Semantic Consistent Weather Transfer

Dominik Hollidt
ETH Zürich

Sichen Li
ETH Zürich

Zhenjie Jiang
ETH Zürich

Antonio Arbues
ETH Zürich

***Abstract*—While benchmarking self-driving car related vision algorithms, using datasets recorded in different seasons of the year is a best practice, but recording such datasets is expensive and time-consuming. Furthermore, the sim-to-real gap is still relevant nowadays.**
Some methods use GANs to transform pictures from one season to another, so datasets can be augmented easily and cheaply.
We propose a method focused on self-driving car datasets, that translates pictures from the overcast weather to the snowy weather. Our approach is based on the WeatherGAN work and uses a self-implemented CycleGAN as its baseline.
We innovate the model definition of WeatherGAN by adding a semantic map to the input of the generator, we include a semantic term in the loss function, and we use a pre-trained feature extractor in the discriminator.
The obtained results do not achieve the level of the CycleGAN baseline because of some difficulty encountered in the training process of the GAN. We propose areas of improvement based on our obtained experience in Section IV.

***Index Terms*—GAN, attention, ResNet, segmentation, driverless dataset**

I. Introduction

The problem of benchmarking and comparing different vision algorithms for autonomous driving tasks is fundamental in order to be able to deploy the most performant one in the real system, as well as to drive towards the correct direction the research process.
In order to do so, one of the basic requirements is to have a dataset available that encompasses different weather conditions as well as scenes recorded in different seasons. Furthermore, it is sometimes also preferable to have the exact same scenario recorded in different seasons to be able to rigorously assess how the algorithm performs in different conditions in the same environment. However, it is almost impossible to replicate the same scene for a car but in different seasons as it would need an exact setting, of objects, actions, movements and velocities.
Few solutions are available to accomplish the previously stated goal. A vehicle with the required sensors can be deployed multiple times to record the dataset under different weather conditions. However, this process is very expensive and time-consuming. To add upon that, most autonomous datasets have less data in winter or lack it completely, which makes training even harder [15, 4, 18]. Other datsets miss weather labels [7] or only provide grayscale images[24]. Further, it is highly unlikely to encounter an exact replication of the same scene in a different weather condition. Another option is to use simulators to model the scenario but research has already shown a gap between simulation and the real world [12, 26, 14, 9, 13].

A rather new approach is based on Generative Adversarial Networks (GANs). They have been recently applied to the problem, making it possible to convert pictures in a certain season to another one [30]. This work has been improved on in WeatherGAN [17]. Using GANs to address appearance changes has already spiked interested in robotics research [25].
In this work, we propose a GAN-based framework to convert city images in overcast weather condition into snowy weather conditions. Thereby we can make up for the lack of winter training data and replicate exact same scenarios in different seasons. Further, by training other season transfers one could make hard scenes easier for networks trained on different data, e.g. transfer winter data to summer for a network trained on summer data. This work takes the WeatherGAN work [17] as reference, and adds up the following contributions:

1) Addition of semantic maps to the input of the Generator model
2) Inclusion of a semantic loss that avoids the training of the semantic network (trained independently)
3) Usage of a pre-trained feature extractor in the Discriminator model
4) Usage of a narrower dataset from autonomous driving.

Fig. 1: Input and output of the CycleGAN baseline.

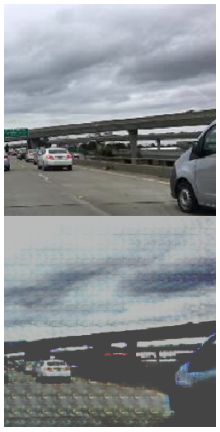


Fig. 2: Input and output of our model.

II. Related Work

The progress GANs showed after their invention in 2014 is incredible [8]. After their invention, GANs were used for two main task. Firstly, the creation of completely new data from an initial noise vector and secondly the transfer of original data to a different domain. In the first case StyleGan showed impressive results and is able to generate impressive artificial faces from a noise together with a latent vector. Mixing the latent vector even allows for style mixing or transfer. However, this work is focused on the image to image translation, where pix2pix models and CycleGan showed one of the first image translation methods via GANs [11, 30]. While pix2pix networks require the training on paired images, CycleGAN's biggest contribution was the unpaired training.

In image-to-image translation problem, the goal is to learn the mapping between an input image x and output image y. However, paired images are often not available and difficult to obtain. CycleGAN [31] tries to circumvent this problem by introducing cycle consistency loss. In CycleGAN, two generator and discriminator pairs are trained simultaneously, where generator G maps x to y, and generator F maps y to x. If G receives y as input or F receives x as input, they should map them back to the original image. CycleGAN incentivizes this behaviour with the cycle consistency loss:

$$\mathscr{L}_{cycle}(G,F) = \mathbb{E}_{x\sim p_{data}(x)}[||F(G(x)) - x||_1] + \mathbb{E}_{y\sim p_{data}(y)}[||G(F(y)) - y||_1].$$

Furthermore, the cycle consistency loss is combined with the adversarial loss, which measures the generator's ability to fool the discriminator to encourage a good mapping.

