Supplementary Material

Sanghyeon Na[1]
sktkdgus98@korea.ac.kr
Seungjoo Yoo[1]
seungjooyoo@korea.ac.kr
Jaegul Choo[2]
jchoo@kaist.ac.kr

[1] Korea University
Seoul, Korea

[2] Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Korea

A Additional Qualitative Results

Figs 1, 2, 3, 4, 5, 6, 7, and 8 are additional qualitative results of MISO.

B Network Architecture

First, we introduce basic blocks, convolutional block, transposed convolutional block, residual block and encoder residual block used in our architecture. Based on these blocks, we introduce our network architectures in Tables. 1 and 2.

Convolutional Block. A convolutional block consists of convolutional layer, normalization layer and non-linear layer denoted as ConvBlock(IN, OUT, K, S, P, norm, non-linear) with notations IN : input channel size, OUT : output channel size, K : kernel size, S : stride, P : padding, norm : normalization type, non-linear : non-linear function type.

Transposed Convolutional Block. A transposed convolutional block is same with the convolutional block but replace convolutional layer with transposed convolutional layer denoted as TransConvBlock(IN, OUT, K, S, P, norm, non-linear) which has same notations with convolutional block.

Residual Block. A residual block consists of two convolutional blocks that the first convolutional block has non-linear layer but the second block does not have. It is denoted as ResBlock(C, K, S, P) with notations C : channel size, K : kernel size, S : stride, P : padding. Concretely, ResBlock(C, K, S, P) = ConvBlock(C, C, K, S, P, IN, ReLU) - ConvBlock(C, C, K, S, P, IN, None) where IN denotes instance normalization. The output tensor of residual block is sum of input tensor and output of the second convolutional block.

Style Encoder. We use the architecture that is used in BicycleGAN [3].

