Style2NeRF: An Unsupervised One-Shot NeRF for Semantic 3D Reconstruction

James Charles[1]
http://www.jjcvision.com
Wim Abbeloos[2]
wim.abbeloos@toyota-europe.com
Daniel Olmeda Reino[2]
daniel.olmeda.reino@toyota-europe.com
Roberto Cipolla[1]
cipolla@eng.cam.ac.uk

[1] Machine Intelligence Lab
Department of Engineering
University of Cambridge
Cambridge, U.K.

[2] Toyota Motor Europe

Abstract

We present Style2NeRF, an unsupervised model for one-shot recovery of 3D pose, shape and appearance of symmetric objects. Style2NeRF contains a transcoder which disentangles 2D representations from pretrained StyleGANs, then maps them to a semantically editable 3D NeRF generator. As such, the generative NeRF inherits StyleGAN's expressiveness and image editing properties, translating them to 3D. We make four key contributions: (i) We provide a novel model to accurately estimate an object's 3D pose, shape and appearance without any human supervision during training; (ii) We show how to map between semantically meaningful 2D and 3D representations using a novel disentangled generative NeRF; (iii) we introduce the *pose and viewpoint ambiguity* problem (suffered by existing 3D GAN methods) and propose a solution improving pose estimation accuracy; (iv) Finally, via transfer learning, we show our model can be trained on real car images where the pose distribution is unknown. Style2NeRF outperforms the state-of-the-art on the CARLA cars dataset as well as a fully supervised model for the task of car pose estimation on ShapeNet-cars and a new dataset of real car images.

1 Introduction

An essential computer vision task in robotics and scene understanding is that of reconstructing 3D objects from single-view RGB images. This problem is ill-conditioned and very hard for computer vision systems. Without prior knowledge there is insufficient information in a single image to recover the 3D object. Machine learning helps in this case as strong priors can be learnt from labelled data and there have been many recent advancements [2, 14, 15, 19, 30, 37]. Often, 3D shape models are constructed offline and used as priors to help constrain the problem. However, these models are difficult to obtain, limited and either require 3D scans or the labour of 3D artists [8, 21, 40]. Some approaches intend to learn the 3D representation from the images themselves or refine the models [2, 15, 19, 37], with all approaches incorporating some form of differentiable rendering.

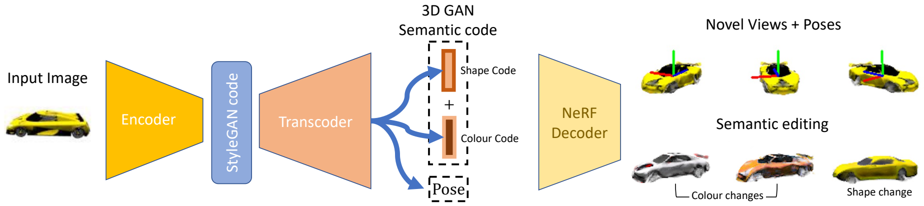


Figure 1: **Style2NeRF:** Our model predicts the 3D pose of a car from a single image while also producing a semantically editable neural radiance field. The model forms a mapping between a pretrained 2D and 3D GAN via a transcoder and uses self-supervision for finetuning. No human supervision is required for training and no pose training labels are necessary.

In recent years neural radiance fields (NeRFs) have become another increasingly popular method of representing a scene in 3D due to their high fidelity. This has spurred a number of works to look at recovering NeRFs from a single image (one-shot NeRF) [18, 24, 29, 32, 35, 38]. Most methods require strong camera pose supervision from multi-view images, either hand or machine annotated, or synthetically derived. However, this paper deals with the even harder setting whereby no labels are provided. This means priors on shape, appearance and pose must be learnt from image information alone. Generative adversairal networks (GANs [11]) have shown potential to learn these priors unsupervised. For instance, StyleGAN [13] representations have been shown to disentangle pose, shape and fine detail naturally, a property which has been used to help lift objects to 3D [12, 17, 28, 33, 39], these methods are 3D aware, but lack multi-view consistency. A recent method EG3D [7] does produce a StyleGAN based NeRF generator which is multi-view consistent. However, this method does not lift images to 3D.

Prior works have also looked at self-supervised pose estimation [22, 25] where no pose labels are provided. However, our method is also designed for shape reconstruction, viewpoint point synthesis and 3D object editing. Our method is also multi-view consistent compared to [25] and does not require matching pairs of training images as in [22].

