Taming Adversarial Domain Transfer with Structural Constraints for Image Enhancement

Elias Vansteenkiste and Patrick Kern
Brighter AI Technologies Gmbh
Torstrasse 177, Berlin
{elias, patrick}@brighter.ai

Figure 1: Our domain transfer techniques applied to the night-to-day, removing rain and removing fog applications

Abstract

The goal of this work is to improve images of traffic scenes that are degraded by natural causes such as fog, rain and limited visibility during the night. For these applications, it is next to impossible to get pixel perfect pairs of the same scene, with and without the degrading conditions. This makes it unsuitable for conventional supervised learning approaches, however, it is easy to collect unpaired images of the scenes in a perfect and in a degraded condition. To enhance the images taken in a poor visibility condition, domain transfer models can be trained to transform an image from the degraded to the clear domain. A well-known concept for unsupervised domain transfer are cycle-consistent generative adversarial models. Unfortunately, the resulting generators often change the structure of the scene. This causes an undesirable change in the semantics. We propose three ways to cope with this problem depending on the type of degradation.

1. Introduction

Generative models can be used to produce new samples from the distribution represented by a given dataset. There are three important approaches to generative modelling: Variational Auto Encoders (VAE) [12], Autoregressive (AR) models [19] and Generative Adversarial Networks (GAN) [6]. The latter is a basic building block in our method. GANs consist of two parts, a generator G and a discriminator D. During training, a vector with values randomly sampled from a normal distribution is fed to the generator network. The generator outputs an image which should be visually indistinguishable from the samples in the dataset. To evaluate this, the discriminator outputs a value that indicates how real a sample looks. Both networks are
trained simultaneously. While the objective for the generator is to fool the discriminator, the objective for the discriminator is to distinguish between real and generated samples.

Generative adversarial training is also commonly used to transform images from one domain to another. The input to the transformation network is an image of one domain. The output of the transformation network should achieve two goals. The transformed image must be indistinguishable from the real samples in the target domain and the transformation should produce semantically correct mappings.

There are several methods for applying adversarial training for a supervised transformation task. An example is superresolution in [13]. The resolution of the input image is increased by the transformation. The objective function is defined by a weighted sum of an adversarial loss and a supervised loss. A second example is transforming satellite photos to maps and segmentation maps of street scenes to photorealistic renders in [8]. In this method the discriminator looks at pairs of images of both domains. It needs to distinguish between pairs with real images from both domains and pairs with a real image from the source domain and a generated image from the target domain.

Both these methods are characterised by the fact that the models are trained in a supervised manner because corresponding pairs of images from both domains are available.

For unsupervised domain transfer the authors in [11] and [21] proposed to train two transformation and two discriminative networks simultaneously. A transformation network is built for each direction, an A2B and a B2A transformer, even though only the A2B transformation is required for some applications. The goal for each discriminator is to distinguish between the real and transformed images for their respective domain. The novel idea is that by transforming an image from one domain to the other and then back it should result in a good reconstruction of the original image. To enforce this, the authors introduced a cycle consistency loss. The total loss is the sum of the generator, discriminator and cycle consistency loss $L_{cyc}$.

$$L = L_G^{A2B} + L_G^{B2A} + L_D^A + L_D^B + L_{cyc} \quad (1)$$

$$L_{cyc} = \lambda_{cyc}^A \cdot \|a - a_r\|_1 + \lambda_{cyc}^B \cdot \|b - b_r\|_1 \quad (2)$$

with $\lambda_{cyc}^A$ and $\lambda_{cyc}^B$, the weights for balancing the consistency loss to the discriminator and generator losses. In [21] the authors use an L1 distance between the original \((a, b)\) and the reconstructed image \((a_r, b_r)\) after a complete cycle, with \(a \in A\) and \(b \in B\). Other unsupervised domain transfer approaches focus on small domain shifts without high level changes or geometric variations [2].

There are three important applications covered in this work, as illustrated in Figure 1. In all of them, the goal is to enhance the images of traffic scenes that are degraded by natural causes. The first application is clearing up rainy traffic scenes, which encompasses removing raindrops and making the overall scene brighter. The second application is transforming foggy traffic scenes. The fine details of the foreground objects should be reconstructed and give a hint of what is obfuscated by the fog in the distance. In the third application the goal is to transform night near-infrared traffic imagery to clear bright daylight images. Since a common denominator of these problems is that it is very hard to collect corresponding paired images from both domains, supervised techniques can not be applied. It is however relatively easy to obtain datasets with unpaired images with degraded visibility and images taken in perfect conditions.

