Unsupervised Image Transformation Learning via Generative Adversarial Networks

Kaiwen Zha[1], Yujun Shen[2], Bolei Zhou[2]
[1]MIT CSAIL [2]The Chinese University of Hong Kong
kzha@mit.edu, {sy116, bzhou}@ie.cuhk.edu.hk

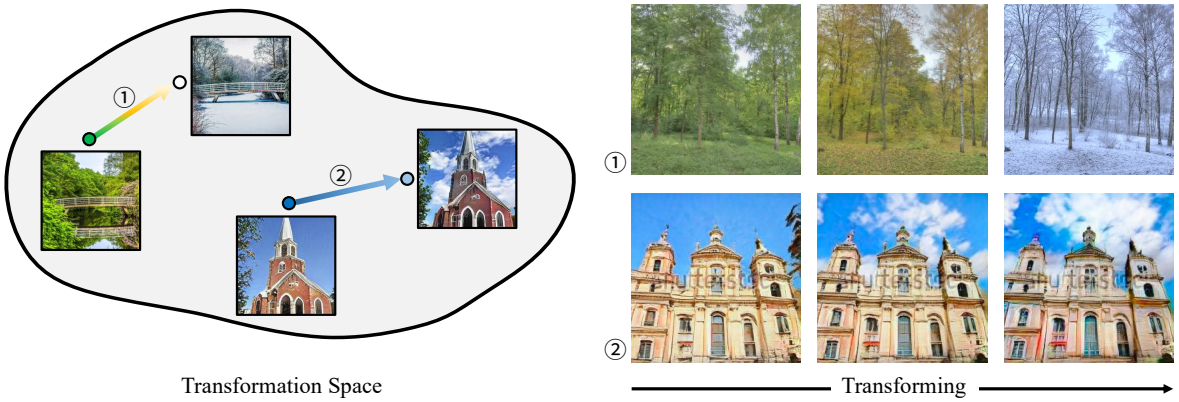


Figure 1. Given an image pair, TrGAN is able to project them onto the learned transformation space and further extract the semantic variation between them. Such variation is then applied to transforming new images. Examples on the right show that TrGAN can both transfer image styles (*i.e.*, changing seasons) and, more importantly, edit image contents (*i.e.*, adding clouds). In addition, the transformation process is semantically continuous, like yielding fall as the intermediate step when transforming from summer to winter.

Abstract

In this work, we study the image transformation problem by learning the underlying transformations from a collection of images using Generative Adversarial Networks (GANs). Specifically, we propose an unsupervised learning framework, termed as TrGAN, to project images onto a transformation space that is shared by the generator and the discriminator. Any two points in this projected space define a transformation that can guide the image generation process, leading to continuous semantic change. By projecting a pair of images onto the transformation space, we are able to adequately extract the semantic variation between them and further apply the extracted semantic to facilitating image editing, including not only transferring image styles (e.g., changing day to night) but also manipulating image contents (e.g., adding clouds in the sky). Code and models are available at https://genforce.github.io/trgan.

1. Introduction

The visual world is constantly changing. The pass of autumn turns the golden color of trees and fields into white, while the sunrise comes to light up the river and sky. Factors such as season and light drive the visual transformations. Recent development of deep models has proven to be remarkable in extracting the explanatory factors in a single image [2], but the machine still lacks the capability of understanding how images are transformed from one to another. For example, given a pair of landscape photos, taken in summer and winter respectively, human can not only tell it is the season that varies between the two, but also foresee the effect of season change for a new set of images. However, it remains challenging for the machine to recognize such an abstract transformation, let alone apply it to transforming new images.

Compared to the traditional classification or generation task, it is more difficult to learn transformations. Firstly, a transformation is often defined by a pair of observations, either of which change may result in a new transformation. Secondly, there are plenty of transformations existing in the visual world thus it is impractical to manually label every single one and train a supervised model. Thirdly, transformation usually follows a continuous semantic variation instead of a binary mapping. A good transformation from

one image to another should result in an interpolation effect that is semantically meaningful. To transform an image, prior work either involves additional labels [38, 22] or can only perform limited transformations with one model [14, 45, 12, 26], limiting the generalization ability.

To overcome these obstacles, we propose *TrGAN*, an unsupervised transformation learner based on Generative Adversarial Networks (GANs). Like other GAN variants, TrGAN employs a generator and a discriminator to compete with each other on the task of image synthesis. Differently, TrGAN introduces a new *transformation space* beyond the original latent space in GANs. Each individual image corresponds to a point in this transformation space, and any two points define a particular transformation. In addition to differentiating real data from synthesized data, the discriminator also learns the mapping from the image space to the transformation space. In this way, given an image pair for inference, we can use the discriminator to embed both of them to the projected space and extract the semantic variation between them, as shown in Fig. 1. Such variation can in turn be used to guide the generation.

Our contributions are summarized as follows:

- We propose TrGAN, by introducing a transformation space shared by the generator and the discriminator, to learn image transformations in an unsupervised manner. Given a pair of images, we are able to adequately extract the semantic variation between them.
- We are able to apply the semantics that are extracted from any customized image pair to transforming new images. More importantly, the transformation can be interpolated continuously and remain semantically meaningful at the intermediate step, outperforming existing style transfer approaches.
- We find that the proposed transformation space is robust to not only support transferring image styles, but also allow editing image contents, as shown in Fig. 1. Furthermore, we study the compositionality of various transformation types, which sheds light on the underlying structure of the transformation space.

2. Related Work

Image Transformation. Image transformation has been a long-standing topic, whose primary goal is to alter images with certain types of variations. Some early studies [34, 10, 37] in this field adopt image analogy to transform images. Laffont *et al.* [22] manually defines 40 transient attributes and employs labeled data for better editing. Recent advance of neural networks enables high-quality image-to-image translation [14, 45, 27] and style transfer [7, 17, 12, 29, 25, 26]. Basically, they borrow the content information from one sample and the style information from another sample (or another domain) to produce a fused image. This idea is further extended to learn a multi-modal image translator [46, 24, 13, 30], improving the synthesis diversity. All these approaches focus on varying the style and appearance of the image while the content remains the same. Some other work [32, 42] designs scene attributes, which is highly related to the content information, to better characterize the object variations in the images. Zhan *et al.* [40] particularly studies how to harmoniously add an object to a given image.

TrGAN differs from existing approaches with three main **improvements**. (i) Unlike prior work [14, 45, 13, 28] that needs to pre-know the transformation type (*e.g.*, labels or domains) before training, TrGAN *unsupervisedly* learns a transformation space to sufficiently discover *all potential* transformations from the training set. It therefore supports characterizing the transformation from any customized image pair (*i.e.*, users can casually choose their own images of interests) rather than relying on categorical labels or numeric parameters that can only define limited transformation types as in [38, 22], which enables broader application scenarios. (ii) Instead of transferring style with a simple one-to-one mapping, TrGAN is able to *continuously and semantically* transform images regarding a particular variation. (iii) As shown in Fig. 1, TrGAN can not only transfer image styles, but also manipulate image contents, benefiting from the diversity and robustness of the learned transformation space.