Up to now only one other such method, Pix2NeRF [4], proposes recovering a NeRF in the above mentioned unsupervised setting. One drawback is that their approach relies upon a known pose distribution of the training data. Similar to their approach, our method also builds upon the work of π-GAN [6]. However, our conditional NeRF is shown to transfer well from synthetic images (where a pose distribution is known) and then capable of training on real images without requiring a known pose prior. Our method is also semantically editable and has an improved inductive bias leading to greater pose estimation performance for symmetric objects. Thus, we believe that our approach is the first unsupervised, conditional and semantically editable NeRF.

Our method called Style2NeRF, shown in Figure 1, leverages rich high level information learnt from 2D StyleGANs and encodes an input image to this latent space. A transcoder maps the the latent vector to camera viewpoint and a 3D semantic code for controlling the NeRF based generative adversarial network (GAN [10]). This code is split into a shape code for globally manipulating the volume density and colour code for globally altering RGB values. The result is a model which can: (i) recover the input pose of the object, (ii) render novel views, (iii) extract 3D shape via volume density querying and (iv) edit the NeRF in semantically meaningful ways. All this without requiring any labelled data for training.

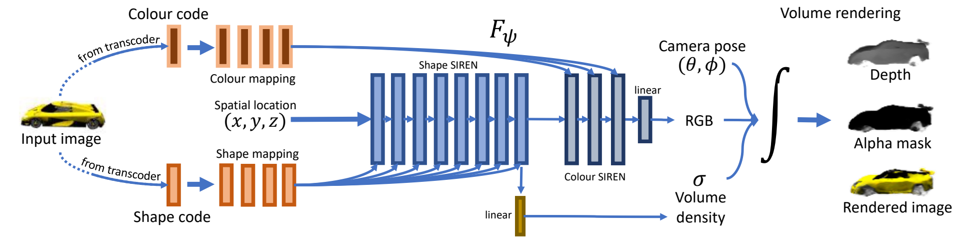


Figure 2: **NeRF decoder:** Architecture of the NeRF decoder illustrated during inference using an example input image.

In summary, we make the following contributions: (i) a new unsupervised conditional one-shot NeRF with object editing properties; (ii) a 2D to 3D GAN transcoder for stylised lifting of images to 3D; (iii) an improved 3D inductive bias for more accurate pose estimation, and (iv) we demonstrate state-of-the-art performance on existing datasets.

2 Method

This section provides an overview of our system and describes in detail the architecture for Style2NeRF. Our system lifts single-view images of pre-segmented objects to 3D in a single forward pass of a neural network. The architecture is broken down into three main stages, as shown in Figure 1: (i) An image is encoded to a StyleGAN latent representation; (ii) from this the transcoder produces a pose and disentangled latent vector of a NeRF based GAN, (iii) finally, a NeRF decoder maps the latent vector to parameters controlling a conditional NeRF model via feature modulation. The final 3D model can be edited by manipulating either the StyleGAN code or the shape and colour components of the 3D GAN latent vector. We next discuss each part of the method in detail.

2.1 StyleGAN Encoder

At the foot of our system is a StyleGAN image encoder giving rise to a rich set of semantically meaningful features. In detail, StyleGAN is a convolutional neural network which maps a latent vector $z \in Z$ drawn from a normal distribution to a realistic image. The latent vector z is first mapped to an intermediate code $w \in W$ which is then transformed to $w^+ = (w_1, w_2, \ldots, w_L) \in W^+$ by L affine transformation layers (one layer per w_i). Once trained, these layers naturally control certain image attributes. Earlier layers have been found to represent camera pose with later layers controlling shape, texture and finer details. When provided with a source image, the task of the encoder is to recover the w^+ which feeds into StyleGAN to produce a close approximation to the source. This process is known as GAN inversion and for this work we use a residual based encoder, named ReStyle [1].

2.2 NeRF Decoder

We represent 3D objects implicitly using a conditional NeRF similar to the π-GAN model of [5]. Our model however. is designed to have a disentangled latent code controlling shape and colour separately.