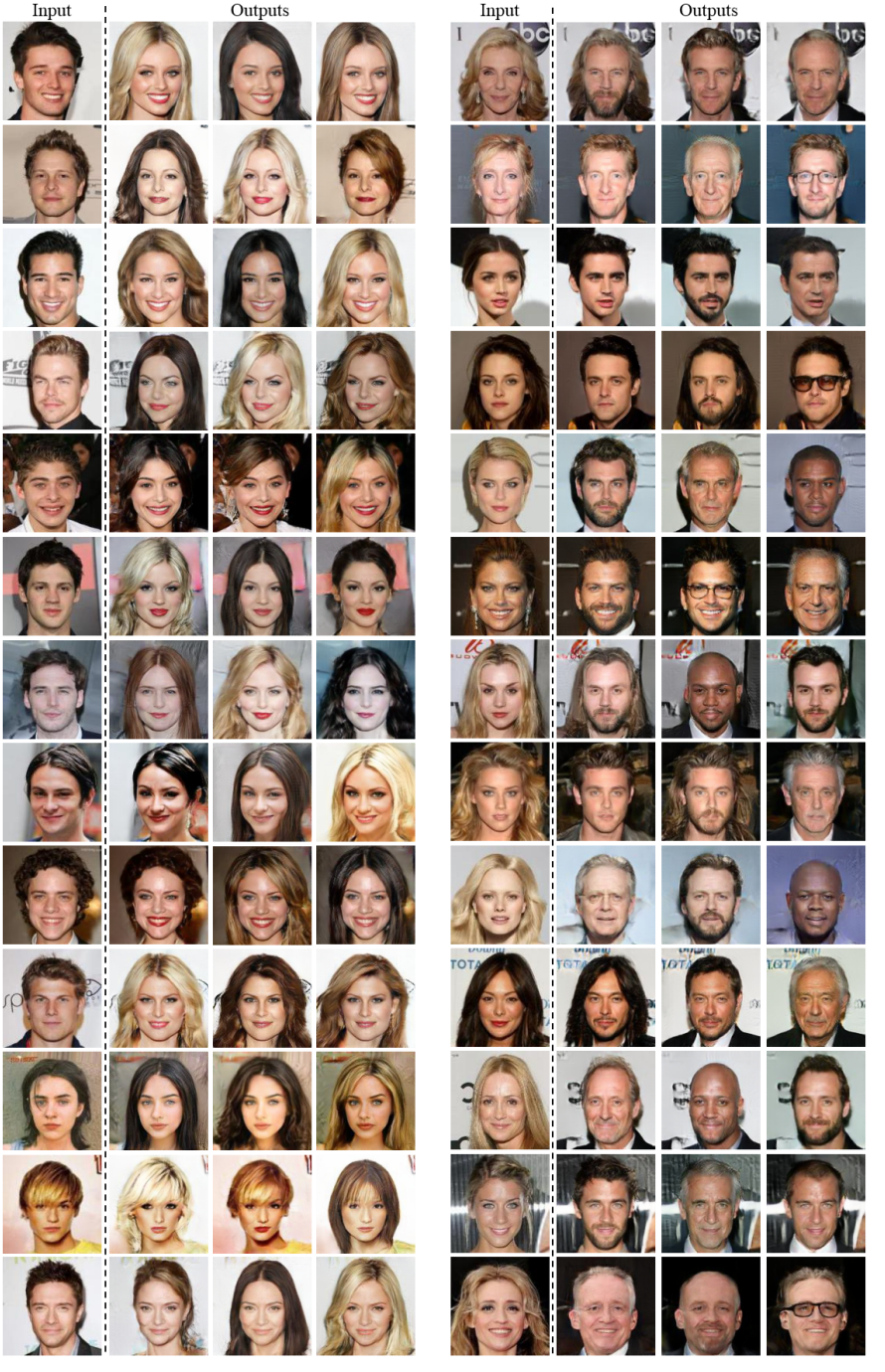

Figure 1: Multimodal image translation on CelebA (left : Male→Female, right : Female→Male).

