SymGAN: Orientation Estimation without Annotation for Symmetric Objects

Phil Ammirato
UNC-Chapel Hill

Jonathan Tremblay
NVIDIA

Ming-Yu Liu
NVIDIA

Alexander C. Berg
UNC-Chapel Hill

Dieter Fox
NVIDIA

Abstract

Training a computer vision system to predict an object's pose is crucial to improving robotic manipulation, where robots can easily locate and then grasp objects. Some of the key challenges in pose estimation lie in obtaining labeled data and handling objects with symmetries. We explore both these problems of viewpoint estimation (object 3D orientation) by proposing a novel unsupervised training paradigm that only requires a 3D model of the object of interest. We show that we can successfully train an orientation detector, which simply consumes an RGB image, in an adversarial training framework, where the discriminator learns to provide a learning signal to retrieve the object orientation using a black-box non differentiable renderer. In order to overcome this non differentiability, we introduce a randomized sampling method to obtain training gradients. To our knowledge this is the first time an adversarial framework is employed to successfully train a viewpoint detector that can handle symmetric objects. Using this training framework we show state of the art results on 3D orientation prediction on T-LESS [12], a challenging dataset for texture-less and symmetric objects.

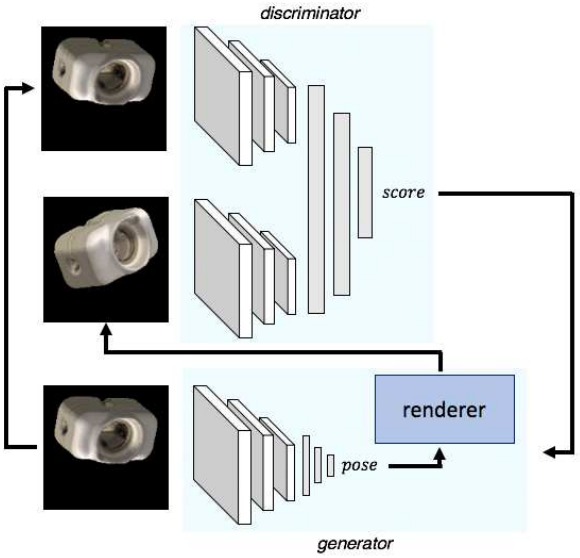


Figure 1. Method overview: The generator, composed of both a pose regressor and a non differentiable renderer, makes a 3D pose prediction and renders the predetermined object at the predicted pose. The discriminator scores how visually similar the poses depicted in the images are. This score is used as training signal for the generator.

1. Introduction

One fundamental problem in any robotics system is knowing the 3D position and orientation (azimuth, elevation and cyclorotation) of objects in the scene, often referred to as 6-DoF (degrees of freedom) pose. Recent advancements in 2d object detection [21, 32] have enabled great progress on estimating position. Recent detectors [43, 46, 38] have been shown to be robust to various factors like occlusion, lighting, clutter, *etc.* Although the problem of object orientation, especially symmetric ones, has received less attention and remains an open challenge.

In general, objects can have varying and complex symmetries, depending on their shape, texture, and occlusions. There exists three different types of symmetries, point (sphere), axes (cylinder) and planes (cube) which can be mixed or partial on any given object. For example, in the case of an untextured mug, where there exists an axis of symmetry on the y-axis, *e.g.*, looking up, has its symmetry disappearing when there is a handle visible. One way to handle object symmetries is to manually specify the symmetries and provide a corresponding training loss [42] or via loss functions that ignore certain aspects of the visual input, such as texture [46, 13]. The former method is labor intensive and non-trivial as certain objects might have partial symmetries. The latter approach has no guarantee that the right representation will be learned.

In this paper we propose a novel approach to detecting the orientation of symmetric objects without the need for explicit symmetry labeling or loss specification. The proposed method only needs a CAD 3d model of the object of interest. The key idea behind our technique is to train a viewpoint regressor using an adversarial training paradigm where the adversarial component learns to provide a loss function that is consistent with an object's symmetries. Similar to generative adversarial networks (GANs)

[9], we train a viewpoint regressor coupled with a 3D renderer, serving as generator[1], simultaneously with a discriminator. The discriminator's goal is to challenge the generator to improve its predictions. Figure 1 shows an overview of our presented method. The generator takes as input a cropped image of the object of interest and returns a predicted pose. The discriminator takes as input two images, the cropped image (the same input used for the generator) and a 3D rendering of the object of interest in the orientation outputted by the generator. The discriminator scores the image pair based on their visual similarity. Since gradients from the rendered images are unavailable, we propose a randomized sampling method to jointly train the discriminator and generator.

Following this section we discuss related work, provide an overview of SymGAN, and experiments on 2D and 3D objects. We summarize our contributions as follows: 1) A new method for training a viewpoint estimator with any kind of object symmetries (including objects without symmetries), 2) A sampling method for training a GAN when gradients are not accessible, and 3) Experiments showing state of the art results on the challenging T-LESS [12] dataset.

