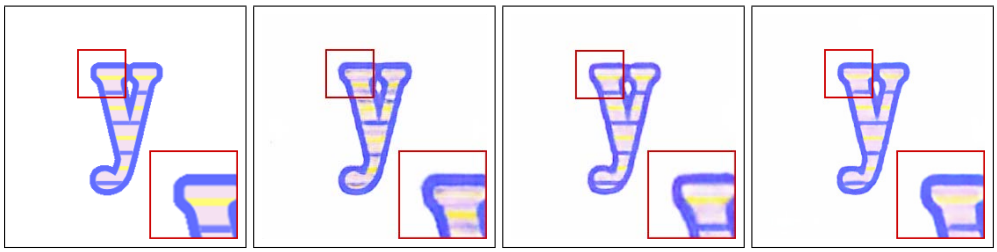
Typography with Decor: Intelligent Text Style Transfer (Supplementary Material)

Wenjing Wang, Jiaying Liu, Shuai Yang, and Zongming Guo
Institute of Computer Science and Technology, Peking University, Beijing, China

Contents

1. Progressively Training Strategy

We exploit a novel progressively training strategy [5] to stabilize and speed up the training process. It's worth noting that different from the original work of [5], we find that a static discriminator architecture leads to clearer edges and texture, as illustrated in Fig. 1. This training strategy leads to fewer artifacts and speeds up the training for more than 10 hours.



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

Figure 1: Effect of different training strategy. (a) Ground Truth. (b) Result with both generator and discriminator static. (c) Result with both generator and discriminator progressively growing. (d) Result with generator growing and discriminator static.

Specifically, the resolution of the images increases from 64×64, 128×128, to 256×256. When resolution increases, the generator changes correspondingly. At 128×128, a 16-layer U-net [6] M, a convolutional block C_1 and a de-convolutional block D_1 comprise the main body of the architecture. Then, a pair of temporary blocks C_{t2} and D_{t2} are used at the front and back side of the main body for converting between RGB images and feature maps. When resolution changes from 128×128 to 256×256, C_{t2} and D_{t2} are deleted, while new layers C_2, D_2, C_3, and D_3 are added to the architecture, as shown in Fig. 2. To smooth the change, new layers fade in smoothly using the same strategy in [5]. For changing from 64×64 to 128×128, the switching strategy is similar.

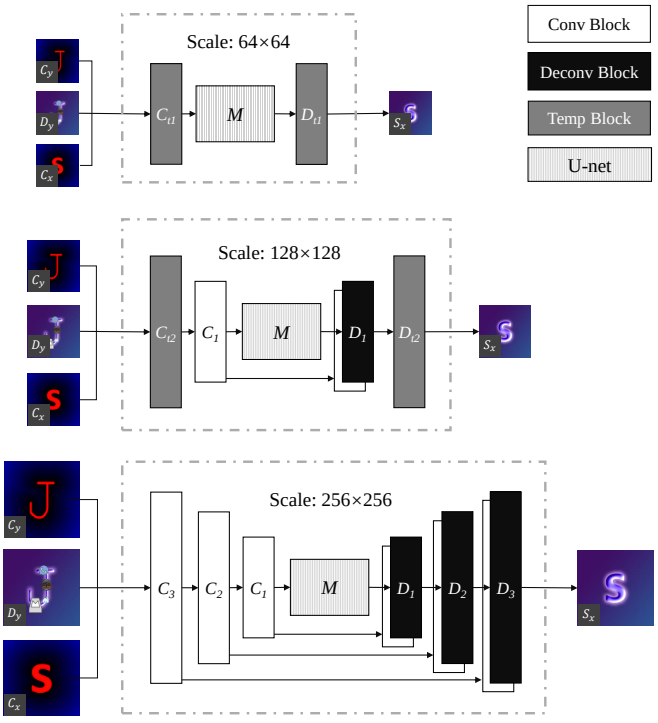


Figure 2: Progressively growing training strategy for image-to-image translation based on Pix2Pix [4]. We incrementally add layers to the generator, and increase the resolution of the images from 64×64, 128×128, to 256×256. Temporary blocks are added for converting between RGB and feature maps.

For the discriminator, instead of gradually deepening the network, we keep the architecture static. We find that growing the discriminator may lead to color bias and artifacts, as illustrated in Fig. 1. This can be explained from two aspects.

On the one hand, if discriminator grows, the ratio of the receptive field to the whole image will approximately stay constant, about $1:4$. However, if the discriminator is set to the deepest at the beginning, the receptive field can cover the whole image when images are at low resolution. Therefore, the discriminator can guide the generator to better perceive global structures and avoid color bias.

On the other hand, appropriately deepening the architecture can strengthen the network. By keeping the architecture static, the discriminator is stronger yet not too strong to break the adversarial training balance. This not only results in better details, but also speeds up the training.

Note that in [4], Isola *et al*. found that large receptive fields lead to artifacts and hurt the visual quality. We solve this problem by switching to the next resolution before the discriminator is too strong.

2. Implementation Details

In this section, we give detailed information about network architecture and parameter configuration. Table. 1 shows the detailed architecture of the transfer network. Table. 2 lists the parameter configuration.

