IoU)* is larger than 0.5 are grouped together and the output is the bounding box with the highest prediction score. Which viewpoint should be associated with this representative bounding box?

One option is to choose the angle of the selected bounding box (BB). This, however, did not yield good results. Instead, we compute the viewpoint vector (in which every possible viewpoint has a score) of BB as follows. Our network computes for each bounding box bb_i a distribution of viewpoints $P_A(bb_i)$ and a classification score $P_C(bb_i)$. We compute the distribution of the viewpoints for BB by summing over the contributions of all the overlapping bounding boxes, weighted by their classification scores:

$$viewpoint\_Score(BB) = \Sigma_i P_A(bb_i) P_C(bb_i). \tag{1}$$

This score vector, of length 360, is associated with BB. Hence, our approach considers the predictions for all the bounding boxes when selecting the viewpoint.

Given this score vector, the viewpoint should be estimated. The score is computed by summing Equation (1) over all the viewpoints within a bin. Following [7,8], this is done for $K = 24$ bins, each representing 15° angles,. Then, the bin selected is the one for which this sum is maximized.

Third, we noticed that small objects are consistently mis-detected by Faster R-CNN, whereas such object do exist in our dataset. To solve it, a minor modification was applied to the network. We added a set of anchors of size 64 pixels, in addition to the existing sizes of $\{128, 256, 512\}$ (anchors are the initial suggestions for the sizes of the bounding boxes). This led to a small increase of training time, but significantly improved the detection results (from 74.3% to 77.8% using mAP) and consequently improved the viewpoint estimation.

3.2 Data

In our problem, we need not only to classify objects, but also to sub-classify each object into viewpoints. This means that a huge number of parameters must be learned, and this in turn requires a large amount of labeled data. Yet, labeled real images are scarce, since viewpoint labeling is extremely difficult.

In [12], a creative procedure was proposed: Given a detected and classified object in an image, the user selects the most similar 3D CAD model (from Google 3D Warehouse [22]) and marks some corresponding key points. The 3D viewpoint is then computed for this object. Since this procedure is expensive, the resulting dataset contains only 30K annotated images that belong to 12 categories. This is the largest dataset with ground truth available today for this task.

To overcome the challenges of training data scarcity, Su *et al.* [7] proposed to augment the dataset with synthetic rendered CAD models from ShapeNet [23]. This allows the creation of as many images as needed for a single model. Random backgrounds from images of SUN397 [24] were added to the rendered images. The images were then cropped to resemble real images taken "in the wild", where the cropping statistics maintained that of VOC2012 [25], creating 2M images. The use of this synthetic data increased the performance by $\sim 2\%$.

We further augmented the training dataset, in accordance with Insight 2, in three manners. First, rather than randomly selecting backgrounds, we chose for each category backgrounds that are realistic for the objects. For instance, boats should not float in living-rooms, but rather be synthesized with backgrounds of oceans or harbors. This change increased the performance only slightly.

More importantly, we augmented the training dataset by horizontally flipping the existing real images. Since the orientation of these images is known, they are used within a new loss function to enforce correct viewpoints (Section 3.3).

Finally, we used unlabeled videos of objects, for which we could exploit the coherency of the motion, to further increase the volume of data and improve the results. We will show in Section 3.3 how to modify the loss function to use these clips for semi-supervised learning.

3.3 Loss

As shown in Figure 3, there are five loss functions in our model, four of which are set by Faster R-CNN. This section focuses on the *viewpoint loss* function, in line of Insights 3 & 4, and shows how to combine it with the other loss functions.

Treating viewpoint estimation as a classification problem, the network predicts the probability of an object to belong to a viewpoint bin (bin= 1°). One problem with this approach is that close viewpoints are located in different bins and bin order is disregarded. In the evaluation, however, the common practice is to divide the space of viewpoints into larger bins (of 15°) [12]. This means that, in contrast to classical classification, if the network errs when estimating a viewpoint, it is better to err by outputting close viewpoints than by outputting faraway ones. Therefore, our loss should address a geometric constraint—the network should produce similar representations for close viewpoints.