Most recently WeatherGAN [17] build upon the idea of CycleGAN, but added domain specific tweaks. The authors noticed that image regions containing so called "WeatherCues" are most affected by weather change, e.g. road might get affected by snow more than the sky. Hence, they decided to split the generator into three parts: one initial translation network, an attention network and a semantic network. Here the translation network has the task of transforming the image to the new weather, however the attention mechanism together with the semantic network would ensure that the image gets only changed in regions where WeatherCues were identified. Lastly the original image would get added to the partially transformed image. While this has the advantage of only focusing weather change to relevant parts of the image, its downside is that effectively three networks have to be trained. Further, one relies on the accurate creation of semantic maps for the semantic module of the generator. The authors used the cycle loss, but also added some perceptual loss for the semantic maps. The discriminator was not covered much by the WeatherGAN work, although their are many incentives to push the discriminator to be more helpful. Richter, Al Haija and Koltun proposed interesting improvements to the discriminator [21]. First of all, they also used a semantic network to ensure that pixels would be transformed semantically consistent and correct. Secondly, they used VGG-16 features together with the semantic label map to create a realism score per pixel. Stacking several of these discriminator blocks allowed each discriminator to focus on a different aspect of the input image. To make up for autonomous data sets often containing the hoods of the car they used a patch sampling technique to reduce artifacts. For the generator modules, the authors reused the input and game engine temporary G-Buffers. To avoid the discriminator outperforming the generator, discriminator back propagation was skipped at random.

Because GAN training can be tricky and unstable several normalization techniques have been proposed [23, 1]. The spectral norm showed good results by stabilizing the training process of the discriminator [19]. The wasserstein loss was proposed to push generator to approximate the observed distribution better [1].

For the purposes of this work, we took inspiration from many the aforementioned ideas, and we successfully implemented many of them.

III. Models and Method

The main contribution of this work is to reduce the computational load of training, when a transfer is only required from one domain to another, e.g. summer to winter. Still, we require a semantic consistency in our images to be useful for autonomous driving challenges. While CycleGAN requires two generators and discriminators (one pair for each direction), our model only requires a single pair. Further, we take inspiration from the WeatherGAN approach and use the attention mechanism to reduce random transformations of the GAN. Moreover, in an attempt to make training more efficient, we remove the semantic network from the generator, but also require a semantic consistency of the input and output image produced by the generator. By restricting the scope of this work to autonomous driving scenario, we expect a better performance compared to the baseline in this environment. In order to have a baseline for comparison, we implemented CycleGAN.

A. Generator

The motivation for the generator is three-fold. First of all, we acknowledge the idea of only changing the image in parts where it is relevant. Secondly, the change should also depend on the semantic map. Lastly, the generator

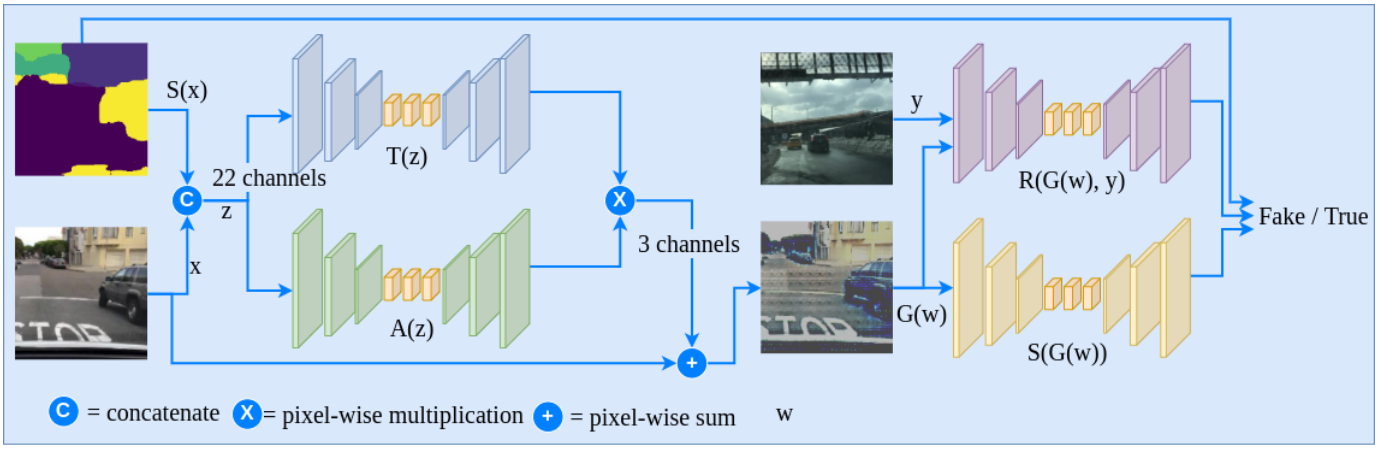


Fig. 3: This is an overview of the architecture of the proposed network

should learn the changes to the image rather than the whole transformation. These three pillars should ideally be achieved while having relatively short training times. The problem with CycleGAN is that it is training two pairs of generators/discriminators. Since the WeatherGAN architecture, which full fills our three targets, is build upon the cycle-loss, it is more demanding during training. Further, we do think that the computational load is increased by the semantic network in the generator. Additionally, this semantic network requires some form of annotated label maps. WeatherGAN uses quite low detail WeatherCue semantic maps as discussed in II. Hence, more performance could be gained by having a finer distinction between regions, i.e. more classes. More details about the generated maps can be found in section III-B. To boost the power of the semantic maps further, we make use of a pre-trained semantic network for high accuracy, detailed semantic maps, as described in the next section. However, these maps serve as additional input to the network rather than being computed during training as in WeatherGAN. We expect this to increase accuracy and reduce training time.