Block	**Layer**	**Parameters**	**Input shape**	**Output shape**
C_3	Conv	4×4, stride=2, padding=1	$3 \times 256 \times 256$	$64 \times 128 \times 128$
	LeakyReLU	0.2	-	-
C_2	Conv	4×4, stride=2, padding=1	$64 \times 128 \times 128$	$128 \times 64 \times 64$
	BatchNorm	-	-	-
	LeakyReLU	0.2	-	-
C_1	Conv	4×4, stride=2, padding=1	$128 \times 64 \times 64$	$256 \times 32 \times 32$
	BatchNorm	-	-	-
	LeakyReLU	0.2	-	-
M	16-layer U-net	-	$256 \times 32 \times 32$	$256 \times 32 \times 32$
D_1	De-Conv	4×4, stride=2, padding=1	$512 \times 32 \times 32$	$128 \times 64 \times 64$
	BatchNorm	-	-	-
	ReLU	-	-	-
D_2	De-Conv	4×4, stride=2, padding=1	$256 \times 64 \times 64$	$64 \times 128 \times 128$
	BatchNorm	-	-	-
	ReLU	-	-	-
D_3	De-Conv	4×4, stride=2, padding=1	$128 \times 128 \times 128$	$3 \times 256 \times 256$
	Tanh	-	-	-
C_{t2}	Conv	4×4, stride=2, padding=1	$3 \times 128 \times 128$	$128 \times 64 \times 64$
	LeakyReLU	0.2	-	-
C_{t1}	Conv	4×4, stride=2, padding=1	$3 \times 64 \times 64$	$256 \times 32 \times 32$
	LeakyReLU	0.2	-	-
D_{t2}	De-Conv	4×4, stride=2, padding=1	$128 \times 64 \times 64$	$3 \times 128 \times 128$
	Tanh	-	-	-
D_{t1}	De-Conv	4×4, stride=2, padding=1	$256 \times 32 \times 32$	$3 \times 64 \times 64$
	Tanh	-	-	-

Table 1: Architecture of the transfer network.

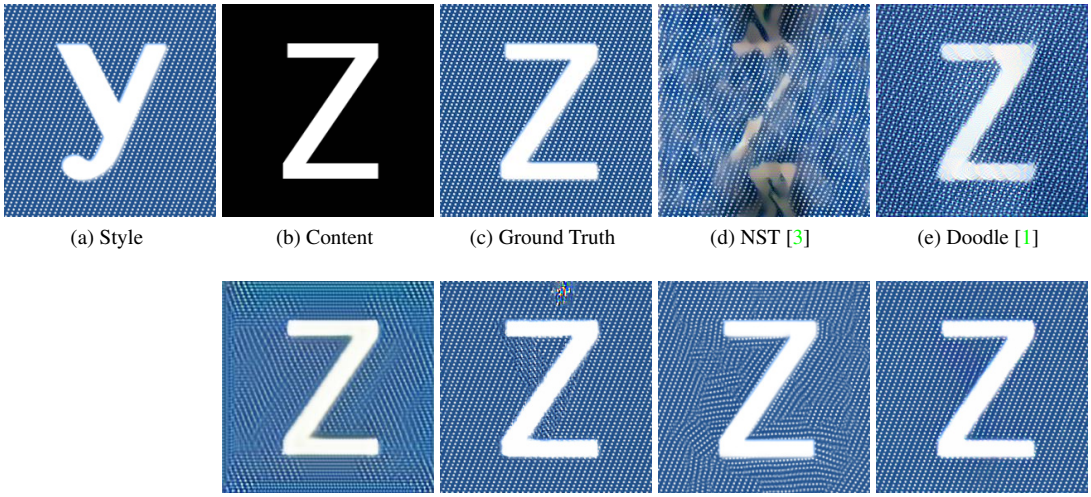
Section	Parameter	Value
3.1	λ_{L1}	1
	λ_{Per}	1
	λ_{seg}	1
	λ_{adv}	0.01
3.2	λ_{adv}	0.01
	λ_{L1}	1
	λ_{GP}	1

Section	Parameter	Value
3.3	$\mathcal{K}_{ws}$	0.06
	$\mathcal{K}_{s}$	0.9
	$\mathcal{K}_{dis}$	5
	λ_{Hor}	1
	λ_{Ver}	1
	λ_{Dis}	1
	λ_{Exi}	1

Table 2: Parameter Configuration.

3. Comparison Results for Text Effects Transfer

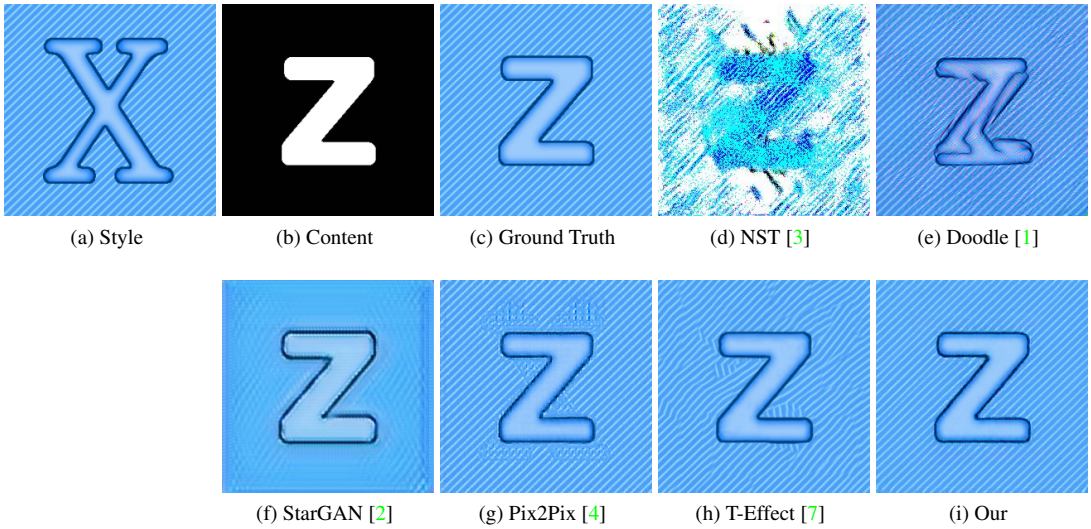
In this section, we present more subjective comparison results for text effects transfer.