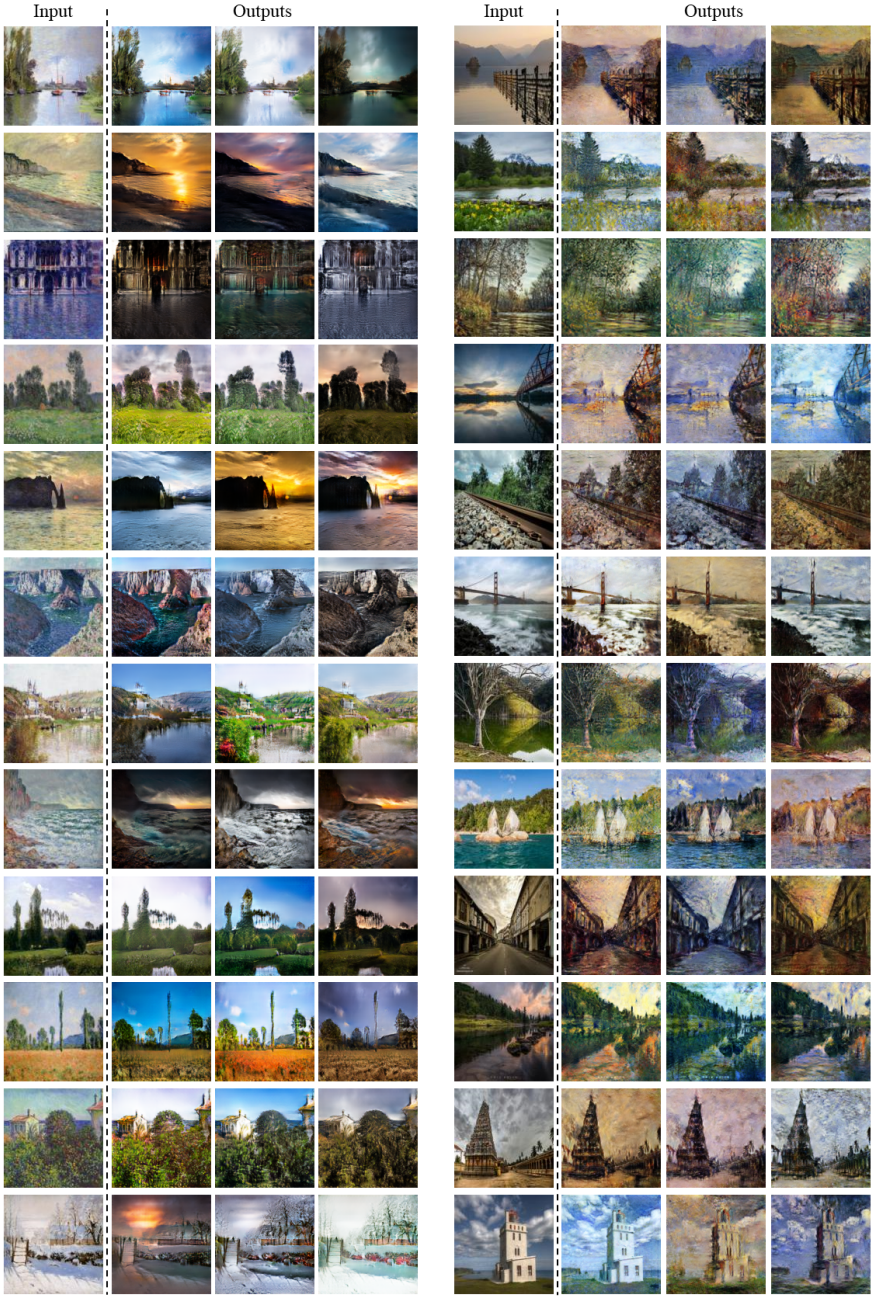

Figure 2: Multimodal image translation on Monet↔Photo (left : Monet→Photo, right : Photo→Monet).

Figure 3: Multimodal image translation on Cat↔Dog (left : Cat→Dog, right : Dog→Cat).