For all these applications, it is of prime importance that the semantics of the traffic situation do not change when the image is enhanced. Cars, pedestrians, buildings and trees need to be in the same location and retain their original shape. This is a problem with the current domain transfer approaches. If the transformers are trained with the cycle consistent adversarial method, the original semantic structure can completely change after transformation. The results are shown in Figure 10 and in more detail in Section 10.

We propose three techniques to retain the structure and semantics of the traffic scene. In the first two techniques, we explicitly force that commonalities between domains are retained during translation.

If the domains are close to each other and the degradation has small features, like noise, then the perception should be the same in both domains (Section 4). Optimising for perception loss during training can guide the transformation network to keep the important features in the image. In case the domains are further apart in terms of color mapping, this does not work as well anymore. In that case forcing the transformer to keep the edges of the original image can provide good results as can be seen in Figure 10 and in Section 5. In the last technique, domain knowledge is exploited by generating additional information that the discriminator network can use to make a better prediction. Even if it is only for one of the domains, it can help the whole cycle. In the night-to-day vision application, we use a pre-trained traffic scene segmentation network. The segmentation network is trained on images of traffic scenes during the day. For the daylight domain, segmentation maps are generated for both the transformed images and the original images and fed to the discriminator (Section 6). Improvements to the transformation network architecture are described in Section 7. In the next we introduce the three benchmark applications.

### 2. Similarities and Difficulty Level

Most of the enhancement tasks require multiple transformation subtasks. We consider three distinctive sub-
tasks. Colour mapping/reconstruction, structural reconstruction and noise reduction. The applications covered in this work require all of the aforementioned subtasks but in different amounts. An overview is shown in Table 1.

Table 1: Difficulty of the transformation subtasks for each application. More plusses indicates a higher difficulty level

| Application       | Noise reduction | Structural reconstruction | Colour mapping/reconstruction |
|-------------------|-----------------|---------------------------|------------------------------|
| Removing fog      | +               | ++                        | +                            |
| Removing rain     | +++             | ++                        | +                            |
| Night to day      | +               | +++                       | +++                          |

Images containing fog require the aforementioned transformations but in a moderate amount. Usually, details in the foreground of the image are clearly visible whereas objects at a further distance lose their colours and structural details in proportion to their distance to the camera. In case of rainy scenes, the biggest problem is the noise made by rain streaks and the deformations made by raindrops. Colours are slightly greyer than in a clear scene.

The night-to-day transformation is the hardest problem. In the black areas the structure is completely lost or the weak features are drowned in noise. A major problem with night to day vision is the dynamic lighting effects caused by the streetlights and car lights, as is illustrated in Figure 2. In very dark scenes, without a lot of street lights, the sky is typically lighter than the surrounding objects. With street lights and car headlights, the sky is typically black. The same holds for tree tops and bushes. Light reflection can cause part of a tree top to lighten up and without street lights it is typically darker than the sky. This makes the colour mapping extremely difficult.

3. Transfigurative and Mapping Artifacts

In this section we describe two common types of artefacts that occur when the vanilla CycleGAN method is used and the main causes of the artefacts.

In the basic unsupervised adversarial learning method for domain transfer two transformers and two discriminators are trained simultaneously. If we train our domain transfer system with a set of unpaired images of daylight and night traffic scenes in a traditional approach, then the goal of the A2B generator is to transform the night image into a day scene and the B discriminator is tasked with making a distinction between real and generated day images. However, it is not explicitly defined that it should keep the original structure in the night scenery. So in practice, the images produced by the generator are completely transfigured. If the system is trained while also trying to minimising the cycle consistency loss [21], then the results improve. Transforming an image from domain A to domain B and transforming it back to domain A should produce an image close to the original image. The A2B transformation should be the inverse of the B2A transformation and vice versa. Some of the structure is implicitly kept because of the cycle consistency loss, but there are still severe changes in the semantics of the scene which makes it practically unusable. The night to day transformation deforms cars and pedestrians as can be seen in Figure 10, 12, 13, 14 and 15. The cycle consistency loss does not guarantee that the semantic structure of the input image will be similar in the different domains. It only imposes that the A2B generator and B2A generator transformations should be inverses of each other.