(a) Style (b) Content (c) Ground Truth (d) NST [3] (e) Doodle [1]

(f) StarGAN [2] (g) Pix2Pix [4] (h) T-Effect [7] (i) Our

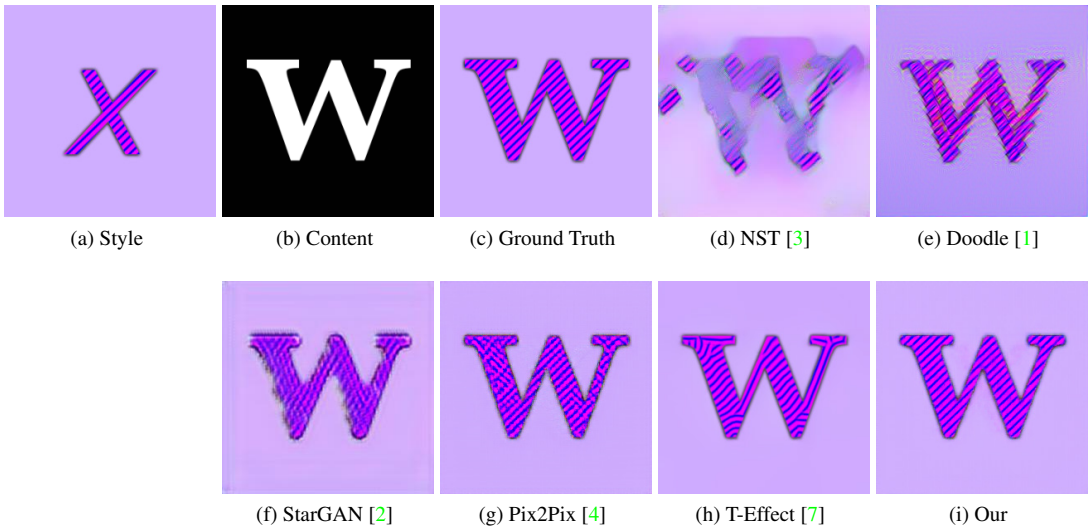
Figure 3: Results for text effects transfer. (a) Input style image. (b) Target content image. (c) Ground truth. (d)-(h) Compared methods. (i) Our result.



(a) Style (b) Content (c) Ground Truth (d) NST [3] (e) Doodle [1]

(f) StarGAN [2] (g) Pix2Pix [4] (h) T-Effect [7] (i) Our

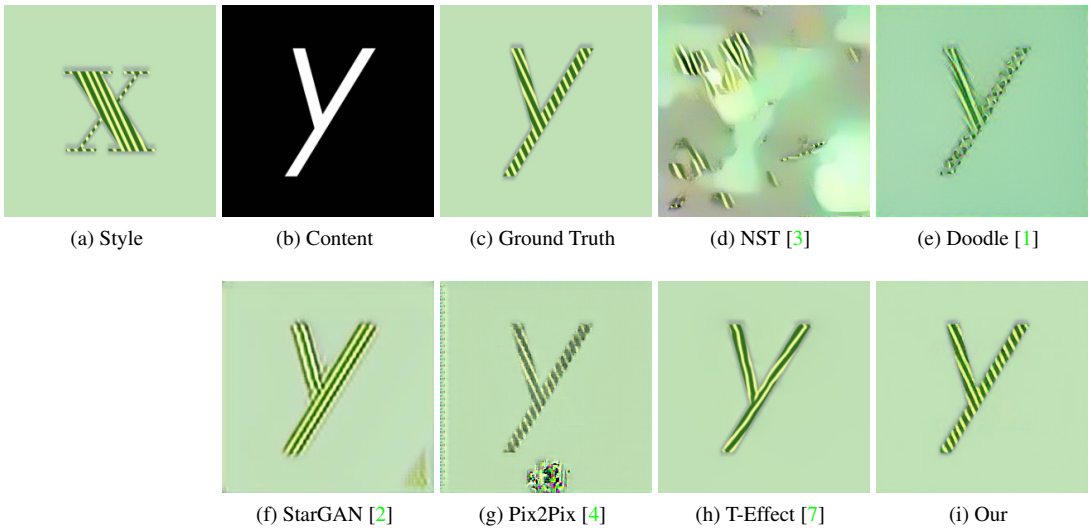
Figure 4: Results for text effects transfer. (a) Input style image. (b) Target content image. (c) Ground truth. (d)-(h) Compared methods. (i) Our result.



(a) Style (b) Content (c) Ground Truth (d) NST [3] (e) Doodle [1]

(f) StarGAN [2] (g) Pix2Pix [4] (h) T-Effect [7] (i) Our

Figure 5: Results for text effects transfer. (a) Input style image. (b) Target content image. (c) Ground truth. (d)-(h) Compared methods. (i) Our result.



(a) Style (b) Content (c) Ground Truth (d) NST [3] (e) Doodle [1]

(f) StarGAN [2] (g) Pix2Pix [4] (h) T-Effect [7] (i) Our

Figure 6: Results for text effects transfer. (a) Input style image. (b) Target content image. (c) Ground truth. (d)-(h) Compared methods. (i) Our result.

4. Comparison Results for Text with Decor

In this section, we present more subjective comparison results for text with decor.

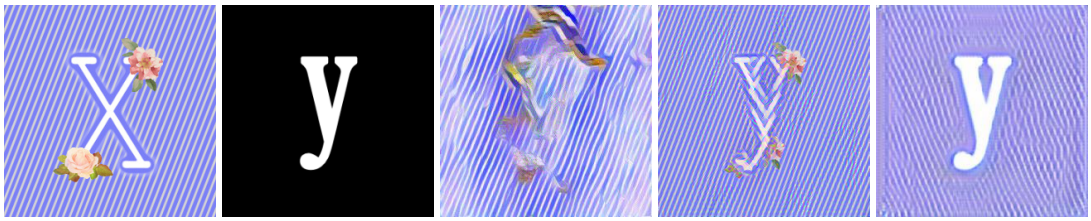
4.1. Results on the Proposed Dataset

We synthesize several text based on text effects from the proposed dataset and use elements from the internet as decor.