Architecture. The full architecture of our NeRF F_ψ is illustrated in Figure 2. As input, F_ψ takes a spatial location (x, y, z) and produces an RGB colour and volume density σ. Using volumetric integration and stratified ray sampling one can render an image, an alpha mask and depth map from any viewpoint [23]. Our NeRF does does not use view direction (as is typical in NeRFs for modelling viewpoint effects) as we found this can cause the model to more easily learn degenerate solutions (Figure 4). A shape SIREN [34] first processes (x,y,z) to output volume density. Next, the penultimate feature layer of this network feeds into a second smaller colour SIREN to produce an RGB colour.

Conditioning the NeRF. F_ψ is conditioned on a latent vector $p \in P$ (drawn from a normal distribution) formed from a concatenation of a shape code s and colour code c, both of 256 dimensions. Object colour and shape can be altered by manipulating c and s separately (see Figure 7(a)). These codes (after running through a mapping network) modulate two separate corresponding SIREN networks.

Pose estimation. Style2NeRF strives to recover the NeRF and camera viewpoint which generated an input image. To recover object pose, the inverse of the camera viewpoint is used. However, this requires all generated objects to be canonicalised and oriented in the same way within the NeRF volume, giving rise to the pose and viewpoint ambiguity problem.

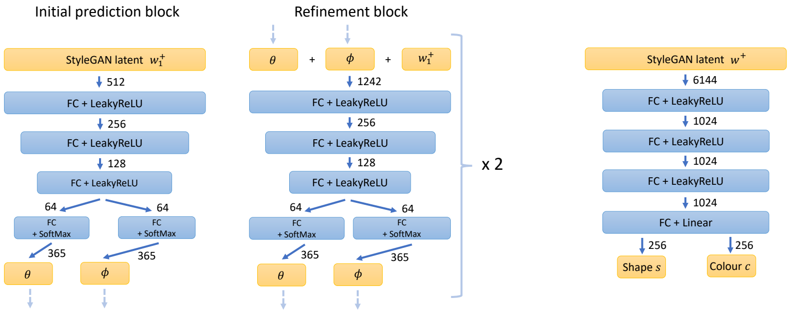
Pose and view point ambiguity. GAN based NeRFs [4, 5, 6, 12] trained without pose labels are free to generate objects at any orientation within the NeRF volume. This means the same viewpoint of an object e.g. front of a car, can be rendered from different camera viewpoints in the NeRF volume. Model regularisation tends to restrict the extent of the ambiguity e.g. it is easier for the model to generate cars in the same orientation, however, there is no direct enforcement. Thus, recovering object pose by camera pose inversion would not be accurate enough for pose estimation. Therefore, we solve this problem by introducing a strong symmetric inductive bias as well as a pose constraint when optimising.

Symmetric inductive bias. By restricting our method to 3D objects with a single plane of symmetry, we force objects in the NeRF volume to align the plane along a specific set of axes. This is done during rendering, rather than shooting straight rays through the volume, instead rays are reflected off the yz-plane. Such a strong inductive bias causes generated objects to precisely align along their planes of symmetry, locking one axis of rotation. To help fix the remaining rotations we introduce a pose constraint (explained in Section 2.4).

2.3 Transcoder

It has been observed in prior works that altering viewpoint by directly manipulating the style code is possible [28, 33]. However, the results are not multi-view consistent. Our decoder remedies this with a transcoder, trained to further disentangle view-point from w^+ and produce a latent code for controlling a multi-view consistent generative NeRF. The transcoder is constructed from two neural networks: (1) a camera pose regressor and (2) a network to map between the latent spaces of a StyleGAN and the 3D GAN based NeRF.

Camera regressor. The camera of the NeRF is assumed to lie on the surface of a sphere with objects centered at it's origin. The camera viewing angle is fully determined by the rotations θ (azimuth) and ϕ (elevation) around the vertical and horizontal axes, respectively. The camera pose regressor is a neural network which processes the first layer latent code w_1 to predict a view direction. An initialisation block makes the first rough 'guess', this guess is



(a) Pose Regressor (b) Style mapping

Figure 3: **Transcoder:** The transcoder consist of two networks: (a) The camera pose regressor, and (b) The style mapping network which converts the full w^+ StyleGAN code to a 3D shape s and colour c code. Both networks consist of stacks of fully connected (FC) layers and activation functions are shown. Further details can be found in the text.

fed to a refinement block which updates the guess by predicting a residule. The refinement block is repeatable and we found two iterations sufficient. Details can be seen in Figure 3(a). The initialisation and and refinement blocks are trained as angle classifiers and repeating allows the model to make a choice between equally probable modes. Angles are defined by 365 equally spaced bins between 0 and 2π.

