Decomposing Image Generation into Layout Prediction and Conditional Synthesis

Anna Volokitin Ender Konukoglu
Luc Van Gool
Computer Vision Laboratory
ETH Zürich
{voanna, kender, vangool}@vision.ee.ethz.ch

Abstract

Learning the distribution of multi-object scenes with Generative Adversarial Networks (GAN) is challenging. Guiding the learning using semantic intermediate representations, which are less complex than images, can be a solution. In this article, we investigate splitting the optimisation of generative adversarial networks into two parts, by first generating a semantic segmentation mask from noise and then translating that segmentation mask into an image. We performed experiments using images from the CityScapes dataset and compared our approach to Progressive Growing of GANs (PGGAN), which uses multiscale growing of networks to guide the learning. Using the lens of a segmentation algorithm to examine the structure of generated images, we find that our method achieves higher structural consistency in latent space interpolations and yields generations with better differentiation between distinct objects, while achieving the same image quality as PGGAN as judged by a user study and a standard GAN evaluation metric.

1. Introduction

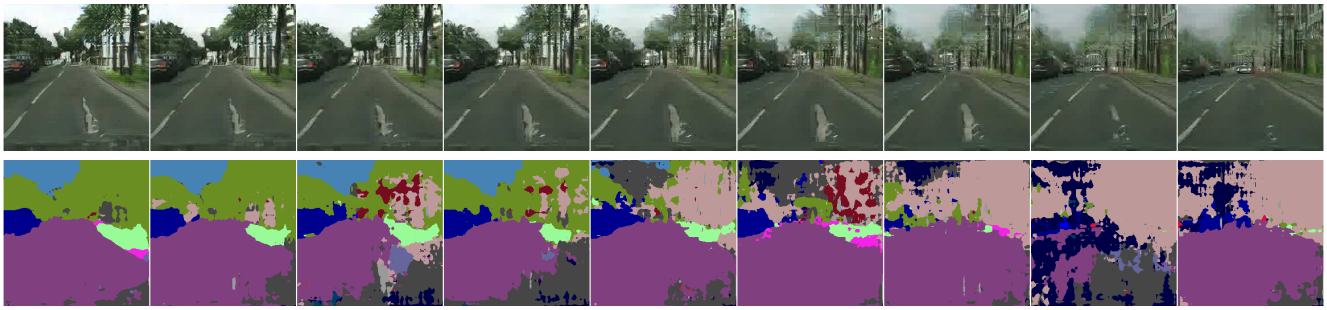
Recently, Generative Adversarial Networks (GANs) [9] have made it possible to synthesise sharp, realistic images from random noise. Much progress has been made in generating higher and higher resolution images, by methods such as Progressive Growing of GANs (PGGAN) [16] and BigGAN [3], that developed innovations to stabilise the training larger and deeper networks.

Despite these advances, challenges remain, in particular in generating images that do not contain one centered object. Often local textures are convincingly modelled while global structure is incoherent. This can lead to a lack of clear separation in the appearance of different generated object classes. To illustrate this, we show interpolated PGGAN generations of street scenes and their DeepLab segmentations [4] in Fig. 1. Large changes in the semantic labels from one image to the next imply that, in some sense, this method has learned to generate patches of "scene-like" texture, without separating image regions into discrete objects. Transitions between images in the sequence end up being transitions between texture patches rather than natural changes in scene composition, unlike a real video sequence and its corresponding segmentation shown in the same figure.