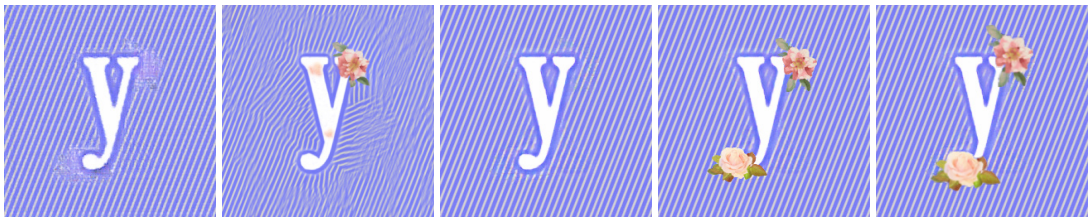
(a) Style (b) Content (c) NST [3] (d) Doodle [1] (e) StarGAN [2]

(f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1 (i) Our 2 (j) Our 3

Figure 7: Results on the proposed dataset for text with decor. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result without combining the decor. (i) Our result with decor simply placed on the original position. (j) Our result with full model.

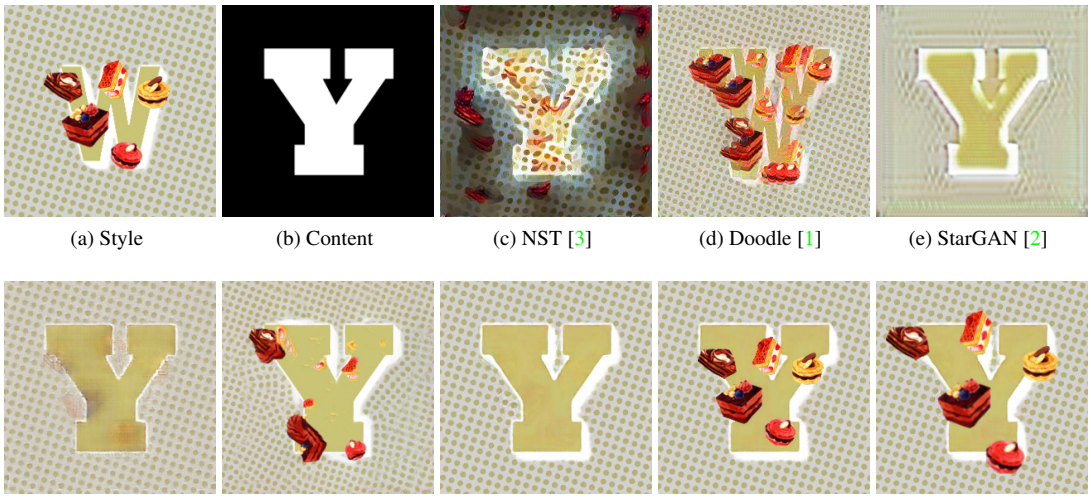


(a) Style (b) Content (c) NST [3] (d) Doodle [1] (e) StarGAN [2]



(f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1 (i) Our 2 (j) Our 3

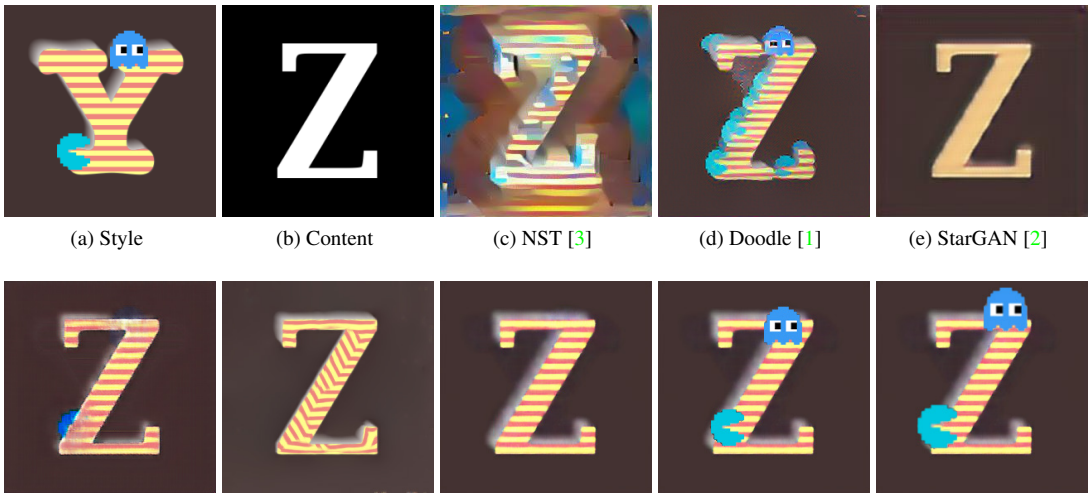
Figure 8: Results on the proposed dataset for text with decor. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result without combining the decor. (i) Our result with decor simply placed on the original position. (j) Our result with full model.



(a) Style (b) Content (c) NST [3] (d) Doodle [1] (e) StarGAN [2]

(f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1 (i) Our 2 (j) Our 3

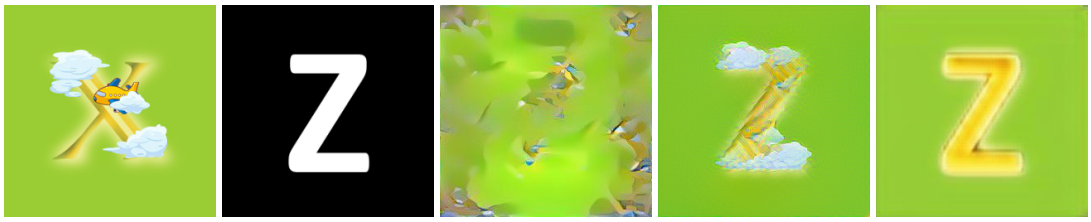
Figure 9: Results on the proposed dataset for text with decor. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result without combining the decor. (i) Our result with decor simply placed on the original position. (j) Our result with full model.