2. Related Work

Object pose estimation is a popular problem in the robotics and computer vision communities, and usually consists of 3D localization and/or 3D orientation estimates to form a full 6D pose. This problem has been addressed by many works using classic computer vision algorithms, often some form of template or feature matching [10, 28, 39, 45]. Other methods have applied classic machine learning techniques [36, 2, 1]. Recently many works using deep learning and convolution neural networks (CNN) have been proposed [17, 27, 46, 43, 22, 38]. A natural way to leverage CNN is to directly regress to pose parameters, this approach leverages data labels directly [2, 26, 46, 30, 19]. Other methods have focused on regressing to cuboid vertexes projection which relies on using PnP [8] to retrieve the final pose [38, 43, 17]. In this work we propose a system that also directly regress to the orientation pose (quaternion).

Most of these methods need specific handling for symmetrical objects, such as special labelling [42]. Kehl *et al.* added a classifier for pseudo symmetric objects, but it still requires hand labeling information about the object [17]. Xiang *et al.* defined a loss function similar to the average distance metric (ADI) [46]. This approach excels under objects with very symmetrical shape, the metric matches each point on the 3D model in the predicted pose with the closest point on from the ground truth pose. Hodan *et al.* showed that using ADI can break under self-occlusion, when the object looks symmetric in two views but some hidden part has moved, *e.g.*, the handle of a mug [13]. Moreover the loss does not take into account visual queues, *e.g.,* given an object that is visually dissimilar, but the shape does not change such as a texture cylinder would get label as symmetrical under the ADI loss. Sundermeyer *et al.* implicitly learn a representation from rendered 3D model views using an auto-encoder [35]. At test time a crop of the object is encoded and then compared via nearest neighbor search to a dictionary of encoded poses to retrieve the final orientation. Their method is limited by the discretization of the object poses, whereas our is technically continuous and outputs a pose directly.

Our work is also part of a larger effort to accomplish training on simulated data while applying on real data, this is also known as the reality gap problem. As such our method is trained on rendered images while tested on real images and only necessitates a 3D CAD model of the object of interest, which is often easier to obtain than hand labelling real-world training data. A popular approach to solve this problem is the usage of domain randomization [40, 41] as an inexpensive method to bridge that gap. This method consists of training a model with extreme visual variety so that when presented with a real-world image the model treats it as another visual variation. It has been successfully applied – though usually needing fine tuning or more structure in the randomization to achieve state of the art results [29, 43] – to car detection [41, 29], pose detection [43, 35], vision based robotics manipulation [40, 42, 16], robotics control [3, 37], and more.

Our proposed method leverages unsupervised adversarial training democratized by the popularity of generative adversarial network (GAN) [9]. The method was originally used for image generation but since has been applied to new domains such as imitation reinforcement learning [11]. Ganin *et al.* have used GANs to learn how a program, *e.g.* a canvas stroke generator, can produce valid looking images of characters. Similar to our work, the gradient loss cannot flow from the discriminator to the generator. They approach this problem by leveraging reinforcement learning, *e.g.*, maximizing the discriminator output as reward. In this work we use GAN to retrieve the distribution of valid orientation poses given an image as well as leveraging a randomized sampling method to obtain signal from the discriminator.

3. Method

In this work, our goal is to learn to predict the orientation, also known as viewpoint estimation of objects without any manual symmetries labelling. Please note that we use pose and orientation interchangeably. Figure 1 shows an overview of our proposed method. First, a generator pro-

[1]Please note that we label our technique GAN as the regressor coupled with the renderer act as a generator even though in spirit it is different than the original definition introduced by Goodfellow *et al.*[9]

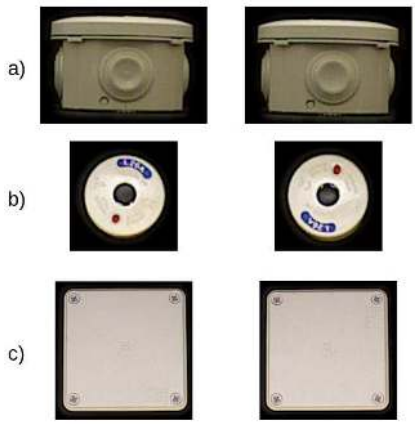


Figure 2. Examples of 'symmetric' objects in the T-LESS[12] dataset. The symmetries range from mostly identical (a), to shape but not texture (b), to very small texture differences (c). This can lead to ambiguities when trying to label data or train a pose estimator.

cesses a cropped image of the object and predicts a quaternion pose. Next a discriminator takes as input two images, the cropped image and a render of the generator's output pose, and outputs how likely the image's poses are a match. In this section we discuss the challenges with respect to viewpoint estimation of symmetrical objects, and the major components needed to train our orientation pose estimator, SymGAN.

3.1. Symmetric Viewpoint Estimation

Estimating the pose of symmetrical object is a non trivial problem, Figure 2 shows two different ground truth poses (row wise) for some objects in T-LESS. In row a) the two poses seem almost perfectly identical in appearance and shape, and it is clear these poses should be labeled as symmetric. In row b) the two poses are clearly visually dissimilar, but metrics like ADI [10] and Visible Surface Discrepancy (VSD) [13], and loss functions based on them [46] will treat these poses as symmetric since the shape is nearly identical. In row c), the poses look almost identical, but a human labeler can see a visual feature in the bottom right corner of the first image (left) moved to the top right corner in the second image (right). This leads to ambiguity in how the symmetry should be labeled. Instead of having to make decisions about what to label, or relying on the shape of 3D models, we develop SymGAN to adaptively learn about visual object symmetries. As such our method is capable of handling ambiguous 'pseudo symmetries' like like Figure 2 c).