To address this, Su *et al.* [7] proposed to use a geometric-aware loss function instead of a regular cross-entropy loss with one-hot label:

$$L_{geom}(\boldsymbol{q}) = -\frac{1}{C}\sum_{k=1}^{360} exp(-\frac{|k_{gt}-k|}{\sigma})log(q(k)). \tag{2}$$

In this equation, $\boldsymbol{q}$ is the viewpoint probability vector of some bounding box, k is a bin index, k_{gt} is the ground truth bin index, $q(k)$ is the probability of bin k, and $\sigma = 3$. Thus, in Equation (2) the commonly used one-hot label is replaced by an exponential decay weight w.r.t the distance between the viewpoints. By doing so, the correlation between predictions of nearby views is "encouraged".

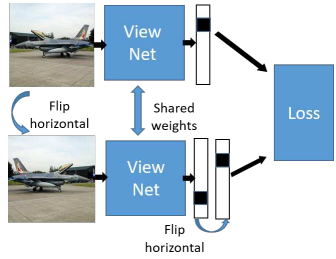


Fig. 4: **Flipped images within a Siamese network**. The loss attempts to minimize the distance between the representations of an image and its flip.

Interestingly, while this loss function was shown to improve the results of [7], it did not improve the results of a later work of [8].

We propose a different loss function, which realizes the geometric constraint. Our loss is based on the fundamental idea of the Siamese architecture [26,27,28], which has the property of bringing similar classes closer together, while increasing the distances between unrelated classes.

Our first attempt was to utilize the contrastive Siamese loss [27], which is applied to the embedded representation of the viewpoint estimation head (before the viewpoint classification layer). Given representations of two images $F(X_1), F(X_2)$ and the L_2 distance between them $D(X_1, X_2) = ||F(X_1) - F(X_2)||_2$, the loss is defined as:

$$L_{contrastive}(D) = (Y)\frac{1}{2}D^2 + (1-Y)\frac{1}{2}\left\{max(0, m-D)\right\}^2. \quad (3)$$

Here, Y is the similarity label, i.e. 1 if the images have close viewpoints (in practice, up to 10°) and 0 otherwise and m is the margin. Thus, pairs whose distance is larger than m will not contribute to the loss. There are two issues that should be addressed when adopting this loss: the choice of the hyper-parameter m and the correct balance between the positive training examples and the negative ones, as this loss is sensitive to their number and to their order. This approach yielded sub-optimal results for a variety of choices of m and numbers/orders.

Therefore, we propose a different & novel Siamese loss, as illustrated in Figure 4. The key idea is to use pairs of an image and its horizontally-flipped image. Since the only difference between these images is the viewpoint and the relation between the viewpoints is known, we define the following loss function:

$$L_{flip}(X, X_{flip}) = L_{geom}(X) + L_{geom}(X_{flip}) + \lambda||F(X) - flip(F(X_{flip}))||_2^2, \quad (4)$$

where L_{geom} is from Equation (2). We expect the L_2 distance term, between the embeddings of an image and the flip of its flipped image, to be close to 0. Note that while previously flipped images were used for data augmentation, we use them within the loss function, in a manner that is unique for pose estimation.

To improve the results further, we adopt the triplet network concept [29,30] and modify its loss to suit our problem. The basic idea is to "encourage" the net-

Loss function (real data)	Score (mAVP24)
Geometric loss, Equation (2)	43.2
Contrastive loss, Equation (3)	42.5
Flip loss, Equation (4)	43.6
Triplet + Geometric loss, Equation (5)+(2)	44.1
Viewpoint loss, Equation (7)	44.4

Table 2: **Results gained by different loss functions.** Equation (7) gives the best performance compared to a variety of loss functions.

work to output similarity-induced embeddings. Three images are provided during training: X^{ref}, X^+, X^-, where X^{ref}, X^+ are from similar classes and X^{ref}, X^- are from dissimilar classes. In [29], the distance between image representations $D(F(X_1), F(X_2))$ is the L_2 distance between them. Let $D^+ = D(X^{ref}, X^+)$, $D^- = D(X^{ref}, X^-)$, and d^+, d^- be the results of applying softmax to D^+, D^- respectively. The larger the difference between the viewpoints, the more dissimilar the classes should be, i.e. $D^+ < D^-$.