Another frequently occurring problem is incorrect mappings. This is a major problem for night to day transformation. An important cause is the dynamic lighting effects caused by the streetlights and car lights, as is illustrated in Figure 2. In very dark scenes, without a lot of street lights, the sky is typically lighter than the surrounding objects. With street lights and car headlights, the sky is typically black. The same holds for tree tops and bushes. Light reflection can cause part of a tree top to lighten up and without street lights it is typically darker than the sky. These problems occur more prominently in domains that are further apart from each other, because an easy colour transformation is not possible. The incorrect mappings are not really discouraged by the CycleGAN loss scheme.

4. Minimising the Perceptual Distance

The first method to retain as much of the original image meaning during domain transfer is minimising the perceptual distance between the generated image and the original image. The perceptual distance is calculated by feeding both the original and transformed image through a pre-trained classification network. The weights of the classification network are fixed. Feature maps at different spatial resolutions are taken and compared. A predefined distance metric is used to compare these feature maps between original and generated image. The final perceptual distance is a weighted sum of the distances between the feature maps at different spatial resolutions. By minimising the perceptual distance we implicitly force the transformed image to contain the same meaning as the original image as perceived by
We are not the first to introduce perceptual distance which has also been called perceptual loss or content loss in previous work. However it has mainly been used in supervised problems. Example applications are increasing the resolution with SRGAN [13] and transferring style from one image to another. A form of perceptual distance was first used in [5]. The authors used layers of pre-trained classification network to generate textures. In [2] Bousmalis et al. use a masked pairwise mean squared error in an unsupervised domain transfer application similar to our approach but with a different metric.

In Figure 3, a schema summarises how the perceptual distance is minimised during domain transfer. The original and transformed image are fed to a pre-trained 19-layer VGGNet. In previous work, the activations of the convolutional layers right before the maxpool layers are chosen. For each feature map the mean squared distance is calculated between the activations caused by feeding the original and transformed image. The total perceptual distance is then a weighted sum of the L2 distances. This perceptual distance is used as a loss function and added to the transformation loss.

\[
L_G^{A2B} = L_G(D_B(G_{A2B}(a))) + \lambda_{feat}^A \cdot \Delta_P(a, b_I)
\]

with \(\Delta_P(a, b_I)\), the perceptual distance between the original image from domain A and the generated image.

This works well for images that are degraded by noise. An example is removing rain drops and streaks. Depending on the size of the noise features we can adjust the weights. If there are only very small noise features we only have to reduce the weight of the first layers. A downside of this approach is that the weights are new hyper parameters. It is cumbersome to optimise and trade off these weights. The higher abstract features are important to keep the high level semantics intact, but lower hierarchy features are important for spatial precision. To overcome this we replaced the VGGNet with a residual network. We only used the activations of the last convolutional layer in a pre-trained 34-layer residual network to determine the perceptual distance. In residual networks smaller features better propagate to higher levels because of the skip connections, so there is already a better implicit weighing between smaller and larger features. The Resnet features performed better than our manually fine tuned VGG-based perceptual distance functions.

In Figure 4 a comparison shows the clear difference between the results of a rain-to-clear transformation network trained with and without minimising perceptual distance. Unfortunately, minimising the perceptual loss only works if the domains are close to each other in terms of structure and if both domains are close to the domain for which the perceptual model is trained. It fails often for the night-to-day application, because the classifier is not trained with near-infrared images of night scenes.

5. Edge Features

A large part of the semantics of an image can be summarised by its edge features. A human can easily distill the content of an image by looking at an edge map of the image, an example is shown in Figure 5. In some domain-transfer
applications it is desirable that the edge features are preserved. An example is the night to day transformation. A counter example is brightening and clarifying rainy scenes. For this application, it is not desirable to keep edges during transformation, because rain streaks have strong edge features.