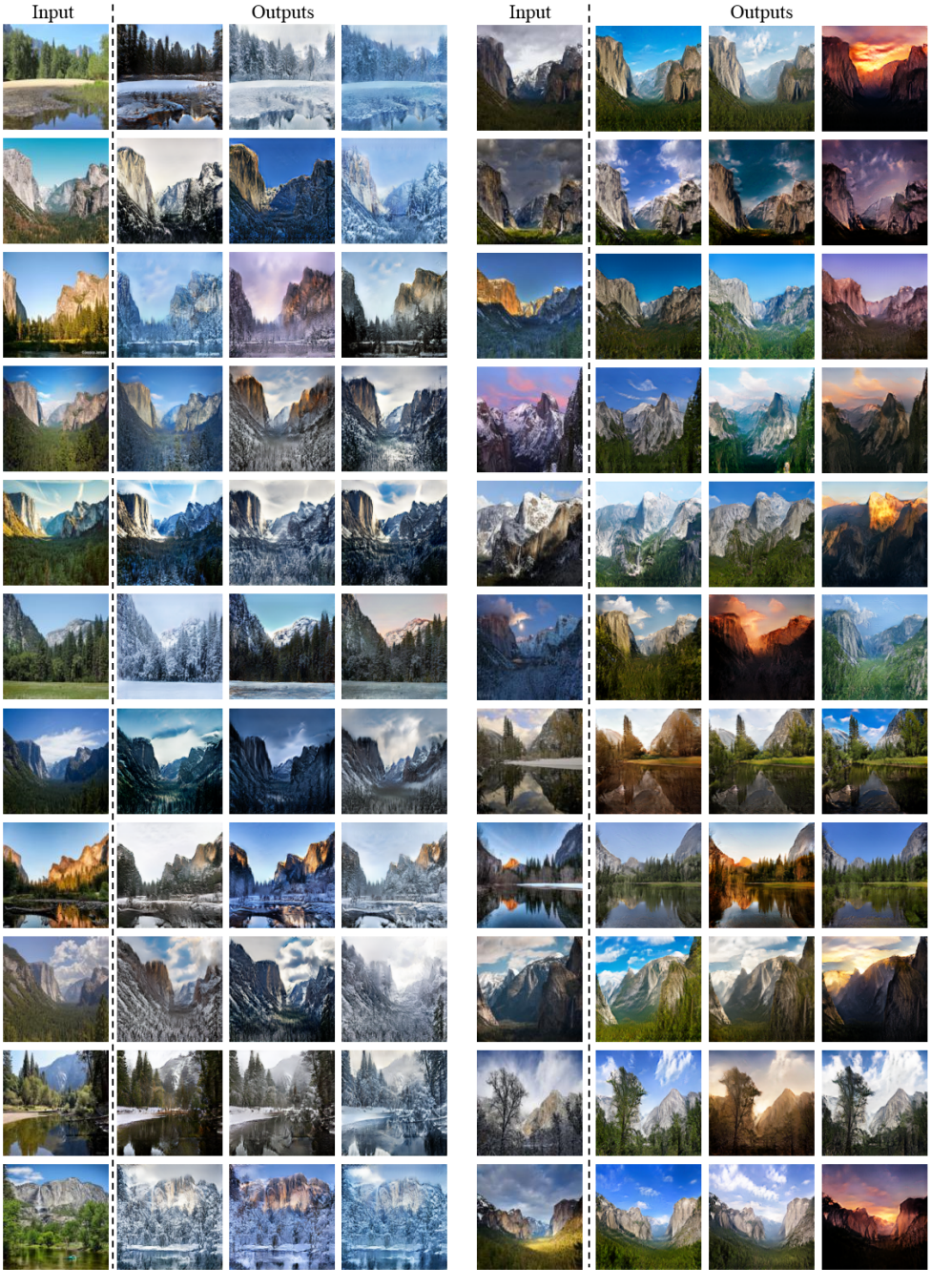

Figure 4: Multimodal image translation on Yosemite (left : Summer→Winter, right : Winter→Summer)