3.2. Conditional Generative Adversary Network

SymGAN's goal is to construct a generative model capable of predicting a target pose, p, conditioned by a target image, $\mathbf{x}_t$. In order to predict such target we introduce a black box render, $\mathcal{R}$, that accepts a pose and a CAD model[2], this function renders the CAD model with the input pose. Formally the task is equivalent to learning $p|\mathbf{x}_t$, *i.e.*, where we aim to output the right pose for a given image. The generation process is depicted in Figure 4 1), where a neural network, *G*, takes as input an image, $\mathcal{R}(p)$, and ouptuts a pose, p_g, that is then rendered, $\mathcal{R}(p_g)$.

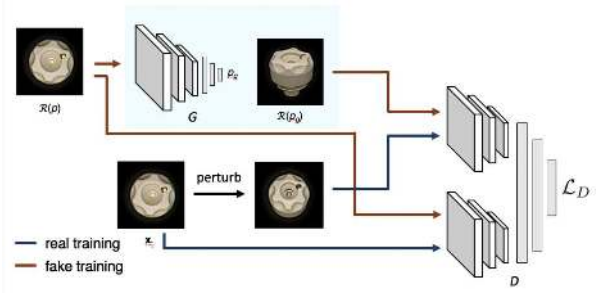


Figure 3. Overview for training the discriminator, training 'fake' images are generated using the generator and shown in dark orange, whereas training 'real' images is depicted in dark blue.

In order to optimize this function (*G*) we employ an adversarial network (*D*) [9]. As such a *generator*, *G*, aims to maximally confuse a *discriminator*, *D*, which is trained to discriminate between $\mathcal{R}(p)$ – images rendered from poses taken from a training set – and render samples generated by the generator output, denoted $\mathcal{R}(p_g)$. In adversarial setting, D is optimized to dissociate real from fake[3] while G is trained to foul D, *e.g.*, producing real like images. As a result the output from the generator gradually becomes more and more similar to the real pose, *e.g.*, $p_g \approx p$. Such as in [7] our method does not require aligned examples from p_g and p compared to other related works [15, 44, 31]. We refer to 'fake' for images generated by *G*'s output and 'real' for images from p.

3.3. Training Procedure

We found that the objective introduced by the original GAN [9] was hard to optimize and thus use L1 loss inspired by Mao *et al.* [23] and metric learning information theory [4], and coupled with zero-centered gradient penalty [33, 24]. In the following we denote the set of training poses as p_d.

Discriminator Here we define the discriminator loss as,

$$\begin{aligned}\mathcal{L}_D \quad = \quad & \mathbb{E}_{p\sim p_d|\mathbf{x}_t}[|D(\mathcal{R}(p),\mathbf{x}_t) - a|] + \\ & \mathbb{E}_{p\sim p_d|\mathbf{x}_t}[|D(\mathcal{R}(G(\mathcal{R}(p))),\mathbf{x}_t) - b|] + Z\end{aligned} \tag{1}$$

where *Z* is the zero-centered gradient penalty [34, 25], *a* is the real data target label and *b* is the fake data target label.

[2]For reading ease we omit it from notation

[3]inputs generated by the generator

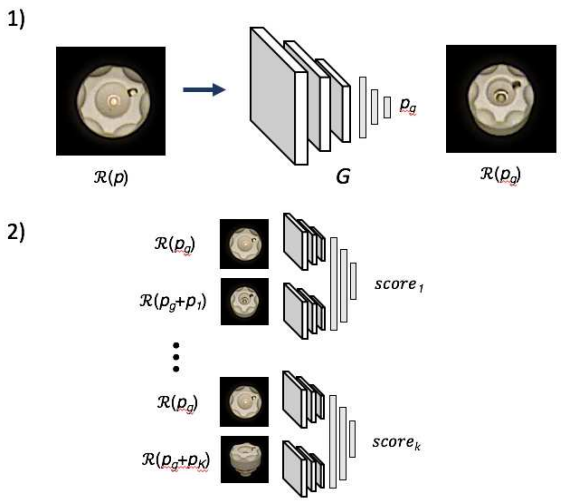


Figure 4. Overview for training the generator 1) we generate a pose, and 2) we sample multiple perturbed poses to find the most valuable as target training.

Figure 3 presents an overview for training the discriminator, 1) We use the generator to output 'fake' images, this is shown by the orange arrows. 2) We use a small perturbation on a rendered sample pose as a 'real' set of images shown by the blue arrows. The 'real' images show poses very close to each other resulting in a low target label. When training 'fake' images, which should show very dissimilar poses, the target is assigned a high value label.