The two main parts of the Generator are the attention mechanism and the translation unit, as visualized in Figure 3. Like in WeatherGAN, the translation unit is supposed to transform the whole image from domain A to domain B, while the attention network takes care of identifying relevant parts of the image. These key parts are changed according to the translation network.
The attention network first scales down the input to lower resolution via Convolutions, BatchNorm2D and MaxPooling layers. This initial condensation of information is followed by two attention modules (as in [29]) of depth three. Finally, the result is up-scaled to its original using convolutions (ConvTranspose2d in PyTorch).

The translation network is a self-implemented ResNet [10] adaptation. The 22-channels image (3 RGB channels and 19 one-hot encoded segmentation channels) input coming directly from the dataset is scaled down via Residual Blocks with a kernel size of 3, to obtain successive outputs with 64, 128, 256 channels. The image is then up-sampled again passing through the channel size of 128, 64, 32, and eventually 3. The Residual Blocks are made of the succession of a 2D-convolutions, batch-norm layer, ReLU activation function, and final max-pooling layer. The convolutional layer encapsulates the information present in each kernel occupancy zone while cutting short on parameters to learn because of their shared nature [16]. The batch-norm layer regularizes the model while avoiding gradient fading or explosion phenomena. The max-pooling layer offers a cheap way to compute solution for translation invariance. On top of that, the Residual Network architecture solves the degradation problem of deep models, easing the training process. During the experiments we used 4 to 9 residual blocks for the translation network.

Lastly, we do a point wise multiplication between the attention and the translated image and add the original image. While we also tried to give the network more expressive power in the blending of the original image, the translated image and the attention map, it did provide good results in practice. The concept was to concatenate the original image, translated image, attention map and the point wise multiplied image to then apply a few layers convolutions on them.

In a nutshell, as showed in Figure 3, the generator gets a [3 + #semantic map channels, H, W] image, performs a translation and modifies the original image only in parts with focus of the attention network.

B. Semantic Segmentation

In order to ease the training effort while keeping compatible semantic segmentation power, we incorporate a start-

of-the-art semantic segmentation model that is pre-trained on a larger and similar dataset, and retrain it with a subset of the BDD100k dataset, to make it more adapted to the autonomous driving scenario.

1) Model: The DeeplabV3 model [3] is a semantic segmentation model that employs the Atrous Spatial Pyramid Pooling (ASPP) module, which applies several atrous convolution with different rates in parallel or in cascade, in order to capture semantic context at multiple scales. The follow-up DeeplabV3+ model [2] improves DeeplabV3 by combining it to form an encoder-decoder structure, as shown in Figure 1 of [2]. By adding a simple decoder module, the object boundaries are highly refined.

2) Dataset: The Cityscapes dataset [4] is composed of stereo video sequences recorded in street scenes from 50 different cities. It contains 5000 images with fine semantic annotations and 20000 images with coarse annotations in 30 classes. The dataset we use, the BDD100k dataset, has similar street views and follows the same class labelling with Cityscapes. Moreover, BDD100k also provides 8000 images with semantic annotations, so that customized models can be conveniently retrained and evaluated.

3) Implementation and evaluation: Based on the torchvision implementation of DeeplabV3 [5] and a follow-up implementation of DeeplabV3+ [6], we take the model with Mobilenet backbone (with much smaller number of parameters compared to ResNet backbone yet still achieving good performance), then retrain it using the available 8000 images with segmentation labels in BDD100k. Among the 8000 images, 7000 are used for training (denoted as *BDD1001k-seg-train*) and 1000 are for validation (denoted as *BDD100k-seg-val*). Table I summarizes the models and evaluations in terms of mean Intersection-Over-Union (mIoU). Figure 4 shows some example segmentation results from the Cityscapes model and the Cityscapes+BDD100k model, compared with original images and ground truth annotations. It can be seen that the retrained model manages to remove spurious fluctuations.

Model on training dataset	**Evaluation dataset**	**mIoU**
DeeplabV3+ on *Cityscapes-train*	*Cityscapes-val*	0.721
DeeplabV3+ on *Cityscapes-train*	*BDD100k-seg-val*	0.377
DeeplabV3+ on *Cityscapes-train* & *BDD100k-seg-train*	*BDD100k-seg-val*	0.574

TABLE I: Semantic segmentation models and results.

C. Discriminator

To ensure realism and semantic consistency the discriminator contains a realism network and our semantic network. The realism network checks for the realism score of the generated image, while the semantic network ensures semantic consistency between the input and the output.

The realism network is the discriminator that's used in CycleGAN. It consists 2d convolutional layers and LeakyReLU activation functions. The output is a 2d matrix of size $\frac{H}{2^4} \times \frac{W}{2^4}$, where $H \times W$ is the size of the input image. Each entry in the matrix has a value between 0 and 1, which represents how indistinguishable the patch of the image the entry covers compare to the real target images, where 1 represents indistinguishable.