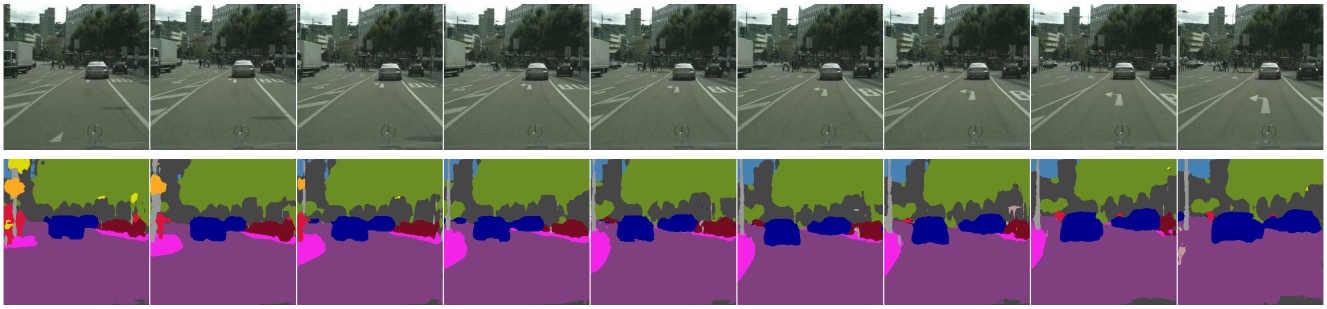
We argue that this lack of separation in object textures can be mitigated by providing an internal representation of the scene structure, thus making allowing the model to generate individual objects more easily. In this work, we decompose the generation process into two stages. First, a GAN generates a semantic segmentation mask that acts as the internal representation of the scene structure. The segmentation mask is then translated into an image using a conditional generation approach. Both PGGAN and our proposed method guide the optimisation of networks during network training, although in our case we use segmentation masks rather than sequentially growing the networks. Fig. 1 shows that this difference in guiding leads to better object separation in the generated images.

We implement the proposed approach as a combination of two existing techniques. To generate segmentation masks, we train a Wasserstein GAN [1] with a DCGAN [21] architecture, extended with two upsampling layers. To texture these segmentation masks, we use pix2pix [14]. We validate the proposed approach using the CityScapes dataset [6], which provides both images and pixel-level segmentation masks. We compare the proposed approach with both the basic GAN without guidance during optimisation and the PGGAN.

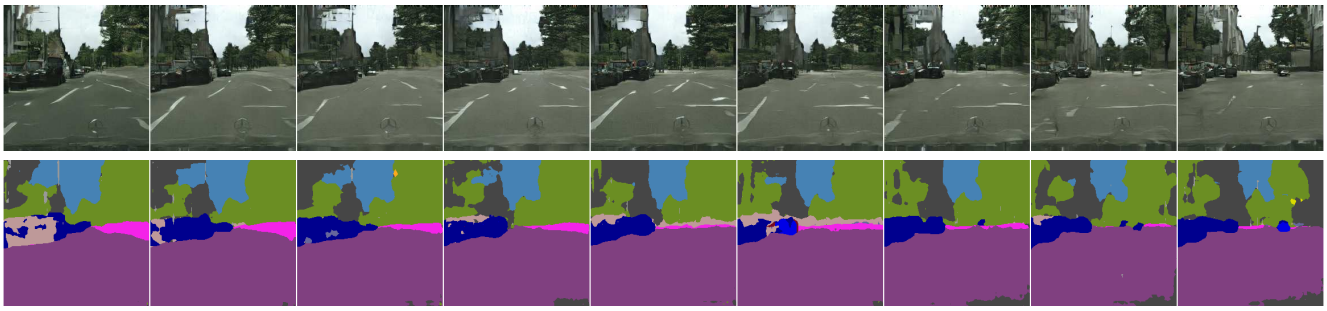
The main findings of this paper as follows: Our method achieves equal image quality in generating multi-object scenes as PGGAN, as measured by a user study and by



Progressive Growing of GANs (PGGAN). top: generated images, bottom: DeepLab segmentations



Real video sequence. top: real images, bottom: DeepLab segmentations



Our method. top: generated images, bottom: DeepLab segmentations

road sidewalk building wall fence tunnel pole traffic light traffic sign vegetation

terrain sky person rider car truck bus train motorcycle bicycle

Figure 1. For each of the 3 rows, top: an interpolation of generated images, bottom: semantic segmentations from DeepLab. The PGGAN-generated images exhibit scene textures corresponding to erratic mixtures of object classes. Our method generates a segmentation map as an intermediate step, and is able to clearly distinguish and place objects with smooth changes during interpolation.

the Frechèt Inception Distance (FID) metric [11]. Furthermore, we use segmentation networks as a tool to probe the structure of generated images and show that our proposed method interpolates in a more semantically meaningful way than PGGAN, and generates images with more clearly differentiated objects.