Generator and gradient estimate We hope to maximize the following loss for the generator,

$$\mathcal{L}_G = \mathbb{E}_{p\sim p_d|\mathbf{x}_t}[|D(\mathcal{R}(G(\mathcal{R}(p))),\mathbf{x}_t) - c|] \quad (2)$$

where c is the generator target. Since we cannot take the gradient from $\mathcal{R}$ we introduce a sampling method for training. We first sample K noise vectors from some distribution, *e.g.,* normal or uniform, giving us a set of noise vectors $\{p_1, p_2, \ldots, p_K\} = S$. We use these noise vectors to perturb the predicted poses and update the generator loss as follows,

$$\begin{gathered}\mathcal{L}_G = \mathbb{E}_{p\sim p_d|\mathbf{x}_t}[|G(\mathcal{R}(p)) - p_q|] \\ p_q = G(\mathcal{R}(p)) + \text{argmin}_{p_i\in S}[D(R(G(\mathcal{R}(p)) + p_i), \mathbf{x}_t)]\end{gathered} \quad (3)$$

Figure 4 shows an overview of the training procedure, 1) generates a pose from an image, and 2) samples poses around the predicted pose. We render all of these poses, pair them with the original image, and get scores from the discriminator. From these scores, the discriminator identifies the pose that is the most visually similar to the original. Finally we take that pose as a target for our generator and take a gradient step with respect to that target.

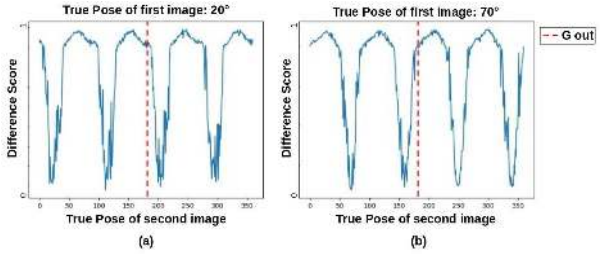


Figure 5. Discriminator's output during training for the solid white 2D square for a fixed input (*(a)* 20° and *(b)* 70°) and a varying one from 0° to 360°.

3.4. Training Details

The adversarial training has been presented, but in practice training GANs can be unstable and challenging [24]. We now go into more details about how we implement a more robust training procedure.

Discriminator In equation 1 the target values, a, for 'real' pairs are based on the distance between the poses of the images, allowing the discriminator to act as a metric function. Larger perturbations lead to larger target values. For 'fake' pairs, target values b are sampled randomly as we cannot be sure how different the poses are, due to unknown symmetries. In practice for a pair of real images a is scaled between 0 and 0.3 based on the perturbation, and b is sampled from a random uniform distribution between 0.8 and 1. To help avoid mode collapse while training, 20% of the time we replace the generator's prediction with a random pose when constructing fake pairs. This ensures the discriminator does not over-fit to the generator's output, which especially early in training can have a narrow range.

Domain Randomization We apply many data augmentations to our training images to allow our model to generalize from synthetic only training data to real test images. We take inspiration from Sundermeyer *et al.* [35] for the bulk of our augmentation. We (1) render synthetic images with random light position, intensity, and specular reflection, (2) insert random backgrounds from Pascal VOC dataset [6], (3) apply random masks to simulate occlusion, and (4) augment the final image with random contrast, brightness, saturation and hue shift.

4. Experiments on 2D Objects

We first apply our system to simple 2D shapes, where we aim to predict a 1D pose that represents the rotation of the shape in the image plane with respect to a fix coordinate frame. This will allow us to better understand how SymGAN learns during training and what its final output looks like.

4.1. Dataset & Training

We investigate 4 shapes: square, square with colored corners, triangle, and decagon; each object have 2, 1, 3, and 5 axes of symmetries respectively. In other words, we can represent the same pose with 4 values, *e.g.,* $0°$, $90°$, $180°$, and $270°$ give the same visual in plane rotation for the square. The shapes are depicted in Figure 6's left column. We generate the 32x32 images as follow: 1) we pick a random scale (0.5 to 1), 2) place the object at a random location, 3) apply a random rotation, 4) paste it on a black background, and 5) add random blur.

The generator and discriminator both use the same architecture. It is composed of 4 convolution layers, with 2 pooling layers, and followed by 1 fully connected layer. We use the Adam optimizer [18] and train for 10,000 iterations with batchsize of 64. Using the same architecture we also trained a pose estimator that directly regress to the pose, we use L2 as loss.

4.2. Results

In order to evaluate this pose predictor we count as success a prediction if it is within one degree of the ground truth while taking into account the symmetries. When training our baseline on the simple square we obtain 10% valid prediction while our proposed method achieves 98%. Our method achieves 96%, 96%, and 82% for the textured square, triangle, and decagon respectively. It is quite intuitive that directly regressing to a multimodal function using L2 would perform worst than a tailored solution. Given the low dimensionality of the in-plane rotation problem, we investigate the behaviour of both the discriminator and generator.