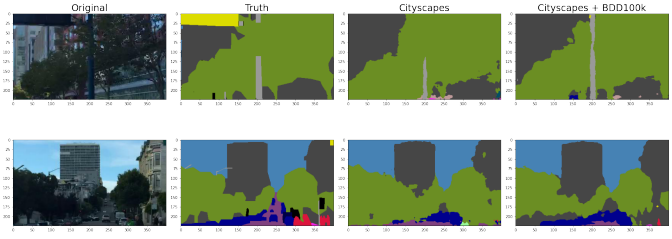


Fig. 4: Segmentation results of both models compared with original input.

In the first attempt, the input to the realism network was boiled down to a 1000 vector by a pretrained ResNet18. Then this vector would be used to do a binary classification of the input image, i.e. fake or real. However, this approach did not work and the network didn't exhibit any learning, and as a result we changed it to match CycleGAN closer, as described above.

D. Loss Function

GANs are trained in a Min-Max game where the Discriminator tries to maximize:

$$\max_D L^D_{adv}(D,G) = E_{x\sim P_{\text{data}}(x)}[\log(1-D(G(x)))] + E_{y\sim P_{\text{data}}(y)}[\log D(y)] \quad (1)$$

The Generator tries to minimize:

$$\min_G L^G_{adv}(D,G) = E_{y\sim P_{\text{data}}(y)}[\log(1-D(G(y)))] \quad (2)$$

More details can be read in [17, 8]. We modified the loss to make use of a weighted sum of three loss functions for the generator. They should ensure semantic consistency, realistic looking images and that realistic looking images stay the same. In more detail, we use the cross entropy loss between the map before the generator transform and the map after the generator transform. A binary cross entropy per pixel in the discriminator image. The identity loss ensures that images of the target domain don't get altered too much and is the L1 loss between a true target domain image and its transformed version. The discriminator just tries to maximise its default objective with binary cross entropy. However, to make the discriminator less overconfident we smoothed the target label to 0.1/0.9 for fake or real respectively.

E. CycleGAN Baseline

For baseline, we implemented CycleGAN with 9 Residual Blocks. With weight 1, 5 and 10 for adversarial loss (GAN loss), cycle consistency and identity loss respectively.

F. Training

We train on the BDD100k dataset [27] with 8770 overcast images and 5549 snowy images. For the baseline we used a resolution of 180x320. The final model was trained on 224x400. To train the generator and discriminator we used Adam optimizer with $\beta_1 = 0.5, \beta_2 = 0.999$. Further, a batch size of 8 and learning rate decay was used. Since the discriminators are prone to outperform the generator, its back propagation was skipped at random, especially since it uses pretrained ResNet features [21]. The training times were all over 24 hours on an Nvidia GTX1080.

IV. Results

This section covers the experiments and the obtained results. Unfortunately our results were rather poor and we could not see the GANs learning hence we only report a few examples of failed image translations in the appendix (see fig. 5 and 6). Further we outline our process of stabilizing the GAN training. We hope this will give an example on what might not be sufficient to train GANs.

The baseline is our own custom CycleGan implementation trained on the BDD100k dataset and achieved comparable results to the ones in the original paper, when visually inspected.
The first experiment was to run the generator with extended blending options, as described in section III, paired with the ResNet18 discriminator to predict a single fake/real label. This one did not seem to produce helpful results.
We tried to improve that one by reusing the discriminator from CycleGan to produce a per-pixel realism score.
Unfortunately, it was not too successful either. Hence we tried to reduce the generators blending options and restricted ourself to a point-wise multiplication between attention and translated output.
This already led to better results, but we tried tuning the loss combination to put more emphasis on the realism loss, since the semantic and identity loss were decreasing a lot more.
To stabilize training, we also added spectral norm, smoothed target labels and skipped the discriminator training more often. However, it didn't help much as the generated images looked almost like the input, as showed in figure 2. In the end we witnessed a higher response of the attention map, see fig. 7. When skipping the discriminator backpropagation to often it up with the real loss decreasing, otherwise it would increase (see fig. 10 and 11). In both cases the generator did not learn any relevant transformation.

V. Discussion

Investigating our results we can see that the CycleGAN achieves some impressive results where one can definitely see snow piling up in the image, as in Figure 1. This is very similar to the expected result. On the other side, the results are quite blurry. We expect that the blurring would become even worse on bigger resolutions.

Patches in the attention maps and images are visible due to the small and odd kernel size we used. Using a stride of two together with a kernel size of three leads to some pixels overlapping in the upscaling by the transposed convolution. Theoretically, models should be able to learn these overlapping but tend to fail often [20]. For this reason, bigger kernel sizes or other up-sampling techniques would produce smooth upscaling results [20]. Our patching artifacts are unfortunately very heavy.

Another issue we noticed was extreme values we encountered by the initial translation network, applying a pointwise multiplication with them let to even bigger values (see 9). We suspected that the more advanced blending mechanism could be at fault for that behaviour. Replacing it with a simple pointwise multiplication like in the original WeatherGan [17] improved training, but we still encountered many high values. Unfortunately applying more normalization to the discriminator also didn't effect them. Hence, we propose to use stronger regularization or a stronger gating behaviour like in LSTM units.