2. Related work

The main idea of this paper, which is decomposing scene generation into semantic maps and image to image translation, has been very nicely explored in concurrent work [2].

Image generation Some works have proposed a hierarchical approach to image generation. LapGAN [7] trains multiple generators and discriminators to operate at different levels of the laplacian image pyramid. [13] use a hierarchy of stacked generators and discriminators. Recently, impressive results have been shown by growing the size of the generator and discriminator by [16], and extended to make use of style conditioning in StyleGAN [17].

Other breakthroughs include BigGAN [3], which does not explicitly use multiscale training, and instead demonstrates that larger models trained with larger minibatch sizes benefit optimisation, and the Self-Attention GAN [26], which provides a module that can account for long range pixel dependencies.

Methods that split the generation of images into stages include the Layered Recursive GAN [25] that splits the generation of images into background and foreground. [18] generate images part by part using multiple generators.

Generating segmentation maps with a GAN has been explored by [8].

Conditional image generation Another relevant line of work is conditional image generation, in particular works using segmentation maps. Pix2pix [14] has shown that image to image translation can give very realistic results in going from a segmentation mask to a real image. Recent improvements, such as CRN [5] and pix2pixHD [23] have shown extensions to higher resolutions. [20] have also shown very convincing translations from segmentation maps to images using a semi-parametric approach. Improvements have also been shown by making use of spatial normalisation in SPADE [19].

Factoring generation into structure and appearance prediction [24] presents work in the same vein as ours. It decomposes the generation into style and structure, where structure takes the form of surface normals.

Aside from image generation conditioned on other images (*e.g.* segmentation maps), another approach is to condition on natural language. [15] generate scene layouts by conditioning on scene graphs and translate these using CRN[5]. A similar approach makes use of conditioning on text directly [12]. In video forecasting, first predicting the future poses of humans and then generating frames has proposed by [22].

3. Method

The proposed approach is composed of two steps: generation of segmentation masks and filling them in with more detailed patterns. The overview of the approach is illustrated in Fig. 2.

Generating segmentation masks with GANs We train a GAN to generate segmentation maps. We use the objective of the Wasserstein GAN [1], with gradient penalty [10]. We use a modified version of DCGAN [21]. The original architecture goes up to 64×64 resolution. To bring our generated segmentation maps up to 256×256 resolution, we add two upsampling layers, each of which have 64 feature channels. Our generator has 5.4 M trainable parameters.

For the discriminator, we use the same approach as described in PatchGAN [14]. We keep the DCGAN discriminator, which operates on 64×64 patches of the segmentation maps, and average over different patches. Our discriminator has 4.4M parameters.

Conditional image synthesis To render the segmentation map, we train a pix2pix [14] with 11.4M parameters. We did not train our models jointly. This work demonstrates an effective combination of existing tools.

Dataset We use the CityScapes dataset [6], because it contains varied scenes and pixelwise semantic annotations. The dataset provides 22000 training images annotated with coarse segmentations, and 3000 annotated with fine segmentations. Examples are shown in Fig. 3.

We take a 1024×1024 crop of the original 1024×2048 images, and downscale it to 256×256. Although this type of cropping does reduce the amount of variation in the scenes, we find that this simplified task is still sufficiently challenging to showcase problems with the generation of multi-object scenes.

4. Experimental settings

We evaluate several configurations for our model. At the output of the generator, we experiment with two alternatives. The first is to encode segmentation maps using the RGB colour scheme used to visualise the CityScapes dataset. Therefore, the output of the generator is simply an RGB image. The second is to use tensors of shape `num classes×width×height`, where the class of every spatial location is encoded as a one-hot vector. In this case, the output of the generator is `num classes`-dimensional and we use a soft-max as the output non-linearity.

Additionally, we experimented with upsampling in the generator using either transposed convolutions or nearest-neighbour upsampling, followed by a convolutional layer.