A common loss, which encourages embeddings of related classes to have small distances and embeddings of unrelated classes to have large distances, is:

$$L_{triplet}(X^{ref}, X^+, X^+) = ||(d^+, 1 - d^-)||_2^2. \tag{5}$$

We found, however, that the distances D get very large values and therefore, applying softmax to them results in d^+, d^- that are very far from each other, even for similar labels. Therefore, we replace D by the *cosine* distance:

$$D(F(x_1), F(x_2)) = \frac{F(x_1) \cdot F(x_2)}{||F(x_1)||_2 ||F(x_2)||_2}. \tag{6}$$

The distances are now in the range $[-1, 1]$, which allows faster training and convergence, since the network does not need to account for changes in the scale of the weights. For *cosine* distance we require $D^+ > D^-$ (instead of $<$), and consequentially the roles of d^+, d^- in Equation (5) should switch.

A minor trick we apply for softmax to produce the range $[0, 1]$, while resolving convergence issues, is to multiply D by a single trainable scalar, as in [31].

Finally, the viewpoint loss is defined as (with $\lambda = 5$):

$$L_{viewpoint}(X^{ref}, X^+, X^-) = L_{triplet}(X^{ref}, X^+, X^+) + \lambda L_{flip}(X^{ref}). \tag{7}$$

Table 2 shows the gains yielded by different combinations of the loss functions discussed above. The combination of the triplet loss and the flip loss (Equation (7)) results in the best performance.

Putting it all together: Finally, the whole network is trained on the sum of all the loss functions from Figure 3, :

$$L_{total} = L^{RPN}_{classification} + L^{RPN}_{regression} + L^{Classifier}_{classification} + L^{Classifier}_{regression} + L_{viewpoint}. \tag{8}$$

The first four terms are from [16] and the last term is from Equation (7).

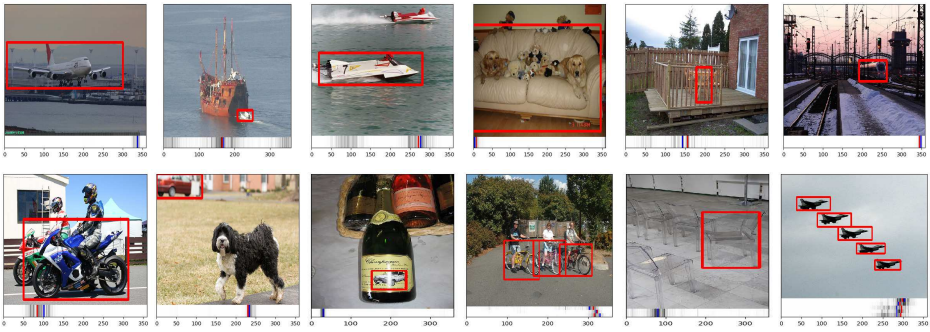


Fig. 5: **Correct viewpoint estimation predictions**. The bar below each image (0°-360°) indicates in blue our highest viewpoint prediction, in red the ground truth, and in black predictions with high confidence. Our prediction and the ground truth fall in the same bin, i.e., within 15° from each other.

4 Results

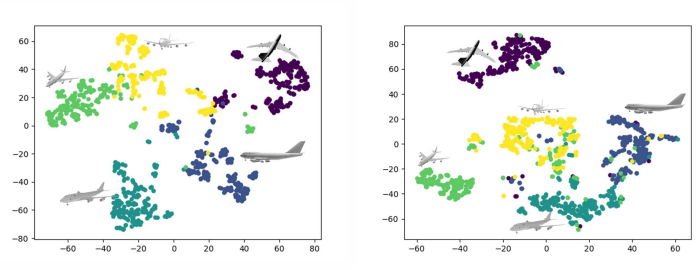
Our evaluation is performed on PASCAL3D+ [12], which contains manually-annotated images from VOC2012 [25] & from ImageNet [32]. All the experiments were conducted using the Keras [33] framework with TensorFlow [34] backend. Figure 5 shows examples of correct predictions made by our model, as well as the detected bounding boxes. The bar below each image indicates in blue our highest viewpoint prediction, in red the ground truth, and in black predictions with high confidence. It can be seen that in most cases our prediction falls in the same bin as the ground truth. Moreover, in most cases the predictions with high confidence (in black) are nicely clustered. The exception is the image of the boats, for which two clusters of 180°-difference are evident. This can be explained by the horizontal near-symmetry of the object.