Generative Adversarial Networks (GANs). Recent years have witnessed the advance of GANs [9] in image synthesis [41, 5, 4, 18, 19, 20]. Starting from a pre-defined latent distribution, GANs model the underlying distribution of the observed data through adversarial training. It has been recently found that GANs can encode various semantics in the latent space, facilitating image manipulation [8, 16, 36, 38]. But they require pre-trained classifier to identify the latent semantics and further use learning-based [44] or optimization-based [1] methods to project an image back to the latent space. The misalignment between the native latent space and the projected space limits their editing capability to some extent [43]. Instead, TrGAN explicitly introduces a transformation space, which is shared by the generator and the discriminator, to directly learn the underlying transformations from training data. Meanwhile, the discriminator takes over the inverse mapping from the image space to the transformation space to support extracting variations between customized image pairs.

3. TrGAN

Fig. 2 illustrates the framework of TrGAN. Besides the competition between the generator and the discriminator, we introduce a transformation space $\mathcal{T}$ to bridge them together. Given a latent code $\mathbf{z} \in \mathcal{Z}$ and a transformation code $\mathbf{t} \in \mathcal{T}$, the generator maps them to a photo-realistic image $\mathbf{x}^{syn}$ that corresponds to a certain point (*i.e.*, $\mathbf{t}$) in

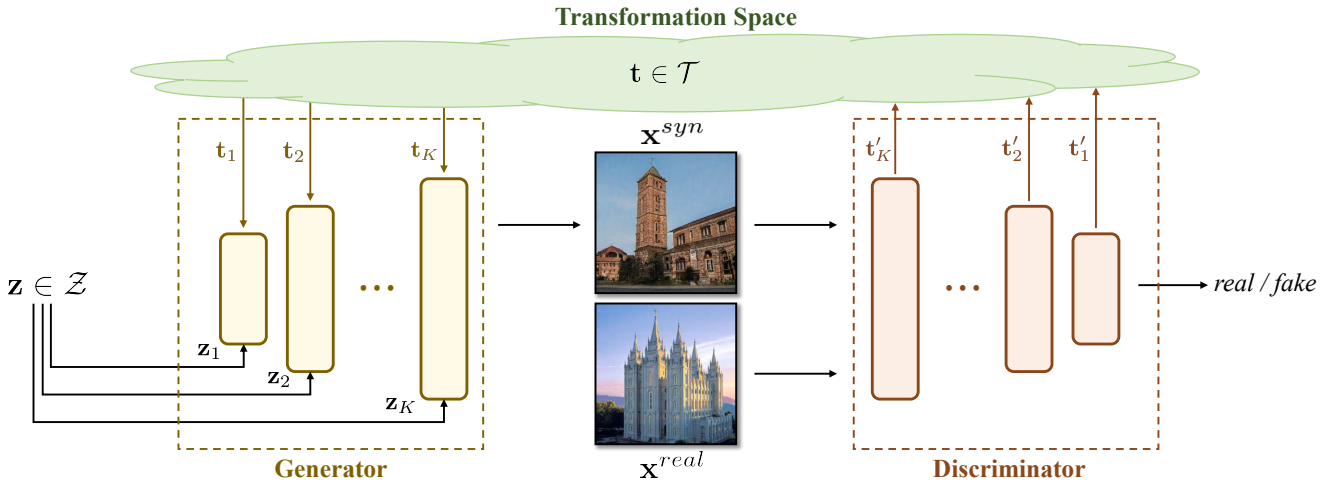


Figure 2. **TrGAN framework**. In addition to the latent space $\mathcal{Z}$, TrGAN introduces a transformation space $\mathcal{T}$ *shared* by the generator and the discriminator. Besides competing with each other on image quality, the discriminator, as the transformation learner, is trained to project any given image to a transformation code, while the generator, as the transformation deployer, learns to employ such a code for controllable synthesis. Both the latent code $\mathbf{z}$ and the transformation code $\mathbf{t}$ are organized in a multi-scale manner across layers.

the transformation space. Meanwhile, the discriminator projects $\mathbf{x}^{syn}$ back to the transformation space and gets $\mathbf{t}'$, the approximation of $\mathbf{t}$. Such design allows the generator and the discriminator to share information. In this way, we can use the discriminator to extract transformation from any input pair and further apply it to controlling the generator.

3.1. Shared Transformation Space

Image transformation can be versatile since one transformation is determined by two images, either of which changing may lead to a completely different transformation. Let's assume a complete graph with N nodes representing N images respectively. There are $\frac{N(N-1)}{2}$ edges in total, each of which stands for a particular transformation type. Along with the dataset growing, the number of transformations will increase dramatically, making it difficult to learn the transformations defined by every single pair. However, these transformations show clear redundancy. Taking season changing as an example, any "summer-winter" pair should correspond to the same variation. Hence, to better model the huge diversity and remove the redundancy, we propose to project images onto a transformation space $\mathcal{T}$ instead of directly learning the transformations themselves. In this way, each image is associated with a particular point in $\mathcal{T}$ and the transformation can be defined by a vector. More importantly, this transformation space is shared by the generator and discriminator, which unifies the transformation learning and deploying process, and enables us to learn multi-scale transformations.

3.2. Transformation Learner

To map the image space to the transformation space, we directly employ the discriminator as the transformation learner considering its encoder structure. Concretely, we use its intermediate layers to learn the transformation projection, which is shown in Fig. 2. Meanwhile, according to the formulation of GANs [9], the discriminator is also assigned with the task to differentiate the real domain $\mathcal{X}^{real}$ from the synthesized domain $\mathcal{X}^{syn}$. Like most advanced GAN variants [4, 19, 20, 35, 31] that employ layer-wise latent codes, we also train the discriminator to output multi-scale transformation codes to capture more fine-grained characteristics. Specifically, given an image $\mathbf{x}$, the discriminator with K projection layers will produce

$$D_{adv}(\mathbf{x}) = p, \tag{1}$$

$$D_{\mathcal{T}}(\mathbf{x}) = \mathbf{t}' \triangleq [\mathbf{t}_1'^{T}, \mathbf{t}_2'^{T}, \cdots, \mathbf{t}_K'^{T}]^T, \tag{2}$$

where p denotes the probability that $\mathbf{x}$ comes from $\mathcal{X}^{real}$ rather than $\mathcal{X}^{syn}$ and $\mathbf{t}'$ is the projected code.

3.3. Transformation Deployer

To better bridge the image space and the transformation space, we expect any point $\mathbf{t} \in \mathcal{T}$ can be projected back to a realistic image, which we call transformation deployment. This is consistent with the primary goal of the generator. In other words, on top of taking a randomly sampled latent code $\mathbf{z} \in \mathcal{Z}$ as the input, the synthesis of the generator also depends on a sampled transformation code $\mathbf{t} \in \mathcal{T}$. Same as the aforementioned transformation projection process, both $\mathbf{z}$ and $\mathbf{t}$ are multi-scale, as $\mathbf{z} \triangleq [\mathbf{z}_1^T, \mathbf{z}_2^T, \cdots, \mathbf{z}_K^T]^T$ and $\mathbf{t} \triangleq [\mathbf{t}_1^T, \mathbf{t}_2^T, \cdots, \mathbf{t}_K^T]^T$. Then, the images are synthesized through $\mathbf{x}^{syn} = G(\mathbf{z}, \mathbf{t})$.