Finally, we also compare results in training with the coarse and fine annotations provided by the CityScapes dataset.

The training settings, e.g. the learning rate, were the same as in the DCGAN paper. All segmentations are computed using a pretrained DeepLabv3 model provided by [4].

4.1. Other methods

To show that the difficulties in capturing the distribution of multi-object scenes with a GAN, we train a model to directly generate scenes from CityScapes. We use the DCGAN architecture with two upsampling layers and refer to this method as DCGAN*. We used a learning rate of 0.00002.

We also compare our method to Progressive Growing of GANs [16] (PGGAN). As mentioned in the introduction, this method constrains and guides the optimisation by jointly growing the generator and discriminator during training. The generator of this model has 23M parameters. We train the model using the provided settings, until the model has seen 150000 images.

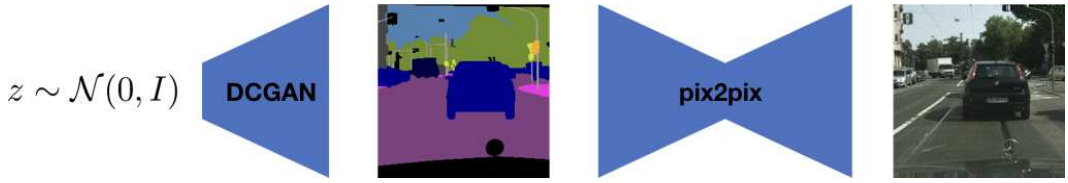


Figure 2. Overview of image generation using segmentation masks as intermediate representations.

Figure 3. Example segmentation maps from CityScapes, finely and coarsely annotated

Since PGGAN does not have the benefit of supervision with intermediate segmentation maps, our comparison is not between two like things. However, the experimental results may still be informative in showing that intermediate guidance with semantic segmentations is useful for generating multi-object scenes with clearly differentiable objects.

5. Results

5.1. Qualitative Evaluation

Example generations from different methods are shown in Fig. 4. Apart from the DCGAN* image samples, all generated image samples have similar quality. Segmentation maps generated by the different variants of our method are indeed all similar to real segmentation maps. Nonetheless, the object shapes are not perfectly realistic, e.g. our method predicts an uneven sidewalk and strange arrangements of traffic signs in (f). This does not prevent realistic translations through pix2pix. Another observation is that our method seems to place fewer pedestrians or cars in the middle of the road compared to real images. We see little difference in the quality of one-hot-encoded and RGB-encoded segmentation masks.

5.2. User Study

To evaluate the realism of generations, we asked Mechanical Turk workers to pick the image "that looks most like a real photograph of a street scene". In each comparison, one image was from our method, and one from PGGAN. Numbers are averaged over 100 comparisons, done by 25 workers each. Table 5.1 shows that users rate PGGAN and variants of our method equally on realism.

5.3. Fréchet Inception Distance

We report the Fréchet Inception Distance [11]. In this metric, features for both the real and the generated images are extracted from the pool_3 layer of an Inception network trained on ImageNet. The distance is then the Wasserstein-2 distance between Gaussians fitted to both classes. We report the values in Table 5.1. Both PGGAN and our methods have lower FID scores than our baseline of directly generating images with DCGAN. Images rendered from generated coarse annotations have FID scores similar to those of PGGAN, whereas the scores for images rendered from our generated fine annotations are higher. This could be due to a lack of fine annotations for training.