As can be seen in Figure 5, the edge map of the daylight scene and the night scene are hard to distinguish. It is difficult to distinguish whether it is based on a daylight or night scene. The one clear difference is that daylight photos typically produce more edges compared to night photos. So the edge features can be used to guide the learning process for the domain transfer models. We propose the following edge feature scheme for the night-to-day. The night-to-day transformer should preserve the edges of the night image. This guides the transformer to produce daylight images with the same structure. For the day-to-night we propose to force the transformer to not introduce new edges, which prevents it from hallucinating new structures and thereby changing the semantics of the scene. For the removing fog application we propose to use the same scheme. The fog-to-clear transformation should preserve even the slightest hint of an edge during the fog-to-clear transformation, but there should not be any new edges introduced by the clear-to-fog transformation.

Several published articles described on a convolutional network architecture for detecting edges in an image. Notable examples are PixelNet [1] and Holistically-Nested Edge Detection (HED) [20]. We tried both and choose the latter to base our edge detection model on. The reason being, HED produced edge maps with more fine edges and details than PixelNet. This is also indicated by Bansal et al. in [1], where both edge detection models are compared. The original HED architecture uses the pre-trained feature layers of a 16-layer VGG network. A VGGNet is relatively memory intensive, so we replaced it with the feature layers of a residual network with 34 layers.

Both the original image \((b)\) and the generated image \((a_f)\) are fed to the edge detection model, as shown in Figure 6. The resulting edge maps are then used to define the following edge feature based losses. They are based on the squared error between both edge maps, because small changes in edge features are relatively less important than major changes. The first one is the edge preservation loss \((L_{EP})\).

\[
\text{err}(b,a_f) = \text{ED}(b) - \text{ED}(a_f) \tag{3}
\]

\[
\text{pos}(b,a_f) = \frac{1 + \text{sign}(\text{err}(b,a_f))}{2} \tag{4}
\]

\[
\text{f}_{\text{bal}} = \sum_{i,j}(1 - \text{ED}(b)(i,j)) \frac{1}{W \cdot H} \tag{5}
\]

\[
L_{EP}(b,a_f) = \text{f}_{\text{bal}} \cdot \|\text{pos}(b,a_f) \cdot \text{err}(b,a_f)\|_2 \tag{6}
\]

with \(\text{ED}\), the edge detection model, \(W\) and \(H\), the width and height of the image.

The second factor indicates that only if there is a positive error, the loss should be taken into account. Only if the edge features are less strong in the transformed image the loss is larger than zero. The transformer should not be punished if the edges are stronger in the target domain. Any hint of an edge could be important for the reconstruction of the scene, and typically they should be made stronger in the night-to-day and fog-to-clear transformation. The first factor is a balancing factor. If there are only a small amount of edges in the edge map, then it is more important that they are preserved when there are a lot of edges in the edge map.

The edge introduction loss \((L_{EI})\) is the second edge feature based loss and it should prevent the introduction of new edges after transformation.

\[
\text{neg}(a,b_f) = \frac{1 - \text{sign}(\text{err}(a,b_f))}{2} \tag{7}
\]

\[
L_{EI} = \sum_{i,j} \frac{\text{ED}(x)(i,j)}{W \cdot H} \cdot \|\text{neg}(a,b_f) \cdot \text{err}(a,b_f)\|_2 \tag{8}
\]

In the second factor the sign function is used to only account the squared error between the edge maps when there is a negative error. This means that this error will only be taken into account when the edges are stronger in the transformed image than in the original image. The first factor is the balancing factor. It is based on the number of edges in the input image. Edges are relatively sparse features.

### 6. Generating Semantically Sound Images

Although transferring the edge features majorly improves the resulting transformed images, there are still some semantic mistakes being produced by the transformer. Common artefacts include buildings with trees built in and trees with some building features in between the branches without there being a building behind the tree. An example can be seen in Figure 11.
To generate images with fewer semantic mistakes, we suggest to exploit expert models of one of the domains, typically the clean domain. We propose to provide more relevant information to the discriminator to implicitly force the transformer to produce images that are semantically correct or at least more plausible.

An ideal candidate for an expert network is a semantic segmentation network. Additionally the segmentation maps have the same resolution as the input image, so they can easily be concatenated with the input image before feeding it to the discriminator. Other expert networks like classifiers could also provide more semantic information to the discriminator. Intermediary feature maps could be fed to the discriminator, but this requires changes to architecture because the feature maps typically have a smaller spatial resolution than the input image.