Style mapping. The style mapping network is a transform $S : W^+ \rightarrow P$ between 2D StyleGAN latent codes and 3D codes governing the response of the NeRF decoder. The architecture is fully described in Figure 3(b).

2.4 Training and losses.

Our model is trained to reconstruct input images using three loss types: (i) reconstruction losses, (ii) an adversarial loss and (iii) a pose constraint. Training happens in two phases: (1) Decoder pretraining, to initialise parameters of our NeRF decoder and (2) Finetuning, where the transcoder and generative NeRF are optimised jointly. The StyleGAN encoder is trained and fixed separately.

Adversarial loss. In the pretraining phase, our NeRF decoder is trained following a similar procedure as π-GAN. Latent shape and colour codes along with camera poses are sampled and rendered images made to progressively "fool" a discriminator. We use the same discriminator architecture as the original π-GAN model. The adversarial loss provided by the discriminator is denoted as L_{adv}. During pretraining an object mask loss and pose constraint are also applied.

Object mask loss. An object mask loss is used to force the rendered alpha masks to match the 'true' mask of the object. This ensures improved shape recovery under restricted training viewpoints. Also it stops the NeRF learning occluders/floaters which is dense matter the NeRF paints with background colour, spoiling the underlining shape. At train time, a pretrained segmentation network is used to segment the NeRF render and this is used as ground truth. The object mask loss L_{mask} is formulated as the cross entropy between the rendered alpha and the predicted segmentation.

Pose constraint. To further enforce NeRF generated objects align well, the principle vector of the NeRF volume is calculated using a differentiable SVD (we use the native implementation in PyTorch). The normalised dot product between this vector and the z-axis of the NeRF coordinate system is used to form the pose constraint loss L_{pose}.

Reconstruction losses. After pretraining, the model is finetuned end to end (other than the encoder). Reconstruction losses can now be used with the NeRF latent codes and poses provided by the transcoder, rather than sampled. Two losses are used: a photometric L2 loss between the pixelwise values of the input and output denoted as L_{photo} and a VGG [36] perceptual loss, denoted as L_{perc}.

Training summary. The objective function for decoder pretraining is then forumlated as: $L_{\text{pre}} = L_{\text{mask}} + L_{\text{adv}} + L_{\text{pose}}$, note that no loss weighting is used. For fine tuning, half the batch uses the reconstruction loss L_{recon} and poses are inferred. But, because the reconstruction losses on their own can cause degenerate solutions and billboard effects, the other half of the batch contains generated images with sampled poses and we apply the adversarial loss L_{adv}, without reconstruction losses.

3 Experiments

Our method is evaluated on three datasets where we measure accuracy of pose estimation, quality of reconstruction and generative performance. Details of the datasets and the evaluations metrics used are provided below.

3.1 Datasets

Pose estimation performance is evaluated on two datasets, SRN-Cars and RealCars, consisting of car images along with their corresponding camera poses. For evaluating generative performance and reconstruction quality we use CARLA.

SRN-Cars. The SRN-Cars dataset contains renderings from 3514 cars sampled from 3D Warehouse with a train/test split across instances [8]. While each model is rendered from 50 random views per object instance, only those views from the top hemisphere are retained for training. The standard test split for benchmarking is left unchanged and consists of 251 views sampled from each car instance based on an Archimedean spiral in the top hemisphere.

RealCars. The RealCars dataset is constructed by sampling 12000 images of segmented car instances from the CompCars dataset. The off-the-shelf PointRend [20] method is applied to each image to segment the car and set the background white. The images are cropped with a square and downsampled to 128 by 128 images. Cropping is centred according to an ellipse fitted to the segmentation mask. Crop size is set so that once downsampled, the scaled minimum axis of the ellipse is on average the same length as ellipses fit to segmentation masks from SRN-Cars. For testing, 400 car images are hand labelled with camera azimuth and elevation by visually adjusting a projected 3D bounding box to the image.