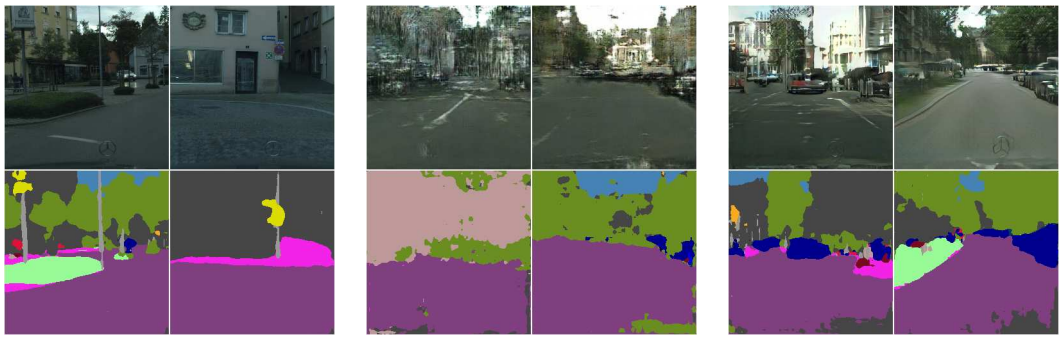
5.4. Latent interpolations

We assume that short video sequences can be approximated by linear paths in the latent space of a good generative model. To check whether a model has this property, we can map the beginning and end of a video sequence to their latent representations, and then interpolate to reconstruct the intermediate frames.

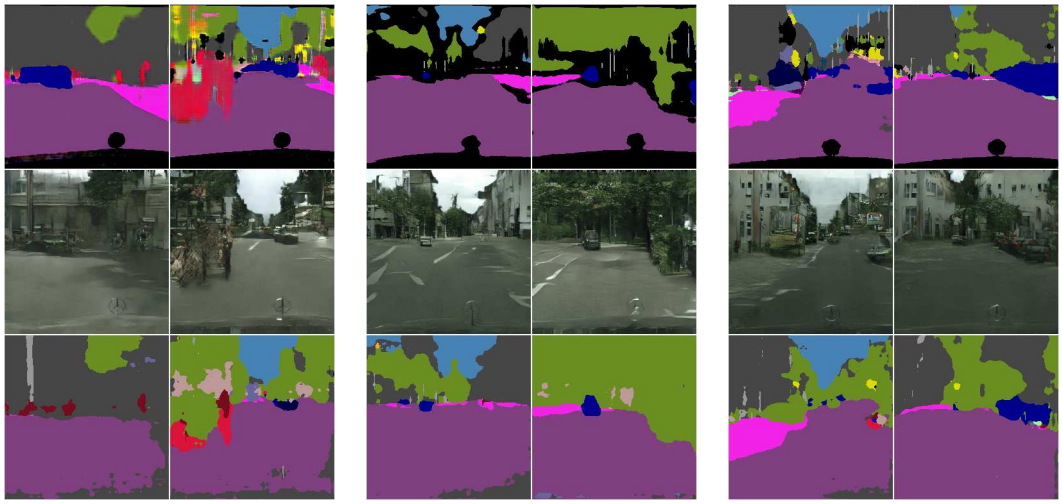
We evaluate our method by segmenting the first and last video frames, and then finding the latent vectors that approximately reconstruct these segmentations. We linearly interpolate between the first and last latent vector to generate intermediate segmentation maps, and render these with pix2pix. To recover the latent representations of segmentation maps generated using our method, we start with a batch of 64 latent vectors and minimise the cross entropy between the generated and target segmentation by optimising the latent vector using using Adam with a learning rate of 0.01 for 100 iterations, and pick the best one. We found that computing the softmax of our generator output (after the output softmax) before computing cross-entropy helped the optimisation. For this experiment we use our method trained on fine annotations with one-hot encoding and upsampling via transposed convolutions.

The results for a 25-frame sequence are shown in Fig. 5.3. Segmentation masks of interpolations created by our method remain quite faithful the structure of the video.

For reference, we adapt this interpolation procedure also to PGGAN, by interpolating between the latent vectors that reconstruct the first and last frames. We then segment these



(a) real images (b) DCGAN* (c) PGGAN



(d) ours, fine RGB seg. (e) ours, coarse one hot seg. nearest upsampling (f) ours, fine one hot seg. transposed conv

Figure 4. Sample generations. Top section: image samples and their segmentations using DeepLab. Bottom section: samples generated by variants of our method. Top row: GAN - generated segmentation masks, middle: filled-in translations of segmentation masks, bottom re-segmentations using DeepLab. The quality of the generated images is similar across the methods, except for DCGAN*, which is worse. In the generated fine annotations, many small objects have incorrect shapes for their class, such as sidewalks and pedestrians.