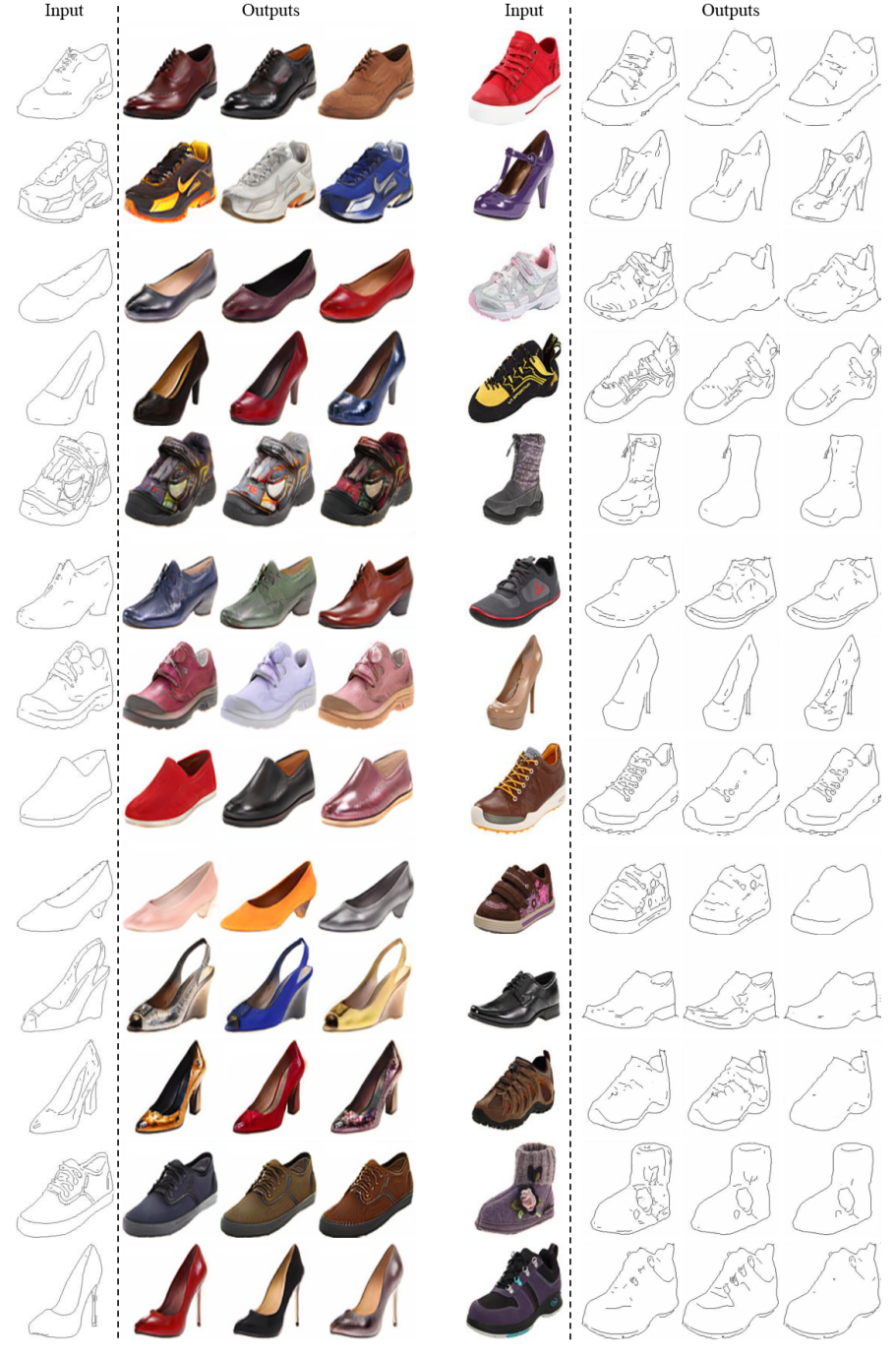

Figure 5: Multimodal image translation on Edges↔Shoes (left : Edges→Shoes, right : Shoes→Edge)

Figure 6: Multimodal image translation on Edges↔Handbags (left : Edges→Handbags, right : Handbags→Edge)

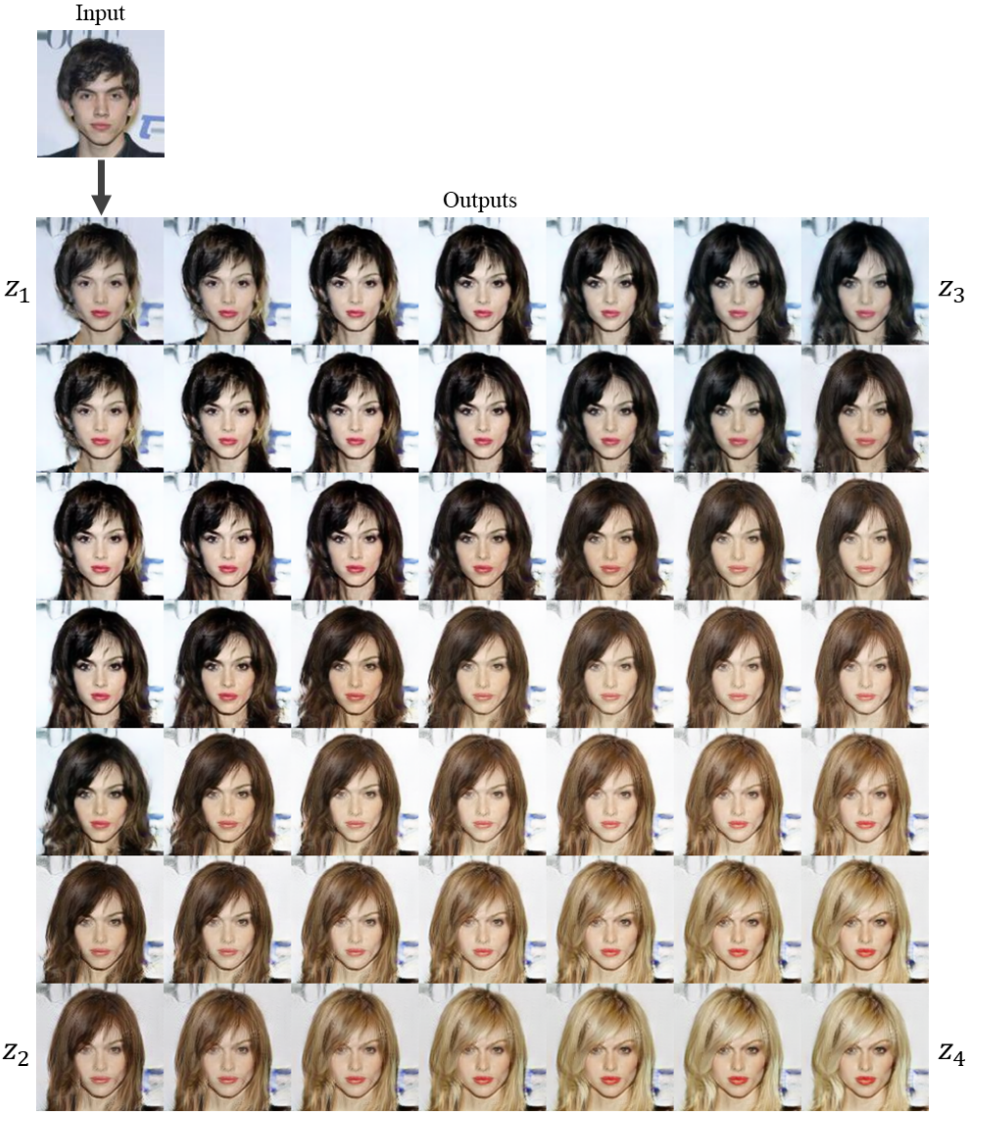

Figure 7: Interpolation between z_1, z_2, z_3, and z_4 on Male→Female.