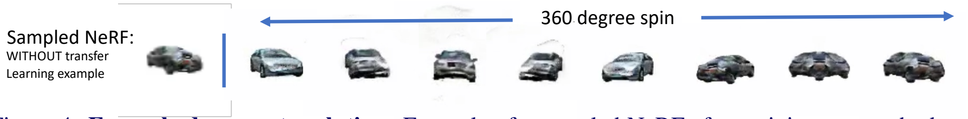


Figure 4: **Example degenerate solution.** Example of a sampled NeRF after training our method on RealCars *without* transfer learning. Notice the billboard effect: different cars are rendered as small images directly in-front of the camera, appearing as if multiview consistency is lost.

CARLA. The CARLA [9] dataset was produced by rendering 16 car types from the CARLA driving simulator at random viewpoints (upper hemisphere only) and textures to produce 10k training images. There is no testing set for this dataset as its intended purpose is for training GANs.

3.2 Evaluation metrics

The following metrics are used durring evaluation:

Pose accuracy. Performance of camera pose estimation is measured as the difference in degrees of rotation from ground truth, separately for azimuth and elevation. When evaluating pose qualitatively, we show the pose of the car (inverse of the camera pose) as a set of coordinate axes. The yellow axis points vertically, the blue axis points towards the car rear and the red axis points out the right side of the car. See the "Novel Views + Poses" element on Figure 1 for an example. Pose estimates from our method are native to the NeRF volume, so before evaluation, a global rotation is applied to offset them to the ground truth (GT) dataset coordinate system. As is usual for self-supervised pose estimation methods [25], we use a sample of 100 labelled images from the GT dataset to learn this bias. In our case, this is simply the median rotation difference.

Image fidelity. The reconstruction quality of model rendered images is evaluated using generative metrics: Inception Score (IS) [31], Frechet Inception Distance (FID) [16], and Kernel Inception Distance (KID) [3] as in prior works [5, 32]. Following the protocol of [4], performance of our approach is also measured in a conditional and unconditional setting.

3.3 Baselines

Our method is compared to the state-of-the-art for 3D aware generative models on the CARLA dataset: HoloGAN [26],GRAF [32] and π-GAN [5]. We also evaluate against Pix2NeRF [4], the only model we know of (other than our own) which is generative, conditional and unsupervised and serves as our direct comparison. For the task of pose estimation we train a strong baseline regression network on SRN-Cars to directly predict elevation and azimuth, fully supervised, using all training data. The network is a ResNet34 backbone with two fully connected layers, one each for elevation and azimuth, to produce 365 different angle classes (as in the last layers of Style2NeRF's camera regressor).

3.4 Training on SRNCars

For pretraining we use the progressive training strategy of π-GAN [5]. Using a resolution of 32x32 with a learning rate of 4e-4 both for the generator and discriminator, with batch size

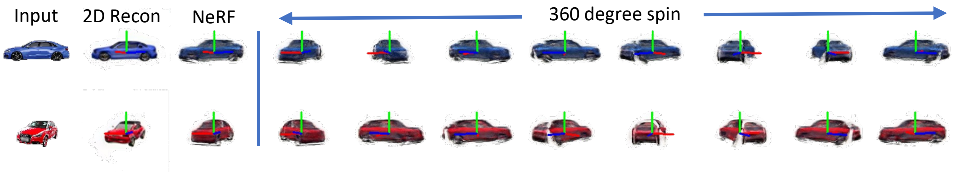


Figure 5: **NeRF recovery on RealCars:** Single-view 3D reconstruction on example images from RealCars, layout is as described in Figure 6.

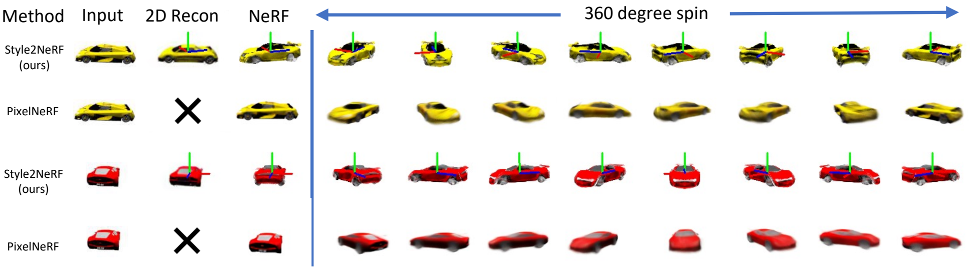


Figure 6: **NeRF recovery on SRNCars:** Single-view 3D reconstruction on example images from SRN-Cars. A comparison agains pixelNeRF is shown. Column 1: Input view, column 2: 2D reconstruction from StyleGAN code (our method only), column 3: NeRF rendering from original viewpoint and column 4-11: NeRF rendering from sampled views of a 360 spin. Inferred poses from our method are shown as a red (right side), green (up) and blue (rear) coordinate axis.