Method	annot.	seg. repr.	upsampling	Fréchet Inception Distance ↓ better	% preferring method over PGGAN ↑ better
PGGAN				63.90	
DCGAN*				302.36	23.9
Ours	coarse	onehot	tr. conv	56.37	49.3
Ours	coarse	onehot	nearest	58.43	49.4
Ours	coarse	RGB	tr. conv	58.33	51.6
Ours	fine	onehot	tr. conv	76.12	52.7
Ours	fine	onehot	nearest	96.33	49.4
Ours	fine	RGB	tr. conv	83.15	51.4

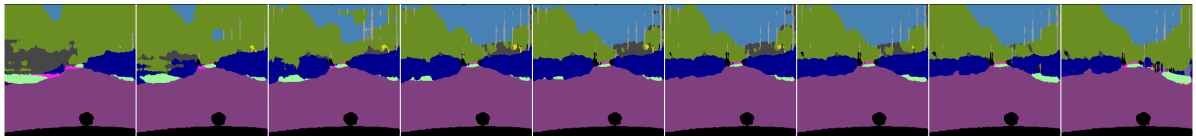
Table 1. Comparison of our method and PGGAN. Our method produces similar quality images as PGGAN for the coarse generation as measured by the Fréchet Inception Distance. Users have an equal preference for both methods when asked to choose which image looked more realistic.

interpolated frames. To recover the latent vectors of images in PGGAN, we also start with 64 initialisations and then minimise the following loss, using Adam with a learning rate of 0.1 for 50 iterations: loss = mean squared er-

original video sequence

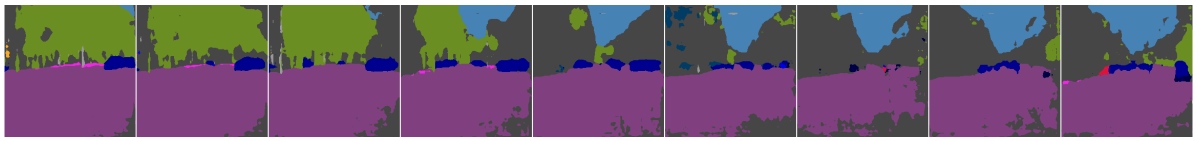
DeepLab segmentation of video sequence

Our reconstruction of the video, by interpolating matched segmentation maps



Our interpolated segmentation maps

PGGAN video reconstruction, interpolating between the reconstructions of the first and last frames



DeepLab segmentation of PGGAN interpolation.

Figure 5. A short video should correspond to a linear walk in the latent manifold. We reconstruct the first and last frames of the video by inverting the GAN of our method (matching our segmentations to the DeepLab segmentations) and of PGGAN (minimising the MSE between generated and true images). We show the results of interpolating between the reconstructed endpoints of the video using both methods. Our layouts are more consistent with the true changes in layout over the video, because of the intermediate object-based representations of scenes.

ror(image, PGGAN(z)) + $0.01||z||^2$, where z is the latent vector.

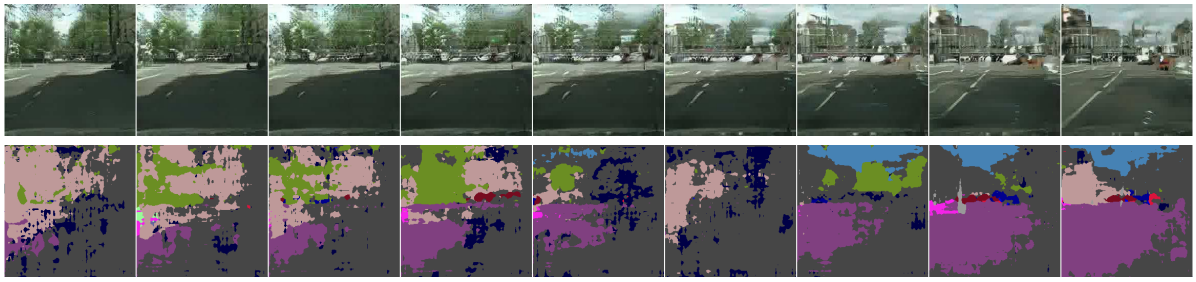
Because inverting a GAN generator is difficult and imprecise, we notice that a region of trees is mapped to trees in the reconstruction of the first frame, but to buildings in the reconstruction of the last frame. Trees and buildings have similar image intensities, leading a failure in their distinction by PGGAN. This leads to an implausible interpolation in which trees morph into buildings.