3.4. Loss Function

In addition to the competition between the generator and the discriminator, we would like the transformation learner

(discriminator) and the transformation deployer (generator) to share the same transformation space $\mathcal{T}$. Accordingly, we train the generator and the discriminator with

$$\begin{aligned}\min_{\Theta_G,\Theta_{D_T}} \max_{\Theta_{D_{adv}}} \mathcal{L} = \ & \mathbb{E}_{\mathbf{x}\sim\mathcal{X}^{real}}[\log D_{adv}(\mathbf{x})] \\ & + \mathbb{E}_{\mathbf{z}\sim\mathcal{Z},\mathbf{t}\sim\mathcal{T}}[\log(1 - D_{adv}(G(\mathbf{z},\mathbf{t})))] \\ & - \lambda\mathbb{E}_{\mathbf{z}\sim\mathcal{Z},\mathbf{t}\sim\mathcal{T}}[\log P(D_T(G(\mathbf{z},\mathbf{t}))|\mathbf{t})], \quad (3)\end{aligned}$$

where λ is the loss weight. $P(D_T(G(\mathbf{z},\mathbf{t}))|\mathbf{t})$ is the approximation distribution of $\mathbf{t}'$ when knowing $\mathbf{t}$. We maximize the mutual information [6] between $\mathbf{t}$ and $\mathbf{t}'$ to force the transformation learner and the transformation deployer to share as much information as possible. Here, we only apply the transformation learner onto synthesized samples since the semantic information contained in real samples is arbitrary under the unsupervised learning setting.

3.5. Transforming Images

After establishing the mapping from the image space to the transformation space, we are able to extract the semantic variation between any paired data and in turn use the discovered semantics to guide the synthesis process. More concretely, given an image pair $(\mathbf{x}_A, \mathbf{x}_B)$, we first use the transformation learner to project them onto the transformation space $\mathcal{T}$ with $\mathbf{t}'_A = D_T(\mathbf{x}_A)$ and $\mathbf{t}'_B = D_T(\mathbf{x}_B)$. These two points $(\mathbf{t}'_A, \mathbf{t}'_B)$ define a transformation type $\mathbf{d}_{AB} = \mathbf{t}'_B - \mathbf{t}'_A$. Then we can use the transformation deployer to transform any arbitrary sampled image $G(\mathbf{z},\mathbf{t})$, following $T(G(\mathbf{z},\mathbf{t})) = G(\mathbf{z},\mathbf{t} + \gamma\mathbf{d}_{AB})$. Here, $T(\cdot)$ and γ denote the transforming operation and the transforming intensity step respectively.

4. Experiments

In this section, we conduct extensive experiments to demonstrate the effectiveness of TrGAN in learning image transformations. Before that, we first introduce the datasets used as well as the implementation details.

4.1. Experimental Settings

Datasets. In our experiments, we validate TrGAN on LSUN church [39], Light Compositing dataset [3], and a time-lapse natural scene dataset we manually collected from The Webcam Clip Art Dataset [23], AMOS [15] and YouTube videos. For each dataset, we hold out 10% images as the test set. ***(i)*** *LSUN dataset* [39] is a large-scale scene image dataset consisting of 7 indoor scene categories and 3 outdoor scene categories. We choose the outdoor church, with $126k$ images in total, as the target set. ***(ii)*** *Light Compositing dataset* [3] contains 6 indoor scene categories: cafe, library, basket, house, sofas, and kitchen, with 112, 83, 129, 149, 32, 127 images respectively. In each scene (originally dark), the images are captured by a fixed camera with the scene partly lighted up by a moving light source. Here, we remove those fully lighted-up images. ***(iii)*** *Time-lapse natural scene dataset* is manually collected by ourselves from The Webcam Clip Art Dataset [23], AMOS [15], and some YouTube videos. It contains more than $100k$ natural images with drastically varying appearances. Here, we shuffle these image sequences into independent images regardless of the temporal correlation.

Implementation Details. We adopt progressive training technique [18] to train our TrGAN, where the resolution progressively grows from 4×4 to 256×256. The transformation codes $\{\mathbf{t}_k\}_{k=1}^K$ and the latent codes $\{\mathbf{z}_k\}_{k=1}^K$ are injected into the generator from the 4×4 resolution layer to the 64×64 layer ($K = 5$). Each transformation code is a 4-D vector, sampled from a uniform distribution $\mathcal{U}(-1, 1)$, while each latent code is a 32-D vector, subject to a normal distribution $\mathcal{N}(0, 1)$. We use Adam optimizer [21] to train both the generator and the discriminator. The learning rate starts from 0.001 and gradually increases to 0.002 with the resolution growing. The loss weight λ is set to 1.0.

4.2. Learning Transformations

In this part, we validate the capability of TrGAN in learning underlying transformations from image collections. We first train models on the three datasets mentioned above and use the transformation learner to project a pair of images, which are not seen in the training process, onto the learned transformation space. We then adopt the extracted transformation to control the image synthesis, as described in Sec. 3.5, to see whether TrGAN can properly identify the variation between the inference image pair.

Fig. 3 shows the results with versatile transformations. We can tell that TrGAN can handle both the relatively small dataset (*i.e.*, Light Compositing dataset [3]) and large-scale datasets (*i.e.*, LSUN church dataset [39] and Time-lapse natural scene dataset) in a completely unsupervised learning manner. Besides, results reveal that TrGAN is able to adequately extract semantically meaningful transformations from the target pair and further apply them to smoothly transforming a third image accordingly. For example, TrGAN successfully captures the style-aware variations, such as "natural light" (from bright to dim), "season" (from summer to fall to winter), "light position" (from left to right), and "brightness" (gradually lighting up the scene from dark), as shown in Fig. 3. It is also able to identify the content-aware variations, such as "altering clouds" and "adding vegetation", outperforming existing approaches that are particularly designed for style transfer [7, 26, 13, 28]. Such a large diversity of transformations are all learned without any annotations, benefiting from the novel transformation space. In this way, we can discover a great number of transformations from customized paired data with strong robustness.

Figure 3. **Diverse transformations** learned by TrGAN. The transformations extracted from the image pairs on the left two columns are applied to the images in **black** boxes. The remaining columns show the continuous transforming results.

4.2.1 Ablation on Transformation Space

As Radford *et al.* [33] has discovered the vector arithmetic property of the latent space of GANs, a straightforward baseline of TrGAN is using the latent space to replace the transformation space. In other words, given a target pair of images, we can project them to the native latent space with GAN inversion approaches [1, 43] and further adopt vector arithmetic for editing. Fig. 4 shows the qualitative comparison where TrGAN demonstrates significant improvement. For example, in the case of adding clouds, the manipulation in the latent space can only blur the sky part yet fail to alter the image contents reasonably. Also, in the second example of Fig. 4, the latent space does not support extracting the shape variation between the input pair. By contrast, after separating the transformation space from the latent space, TrGAN can put more effort into the learning of image transformations, with much stronger generalization ability.