The common evaluation metric for this problem is the *mean Average View Precision (mAVP)* [12]. Briefly, in AVP, the output from the detector is considered to be correct if the bounding boxes' overlap is larger than 50% **AND** the viewpoint is correct. The AVP is defined as the area under the Viewpoint Precision-Recall (VPR) curve. It is therefore a joint metric both for detection and for viewpoint estimation. Following previous work, we compare our results based on the discrete AVP with $K = 24$ viewpoint bins.

4.1 Training

Our model was initialized with weights from Faster R-CNN, trained on VOC2012 & VOC2007 datasets. The weights of the viewpoint estimation head were initialized with Xavier initialization [35]. Adam optimizer [36] was used with the learning rate set to $lr = 10^{-4}, \beta_1 = 0.9, \beta_2 = 0.999$, unless otherwise specified.

Each training step was performed using a single image, from which we took a mini-batch of 32 region proposals, out of the proposals the network had made.



(a) Triplet loss embedding (b) Geometric loss embedding

Fig. 6: **Embeddings.** Each point represents an image embedding in the feature space (using t-SNE), where color corresponds to the ground-truth viewpoint bin. The triplet loss manages not only to separate the viewpoint bins better than the geometric loss, but also to better reflect the circular nature of the problem.

By default, half of the regions included the objects and half did not; however if the network did not provide enough regions of objects, we padded the mini-batch with more background regions.

We started the training with the synthetic data, followed by the real data. For the synthetic data, we created $\sim 100K$ synthetic images per category. We fixed the weights of the detection & classification network, since we noticed that synthetic data decreased the detection results significantly. We fine-tuned only the viewpoint estimation head, training for $200K$ iterations.

As real data, we used the $22K$ training images from the annotated data of PASCAL3D+. We augmented the data by horizontally-flipped images. We started the training with the weights gained from the synthetic training and fine-tuned the whole network. Our model was trained for $200K$ iterations. Then, we reduced the learning rate by a factor of 10 and continued the training only for the viewpoint estimation head, for $150K$ iterations.

Training with our triplet loss. At every iteration, we randomly selected a class and a reference image from this class. As a positive example, an image from the same class, whose viewpoint is within 5° from the reference, was chosen. As a negative example, we started from "easy" images (from the same class, but faraway viewpoints) and worked our way to more difficult ones. Specifically, for the first 100K triplets, we sampled the distance to the reference from a Gaussian centered at 100° with std of 20° and selected a suitable image. After the loss has stabilized, we sampled from a Gaussian centered at 15° with std of 2°.

Figure 6 shows the 2D embedding of the airplane class, using t-SNE [37], as resulted from the triplet network. Similar viewpoints are better clustered when using our loss than when using the geometric loss of [7]. Moreover, the points are structured in a more "circular" shape, which reflects the circular nature of our problem, as explained in [7]. Thus, the triplet loss not only better separates the embeddings, but also manages to push the features outward.

Training with videos. The key idea behind the use of videos is that though the viewpoint is unknown, we do know that viewpoints of subsequent frames should be similar. To realize this idea, we use the triplet architecture, this time within a semi-supervised learning scheme. We downloaded 100 unannotated YouTube videos that contain objects from our categories, for which it is unknown whether the object appears in a frame or not. All these videos have large motions, such as a landing airplane or bike racing. In addition, 10 videos per class were utilized, each containing a single object that slowly rotates 360°. For them, it is guaranteed that the object appears in all the frames and that all the viewpoints are sampled. Each video is a few minutes long, containing thousands of frames.