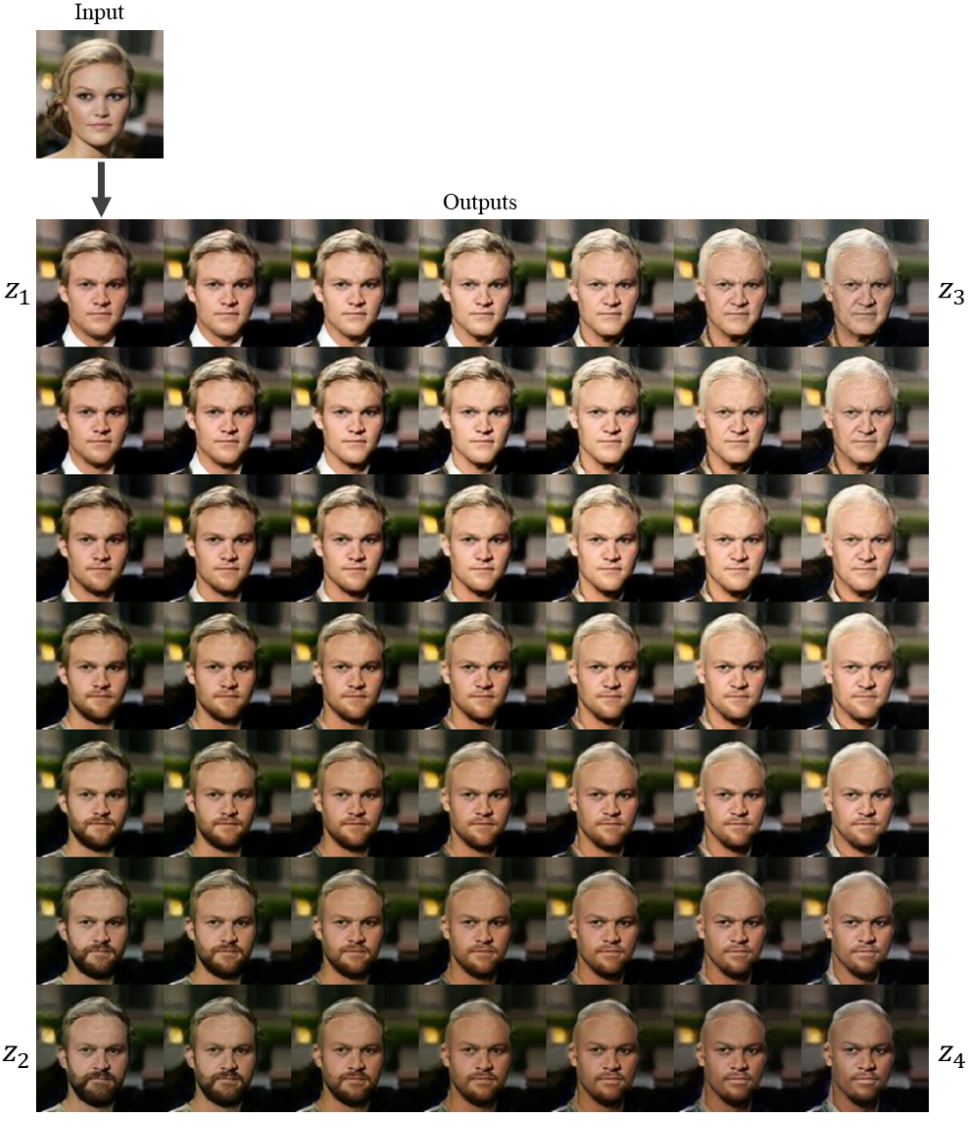

Figure 8: Interpolation between z_1, z_2, z_3, and z_4 on Female→Male.