Tab. 1 shows the quantitative comparison results. Here, we pick 10 image pairs as the target transformations. For each pair, we use the extracted transformation to manipulate 10 images by 5 steps. Then, we use Frechet Inception Distance (FID) [11] as the metric to measure the synthesis quality from each method. We also conduct a user study on the Amazon Mechanical Turk (AMT) platform to evaluate TrGAN against the baseline approach. Ten workers are asked two questions for each synthesis, *i.e.*, (i) which

Figure 4. **Qualitative comparison between the transformation space and the latent space.** (a) Transforming images by projecting the target image pair (left two columns) to the proposed *transformation space* and using it to manipulate the raw synthesis in **black** boxes. (b) Transforming images by inverting [1] the target pair of images to the *latent space* of conventional GANs and performing manipulation through vector arithmetic [33]. Samples in **yellow** boxes indicate the reconstruction results from the inverted codes.

method produces images more realistically, and (ii) which method transforms the images more semantically meaningfully. As suggested in Tab. 1, the novel transformation space is far more competitive than the vanilla latent space.

4.3. Real Image Editing

To enable real image editing with the proposed transformation space, we propose a rerendering module on top of TrGAN.

Rerendering module. Our rerendering module follows an encoder-decoder architecture to edit a real image. Different from most style transfer approaches [7, 12, 26], which require a style image as input to stylize the target image, our module is designed based on the proposed transformation space. More concretely, we borrow the multi-scale intermediate features from the generator, which is explicitly controlled by the layer-wise transformation codes, and feed them to the rerendering module to produce a photo-realistic image. Here we train this module from scratch with a well-trained TrGAN generator. Details about the rerendering module can be found in **Appendix B**.

As shown in Fig. 5 (e), our rerendering module can successfully transfer the season and daylight of real images based on the variations extracted from the reference pairs. More importantly, the transforming process is semantics-aware. For example, when altering an image from summer to winter, the learned transformation space can properly interpolate a point that corresponds to the fall season.

Table 1. Quantitative comparison between the novel transformation space proposed in TrGAN and the latent space of conventional GANs.

	FID	More realistic	More semantically meaningful
Latent space	33.69	9.4%	4.8%
Transformation space	**14.57**	**90.6%**	**95.2%**

Comparison with style transfer alternatives. We compare our method with existing style transfer approaches, including Reinhard *et al.* [34], Huang *et al.* [12] and Li *et al.* [26]. Here, for style transfer alternatives, we interpolate their style features for continuous transformation. Fig. 5 and Tab. 2 display the qualitative and quantitative comparison results respectively. We can observe that the transformation obtained by TrGAN is much closer to the ground-truth sequence, especially at the intermediate interpolation steps. Different from other algorithms that merely interpolate the color tones, TrGAN can produce semantically meaningful results in the continuous transforming process, like fall between summer and winter and dusk between dawn and night, benefiting from the unsupervisedly learned transformation space. This is also reflected in the user study in Tab. 2.

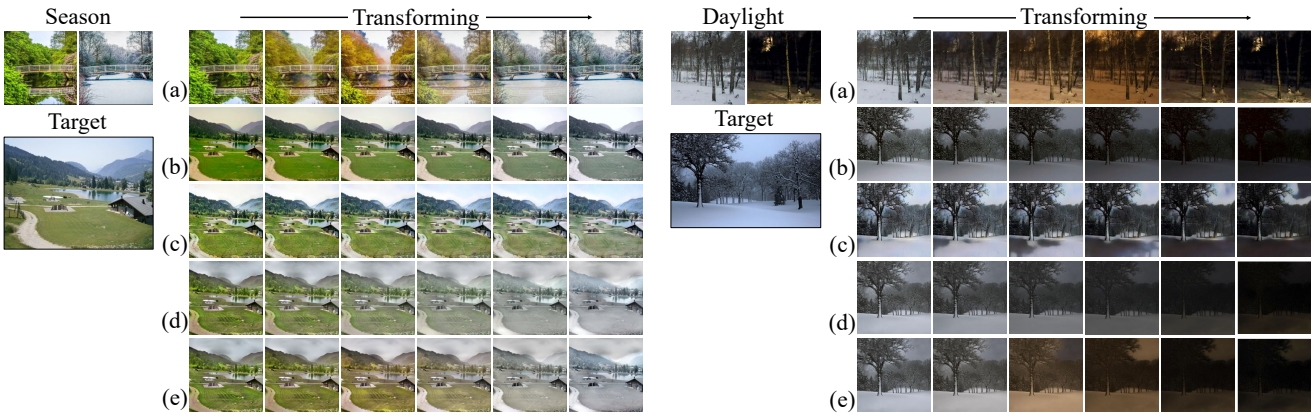


Figure 5. **Qualitative comparison with the style transfer alternatives**. For each example, the variation between the top left two images is applied to the target real image. Each row shows a sequence of varying images, indicating the gradual transforming process. From top to bottom: (a) Ground-truth sequence in the test set; (b) Reinhard *et al.* [34]; (c) Huang *et al.* [12]; (d) Li *et al.* [26]; (e) Ours. We beat other competitors, especially at the intermediate steps, by learning a more semantically meaningful transformation. Zoom in for details.

4.4. Analysis on the Transformation Space

In this section, we dive deeper into the proposed transformation space. We first explore its compositionality to see how it performs in combining different transformations. We then conduct a layer-wise analysis that sheds light on how the transformation space is organized from the layer perspective. We finally demonstrate the robustness of the transformation space in handling highly unrelated images.

Compositionality of the transformation space. Given two pairs of images $(\mathbf{x}_A, \mathbf{x}_B)$ and $(\mathbf{x}_C, \mathbf{x}_D)$, our transformation learner can project them onto the transformation space, getting $(\mathbf{t}'_A, \mathbf{t}'_B)$ and $(\mathbf{t}'_C, \mathbf{t}'_D)$. Here, we want to validate how the two transformation types $\mathbf{d}_{AB} = \mathbf{t}'_B - \mathbf{t}'_A$ and $\mathbf{d}_{CD} = \mathbf{t}'_D - \mathbf{t}'_C$ can be combined together for a fused image transformation. Fig. 6 gives two examples. We can tell that transformations learned by TrGAN are flexible for composition. Taking content editing as an example, we can add clouds and vegetation onto the image simultaneously, as shown in Fig. 6a, Besides, we can also modulate two style-aware factors (*i.e.*, season and daylight) at the same time, which is presented in Fig. 6b.

Disentanglement of the transformation space. Recall that both the transformation learner and the transformation deployer employ multi-scale transformation codes. In this part, we explore how the transformation space is organized across different layers. In Fig. 7, we show the results by altering the transformation codes along a certain direction but from different layers. It turns out that codes at different layers tend to correspond to different transformation controls that are disentangled from each other. Taking the right sample in Fig. 7 as an example, manipulation at the first layer brightens the bottom left area, while manipulation at the second layer lights up the top left part. Furthermore, jointly modulating the codes at these two layers can increase the brightness of the entire building on the left.

Table 2. Quantitative comparison between TrGAN and style transfer alternatives.

	FID	More realistic	More semantically meaningful
Reinhard *et al.* [34]	22.78	27.0%	11.0%
Huang *et al.* [12]	43.66	4.0%	3.0%
Li *et al.* [26]	18.43	32.5%	14.5%
TrGAN (ours)	**17.75**	**36.5%**	**71.5%**

Robustness of the transformation space. Sometimes, the underlying variation is not easy to spot if the pair images are not vastly related, *e.g.*, a summer picture and a winter picture that are taken at two different places. Here, we would like to evaluate whether TrGAN can handle such hard cases. As shown in Fig. 8, our model can still extract semantically meaningful transformations even if the input pair images are highly unrelated. For example, in Fig. 8a, our model can adequately discover the variations of the church shape as well as the camera angle from the reference pair. Similarly, in Fig. 8b, the season transformation can be well distilled from the target pair of images regardless of the context difference. These results verify the robustness of the learned transformation space and show that our method enables broader application scenarios.