At each training iteration, a triplet was chosen, where the reference frame was randomly selected from some video, the positive frame was its adjacent frame (assuming that the viewpoint did not change much) and the negative frame was taken from a later frame in this video. The weights were initialized to the results of the regular triplet loss discussed above. The only labeling performed was the class of the object in the video and an estimation of the gap needed for the negative frame. We note that our viewpoint loss function is a combination of the flip loss and the triplet loss (Equation (7)), yet videos are not associated with labels that allow us to compute the flip loss. Therefore, when using videos, the flip loss term used a random real image, rather than a video frame.

4.2 Comparison with state-of-the-art results

Table 3 "zooms into" the findings of Table 1, showing the gains for the different classes, attributed to the different components of our model. The upper part of the table shows the results of previous works. The middle part shows the results of applying Insights 1,4,5. In particular, when replacing VGG by ResNet and maintaining the same loss and data as [8], the results improve from 36.1 of [8] to 39.5, using both real and synthetic data (and to 37.6 using only real data). By using the geometric loss from Equation (2), the performance is improved to 40.6. When choosing the bin that integrates on the distribution of the viewpoints within it (Equation (1)), instead of choosing the maximum activation bin, the result is further improved to 43.2. This nice improvement can be explained by noting that the bin integral method can be considered as noise reduction, which is beneficial especially for noisy classes, such as bicycles and motorbikes.

The lower part of the table shows the influence of Insights 2,3, assuming that our model uses ResNet/Faster R-CNN, was trained on synthetic & real data and chooses the bin using an integral on the distribution. Our viewpoint loss improves the results by 1.2%; the video data further improves by 1.5%. Overall we achieved improvement of 9.8% compared to the current state-of-the-art results.

We note that different methods improve different categories. For instance, the integration method vastly improves the motorbike and the bicycle classes. In these classes there are many images that contain more than one object from the class, which are very close to one another. Their detected bounding boxes overlap and objects produce "noisy viewpoints" for others. When integrating over all the bounding boxes, some of which do contain a single object, this noise is reduced.

Method	aero	bicycle	boat	bus	car	chair	table	mbike	sofa	train	TV	mAVP24
[7]:AlexNet/R-CNN-Geometry-synthetic+real	21.5	22.0	4.1	38.6	25.5	7.4	11.0	24.4	15.0	28.0	19.8	19.8
[9]: VGG/R-CNN-Classification-real	37.0	33.4	10.0	54.1	40.0	17.5	19.9	34.3	28.9	43.9	22.7	31.1
[8]: VGG/Fast R-CNN-Classification-synthetic+real	43.2	39.4	16.8	61.0	44.2	13.5	29.4	37.5	33.5	46.6	32.5	36.1
Ours: ResNet/Faster-Classification-real	41.6	33.7	20.6	65.3	45.4	17.9	33.8	36	34.5	48.6	36.6	37.6
Ours: ResNet/Faster-Classification-synthetic+real	43.6	37.1	19.9	68.5	48.6	19.8	37.1	34.2	38.2	48.3	39.6	39.5
Ours: ResNet/Faster-Geometry-synthetic+real	43.9	35.4	20.9	70.3	51.5	20.0	38.6	34.0	41.6	50.4	40.0	40.6
Ours: ResNet/Faster-Geometry-synthetic+real-integral	43.5	41.2	23.9	68.4	52.7	22.4	41.9	42.0	44.1	50.3	45.0	43.2
Ours: all the above +Viewpoint loss	46.6	41.1	**23.9**	72.6	53.5	22.5	42.6	42.0	44.2	**54.6**	44.8	44.4
Ours: all the above +Viewpoint loss-video data	**47.7**	**42.5**	23.8	**74.8**	**54.7**	**25.9**	**42.8**	**43.5**	**46.3**	54.6	**47.9**	**45.9**

Table 3: **The number of correctly-estimated viewpoints is 25% higher than SOTA results**, improving from 36.1 of [8] to 45.9, on PASCAL3D+.