30 and 23 samples per ray. Learning rate is dropped to 2e-4 and then 1e-4 after 10k and 100k iterations, respectively. Pretraining is stopped at 150k iterations. Finetuning is the same, but with batch size set at 15 and number of samples per ray at 24 and a transcoder learning rate of 1e-4 throughout.

3.5 Training on RealCars

When training 3D aware generative models on data without pose labels, it has been found important to know the underlying camera pose distribution to avoid degenerative solutions [4], see Figure 4. Some methods seek to learn the distributions [27] but this can be difficult to tune. For Style2NeRF we can use StyleGANs ability to generate out of domain images and the Restyle encoder's ability to invert them. Thus, after pretraining on SRN-Cars (known pose distribution) our model transfers to RealCars (unknown pose distribution) naturally and can then be finetuned using reconstruction losses. The transfer works well and results can be seen in Figure 5 (more in supplementary). The training schedule here is the same as for SRNCars.

4 Results

NeRF recovery. Example results of Style2NeRF applied to test input images from SRN-Cars is shown in Figure 6 (more in supplementary). The 2D reconstruction is from the StyleGAN encoder, the NeRF column shows the result after transcoding and rendering to the same viewpoint. Novel views synthesised by the NeRF are shown as samples from a 360 degree spin around the NeRF's volumetric origin. Also shown are the ground truth and predicted car poses overlaid as coordinate axes on the corresponding images. We compare qualitatively against pixelNeRF which has full access to pose labels during training and

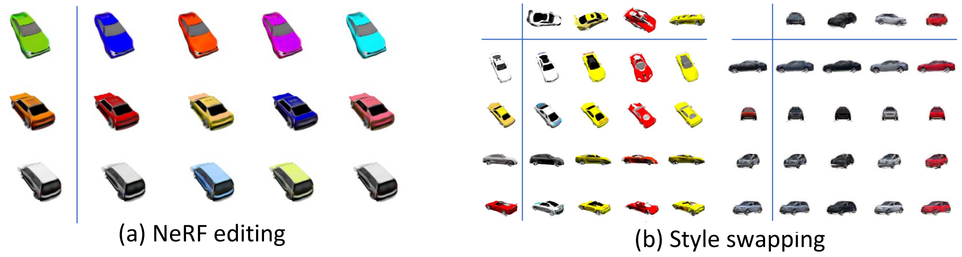


Figure 7: **Object editing:** (a) First column: Cars generated by sampling NeRF codes. Second to forth columns: re-sampling color codes c only, notice how shape remains unchanged. (b) First column and row: Embedded cars into NeRF space. Second to forth column: Style transfer by re-placing latent code of last 4 layers of style code and re-embedding to NeRF using Style2NeRF transcoder.

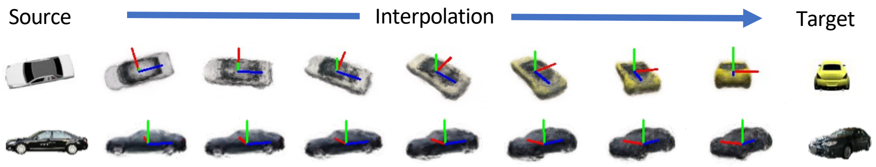


Figure 8: **Interpolation:** Linear interpolation between 3D semantic codes of a source and target image. Upper row is from the SRNCars dataset and lower row is from the RealCars dataset.

at inference for producing the spin images. In comparison it can be seen that pixelNeRF, although producing good results under similar viewpoints, lacks the detail of Style2NeRF which can hypothesis strong structural detail in unseen views. Our method is also able to infer the pose of the vehicle which pixelNeRF cannot. A similar visualisation is shown for a model trained on RealCars, with more examples in the supplementary.