5. Conclusion

In this paper, we present TrGAN to learn the underlying transformations from images in an unsupervised learning manner. Given an image pair, we can adequately extract the semantic variation between them and further apply it to guiding the synthesis. Experimental results demonstrate the composition property as well as the versatility of the proposed transformation space in TrGAN.

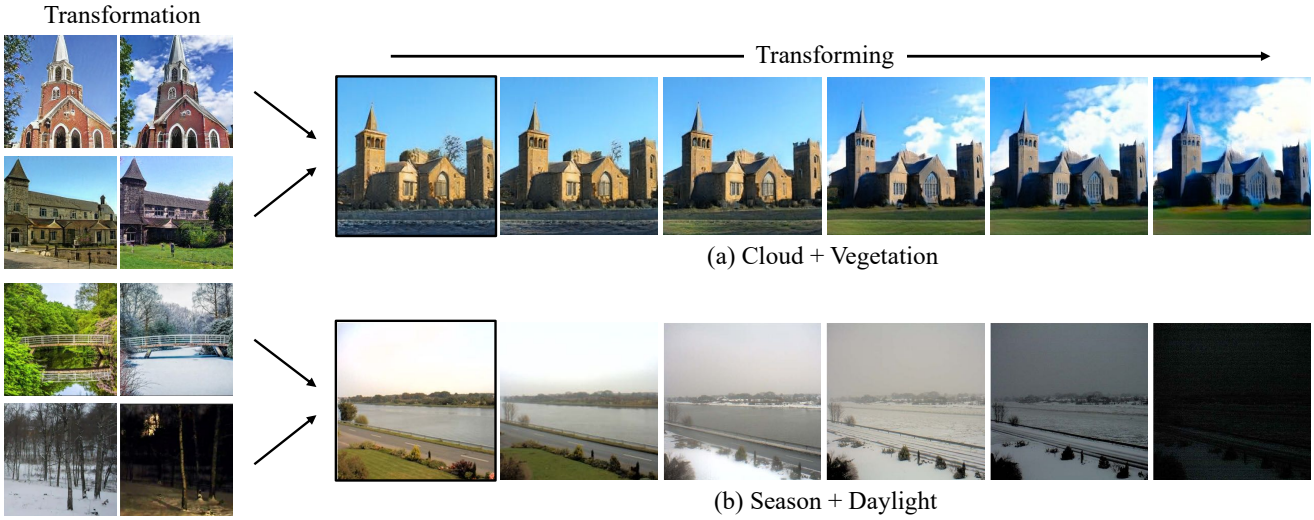


Figure 6. **Composition of two independent transformations**. The first two rows on the left are two independent transformations (upper: cloud, lower: vegetation) that modify the image contents, whilst the last two rows on the left are transformations (upper: season, lower: daylight) that change the image styles. On the right show the transforming sequences, where the **black** boxes highlight the base images.

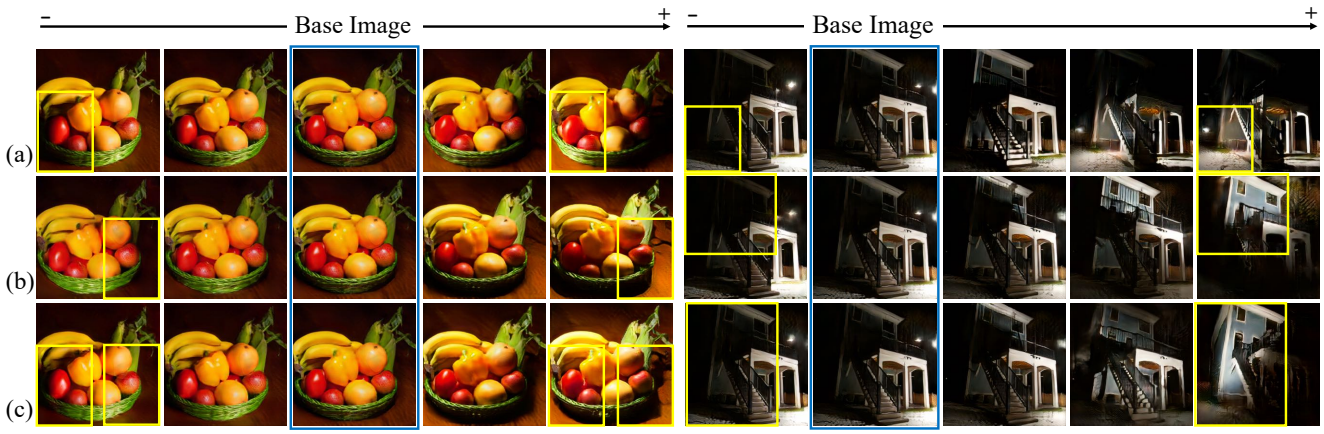


Figure 7. **Layer-wise analysis** of the transformation space. From top to bottom: (a) Transformations by manipulating $\mathbf{t}_1$ (*i.e.*, the transformation code fed into the first layer) along a certain direction upon the base image; (b) Transformations by manipulating $\mathbf{t}_2$ along the same direction; (c) Transformations by jointly manipulating $\mathbf{t}_1$ and $\mathbf{t}_2$. **Blue** boxes highlight the base images. **Yellow** boxes are used to track the changes of lighting conditions. We can observe that the transformation codes from different layers tend to control the lighting conditions of different areas.

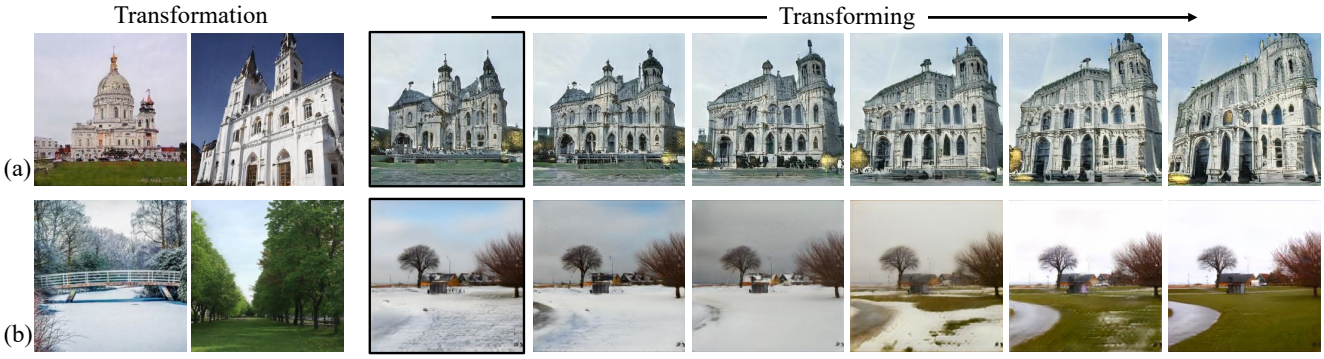


Figure 8. **Transformations discovered from unrelated image pairs**. On each row, the left two images are the target image pair that are not related to each other. The image sequence on the right shows the results by extracting the transformation from the reference pair and further utilizing it to transform images. The **black** boxes highlight the base images. We can see that TrGAN can adequately capture the variations of church shape and season from unrelated pairs, demonstrating the robustness of the learned transformation space.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Int. Conf. Comput. Vis.*, 2019. 2, 5, 6