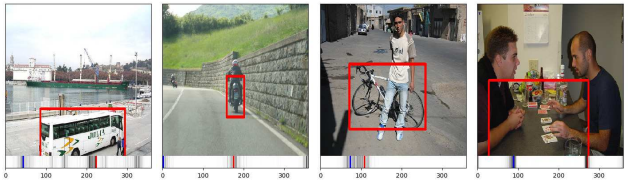


Fig. 7: **Typical false viewpoint estimation cases.** Our predictions for the bus and the motorbike is 180°-opposite; for the bicycle our prediction matches the viewpoint of the handlebar rather than that of the main frame; the table should have two correct viewpoints, of which our viewpoint estimator chose one.

Moreover, the viewpoint distribution in PASCAL3D+ for these classes is more uniform than in most other classes. Hence, our network has no bias and tends to assign probabilities to all the object's symmetries. This is beneficial for the bin integral method, since collecting more information improves prediction.

The flip loss shows improvements mainly for the "rectangular" objects, such as buses, trains and tables. We infer that flipping indeed helps the network in resolving some of the symmetry ambiguities, as desired.

The viewpoint (triplet/flip) loss is mostly beneficial for classes for which the geometric loss errs by 180° (object facing backward/forward), such as airplanes, buses and trains. A possible explanation is that unlike the geometric loss, which relates between close viewpoints through its use of Gaussian weights, the viewpoint loss relates also between faraway viewpoints.

The use of video clips improves the classes for which we had videos that contain almost all viewpoints.

Limitations: Figure 7 illustrates some typical failures. For the bus and the motorbike, the failures are due to backward/forward symmetry—our model predicted the 180°-opposite viewpoint. The false prediction for the bicycle is due to the handlebar position, which is not aligned with the main frame. The table illustrates a case where two viewpoints are equally correct (i.e. there is no front & back to a rectangular table), but our algorithm chose one viewpoint whereas the ground truth is the other.

Estimating all Euler angles: Following [8], throughout this paper only the azimuth is predicted. This is so because of two reasons: (i) Our method is general and can be applied to the other Euler angles, by simply adding a fully-connected layer for each angle; (ii) Unlike the azimuth distribution, the elevation & tilt distributions have little variation (about 85% of the images are within 20°).

Nevertheless, we achieve state-of-the-art results also when all 3 angles are considered. In [7,9], all Euler angles are predicted and the accuracy is measured on the three angles jointly, as follows. (1) *MedErr* calculates the median error between the predicted rotation matrix to the ground truth one (the lower the better) and (2) $Acc_{\frac{\pi}{6}}$ computes the fraction of instances whose predicted viewpoint is within a fixed threshold of the target viewpoint (the higher the better).

On these metrics we gain 23% improvement with *MedErr* (15.6 [9], 11.7 [7], 8.9 ours) and 8.5% improvement with $Acc_{\frac{\pi}{6}}$ (0.76 [9], 0.82 [7], 0.89 ours).

5 Conclusions

This paper has addressed the task of viewpoint estimation of an object in an image. It provides five insights, which regard all the components of the network: the architecture, the training data, the loss function, and the integration of the results.

Based on these insights, a network was designed such that: (i) The architecture jointly solves detection, classification, and pose estimation, using the most advanced CNN for performing the two former tasks. (ii) To handle the shortage in labeled data, the paper proposes to add both videos and flipped images to the training stage. (iii) A novel loss function that takes into account both the geometric nature of the problem, as well as the constraints posed by videos and flipped images, is introduced. (iv) While previous works predicted the viewpoint using the maximum activation, we propose an integration scheme for prediction.

Our network improves the state-of-the-art results for this problem on PASCAL3D+ by 9.8%. The paper carefully analyzes the influence of each component on the overall performance.

Future directions: Our viewpoint estimation is based only on the information within the bounding box. However, information from the full image can be helpful. For instance, the wave direction may assist in determining the boat's viewpoint, or passengers on the platform may indicate the train's direction.