(a) Style (b) Content (c) NST [3] (d) Doodle [1] (e) StarGAN [2]

(f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1 (i) Our 2 (j) Our 3

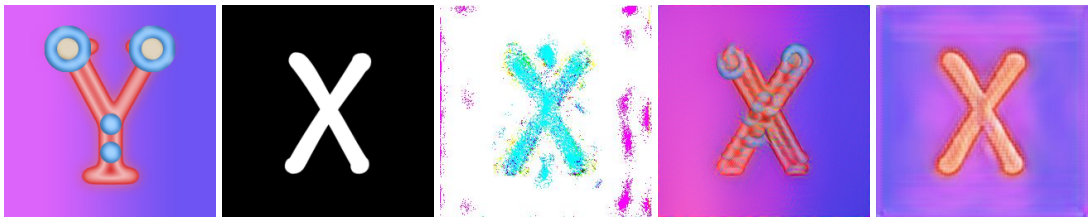
Figure 10: Results on the proposed dataset for text with decor. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result without combining the decor. (i) Our result with decor simply placed on the original position. (j) Our result with full model.



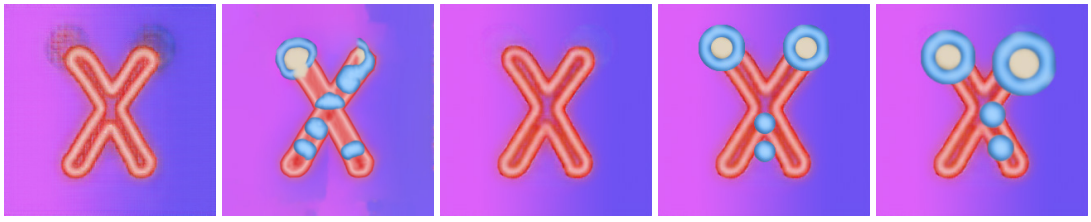
(a) Style (b) Content (c) NST [3] (d) Doodle [1] (e) StarGAN [2]

(f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1 (i) Our 2 (j) Our 3

Figure 11: Results on the proposed dataset for text with decor. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result without combining the decor. (i) Our result with decor simply placed on the original position. (j) Our result with full model.



(a) Style (b) Content (c) NST [3] (d) Doodle [1] (e) StarGAN [2]



(f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1 (i) Our 2 (j) Our 3

Figure 12: Results on the proposed dataset for text with decor. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result without combining the decor. (i) Our result with decor simply placed on the original position. (j) Our result with full model.

4.2. Results on Custom Text Effects

We present more results for custom text with decor.

(a) Style (b) Content (c) Doodle [1] (d) T-Effect [7] (e) Our

Figure 13: Results on custom artistic text. (a) Input style image. (b) Target content image. (c)-(d) Compared methods. (e) Our result.

(a) Style (b) Content (c) Doodle [1] (d) T-Effect [7] (e) Our

Figure 14: Results on custom artistic text. (a) Input style image. (b) Target content image. (c)-(d) Compared methods. (e) Our result.

(a) Style (b) Content (c) Doodle [1] (d) T-Effect [7] (e) Our

Figure 15: Results on custom artistic text. (a) Input style image. (b) Target content image. (c)-(d) Compared methods. (e) Our result.

5. Failure Cases

Fig. 16 shows a failure case where all methods fail to reconstruct the complex 3D structure.

(a) Style (b) Content (c) NST [3] (d) Doodle [1]

(e) StarGAN [2] (f) Pix2Pix [4] (g) T-Effect [7] (h) Our 1

Figure 16: A failure case where all methods fail to reconstruct the complex 3D structure. (a) Input style image. (b) Target content image. (c)-(g) Compared methods. (h) Our result.

6. Quantitative Evaluation

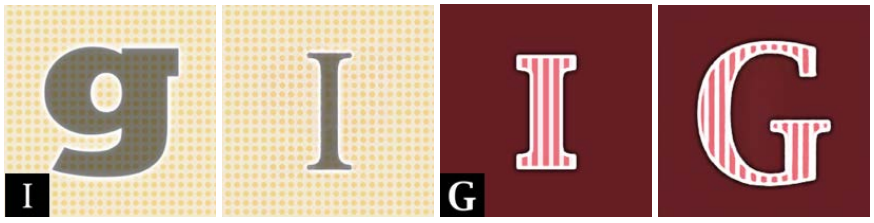
We conducted a user study where 51 observers were given images pairs and asked to select which one is of the best style similarity. The mean preference ratio over 9 test cases (Figs. 13-14 and Sup-Figs. 13-15) is shown in Table 3, where our method performs the best.

Table 3: Quantitative evaluation of style similarity.

	Doodle	**T-Effect**	**Proposed**
Preference Ratio	0.1743	0.4564	0.8693

7. Results with large glyph difference.

Our model is quite robust to fonts and languages, as shown in Fig. 17, verifying that our network has a wide applicability.



(a) Input (b) Output (c) Input (d) Output

Figure 17: Results for large glyph difference.

8. Example Style Images of the Proposed Dataset.

In this section, we illustrate all kinds of text effects in our dataset.

Figure 18: Example style images from the proposed dataset.