[2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013. 1

[3] Ivaylo Boyadzhiev, Sylvain Paris, and Kavita Bala. User-assisted image compositing for photographic lighting. *ACM Trans. Graph.*, 2013. 4

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *Int. Conf. Learn. Represent.*, 2019. 2, 3

[5] Ting Chen, Mario Lucic, Neil Houlsby, and Sylvain Gelly. On self modulation for generative adversarial networks. *arXiv preprint arXiv:1810.01365*, 2018. 2

[6] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Adv. Neural Inform. Process. Syst.*, 2016. 4

[7] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. 2, 4, 6

[8] Lore Goetschalckx, Alex Andonian, Aude Oliva, and Phillip Isola. Ganalyze: Toward visual definitions of cognitive image properties. In *Int. Conf. Comput. Vis.*, 2019. 2

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Adv. Neural Inform. Process. Syst.*, 2014. 2, 3

[10] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001. 2

[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 *Adv. Neural Inform. Process. Syst.*, 2017. 5

[12] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Int. Conf. Comput. Vis.*, 2017. 2, 6, 7, 10

[13] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Eur. Conf. Comput. Vis.*, 2018. 2, 4

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[15] Nathan Jacobs, Walker Burgin, Nick Fridrich, Austin Abrams, Kylia Miskell, Bobby H Braswell, Andrew D Richardson, and Robert Pless. The global network of outdoor webcams: properties and applications. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2009. 4

[16] Ali Jahanian, Lucy Chai, and Phillip Isola. On the"steerability" of generative adversarial networks. *Int. Conf. Learn. Represent.*, 2020. 2

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Eur. Conf. Comput. Vis.*, 2016. 2

[18] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *Int. Conf. Learn. Represent.*, 2018. 2, 4

[19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2, 3, 10

[20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2, 3

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Int. Conf. Learn. Represent.*, 2015. 4

[22] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Trans. Graph.*, 2014. 2

[23] Jean-François Lalonde, Alexei A Efros, and Srinivasa G Narasimhan. Webcam clip art: Appearance and illuminant transfer from time-lapse sequences. *ACM Trans. Graph.*, 2009. 4

[24] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Eur. Conf. Comput. Vis.*, 2018. 2

[25] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Adv. Neural Inform. Process. Syst.*, 2017. 2

[26] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *Eur. Conf. Comput. Vis.*, 2018. 2, 4, 6, 7, 10

[27] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Adv. Neural Inform. Process. Syst.*, 2017. 2

[28] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. In *Int. Conf. Comput. Vis.*, 2019. 2, 4

[29] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[30] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

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

[32] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2012. 2

[33] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *Int. Conf. Learn. Represent.*, 2016. 5, 6

[34] Erik Reinhard, Michael Adhikhmin, Bruce Gooch, and Peter Shirley. Color transfer between images. *IEEE Computer graphics and applications*, 2001. 2, 6, 7

[35] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Int. Conf. Comput. Vis.*, 2019. 3

[36] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[37] Yichang Shih, Sylvain Paris, Frédo Durand, and William T Freeman. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Trans. Graph.*, 2013. 2

[38] Ceyuan Yang, Yujun Shen, and Bolei Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis. *arXiv preprint arXiv:1911.09267*, 2019. 2

[39] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 4

[40] Fangneng Zhan, Hongyuan Zhu, and Shijian Lu. Spatial fusion gan for image synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[41] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019. 2

[42] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Adv. Neural Inform. Process. Syst.*, 2014. 2

[43] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *Eur. Conf. Comput. Vis.*, 2020. 2, 5

[44] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *Eur. Conf. Comput. Vis.*, 2016. 2

[45] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Int. Conf. Comput. Vis.*, 2017. 2

[46] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Adv. Neural Inform. Process. Syst.*, 2017. 2

Appendix

A. Overview

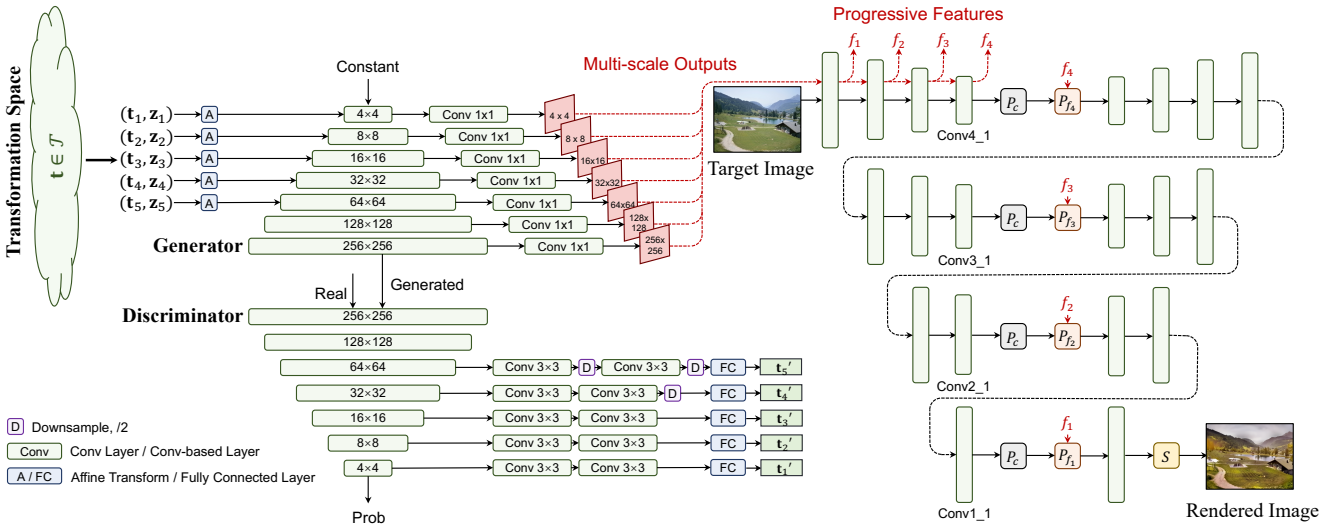
In this supplementary material, we first introduce the details of our proposed TrGAN framework in Appendix B. We then provide more results on the diverse transformations learned by TrGAN in Appendix C.

B. TrGAN Framework

Fig. 9 illustrates the detailed architecture of TrGAN as well as the rerendering module. Recall that TrGAN introduces a novel transformation space $\mathcal{T}$, which is shared by both the discriminator (transformation learner) and the generator (transformation deployer). In the generator which starts with a constant feature map [19], the transformation code $\mathbf{t} \in \mathcal{T}$, concatenated with a sampled latent code $\mathbf{z} \in \mathcal{Z}$, is fed into each convolutional layer using Adaptive Instance Normalization (AdaIN) [12]. Here, we use an affine function $A(\cdot)$ to align the dimension of the combined code $(\mathbf{t}, \mathbf{z})$ with the number of feature channels in a particular layer. In the discriminator which aims at differentiating real images from fake ones, two more convolutional layers as well as one more fully-connected layer follow each layer to project the synthesized image back to the transformation space and get $\mathbf{t}'$. These additional convolutional layers and fully-connected layers refer to $D_T(\cdot)$ in the main paper.