In Figure 7 there is an illustration of the proposal. If we have a pre-trained segmentation model for the clear domain, we can use it to generate segmentation maps for both the original images in the clear domain and the generated fake images. We feed images and their respective segmentation maps to the domain discriminator. The domain discriminator is then trained to distinguish between real and generated images. If the segmentation map of the generated image does not make sense, it will be easier for the discriminator to detect the fake images. The result is an image with fewer semantic mistakes. An additional advantage of feeding the segmentation maps to the discriminator is that they stabilise the training process. The segmentation maps are high-quality features that are not trainable.

Several publications focus on segmenting traffic scene imagery. Approaches to semantic segmentation have been published in [9], [15] and [16]. We chose a FC-DenseNet architecture, because it is a recent good performing segmentation architecture and requires few parameters. We trained it to segment the traffic imagery in the CityScapes dataset [4]. One small architectural change helps the discriminator even more. A segmentation network typically outputs a binary segmentation map per class. The last non-linearities of a FC-DenseNet are pixel-wise softmax. We changed this to sigmoid nonlinearities. It enables the model to output whether a pixel is part of both a building and a tree, which is not possible in real images, but it can occur for generated images and this helps the discriminator to find inconsistencies.

7. Improvements to the Architecture

There are two common transformation network architectures for Cycle GAN. A U-Net based architecture [16], an architecture originally used for segmentation tasks, and an architecture built with residual blocks [7]. We propose to use an architecture built based on the FC-DenseNet model [9], which is also used for segmentation. It outperforms U-Net based segmentation networks and is based on the dense block architecture.

We tweaked the architecture to make it more suitable for transforming images. In the original FC-DenseNet architecture, the spatial resolution is increased by transition up modules. The transition up module consists of a 3x3 transposed convolution layer. The upsampled feature maps are concatenated with the feature maps coming from the skip connection. Together they form the input to a new dense block. We replaced the transposed convolution upsample method with sub-pixel convolutional upsampling, which is first introduced in [18] for increasing the resolution of video and images.

For the night to day application, we noticed some artefacts where a large part of the scene is black. The transformer defaults to the most probable mapping for black areas, which is typically the sky or tree top patches. If this black patch does not occur at the top of the image this can lead to unrealistic transformations. To overcome this issue the field of view is maximally enlarged. Adding more convolutional layers increases the field of view linearly, while introducing more parameters and a higher memory consumption. Adding more downsampling levels is more effective, because it quadratically increases the field of view. However, it is limited by the size of the input patch. We increase the number of levels to match the size of the input patch and keep the same number of layers per level. The number of downsampling levels and the training crop sizes used during training is reported in Table 7.

8. Experiments

8.1. Datasets

To create a labeled dataset we used a simulation environment based on GTA V v1.4 enhanced by several mods to increase photorealism and weather simulations. We further modified GTA to be controllable by an external script. It allowed us to automatically collect images from many different sceneries and weather conditions. Without a simulation, it would not be feasible to gather image pairs of different
weather conditions. Even though all experiments are performed under unsupervised conditions, having image pairs in the validation set allows to better assess the performance of the proposed techniques. For the night scenes, we developed a shader for an infrared night vision effect, based on [3]. For fog we used the volumetric effect provided by the NaturalVision mod. The rainy weather condition in modified GTA engine did not provide a convincing effect, so we used part of the public dataset called Synthia1 [17]. Synthia1 contains images of traffic scenes. Some of the images are taken during rain showers. The images have strong rain streaks including raindrops on the camera lens. After selection and preprocessing, the dataset contains roughly 500 rainy and 500 sunny training images. The test set contains 75 images each. The downside is that the dataset does not have corresponding pairs.

We also collected a dataset with real world night images. Images were captured while driving around in an urban environment. During the day a regular RGB camera and during the night a Near Infra Red camera was used. After curating the dataset contains around 5000 night images and 5000 day images.

8.2. Quantitative Comparison

The proposed techniques are benchmarked with the labeled datasets. We compare four different configurations.

- **Cycle+Resnet**: the baseline configuration with a basic cycle-consistency adversarial approach with a generator built out of residual blocks
- **Edge+ResNet**: In this configuration the edge losses are added during training
- **Edge+FCDenseNet**: The edge losses and our new FC-DenseNet based transformation architecture described in the previous section.
- **Perc+FCDenseNet**: The new FCDenseNet based transformation architecture trained while minimising the perceptual distance

For the exact parameters of the configurations we refer to the additional materials section.