During training we monitored the gradients and could not see any vanishing or exploding gradients. Further, despite tweaking the combination of losses, adapting the ratio of discriminator skips, including spectral norm, changing the discriminator, modifying the blending mechanism in the generator, changing the depth of the translation network, using smoothed target labels for the discriminator and modifying batch sizes, the results of our GAN-experiments leave a lot of room for improvement, but since GAN training's are prone to instabilities, more time would be needed to get the desired results. We propose the following areas of improvement:

- use a more suited loss, e.g. Wasserstein loss
- improve the discriminator following [21]
- use a different translation network, e.g. U-Net [22] or HRNetV2 [28], as they showed good results already [21]
- improve upscaling units to reduce patching effect as described in [20]
- skip the discriminator training based on performance to keep it as the same level of the generator [21]
- use stronger generator normalization or more advanced gating behaviour

VI. Summary

This work focused on the proposition of a specific GAN model for the conversion of self-driving cars visual datasets from an overcast weather to snow.

The model in Figure 3 was developed and implemented. Its main innovations over WeatherGAN figure: the employment of a separate segmentation network to generate the segmentation maps, the semantic loss term that avoids training the semantic network, and the use of a pre-trained feature extractor in the discriminator.

The developed GAN model turned out to be very difficult to train, and while many problems were successfully overcome, many others were not, leading to results that don't match the self-implemented CycleGAN baseline.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, June 2017, pp. 214–223. URL: https://proceedings.mlr.press/v70/arjovsky17a.html.

[2] Liang-Chieh Chen et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 801–818.

[3] Liang-Chieh Chen et al. "Rethinking atrous convolution for semantic image segmentation". In: *arXiv preprint arXiv:1706.05587* (2017).

[4] Marius Cordts et al. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[5] *DeepLabv3Plus-Pytorch*. https://github.com/pytorch/vision/blob/main/torchvision/models/segmentation/deeplabv3.py. Accessed: 2022-01-04.

[6] *DeepLabv3Plus-Pytorch*. https://github.com/VainF/DeepLabV3Plus-Pytorch. Accessed: 2022-01-04.

[7] Jakob Geyer et al. "A2D2: Audi Autonomous Driving Dataset". In: (2020). arXiv: 2004.06320 [cs.CV]. URL: https://www.a2d2.audi.

[8] Ian J. Goodfellow et al. "Generative Adversarial Networks". en. In: *arXiv:1406.2661 [cs, stat]* (June 2014). arXiv: 1406.2661. URL: http://arxiv.org/abs/1406.2661 (visited on 01/02/2022).

[9] Winter Guerra et al. "FlightGoggles: Photorealistic Sensor Simulation for Perception-driven Robotics using Photogrammetry and Virtual Reality". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866. Nov. 2019, pp. 6941–6948. DOI: 10.1109/IROS40897.2019.8968116.

[10] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[11] Phillip Isola et al. "Image-to-Image Translation with Conditional Adversarial Networks". In: *arXiv:1611.07004 [cs]* (Nov. 26, 2018). arXiv: 1611.07004. URL: http://arxiv.org/abs/1611.07004 (visited on 01/02/2022).

[12] Theo Jaunet et al. "SIM2REALVIZ: Visualizing the Sim2Real Gap in Robot Ego-Pose Estimation". en. In: *arXiv:2109.11801 [cs]* (Dec. 2021). arXiv: 2109.11801. URL: http://arxiv.org/abs/2109.11801 (visited on 01/04/2022).

[13] Abhishek Kadian et al. "Sim2Real Predictivity: Does Evaluation in Simulation Predict Real-World Performance?" In: *IEEE Robotics and Automation Letters* 5.4 (Oct. 2020). Conference Name: IEEE Robotics and Automation Letters, pp. 6670–6677. ISSN: 2377-3766. DOI: 10.1109/LRA.2020.3013848.

[14] Eric Kolve et al. "AI2-THOR: An Interactive 3D Environment for Visual AI". en. In: (), p. 4.

[15] Måns Larsson et al. "A Cross-Season Correspondence Dataset for Robust Semantic Segmentation". en. In: *arXiv:1903.06916 [cs]* (Aug. 2019). arXiv: 1903.06916. URL: http://arxiv.org/abs/1903.06916 (visited on 11/21/2021).

[16] Y. Lecun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[17] Xuelong Li, Kai Kou, and Bin Zhao. "Weather GAN: Multi-Domain Weather Translation Using Generative Adversarial Networks". In: *CoRR* abs/2103.05422 (2021). arXiv: 2103.05422. URL: https://arxiv.org/abs/2103.05422.

[18] Yiyi Liao, Jun Xie, and Andreas Geiger. "KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D". In: *arXiv.org* 2109.13410 (2021).

[19] Takeru Miyato et al. "Spectral Normalization for Generative Adversarial Networks". In: *arXiv:1802.05957 [cs, stat]* (Feb. 16, 2018). arXiv: 1802.05957. URL: http://arxiv.org/abs/1802.05957 (visited on 12/24/2021).

[20] Augustus Odena, Vincent Dumoulin, and Chris Olah. "Deconvolution and Checkerboard Artifacts". In: *Distill* (2016). DOI: 10.23915/distill.00003. URL: http://distill.pub/2016/deconv-checkerboard.

[21] Stephan R Richter, Hassan Abu AlHaija, and Vladlen Koltun. "Enhancing photorealism enhancement". In: (), p. 16.

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[23] Tim Salimans and Diederik P. Kingma. *Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks*. 2016. arXiv: 1602.07868 [cs.LG].