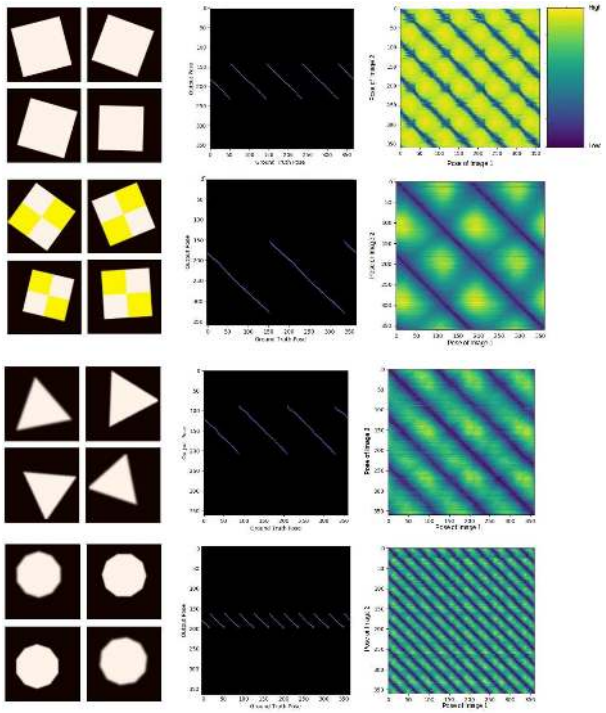


Figure 6. Inputs (first column), outputs from generator (second column), and outputs from discriminator (third column) for our experiments on 2D shapes.

Discriminator Figure 5 shows the discriminator decision boundary when fixing one image – left is set at $20°$ and right is set at $70°$ – and varying the second image rotation from $0°$ to $360°$ (x-axis). The blue line shows the discriminator's output when comparing the input image to a different pose, a low value represents a similar pose and a high value a dissimilar one. This suggests that the discriminator has visually learned to distinguish between the different symmetries. For example, on the left image all symmetric poses to the input, ($\{20°, 110°, 200°, 290°\}$), are assigned low scores, while other non-symmetric poses are assigned high scores elsewhere. The generator's output for these images, in the middle stages of training, is shown by a red dashed line.

As previously described, when training on this input image, we render the object at k random poses centered around the predicted pose. We then pair them with the input image and get difference scores from the discriminator. The pose with the best (lowest) difference score from the discriminator will be chosen as the training target for the generator.

As we can see in the graph, the pose with the best score will be something greater than the generator's output (180). Through many training iterations the generator's output will slowly be pushed towards the valley at 200, where it will eventually converge.

Generator The middle column in Figure 6 shows the generator's output (y-axis) at each ground truth pose (x-axis) after training has converged. A good way to interpret the plot is by drawing a horizontal line across the plot. Take the line $y = 200$ for the first row as example, it crosses the model output at $\{20°, 110°, 200°, 290°\}$. This means that for those four ground truth poses the model outputs 200, which is the desired behavior, since all four poses are indistinguishable from each other. The choice of 200 over any other of the symmetry values is arbitrary and it is a result from the model initialization.

The third column in Figure 6 describes the discriminator's output as a heatmap for all possible pairs of poses, after training has converged. For example, on the second image on the first row, the point $(x = 30, y = 50)$ corresponds to the discriminator's output for a pair of images where the first image has the square at $30°$ and the second image at $50°$. To interpret this plot imagine, again, tracing a horizontal (or vertical) line anywhere on the graph. As such, on the

third image on the first row, if we were to trace $y = 200$ there would be 4 valleys where the discriminator's score is low, *e.g.,* at $\{20, 110, 200, 290\}$, corresponding to the four symmetries on the square. Everywhere else the discriminator's score is high, as it has learned to assign high values to visually dissimilar poses.

5. Experiments in 3D: T-LESS

We next test our system on the 3D orientation prediction problem on the T-LESS dataset [12] from the SixD Pose Challenge [14]. T-LESS consists of 30 object instances most of which have limited texture and some level of symmetry. One challenge with any set of real objects is a varying level of symmetries, which can make manual labeling of symmetries even more difficult and ambiguous.

We use the VSD [13] metric for evaluation, a metric which is robust to shape (but not texture) symmetries and is widely used by other works on this dataset.

5.1. Training Procedure

We make some changes in our training implementation to be more practical for 3D objects. Each object has a reconstructed 3D model available, from which we render our training images. As discussed in Section 3, we have a renderer in our training loop, generating training images on the fly. While this is still possible in the 3D setting, we found pre-rendering training images improved training speed and did not greatly reduce performance. As such, we render 10,000 images per object each with a random pose on a black background, randomly changing the lighting location and intensity per our domain randomization approach.

To simulate the renderer during training, we build a nearest neighbor graph based on each training image's ground truth pose. Then, when we need to generate an image with pose p, we simply query for the training image that is the nearest neighbor to p. When we need to sample K images around some pose p, we simply query for the N, where $N > K$, nearest neighbors of p and sample K of those.

For example, when training the discriminator with a pair of 'real' images, we first pick a random pose, p, and get the training image with the closest pose to p. Follow we query $N = 10$ nearest training images to p, and choose one of them randomly. This gives us a pair of images in which the object has similar pose but not exact. On the other hand, for a pair of 'fake' images, it is important that we sample far away enough from the generator's output to explore the possible pose space, but not so far away that the sampled poses are just completely random. As such, using an output pose from the generator, p, we choose to sample $K = 10$ images randomly from the $N = 200$ nearest neighbors of p. After we fetch the rendered images using the nearest neighbor graph, we apply our other data augmentations in the training loop.