Object editing. One can edit lifted objects using Style2NeRF either by manipulating the 3D GAN or StyleGAN codes, we show examples of both methods in Figure 7(a) and Figure 7(b) respectively. Both codes have semantic properties, in Figure 7(a) editing the colour of cars without altering the shape is demonstrated. In Figure 7(b) the 2D StyleGAN code is used for style transfer from cars in the top row to cars in the first column, restyling cars without effecting shape.

Interpolation. As a demonstration of Style2NeRF's well behaved latent space, illustrated in Figure 8 are linear interpolations between the 3D GAN latent codes and poses inferred on a source and target image. Notice the smooth change in colour, shape and pose between source and target with car objects produced at every step of the interpolation and rotated sensibly (more examples shown in supplementary).

Pose estimation. Pose estimation accuracy is evaluated on all test images from SRN-Cars and RealCars. Angular absolute mean error is reported in degrees in Table 1 for elevation (Elev.) and azimuth (Azi.). Surprisingly, Style2NeRF outperforms the fully supervised baseline for both elevation and azimuth on SRN-Cars and has better azimuth estimates on RealCars, while being very competitive in elevation. We hypothesis that this is because Style2NeRF is trained as a generator and during finetuning the transcoder can therefore observe generated cars in addition to those in the train set. An ablation of Style2NeRF without the symmetric inductive bias and pose constraint (Style2NeRF w/o con.) shows larger errors, indicating these additions alleviate the 'pose and viewpoint' ambiguity problem.

SRNCars - Rotation error (degrees)

Pose regressor	Elev.	Azi.	min-Azi.
Baseline (ResNet34)	14	48	13
Style2NeRF w/o con.	10	85	16
Style2NeRF (ours)	**9**	**45**	**8**

RealCars - Rotation error (degrees)

Pose regressor	Elev.	Azi.	min-Azi.
Baseline (ResNet34)	**1**	77	41
Style2NeRF w/o con	6	114	32
Style2NeRF (ours)	3	**70**	**19**

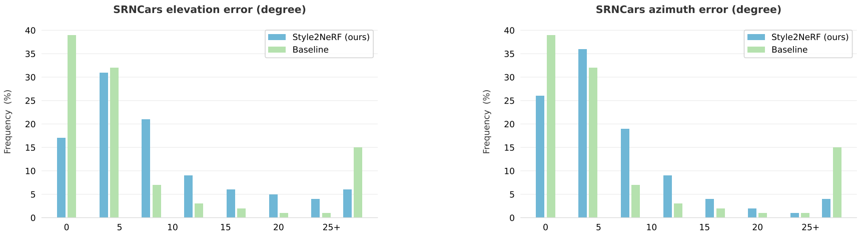


Table 1: **Pose estimation evaluation** on SRNCars and RealCars datasets with rotation errors shown in degrees. Bar charts show distribution of rotation errors on SRNCars.

	64 × 64			128 × 128		
Method	FID ↓	KID ↓	IS ↑	FID ↓	KID ↓	IS ↑
HoloGAN [26]	134	9.70	-	67.5	3.95	3.52
GRAF [32]	30	0.91	-	41.7	2.43	3.70
π-GAN [5]	13.59	0.34	3.85	29.2	1.36	4.27
Pix2NeRF unconditional	10.54	0.37	3.95	27.23	1.43	4.38
Pix2NeRF conditional	12.06	0.44	3.81	38.51	2.37	3.89
Style2NeRF (ours) unconditional	**9.29**	**0.33**	**3.98**	**21.93**	**1.03**	**4.57**
Style2NeRF (ours) conditional	11.03	0.43	3.82	35.84	2.00	3.86

Table 2: Quantitative results on CARLA [9].

All methods struggle with front to back confusion of cars, resulting in what appears to be noisy pose estimates for azimuth. Therefore, we show front to back confusion is the true cause by reporting the minimum azimuth error (min-Azi.) after flipping the pose from front to back. This results in an only 8 degree average error for our method on SRN-Cars. For RealCars this is a 19 degree azimuth error, far superior to the baseline 41 degree error, showing our method can generalise better across datasets. The distributions of the elevation and minimum azimuth error is also shown for SRN-Cars below Table 1. This reveals that when the baseline works well, pose estimates are more accurate than Style2NeRF. However, as cases become more difficult Style2NeRF quickly becomes more reliable.