[24] P. Wenzel et al. “4Seasons: A Cross-Season Dataset for Multi-Weather SLAM in Autonomous Driving”. In: *Proceedings of the German Conference on Pattern Recognition (GCPR)*. 2020.

[25] Markus Wulfmeier, Alex Bewley, and Ingmar Posner. “Addressing Appearance Change in Outdoor Robotics with Adversarial Domain Adaptation”. en. In: *arXiv:1703.01461 [cs]* (Sept. 2017). arXiv: 1703.01461. URL: http://arxiv.org/abs/1703.01461 (visited on 12/22/2021).

[26] Fei Xia et al. “Gibson Env: Real-World Perception for Embodied Agents”. en. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT: IEEE, June 2018, pp. 9068–9079. ISBN: 978-1-5386-6420-9. DOI: 10.1109/CVPR.2018.00945. URL: https://ieeexplore.ieee.org/document/8579043/ (visited on 12/22/2021).

[27] Fisher Yu et al. “BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning”. In: *arXiv:1805.04687 [cs]* (Apr. 8, 2020). arXiv: 1805.04687. URL: http://arxiv.org/abs/1805.04687 (visited on 01/03/2022).

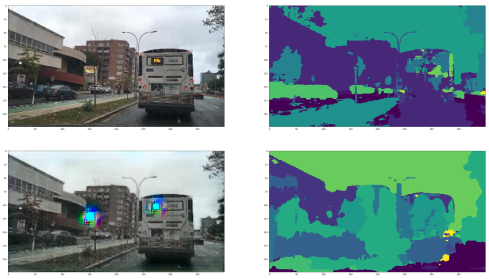
[28] Yuhui Yuan et al. “Segmentation Transformer: Object-Contextual Representations for Semantic Segmentation”. en. In: *arXiv:1909.11065 [cs]* (Apr. 2021). arXiv: 1909.11065. URL: http://arxiv.org/abs/1909.11065 (visited on 01/04/2022).

[29] Han Zhang et al. “Self-Attention Generative Adversarial Networks”. In: *arXiv:1805.08318 [cs, stat]* (June 14, 2019). arXiv: 1805.08318. URL: http://arxiv.org/abs/1805.08318 (visited on 01/03/2022).

[30] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017 IEEE International Conference on Computer Vision (ICCV). Venice: IEEE, Oct. 2017, pp. 2242–2251. ISBN: 978-1-5386-1032-9. DOI: 10.1109/ICCV.2017.244. URL: http://ieeexplore.ieee.org/document/8237506/ (visited on 11/21/2021).

[31] Jun-Yan Zhu et al. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. 2020. arXiv: 1703.10593 [cs.CV].

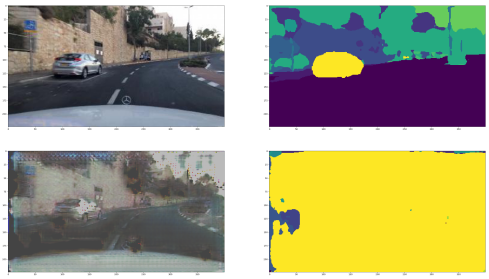
APPENDIX



(a)

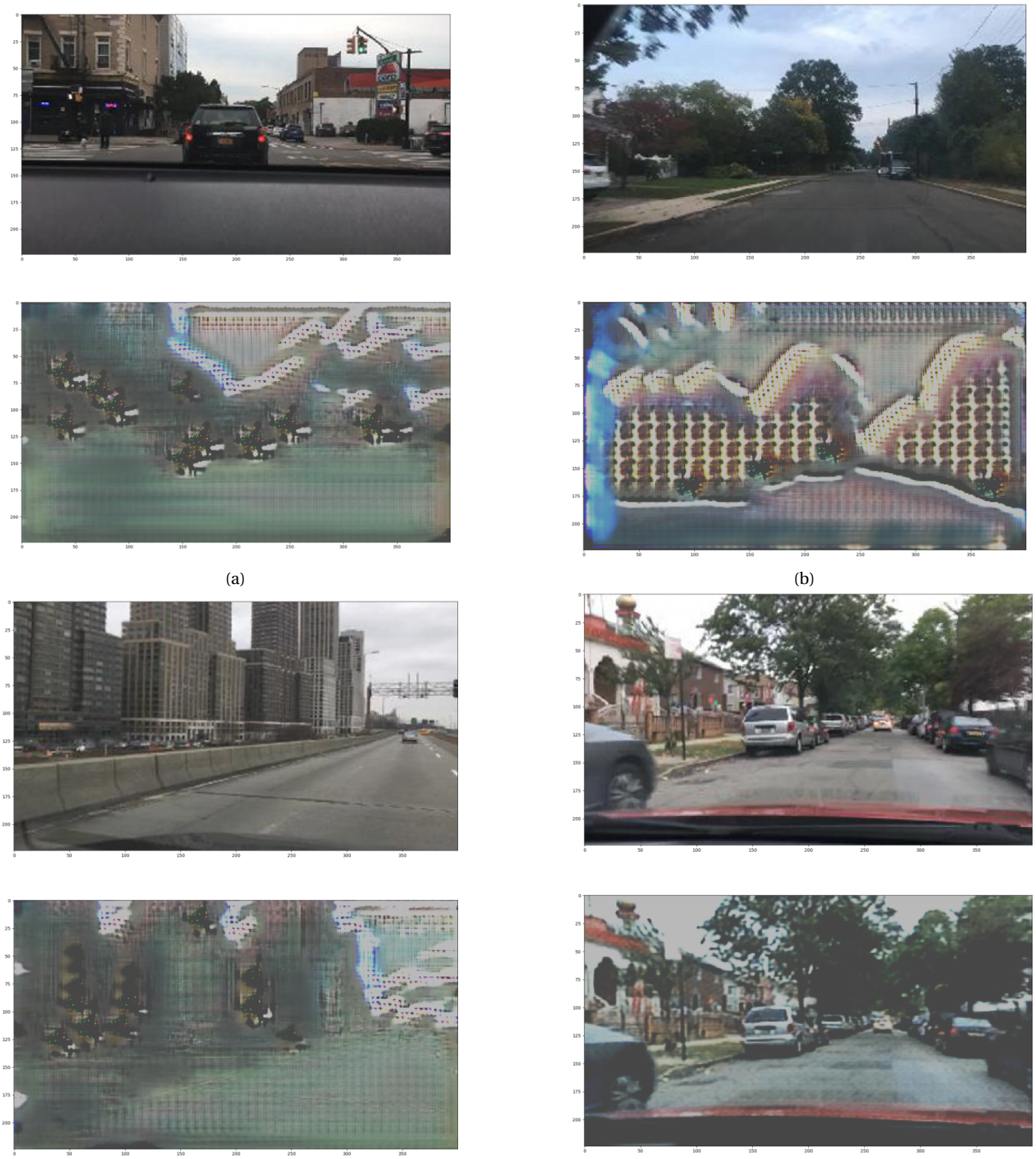
(b)