Please note that this process does not limit what the discriminator sees as 'real' pairs. Consider two poses that are far apart in pose space, (p_1, p_2). If (p_1, p_2) are visually ambiguous, $(p_1 + \epsilon, p_2)$ are also visually ambiguous for some small enough ϵ. Therefore $(p_1, p_1 + \epsilon)$ provides similar visual information as (p_1, p_2). $(p_1, p_1 + \epsilon)$ are close in pose space and will be used as a 'real' pair in training, effectively including (p_1, p_2) as a 'real' pair in training. If (p_1, p_2) are not visually ambiguous, then it is correct to not include them as a 'real' pair during training.

5.2. Model Details

Our generator takes as input a 128x128 crop of an object. It first passes the image through a feature extractor based on the FLowNetSimple architecture [5], followed by three fully connected layers to predict the objects pose as a quaternion. The discriminator also uses the same architecture, but only outputs a single score. We train both using Adam with a learning rate of $1e^{-4}$ and batchsize of 32. We stop training when we see performance converge on the training data for the object of interest, usually within 50,000 to 100,000 update iterations. We trained three models per object and keep the one that performed best on the training data to evaluate on the test data.

5.3. Evaluation

In order to evaluate our method on T-LESS we use Visible Surface Discrepancy (VSD) as it is an ambiguity-invariant pose error function that is determined by the distance between the estimated and ground truth visible object depth surfaces. We compare our method to a *baseline* that we trained to directly regress to the object pose and the Augmented Auto Encoder (AAE) from [35]. The former uses the same architecture as the pose predictor in SymGAN but it is trained using annotated real poses from T-LESS and does not have a discriminator or renderer in the loop. The latter is a method trained for 3D orientation retrieval using an auto encoder method.

Our focus is only on the 3D orientation portion of the 6D pose challenge, and so during evaluation we combine our 3D orientation prediction with the ground truth 3D translation. As input to our network, we use the ground truth bounding boxes, and when multiple instances of one object are present in one scene we choose the instance with the highest visibility similar to [35]. We omit showing refined poses using methods such as ICP as we want to focus on the RGB component of our system.

5.4. Results

Figure 9 shows qualitative results randomly selected from T-LESS scene 1, it depicts 4 objects: 02, 25, 29, and 30 shown in blue, yellow, orange and purple respectively. As in the SIXD challenge, we report the recall of correct 6D

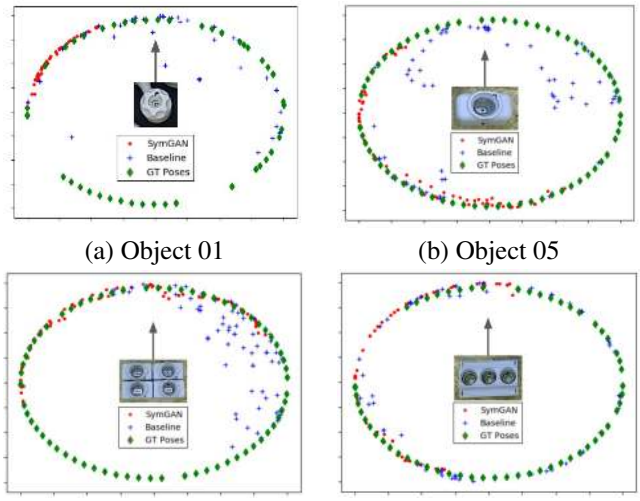
Table 1. Results on T-LESS Dataset T-LESS: Object recall for $err_{vsd} < 0.3$ on all Primesense test scenes. The italic number depict the objects with axes of symmetries.

object	baseline	Sundermeyer [35]	SymGAN (ours)
1	65.0	12.33	**75.5**
2	80.2	11.23	**88.0**
3	**85.2**	13.11	84.0
4	62.2	12.71	**66.2**
5	58.5	66.70	**73.0**
6	**65.5**	52.30	65.1
7	11.8	**36.58**	22.2
8	4.6	22.05	**42.0**
9	20.6	46.49	**47.4**
10	3.6	**14.31**	13.3
11	**66.3**	15.01	66.0
12	40.3	31.34	**49.4**
13	**69.0**	13.60	66.6
14	36.0	**45.32**	34.3
15	54.0	50.00	**58.2**
16	78.0	36.09	**80.6**
17	68.0	**81.11**	54.3
18	**67.0**	52.62	63.0
19	23.5	**50.75**	28.5
20	7.8	**37.75**	12.4
21	35.0	**50.89**	33.3
22	11.0	**47.60**	9.6
23	15.0	**35.18**	17.0
24.0	67.75	11.24	**68.2**
25	60.5	**37.12**	35.0
26	**49.0**	28.33	43.3
27	10.0	21.86	**40.0**
28	43.4	42.58	**54.2**
29	23.0	**57.01**	38.5
30.0	90.3	70.42	**92.0**
avg.	45.74	36.79	**50.74**

object poses at $err_{vsd} < 0.3$ with tolerance set at 20 mm and objects with visibility greater than 10% [14]. When comparing our method to similar RGB based method, we can see that our method, with 50.74 avg., learns a better orientation retrieval than previous work[4] with 36.79 avg. It is also interesting to point out that our baseline, with 45.74 avg. is also stronger.