9. Text Effects Transfer Results for All the Text Effects in the Proposed Dataset.

In this section, we illustrate our text effects transfer results for all the text effects in the proposed dataset. For each style, we use 'a' to 'v' for training and 'w' to 'z' for testing.

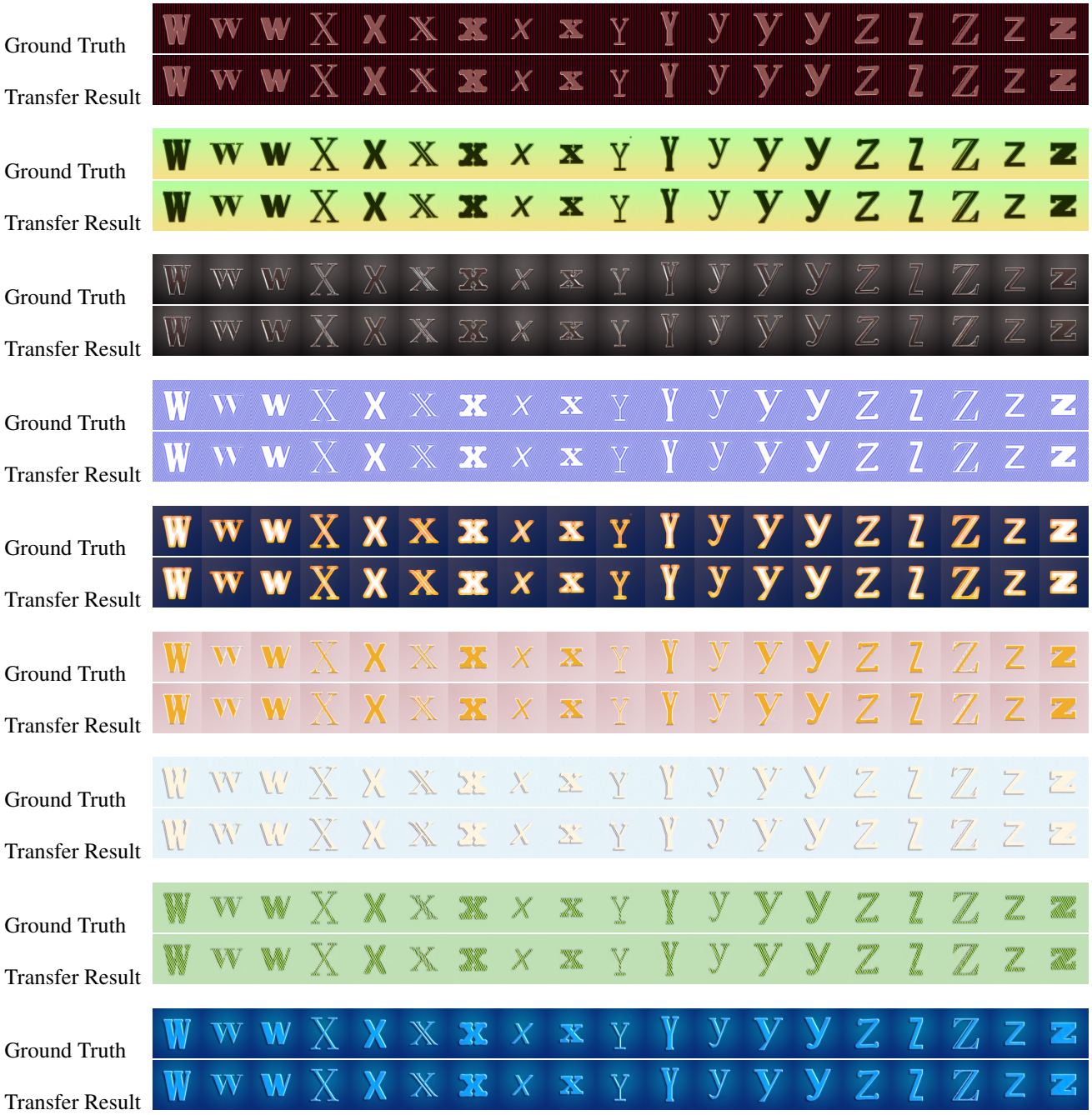


Figure 19: Our text effects transfer results. For each group, the first row shows the ground truth images in our dataset. The second row shows our transfer results by transferring the text effects from the example style image in Fig. 18.