(c) Weird artifacts but no visible transformation.



(d) The attention map learned to remove some parts of the image, but the image from the translation network image is black. Hence the black artifacts.

Fig. 5: Four failed examples of our GAN. Top left original image, bottom left transformed image, top right semantic map before, bottom right map after transformation. Semantic map are the most stable. The transformation fails.



(a) (b) (c) (d)

Fig. 6: Failed examples where the attention realises focus areas, but the translation network does not provide good fill in information as seen in fig. 8. The top is the input the bottom the transformed image.

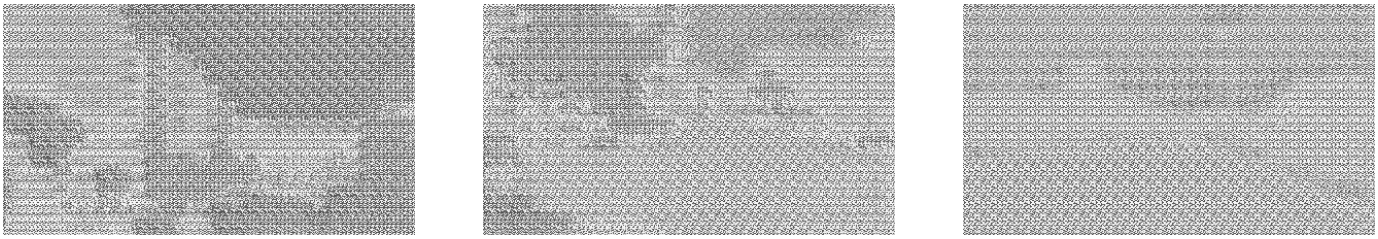


Fig. 7: The attention picks up on some parts of the environment.

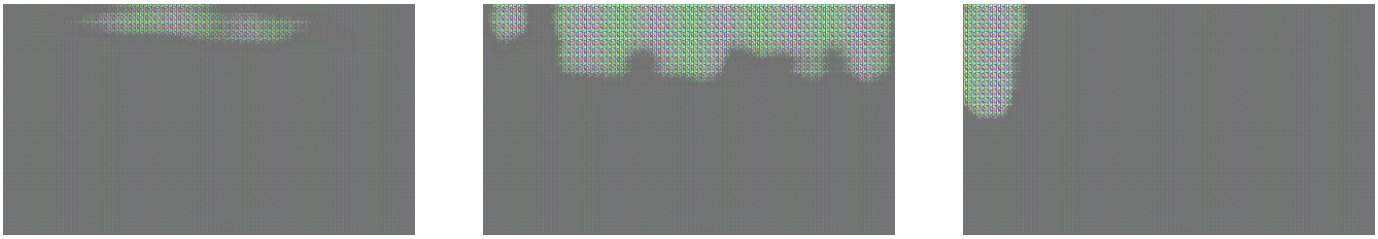


Fig. 8: The translation network is not learning any useful transformation.