To benchmark the configurations the perceptual distance between the transformed image and the real image is calculated. We used a pre-trained 16-layer VGGNet and used equal weights for the different feature maps.

In Figure 8 and Figure 9 the box plots summarise the results of the different configurations. For both applications, our proposed techniques clearly outperform the baseline and the Edge+FCDenseNet configuration is the best performing configuration. Another similarity is that the FC-DenseNet architecture also outperforms the ResNet based architecture. The biggest difference between the both tasks is that minimising the perceptual distance works almost as good as the Edge+FCDenseNet configuration for removing fog, but clearly does not help much in case of the night-to-day transformation.

We also observed that there is a large difference in average perceptual distance for the two applications. The average distance for the night-to-day transformation is higher than the fog-to-clear transformation. This is an indication of how difficult the night-to-day transformation is compared to the fog-to-clear transformation.

8.3. Qualitative Comparison

In the qualitative comparison we compare the Edge+FCDenseNet configuration with the original Cycle+Resnet configuration on the real image dataset for the night-to-day transformation task. A few samples are shown in Figure 10. For more high resolution samples we refer to the additional materials section. The transformation results of the Cycle+Resnet approach typically have transmutation artefacts. The model hallucinates new buildings and changes the perspective in the image. Cars are deformed and the semantics of the scene are changed. Our new technique clearly outperforms Cycle+Resnet. The structure is not changed. The scene is overall brighter and it is easier to understand the semantics of the scene.
It still has some problems with the glares coming from street lights. They produce sharp edges, as can be seen in Figure 5, and these edges are enforced in our training process. This leads to unnatural transitions between air and tree tops. Another problem is the completely dark areas where our model sometimes hallucinates dark green tree foliage.

To further improve the semantics of the produced images we proposed to feed semantic segmentation maps to the discriminator. In Figure 11 some samples are shown. We compare the Edge+FCDenseNet configuration with and without the semantic maps feed to the discriminator. For some samples this helped the colour precision as can be seen in the top two rows in Figure 11. If we further zoom in on the images in the second row, then you can observe trees illuminated by street lights have a white colour with the old configuration and in the new configuration this changed to green. The train tracks that were coloured green are now correctly coloured grey. However, for half of the images the semantic feed did not really improve the result as can be seen in the bottom row images.

9. Conclusion

We introduced new techniques to improve the training of domain transfer models, targeted to enhance images of traffic scenes taken in conditions with limited visibility. We covered degraded visibility caused by rain, fog and the lack of light during the night.

For domains that are close enough in terms of colouring, we propose to force the generator to minimise the perceptual distance between the original image and the transformed image. This is applicable to images of rainy and foggy scenes. For domains that are further apart, like daylight and near infrared images during the night, we propose two separate techniques that can be used in tandem. In the first technique we use the edge features of the original image and the transformed image. It can be beneficial to force the same edge features after transformation or to prevent any new edge features from emerging during transformation. In the second technique, the discriminator is provided with high quality fixed non trainable features. A pre-trained semantic segmentation model is an ideal candidate because it gives pixel-wise high quality features that can be easily merged together with the image and fed to discriminator.

We also showed that synthetic datasets generated by gaming engines can help guide us to develop and train a better domain transformation models. We generated images of scenes taken in poor visibility conditions and the same scene in perfect conditions. This allows us to benchmark our unsupervised domain transfer approaches. Finally, we validated the results on a datasets with images of real traffic scenes during the day and the night and showed superior performance compared to the original cycle-consistent adversarial approach.

References

[1] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. arXiv:1702.06506, 2017.
[2] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. arXiv preprint arXiv:1612.05424, 2016.
[3] M. Bugała. Implementing the night vision effect with the use of a depth map. *Szybkobieżne Pojazdy Gasienicowe*, (2(40)):89–95, 2016. 7

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016. 6

[5] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015. 4

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1

[7] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016. 6

[8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016. 2

[9] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1175–1183. IEEE, 2017. 6

[10] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[11] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *CoRR*, abs/1703.05192, 2017. 2

[12] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[13] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016. 2, 4