Although the generations of our model and PGGAN are of similar quality, this experiment shows that having a very crude world model in the form of semantic segmentations helps our method generate samples similar to true missing frames.

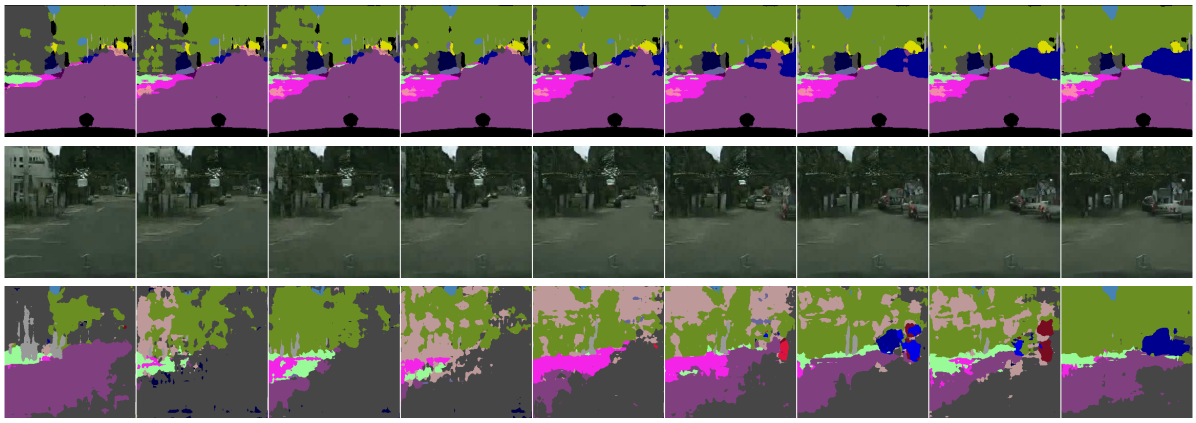
5.5. Segmenting interpolations

Here we discuss further the segmented interpolations in Fig. 1, and provide further examples in Fig 5.4. In other sampled sequences, we observe the same noisy pattern in the segmentations of interpolations. In particular, the pink region, which have the label *fence*, are often predicted, probably due to the blocky artefacts in the generated images.

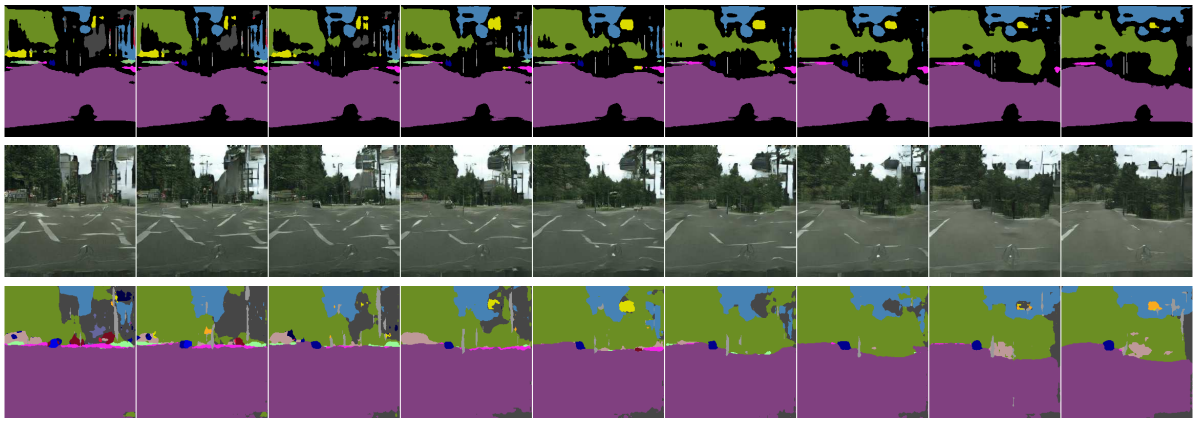
Interpolations generated by our method, where the seg-



Progressive Growing of GANs (PGGAN). top: generated images, bottom: DeepLab segmentations.