Image fidelity. Results of reconstruction and generative performances on CARLA are show in Table 2. For the conditional and unconditional mode of evaluation our method outperforms the state-of-the-art by a significant margin. Our approach performs significantly better in all cases across all metrics.

5 Conclusions

In summary, we introduce Style2NeRF, an unsupervised single view NeRF method for recovering the pose, shape and appearance of symmetric objects. When evaluating on SRN-Cars and our newly introduced RealCars dataset, our model outperforms a standard fully supervised model. When evaluating on the CARLA cars dataset Style2NeRF beats the state-of-the-art across all metrics. We show our model can generalise well from synthetic to real datasets via transfer learning and that the NeRFs produced are semantically editable.

References

[1] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *CVPR*, 2021.

[2] Benjamin Biggs, Oliver Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out? 3d animal reconstruction with expectation maximization in the loop. In *ECCV*, 2020.

[3] Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2021.

[4] Shengqu Cai, Anton Obukhov, Dengxin Dai, and Luc Van Gool. Pix2nerf: Unsupervised conditional p-gan for single image to neural radiance fields translation. In *CVPR*, 2022.

[5] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. *arXiv preprint arXiv:2012.00926*, 2020.

[6] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. *arXiv preprint arXiv:2112.07945*, 2021.

[7] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022.

[8] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[9] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Y. Bengio. Generative adversarial networks. In *NeurIPS*, 2014.

[11] Ian J Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv preprint arXiv:1312.6082*, 2013.

[12] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021.

[13] Ligong Han, Sri Harsha Musunuri, Martin Renqiang Min, Ruijiang Gao, Yu Tian, and Dimitris Metaxas. Ae-stylegan: Improved training of style-based auto-encoders. In *WACV*, 2022.

[14] Paul Henderson, Vagia Tsiminaki, and Christoph H Lampert. Leveraging 2d data to learn textured 3d mesh generation. In *CVPR*, 2020.

[15] Philipp Henzler, Jeremy Reizenstein, Patrick Labatut, Roman Shapovalov, Tobias Ritschel, Andrea Vedaldi, and David Novotny. Unsupervised learning of 3d object categories from videos in the wild. In *CVPR*, 2021.

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

[17] Sungmin Hong, Razvan Marinescu, Adrian V Dalca, Anna K Bonkhoff, Martin Bretzner, Natalia S Rost, and Polina Golland. 3d-stylegan: A style-based generative adversarial network for generative modeling of three-dimensional medical images. In *Deep Generative Models, and Data Augmentation, Labelling, and Imperfections*, pages 24–34. 2021.

[18] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. *arXiv preprint arXiv:2109.01750*, 2021.

[19] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018.

[20] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020.

[21] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *TOG*, 2015.

[22] Octave Mariotti, Oisin Mac Aodha, and Hakan Bilen. Viewnet: Unsupervised viewpoint estimation from conditional generation. In *ICCV*, 2021.

[23] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[24] Norman Müller, Andrea Simonelli, Lorenzo Porzi, Samuel Rota Bulò, Matthias Nießner, and Peter Kontschieder. Autorf: Learning 3d object radiance fields from single view observations. In *CVPR*, 2022.

[25] 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 *CVPR*, 2020.

[26] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *ICCV*, 2019.

[27] Michael Niemeyer and Andreas Geiger. Campari: Camera-aware decomposed generative neural radiance fields. In *3DV*, 2021.

[28] Xingang Pan, Bo Dai, Ziwei Liu, Chen Change Loy, and Ping Luo. Do 2d gans know 3d shape? unsupervised 3d shape reconstruction from 2d image gans. *arXiv preprint arXiv:2011.00844*, 2020.

[29] Konstantinos Rematas, Ricardo Martin-Brualla, and Vittorio Ferrari. Sharf: Shape-conditioned radiance fields from a single view. In *ICML*, 2021.

[30] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *CVPR*, 2019.

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, 2016.

[32] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *NeurIPS*, 2020.

[33] Yichun Shi, Divyansh Aggarwal, and Anil K Jain. Lifting 2d stylegan for 3d-aware face generation. In *CVPR*, 2021.

[34] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020.

[35] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d scene representation and rendering. In *arXiv:2010.04595*, 2020.

[36] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *CVPR*, 2020.

[37] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *CVPR*, 2020.

[38] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021.

[39] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *ICLR*, 2021.

[40] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *CVPR*, 2017.