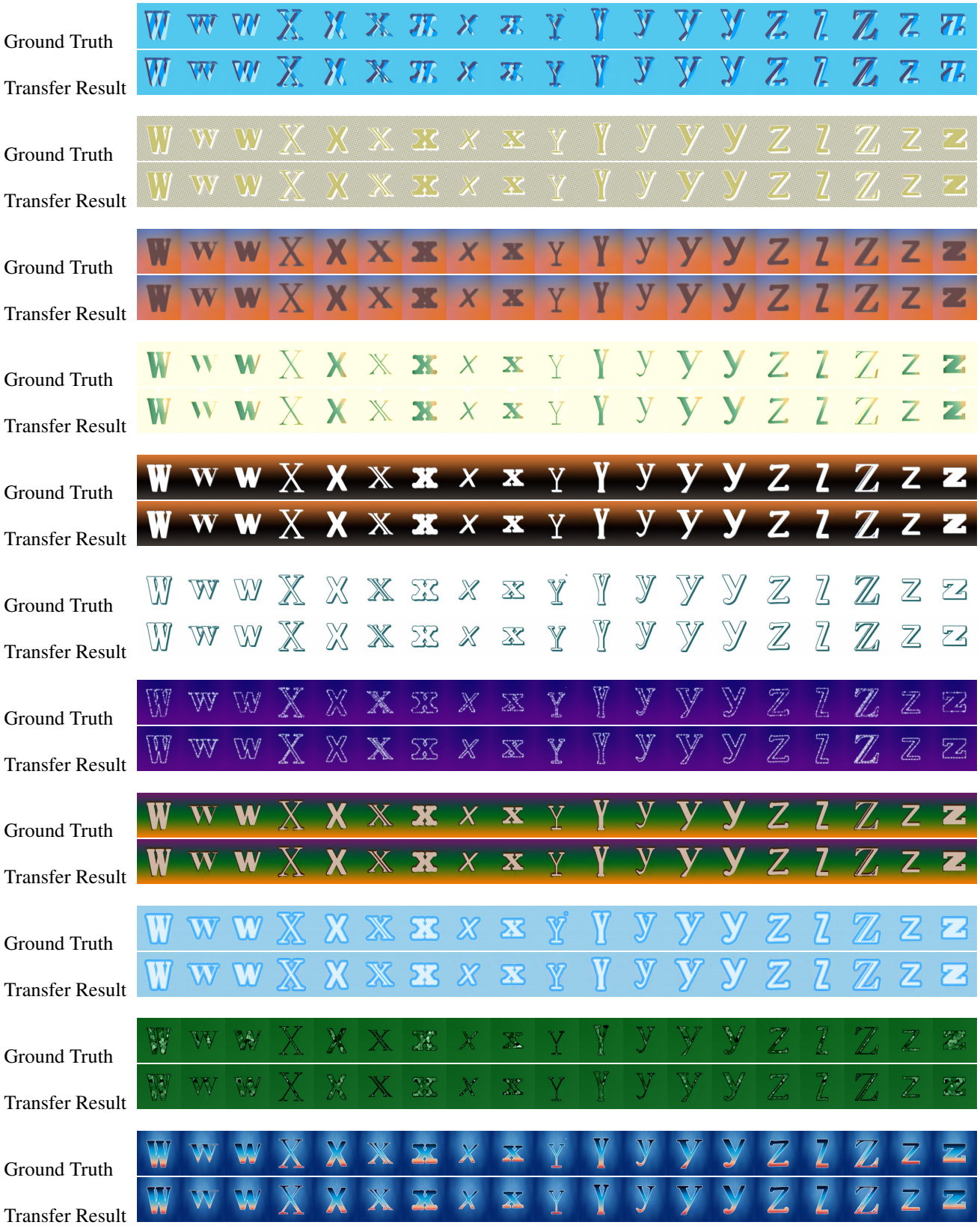


Figure 20: Our text effects transfer results. For each group, the first row shows the ground truth images in our dataset. The second row shows our transfer results by transferring the text effects from the example style image in Fig. 18.

Figure 21: Our text effects transfer results. For each group, the first row shows the ground truth images in our dataset. The second row shows our transfer results by transferring the text effects from the example style image in Fig. 18.