Ours. top: generated fine seg. maps, middle: rendered images, bottom: DeepLab segmentations.



Ours. top: generated coarse seg. maps, middle: rendered images, bottom: DeepLab segmentations.

Figure 6. Both PGGAN and our method trained with fine annotations create scene textures with ambiguous object classes. Our method with coarse annotations is able to clearly distinguish and place objects in a way that changes smoothly during interpolations.

mentation maps are fine, also produce very noisy segmentation maps. There are two reasons why this might happen. First, the training set of fine annotations contains only 3000 images, compared to 22000 images of coarse annotations. The generator might have overfit to these examples. Secondly, the fine annotations are much more complex than the coarse ones, and contain many small objects. Since our generated fine segmentation maps are not perfect, the many small noisy objects may get rendered into something unrecognisable by pix2pix.

When we use our method trained with coarse annotations, however, we see a much smoother and cleaner pattern of segmentations from DeepLab. The source segmentations generated by our method, and the re-segmentations of our rendered images by DeepLab visually seem to match well, which is not the case with fine annotations.

Most importantly, PGGAN tends to let objects appear and dissolve without much physical consistency, whereas our approach will keep them in existence, merely moving them around. These transitions can be seen much more clearly in video sequences generated from random walks in the latent space, which can be viewed in the Supplementary

Material or at https://imgur.com/a/NlP9Lzn.

6. Discussion

We have proposed an alternative method for training GANs to generate images with multiple objects. Segmentation maps are less complex than than natural images, and GANs have an easier time learning them than high resolution images directly. Incorporating segmentation maps into GAN training improves interpolations. Interpolations produced by our method are aware of scene layouts, and show a higher structural consistency between frames than PGGAN. These preliminary results suggest that using intermediate representations allows for the generation of scenes with more distinct objects as revealed by their DeepLab segmentations, while having equal realism to PGGAN generations, as measured by both our user study and the FID metric.

Acknowledgement

Part of this work has been financed through a generous gift by Google.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 1, 3

[2] Samaneh Azadi, Michael Tschannen, Eric Tzeng, Sylvain Gelly, Trevor Darrell, and Mario Lucic. Semantic bottleneck scene generation. *arXiv preprint arXiv:1911.11357*, 2019. 2

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018. 1, 2

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv:1802.02611*, 2018. 1, 3

[5] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 1, 2017. 3

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 3

[7] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015. 2

[8] Emanuele Ghelfi, Paolo Galeone, Michele De Simoni, and Federico Di Mattia. Adversarial pixel-level generation of semantic images. *arXiv preprint arXiv:1906.12195*, 2019. 3

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1

[10] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017. 3

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6626–6637. Curran Associates, Inc., 2017. 2, 4

[12] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7986–7994, 2018. 3

[13] Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. Stacked generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 4, 2017. 2

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017. 1, 3

[15] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[16] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 1, 2, 3

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 2

[18] Hanock Kwak and Byoung-Tak Zhang. Generating images part by part with composite generative adversarial networks. *arXiv preprint arXiv:1607.05387*, 2016. 3

[19] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019. 3

[20] Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[21] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolu-

tional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 1, 3

[22] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[23] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. *arXiv preprint arXiv:1711.11585*, 2017. 3

[24] Xiaolong Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. In *European Conference on Computer Vision*, pages 318–335. Springer, 2016. 3

[25] Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. Lr-gan: Layered recursive generative adversarial networks for image generation. *International Conference on Learning Representations*, 2017. 3

[26] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pages 7354–7363, 2019. 2