We further introduce a rerendering module to enable real image editing. As shown in Fig. 9, our rerendering module follows an encoder-decoder structure, which is organized in a multi-level manner. To better connect the rerendering module with the learned transformations, the rerendering module takes the intermediate spatial feature maps (*i.e.*, "multi-scale outputs" in Fig. 9) from the transformation deployer as the inputs. These feature maps are all upsampled to the largest scale, and the first-level encoder will extract the "progressive features" $\{f_i\}$ from them, as shown in Fig. 9. The encoder and decoder from each level are connected with a whitening function $P_c(\cdot)$ and a coloring function $P_{f_i}(\cdot)$ [26]. Here, the coloring function $P_{f_i}(\cdot)$ is based on an averaged feature map which averages all channels from f_i. Finally, the output from the last-level decoder will be smoothed with a smoothing operation $S(\cdot)$. The rerendering module is trained from scratch based on the pre-trained generator.

Discussion. Since the rerendering module mainly targets at image stylization, it can typically handle style-based transformation (*e.g.*, changing the season or daylight), but fails to deal with object-related image editing (*e.g.*, adding clouds or vegetation). There are two possible solution to tackle this obstacle. First is to invert a given image back to the original latent space $\mathcal{Z}$ and get $\mathbf{z}'$. Since we already have the transformation learner to extract the transformation code



(a) Transformation Learner & Deployer **(b) Rerendering Module**

Figure 9. **Detailed structure of TrGAN.** In addition to the latent space $\mathcal{Z}$ that is commonly used in conventional GANs, TrGAN introduces a transformation space $\mathcal{T}$ that is *shared* by the generator and the discriminator. Besides competing with each other on image domains (*i.e.*, real or fake), the discriminator, as the transformation learner, is also trained to extract the pairwise transformation from input images while the generator, as the transformation deployer, learns to employ such a transformation for controllable image synthesis. Both the latent code $\mathbf{z}$ and the transformation code $\mathbf{t}$ are organized in a multi-scale manner across layers. A rerendering module is built upon the transformation deployer to further transform real images.

$\mathbf{t}'$ from it. We can use $(\mathbf{t}', \mathbf{z}')$ to well reconstruct the target image and further modulate $\mathbf{t}'$ for image transforming. Second is to re-design the rerendering module and make it applicable to image-to-image translation beyond color tones. This may require manually labelled paired data for training. Nevertheless, TrGAN still provides a very promising way to unsupervisedly learn the transformation from image pairs. It also points out a new direction in transforming images, where users can *customize their own transformation pair with no need to re-train the model*.

C. More Image Transforming Results

In this section, we show more image transforming results using TrGAN. Recall that TrGAN is capable of extracting the transformation from an image pair and further apply such transformation for controllable image synthesis. It is worth noting that TrGAN can not only transfer image styles (*e.g.*, changing the season of a landscape), but also modulate the objects inside the image (*e.g.*, adding clouds in the sky). Fig. 10, Fig. 11, and Fig. 12 show the light position, the cloud and the vegetation variations extracted by TrGAN respectively. The transforming samples suggest that TrGAN can accurately capture the semantic variations between the input pairs and further utilize them to control the synthesis process. We are even able to compose the transformations extracted from two independent pairs, as shown in Fig. 13, validating the compositing property of the learned transformation space $\mathcal{T}$. Fig. 14 shows the season and daylight transformations on real images, with the help of the rerendering module. we can see that TrGAN is also able to transfer image styles with *continuous semantic change* instead of merely interpolating the color tones. For example, we manage to synthesize images in autumn when transforming from summer to winter, and synthesize images in dusk when transforming from morning to night. Finally, in Fig. 15, we evaluate TrGAN by taking highly unrelated images as the input pair. In particular, they are with different church types, as well as different shapes and visual angles. In this case, the proposed TrGAN can still convincingly discover the meaningful variations between them, verifying the robustness of the learned transformation space and showing that our method enables broader application scenarios.

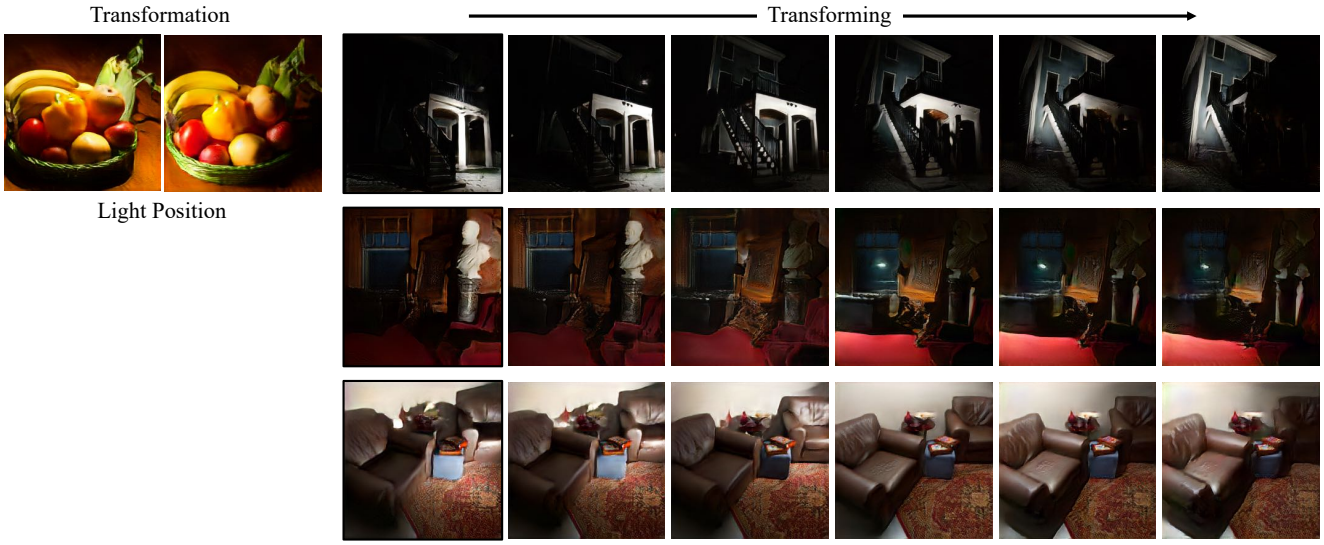


Figure 10. **Altering light position.** The input pair is shown on the left (light position from right to left) and transforming samples are shown on the right.