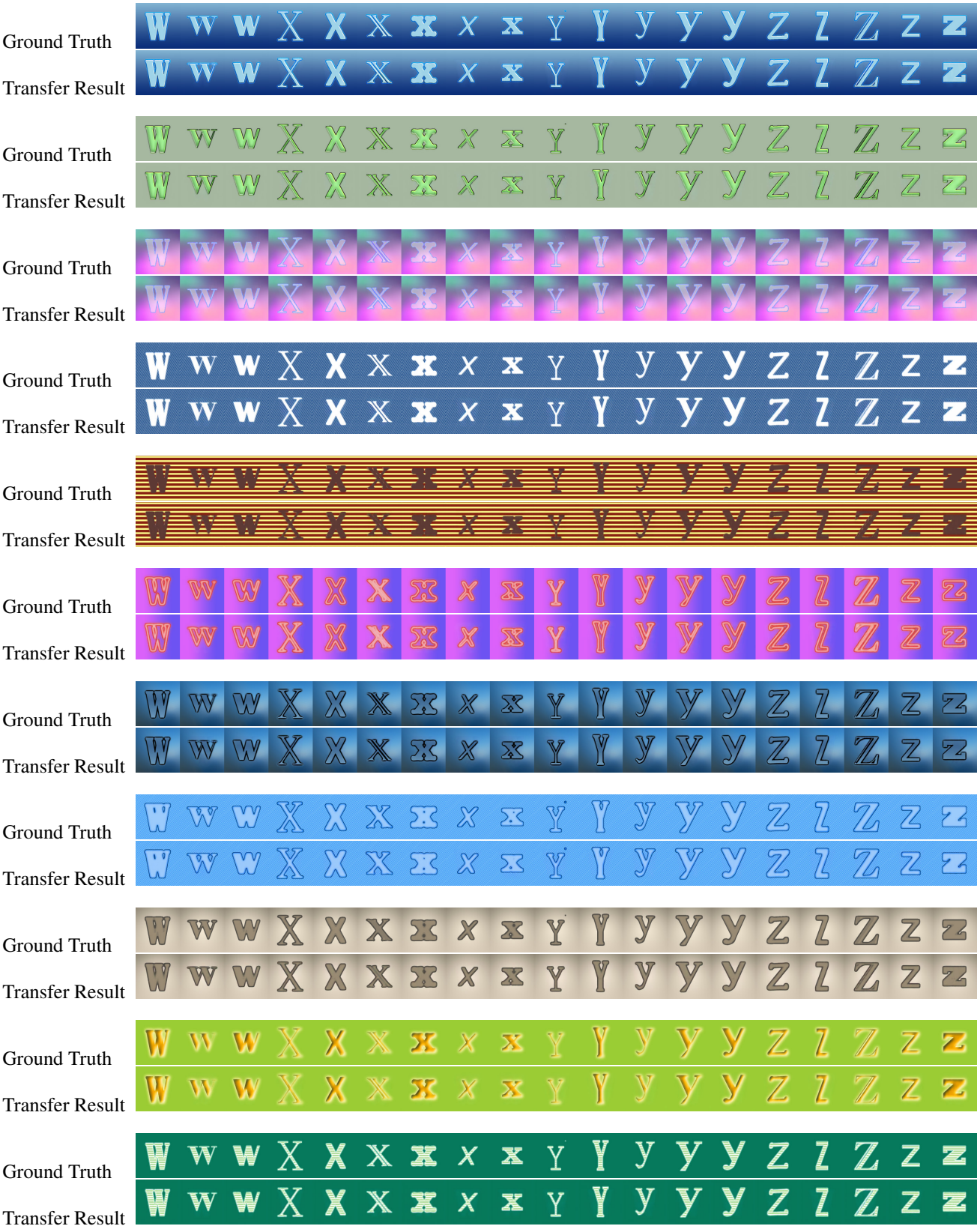


Figure 22: Our text effects transfer results. For each group, the first row shows the ground truth images in our dataset. The second row shows our transfer results by transferring the text effects from the example style image in Fig. 18.

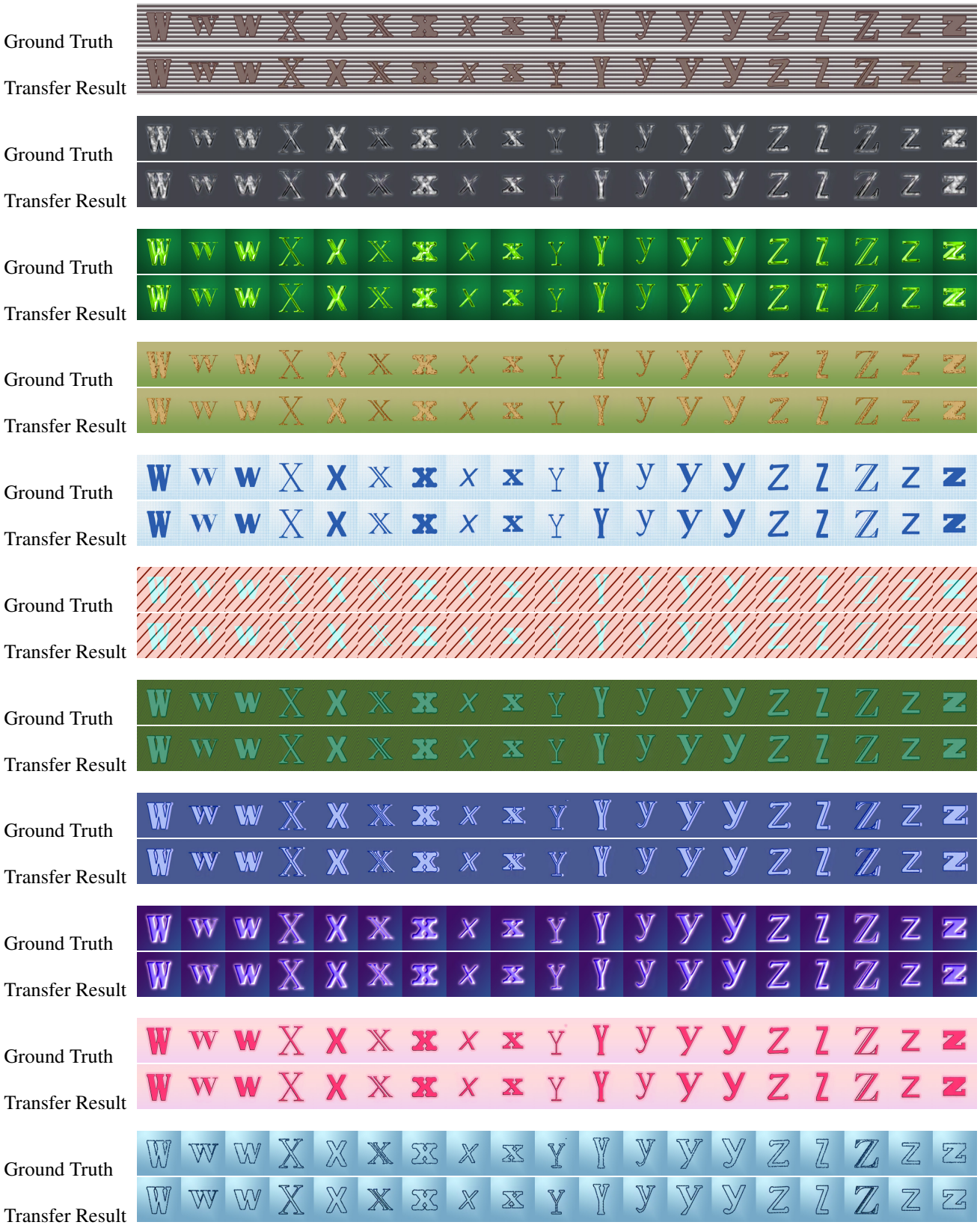


Figure 23: Our text effects transfer results. For each group, the first row shows the ground truth images in our dataset. The second row shows our transfer results by transferring the text effects from the example style image in Fig. 18.

Figure 24: Our text effects transfer results. For each group, the first row shows the ground truth images in our dataset. The second row shows our transfer results by transferring the text effects from the example style image in Fig. 18.

References

[1] A. J. Champandard. Semantic style transfer and turning two-bit doodles into fine artworks. *arXiv preprint arXiv:1603.01768*, 2016. 4, 5, 6, 7, 8, 9, 10

[2] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *arXiv preprint*, 1711, 2017. 4, 5, 6, 7, 8, 10

[3] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016. 4, 5, 6, 7, 8, 10

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017. 2, 4, 5, 6, 7, 8, 10

[5] 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. 1, 2

[6] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[7] S. Yang, J. Liu, Z. Lian, and Z. Guo. Awesome typography: Statistics-based text effects transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7464–7473, 2017. 4, 5, 6, 7, 8, 9, 10