In Table 1 we have also identified, using italic, the objects that have axes of symmetries, like object 02 (blue) in Figure 9. If we isolate these and compute their *vsd* we obtain 67.60 avg. and 15.8 std. which is better results compared to objects with plane symmetries (non-italic objects) with 35.94 avg. and 18.6 std. When computed on [35] we do not obtain any significant distinction where there method got 32.58 avg. and 23.9 std., and 40.47 avg. and 14.1 std. for axes and planes symmetries respectively.

[4] [35] was state of the art on T-LESS when this paper was written



(a) Object 01 (b) Object 05

(c) Object 08 (d) Object 09

Figure 7. How different models' (SymGAN, our baseline) outputs change as an object is rotated in a plane (approximately). The arrow depicts the orientation of the example image in the center, the object is then rotated 360 degrees around an axis coming out of the page, and the models' 3D pose outputs are project to a plane and plotted. GT means ground truth pose.

This might suggests multiple hypothesis: 1) objects with axes of symmetries are easier to learn as it reduces the number of dimensions when trained with a discriminator, and 2) The discriminator might suffer more of mode collapse on objects with planes of symmetries. Nonetheless, SymGAN learns based on visual appearance of the objects, directly from RGB images, and so does not suffer from relying solely on shape like previous methods.

5.5. Results on Symmetric Views

In this subsection we investigate SymGAN performance on some objects with respect to a specific axis of symmetry. We choose four objects from the TLESS dataset, for each object we select a set of views from the test set such that the views cover a 360 degree rotation around a top down view of the object. This is visualized in Figure 7, at the center of each plot there is an image of the object, the selected test views correspond roughly to rotating that image around an axis coming out of the page. In practice the camera in TLESS is not directly above the objects, but it is very close to it. See the supplementary material for more examples of the views used for this experiment.

We project the 3D predict poses (red and blue for SymGAN and baseline respectively), and the ground truth pose (green), to a 2D plane for easy visualization. The accuracy (object recall for $err_{vsd} < 0.3$) results of the two methods

Table 2. Object recall for $err_{vsd} < 0.3$ on restricted set of views from the test set. The objects and selected views are exactly those visualized in Figure 7. SymGAN is able to outperform the direct regression baseline on these symmetric views by a greater margin than on all test views, illustrating SymGan's effectiveness on symmetric objects.

Object	SymGAN	Baseline
01	93.0	45.8
05	90.2	40.2
08	97.2	15.2
09	83.3	6.9

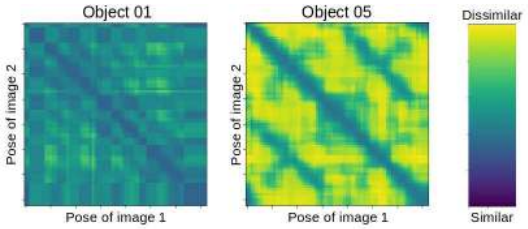


Figure 8. Visualization of discriminator outputs on pairs of images from the restricted testing used in Tabel 2. Similar to the plots in Figure 6, column 3. Object 01's response is smooth and contains low values indicating most of the views are symmetric. Object 05's response shows a clear pattern indication a 180 degree symmetry, a noisier version of the square with colored corners from Figure 6.

on these restricted testing sets are shown in Table 2. Notice that SymGAN is able to achieve over 83% on all the objects, plots (b) (c) and (d) in Figure 7 show rectangular type objects, each with a symmetry at 180 degrees meaning half of the poses are redundant. SymGAN is able to learn this distribution, and it's outputs only cover about half of the ground pose space. The baseline is not able to learn the object's symmetries and outputs a wide range of (often incorrect) poses. Plot (a) shows an object that is symmetrical in every view of this restricted test set. Ideally SymGAN would output a single point, but its output here is fairly restricted.

It should be noted that while none of the objects are exactly symmetric due to some very small features they are generally considered symmetric in the literature, and are symmetric with respect to our model's discriminablilty. As stated in section 3, our method will adapt with the capacity of the model used.

Overall we can conclude that SymGAN does indeed learn objects' symmetries. The performance gain over the baseline on the restricted test set is larger than the performance gain over the entire test set, further validating SymGANs advantage on symmetric views.

6. Discussion & Conclusion

Leveraging adversarial training we have shown that it is possible to train a state of art viewpoint (3D orientation) regressor for untextured symmetrical objects without any hand labelling of object symmetries (or poses). SymGAN does not estimate the full multi-modal distribution over symmetric poses, but is able to improve the point estimate state-of-the-art in 3D orientation. As our method only requires a 3D model and involves a non differentiable black box, we introduced a sampling procedure for retrieving training gradients.

Moreover, we are also interested in learning the full 6D pose using SymGAN, investigating best GAN losses as a function of the type of symmetries, and using a differentiable renderer [20].