[14] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 6

[16] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. 6

[17] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3234–3243, 2016. 7

[18] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016. 6

[19] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixel-cnn decoders. In *Advances in Neural Information Processing Systems*, pages 4790–4798, 2016. 1

[20] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015. 5

[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017. 2, 3
10. Additional Results and Documentation

10.1. Parameter Values

In this section we describe the parameter values used to generate the images produced in this work. In Table 9 an overview is given of the most important parameters and their symbol or abbreviation. The parameters that stay constant for all our configurations are summarised in Table 2. We mainly consider three configurations in this work. For each configuration the parameter values are listed in a separate table. Table 3 contains the parameter values for the baseline configuration with the cycle consistency loss. Table 4 contains the parameter values for the configuration with an additional perceptual distance loss, which corresponds to the Perc+FCDenseNet configuration in the experiments section. Table 4 lists the parameter values for the configuration with the additional edge preservation and edge introduction losses, which corresponds with the Edge+FCDenseNet configuration in the experiments section. The constant parameters and the parameters that have a value of zero are omitted for clarity.

Table 2: Constant parameters

| Parameter | Value |
|-----------|-------|
| $N_{iter}$ | 100   |
| $N_{iter,decay}$ | 100 |
| $L_r$ | 0.0002 |
| $\beta_1$ | 0.5 |
| Pool size | 50 |

Table 3: Training parameter values for the cycle approach for the real night to day task.

| Parameter | Value |
|-----------|-------|
| $\lambda_{cy,A}$ | 10 |
| $\lambda_{cy,B}$ | 10 |

Table 4: Training parameter values for the cycle + $\Delta P$ approach.

| Parameter | Value |
|-----------|-------|
| $\lambda_{cy,A}$ | 10 |
| $\lambda_{cy,B}$ | 10 |
| $\lambda_{p,afb}$ | 0.25 |
| $\lambda_{p,bfa}$ | 0.25 |
| $\lambda_{p,farb}$ | 0.25 |
| $\lambda_{p,fbra}$ | 0.25 |

During training the images in the datasets are first down-sized and a random crop is taken. The datasets with synthetically generated images were preprocessed so that down-sizing was not necessary anymore. The load sizes and the size of the cropped patches for each configuration are summarised in the following table.

Table 5: Training parameter values for the cycle + edge approach.

| Parameter | Value |
|-----------|-------|
| $\lambda_{cy,A}$ | 10 |
| $\lambda_{cy,B}$ | 5 |
| $\lambda_{ep,afb}$ | 100 |
| $\lambda_{ep,bfa}$ | 100 |
| $\lambda_{ei,bfa}$ | 10 |
| $\lambda_{ei,fbra}$ | 10 |

Table 6: Resolutions of the images and crop sizes during training.

|               | real/synth. | original | load | crop    |
|---------------|-------------|----------|------|---------|
| night2day     | real        | 1920x1080| 512x288| 256x256 |
| night2day     | synth       | 256x256  | 256x256| 192x192 |
| removing fog  | synth       | 256x256  | 256x256| 192x192 |
| removing rain | synth       | 256x256  | 256x256| 192x192 |

Table 7: Crop sizes and number of scales in the FCDenseNet architecture.

|               | real/synth. | crop size | # scales |
|---------------|-------------|-----------|----------|
| night2day     | real        | 256x256   | 8        |
| night2day     | synth       | 192x192   | 6        |
| removing fog  | synth       | 192x192   | 6        |
| removing rain | synth       | 192x192   | 6        |

10.2. Runtime

The new approaches, cycle + $\Delta P$ and cycle + edge require more memory and runtime during training. Each original and transformed image has to be processed by a pre-trained classifier or edge detection model respectively. The extra forward passes through the classification/edge detection model are the cause for the extra runtime. Additionally the classification/edge detection models are loaded in the GPU so this requires more VRAM. As a consequence the maximal batch sizes decreases, which further impacts the runtime. The total runtimes and maximum batch sizes for the different approaches are reported in Table 8 for the real night-to-day transformation task. We used p3.8xlarge instances from Amazon Web Service. These instances contain 4 NVIDIA Tesla V100 GPUs. For inference the runtime of the new approaches is very similar to the original cycle method, because we only have to perform a forward pass through the transformation network, which is an advantage if the transformers are deployed in real life applications.