Second, the available dataset should be enhanced. As the improved performance due to our additional data may imply, larger datasets are likely to benefit viewpoint estimation. Moreover, better annotation methods are necessary, as currently some images are falsely annotated, which biases both training and testing. Finally, for certain types of objects (e.g. circular tables or the table in Figure 7), any attempt to define a single ground-truth viewpoint is doomed to fail. Such special cases should be given proper attention.

Acknowledgements: We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPU, as well as the Ollendorff Foundation.

References

1. Huttenlocher, D.P.: Object recognition using alignment. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (1987) 102–111 1
2. Lowe, D.G.: The viewpoint consistency constraint. International Journal of Computer Vision (IJCV) **1**(1) (1987) 57–72 1
3. Lowe, D.G., et al.: Fitting parameterized three-dimensional models to images. IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI) **13**(5) (1991) 441–450 1
4. Huttenlocher, D.P., Ullman, S.: Recognizing solid objects by alignment with an image. International Journal of Computer Vision (IJCV) **5**(2) (1990) 195–212 1
5. Choi, C., Taguchi, Y., Tuzel, O., Liu, M.Y., Ramalingam, S.: Voting-based pose estimation for robotic assembly using a 3D sensor. In: IEEE International Conference on Robotics and Automation (ICRA). (2012) 1724–1731 1
6. Marchand, E., Uchiyama, H., Spindler, F.: Pose estimation for augmented reality: a hands-on survey. IEEE Transactions on Visualization and Computer Graphics **22**(12) (2016) 2633–2651 1
7. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3d model views. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (2015) 2686–2694 1, 2, 3, 4, 6, 7, 8, 11, 13, 14
8. Massa, F., Marlet, R., Aubry, M.: Crafting a multi-task CNN for viewpoint estimation. arXiv:1609.03894 (2016) 1, 3, 4, 6, 8, 12, 13, 14
9. Tulsiani, S., Malik, J.: Viewpoints and keypoints. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2015) 1510–1519 1, 2, 3, 4, 13, 14
10. Penedones, H., Collobert, R., Fleuret, F., Grangier, D.: Improving object classification using pose information. Technical report, Idiap (2012) 1
11. Osadchy, M., Cun, Y.L., Miller, M.L.: Synergistic face detection and pose estimation with energy-based models. Journal of Machine Learning Research (2007) 1197–1215 1
12. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond PASCAL: A benchmark for 3D object detection in the wild. In: IEEE Winter Conference on Applications of Computer Vision (WACV). (2014) 2, 6, 7, 10
13. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR). (2014) 580–587 3
14. Pinheiro, P.O., Collobert, R., Dollár, P.: Learning to segment object candidates. In: Advances in Neural Information Processing Systems (NIPS). (2015) 1990–1998 3
15. Girshick, R.: Fast r-cnn. arXiv:1504.08083 (2015) 3
16. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NIPS). (2015) 91–99 3, 4, 9
17. Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (2015) 2650–2658 3
18. Gkioxari, G., Girshick, R., Malik, J.: Contextual action recognition with r* cnn. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (2015) 1080–1088 3

19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS). (2012) 3
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 (2014) 3
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016) 770–778 3, 4
22. : Google 3D warehouse. http://sketchup.google.com/3dwarehouse 6
23. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 (2015) 7
24. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2010) 3485–3492 7
25. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. International Journal of Computer Vision (IJCV) **88**(2) (2010) 303–338 7, 10
26. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a" siamese" time delay neural network. In: Advances in Neural Information Processing Systems(NIPS). (1994) 737–744 8
27. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2005) 539–546 8
28. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2006) 1735–1742 8
29. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition, Springer (2015) 84–92 8, 9
30. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2014) 1386–1393 8
31. Hoffer, E., Hubara, I., Soudry, D.: Fix your classifier: the marginal value of training the last weight layer. arXiv:1801.04540 (2018) 9
32. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2009) 248–255 10
33. Chollet, F., et al.: Keras. https://github.com/keras-team/keras (2015) 10
34. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems. https://www.tensorflow.org (2015) 10
35. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. (2010) 249–256 10

36. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014) 10
37. Maaten, L.v.d., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research **9** (2008) 2579–2605 11