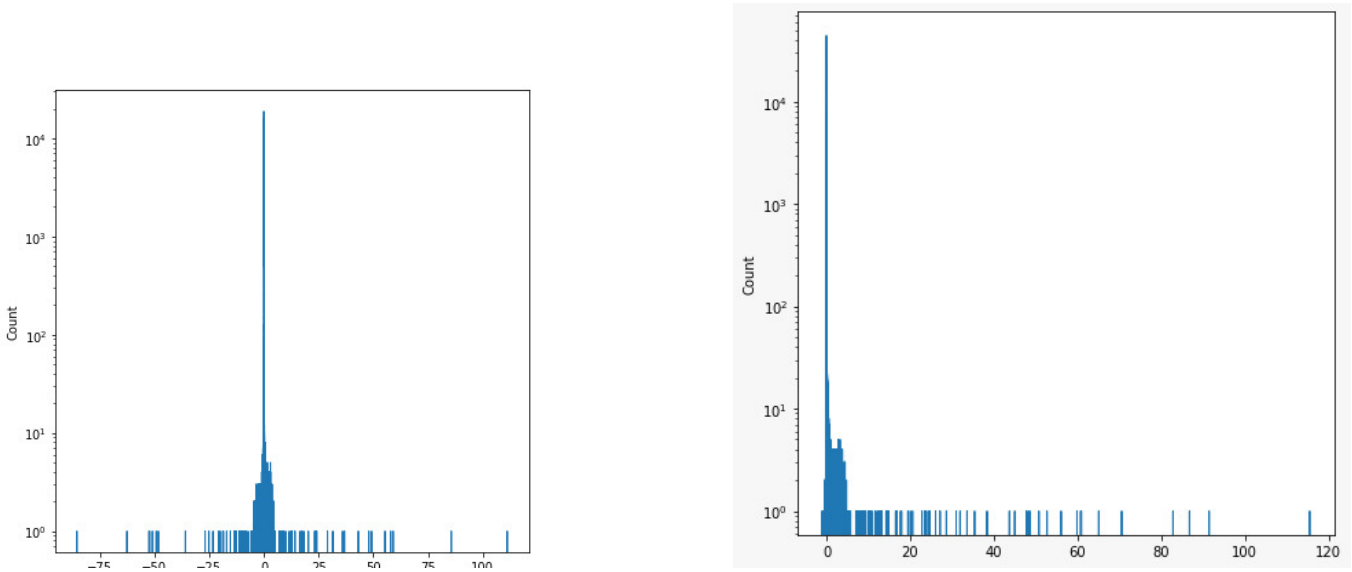


Fig. 9: Left example of point wise multiplication histogram. Right example of translation histogram. We see extreme values despite many being zero.

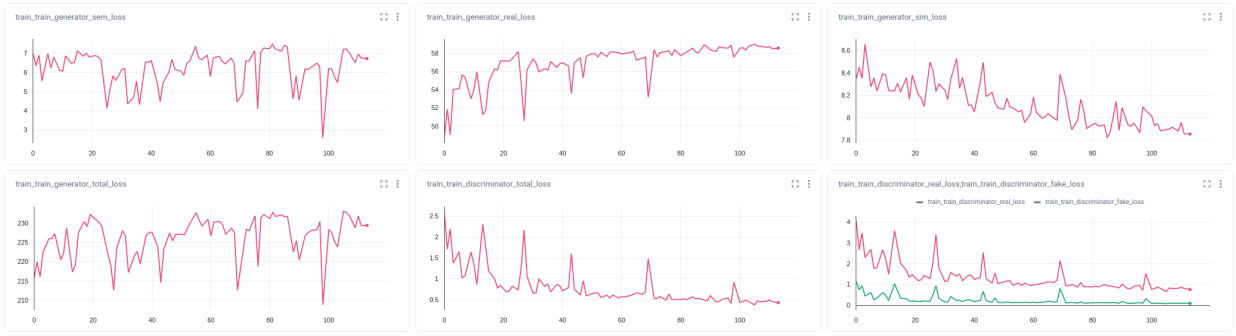


Fig. 10: Typical loss plot per epoch when skipping the discriminator rarely, e.g. p=0.1. Generator real loss increases but is not creating realistic transformed images. Here the discriminator has a lower loss for fake images and is outperforming the generator.

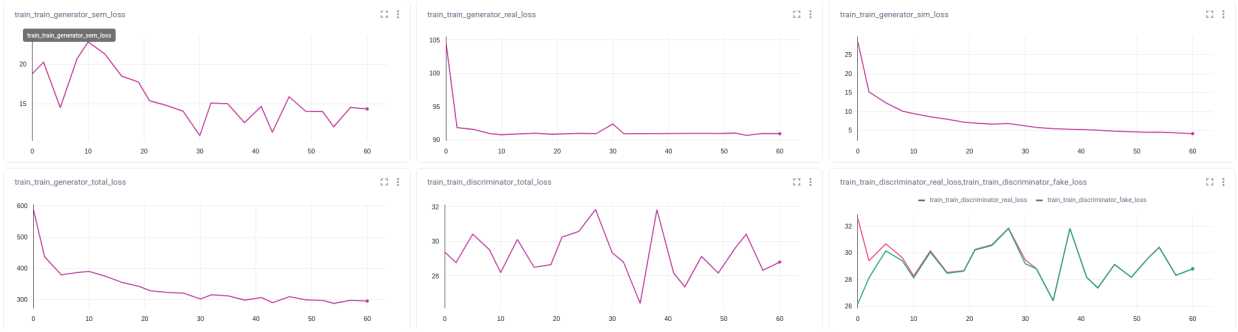


Fig. 11: Typical loss plot per epoch when skipping the discriminator more often e.g. p=0.5. Generator real loss goes to zero but is not creating realistic transformed images. Here the discriminator has a similar loss for real and fake images.