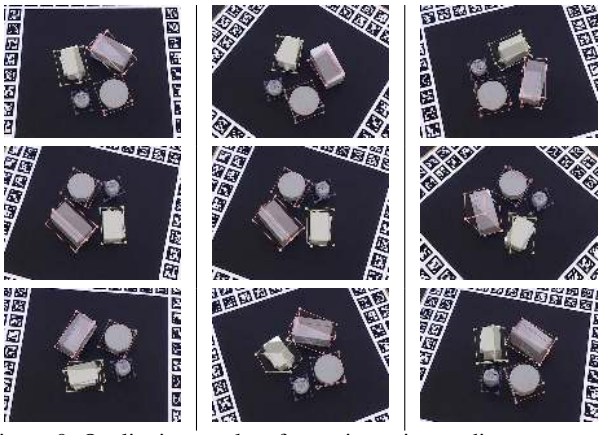


Figure 9. Qualitative results of our viewpoint predictor on scene 1 from T-LESS dataset [12]. The scene is composed of object id. 02, 25, 29, and 30 shown in blue, yellow, orange and purple respectively.

Acknowledgement

The authors would like to thank Thang To for his help in rendering different images. Part of this research was supported by NSF grants 1526367 and 1452851.

References

[1] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *ECCV*, 2014.

[2] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, and C. Rother. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *CVPR*, 2016.

[3] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *arXiv:1810.05687*, 2018.

[4] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.

[5] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. Smagt, D. Cremers, and T. Brox. FlowNet:

Learning optical flow with convolutional networks. In *ICCV*, 2015.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[7] Y. Ganin, T. Kulkarni, I. Babuschkin, S. Eslami, and O. Vinyals. Synthesizing programs for images using reinforced adversarial learning. 2018.

[8] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, Aug. 2003.

[9] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *Advances in neural information processing systems (NIPS)*, 2014.

[10] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient response maps for real-time detection of textureless objects. *PAMI*, 34(5):876–888, 2012.

[11] J. Ho and S. Ermon. Generative adversarial imitation learning. In *NIPS*, 2016.

[12] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. In *WACV*, 2017.

[13] T. Hodan, J. Matas, and S. Obdrzalek. On evaluation of 6d object pose estimation. In *ECCVW*, 2016.

[14] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari10, T.-K. Kim, J. Matas, and C. Rother. Bop: Benchmark for 6d object pose estimation. In *European Conference on Computer Vision (ECCV)*, 2018.

[15] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[16] S. James, A. J. Davison, and E. Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. In *arXiv:1707.02267*, 2017.

[17] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In *ICCV*, pages 1521–1529, 2017.

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[19] A. Kundu, Y. Li, and J. M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *CVPR*, 2018.

[20] T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen. Differentiable monte carlo ray tracing through edge sampling. In *SIGGRAPH Asia*, 2018.

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.

[22] F. Manhardt, W. Kehl, N. Navab, and F. Tombari. Deep model-based 6d pose refinement in rgb. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[23] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *ICCV*, 2017.

[24] L. Mescheder, A. Geiger, and S. Nowozin. Which training methods for gans do actually converge?, 2018.

[25] L. Mescheder, A. Geiger, and S. Nowozin. Which training methods for gans do actually converge? In *arXiv 1801.04406*, 2018.

[26] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká. 3D bounding box estimation using deep learning and geometry. In *CVPR*, 2017.

[27] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká. 3D bounding box estimation using deep learning and geometry. In *arXiv 1612.00496*, 2016.

[28] K. Pauwels, L. Rubio, and E. Ros. Real-time pose detection and tracking of hundreds of objects. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(12), 2016.

[29] A. Prakash, S. Boochoon, M. Brophy, D. Acuna, E. Cameracci, G. State, O. Shapira, and S. Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *arXiv:1810.10093*, 2018.

[30] M. Rad and V. Lepetit. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In *ICCV*, 2017.

[31] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text-to-image synthesis. In *ICML*, 2016.

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[33] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann. Stabilizing training of generative adversarial networks through regularization, 2017.

[34] L. A. N. S. Roth, K. and T. Hofmann. Stabi- lizing training of generative adversarial networks through regularization. In *Advances in Neural Information Pro- cessing Systems (NIPS)*, 2017.

[35] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *ECCV*, 2018.

[36] D. J. Tan, N. Navab, and F. Tombari. 6d object pose estimation with depth images: A seamless approach for robotic interaction and augmented reality. 2017.

[37] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In *CoRL*, 2018.

[38] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6D object pose prediction. In *CVPR*, 2018.

[39] H. Tjaden, U. Schwanecke, and E. Schmer. Real-time monocular pose estimation of 3D objects using temporally consistent local color histograms. In *ICCV*, pages 124–132, 2017.

[40] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017.

[41] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *CVPR Workshop on Autonomous Driving (WAD)*, 2018.

[42] J. Tremblay, T. To, A. Molchanov, S. Tyree, J. Kautz, and S. Birchfield. Synthetically trained neural networks for learning human-readable plans from real-world demonstrations. In *ICRA*, 2018.

[43] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *CoRL*, 2018.

[44] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018.

[45] P. Wohlhart and V. Lepetit. Learning Descriptors for Object Recognition and 3D Pose Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[46] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *RSS*, 2018.