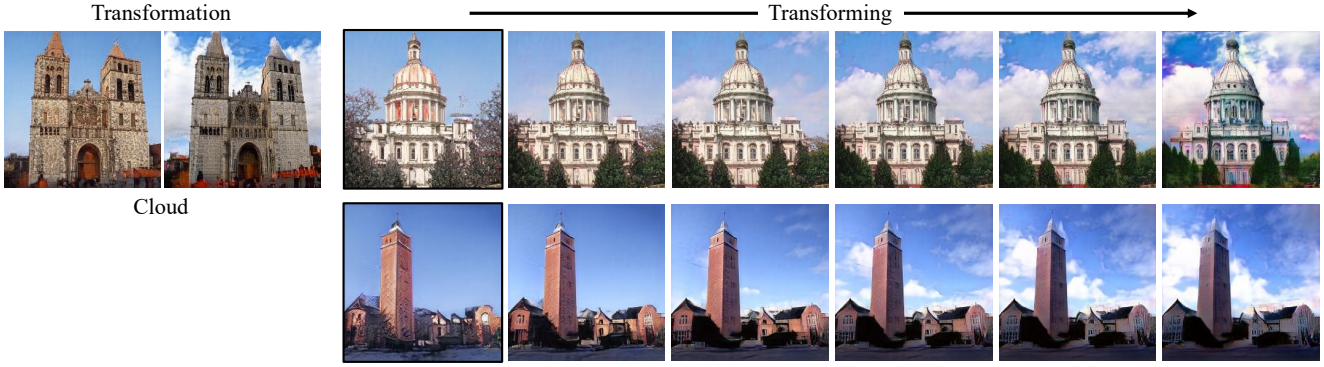


Figure 11. **Adding clouds.** The input image pair is shown on the left and transforming samples are shown on the right.

Figure 12. **Adding vegetation.** The input image pair is shown on the left and transforming samples are shown on the right.

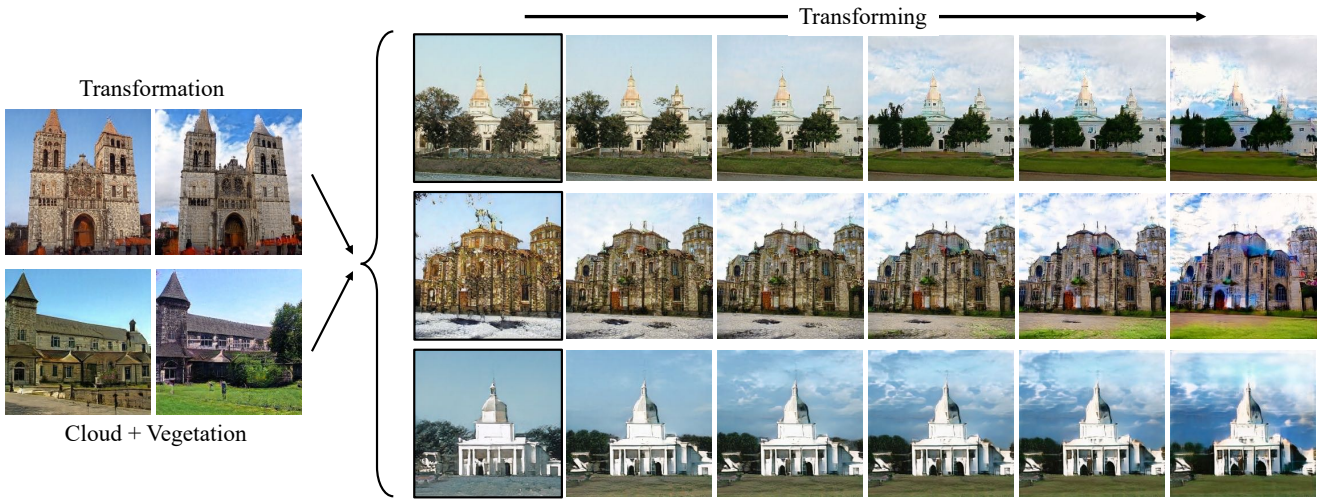


Figure 13. **Simultaneously adding clouds and vegetation.** Two independent image pairs (adding clouds, adding vegetation) are embedded to the transformation space and composed together to transform other images.

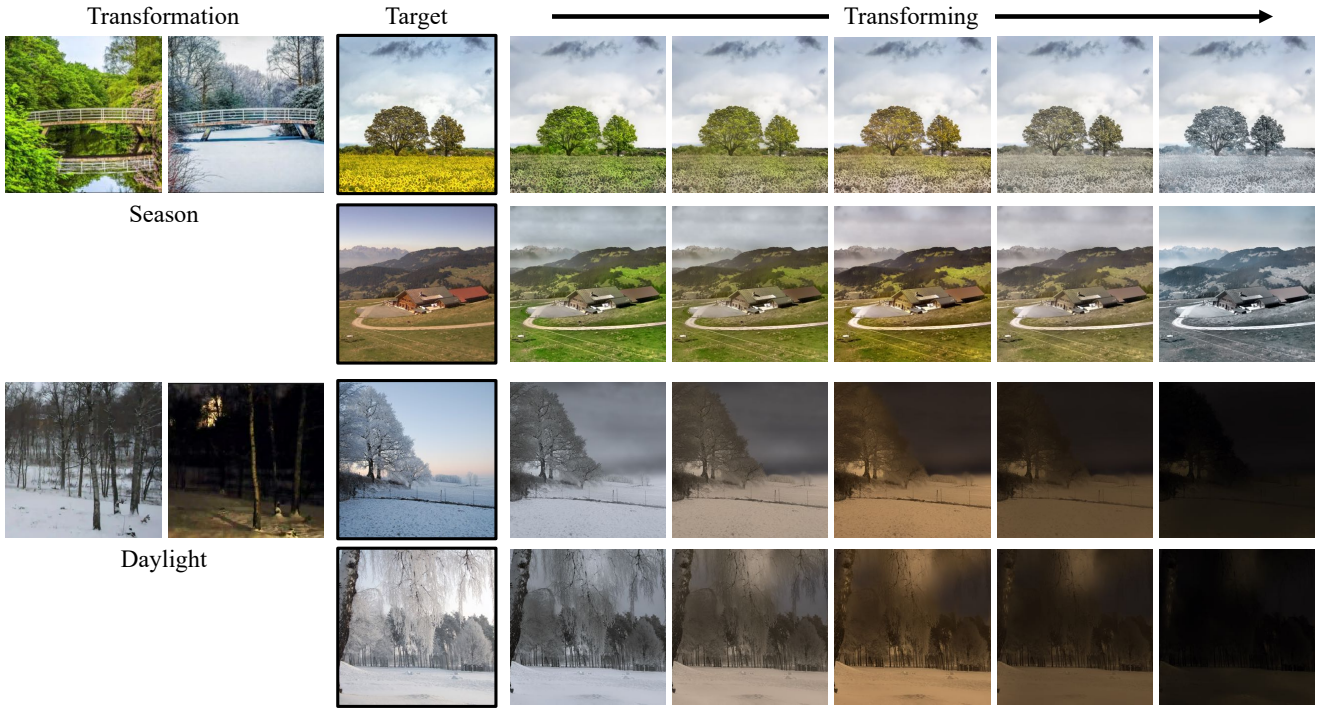


Figure 14. **Transforming season and daylight of real images.** For each image group (top two rows and bottom two rows), the input pair is shown on the left, **black** boxes highlight the target real images, and transforming results are shown on the right. Our transformation goes beyond merely interpolating color tones. Instead, we can transform season and daylight semantically, *i.e.*, yielding autumn between summer and winter, and producing dusk between morning and night.

Figure 15. **Transformations discovered from unrelated image pairs.** The input image pairs on the left are of different church types, shapes and visual angles. The transforming samples on the right successfully capture such variations, demonstrating the robustness of the proposed transformation space.