Table 8: Runtime of the train process for the night to day transformation on real data. Training server contains 4 x NVIDIA Tesla V100

| Approach      | Total train time | Max. batch size |
|---------------|------------------|-----------------|
| cycle         | 18h21m           | 28              |
| cycle + $\Delta P$ | 24h49m          | 16              |
| cycle + edge  | 26h00m           | 16              |
10.3. Additional Results

The results in the paper are presented in a smaller resolution due to space constraints. In this section we provide extra high resolution results for easier inspection. For each task the cycle, cycle + ΔP and cycle + edge are compared side-by-side with the original image and for the synthetic datasets the corresponding image in the target domain is also shown. The results for the night to day task are shown in Figure 12, 13, 14 and 15 for the dataset with real images and in Figure 18 and 19 for the dataset with synthetically generated images. In Figure 16 and 17 the transformed images for the removing fog application are compared. The results in the figures are the complete images processed in one time. Although the transformers are trained with slightly smaller crops, the results do not degrade a lot.
Table 9: Overview of the parameters

| Symbol/Abbrev. | Description |
|---------------|-------------|
| \( \lambda_{cy,A} \) | weight for the cycle consistency loss between an original image from domain A and its reconstruction. \((a 
\Rightarrow fb 
\Rightarrow ra)\) |
| \( \lambda_{cy,B} \) | weight for the cycle consistency loss between an original image from domain B and its reconstruction. \((b 
\Rightarrow fa 
\Rightarrow rb)\) |
| \( \lambda_{p,afb} \) | weight for the perceptual loss between an original image from domain A and its transformed fake image in domain B |
| \( \lambda_{p,bfa} \) | weight for the perceptual loss between an original image from domain B and its transformed fake image in domain A |
| \( \lambda_{p,farb} \) | weight for the perceptual loss between a transformed fake image in domain B and the reconstructed image in domain A |
| \( \lambda_{p,fbra} \) | weight for the perceptual loss between a transformed fake image in domain B and the reconstructed image in domain A |
| \( \lambda_{p},afb \) | weight for the perceptual loss between an original image from domain A and its transformed fake image in domain B |
| \( \lambda_{p},bfa \) | weight for the perceptual loss between an original image from domain B and its transformed fake image in domain A |
| \( \lambda_{ep,afb} \) | weight for the edge preservation loss between an original image from domain A and its transformed fake image in domain B |
| \( \lambda_{ep,afb} \) | weight for the edge preservation loss between a transformed fake image in domain B and the reconstructed image in domain A |
| \( \lambda_{ep,farb} \) | weight for the edge preservation loss between a transformed fake image in domain B and the reconstructed image in domain A |
| \( \lambda_{ep,fbra} \) | weight for the edge preservation loss between a transformed fake image in domain B and the reconstructed image in domain A |
| \( \lambda_{ei,afb} \) | weight for the edge introduction loss between an original image from domain A and its transformed fake image in domain B |
| \( \lambda_{ei,bfa} \) | weight for the edge introduction loss between an original image from domain B and its transformed fake image in domain A |
| \( \lambda_{ei,farb} \) | weight for the edge introduction loss between a transformed fake image in domain B and the reconstructed image in domain A |
| \( \lambda_{ei,fbra} \) | weight for the edge introduction loss between a transformed fake image in domain B and the reconstructed image in domain A |
| Load size | input images are scaled to this size |
| Crop size | the size of the random crop taken out of the resized image during training |
| \( N_{iter} \) | Number of iterations at start learning rate |
| \( N_{iter,decay} \) | Number of iterations to linearly decay learning rate to zero |
| \( Lr \) | Initial learning rate for adam |
| \( \beta_1 \) | Momentum term of adam |
| Pool size | The size of image buffer that stores previously generated images for the discriminator |
Figure 12: Different approaches for the night to day transformation on real images, part 1/4
Figure 13: Different approaches for the night to day transformation on real images, part 2/4
Figure 14: Different approaches for the night to day transformation on real images, part 3/4
Figure 15: Different approaches for the night to day transformation on real images, part 4/4
Figure 16: Different approaches for removing fog, part 1/2
Figure 17: Different approaches for removing fog, part 2/2
Figure 18: Different approaches for the night to day transformation on synthetically generated images, part 1/2
Figure 19: Different approaches for the night to day transformation on synthetically generated images, part 2/2