Input → Output	Layer Configuration
(N, 3, h, w) → (N, 64, $\frac{h}{2}$, $\frac{w}{2}$)	ConvBlock (IN=3, OUT=64, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 64, $\frac{h}{2}$, $\frac{w}{2}$) → (N, 128, $\frac{h}{4}$, $\frac{w}{4}$)	ConvBlock (IN=64, OUT=128, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 128, $\frac{h}{4}$, $\frac{w}{4}$) → (N, 256, $\frac{h}{8}$, $\frac{w}{8}$)	ConvBlock (IN=128, OUT=256, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 256, $\frac{h}{8}$, $\frac{w}{8}$) → (N, 512, $\frac{h}{16}$, $\frac{w}{16}$)	ConvBlock (IN=256, OUT=512, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 512, $\frac{h}{16}$, $\frac{w}{16}$) → (N, 1024, $\frac{h}{32}$, $\frac{w}{32}$)	ConvBlock (IN=512, OUT=1024, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 1024, $\frac{h}{32}$, $\frac{w}{32}$) → (N, 1, $\frac{h}{32}$, $\frac{w}{32}$)	ConvBlock (IN=1024, OUT=1, K=3x3, S=1, P=1, None, None)
(N, 3, h, w) → (N, 64, $\frac{h}{2}$, $\frac{w}{2}$)	ConvBlock (IN=3, OUT=64, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 64, $\frac{h}{2}$, $\frac{w}{2}$) → (N, 128, $\frac{h}{4}$, $\frac{w}{4}$)	ConvBlock (IN=64, OUT=128, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 128, $\frac{h}{4}$, $\frac{w}{4}$) → (N, 256, $\frac{h}{8}$, $\frac{w}{8}$)	ConvBlock (IN=128, OUT=256, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 256, $\frac{h}{8}$, $\frac{w}{8}$) → (N, 512, $\frac{h}{16}$, $\frac{w}{16}$)	ConvBlock (IN=256, OUT=512, K=4x4, S=2, P=1, SN, Leaky ReLU)
(N, 512, $\frac{h}{16}$, $\frac{w}{16}$) → (N, 1, $\frac{h}{16}$, $\frac{w}{16}$)	ConvBlock (IN=512, OUT=1, K=3x3, S=1, P=1, None, None)

Table 1: The architecture of two independent discriminator. SN denotes spectral normalization [2].

Input → Output	Layer Configuration
(N, $3+n_z$, h, w) → (N, 96, h, w)	ConvBlock (IN=$3+n_z$, OUT=96, K=7x7, S=1, P=3, IN, ReLU)
(N, 96, h, w) → (N, 192, $\frac{h}{2}$, $\frac{w}{2}$)	ConvBlock (IN=96, OUT=192, K=4x4, S=2, P=1, IN, ReLU)
(N, 192, $\frac{h}{2}$, $\frac{w}{2}$) → (N, 384, $\frac{h}{4}$, $\frac{w}{4}$)	ConvBlock (IN=192, OUT=384, K=4x4, S=2, P=1, IN, ReLU)
(N, 384, $\frac{h}{4}$, $\frac{w}{4}$) → (N, 384, $\frac{h}{4}$, $\frac{w}{4}$)	ResBlock (C=384, K=3x3, S=1, P=1)
(N, 384, $\frac{h}{4}$, $\frac{w}{4}$) → (N, 384, $\frac{h}{4}$, $\frac{w}{4}$)	ResBlock (C=384, K=3x3, S=1, P=1)
(N, 384, $\frac{h}{4}$, $\frac{w}{4}$) → (N, 384, $\frac{h}{4}$, $\frac{w}{4}$)	ResBlock (C=384, K=3x3, S=1, P=1)
(N, 384, $\frac{h}{4}$, $\frac{w}{4}$) → (N, 192, $\frac{h}{2}$, $\frac{w}{2}$)	TransConvBlock (IN=384, OUT=192, K=4x4, S=2, P=1, IN, ReLU)
(N, 192, $\frac{h}{2}$, $\frac{w}{2}$) → (N, 96, h, w)	TransConvBlock (IN=192, OUT=96, K=4x4, S=2, P=1, IN, ReLU)
(N, 96, h, w) → (N, 3, h, w)	TransConvBlock (IN=96, OUT=3, K=4x4, S=2, P=1, None, TanH)

Table 2: The architecture of generator. IN denotes instance normalization

	λ_{adv}	λ_{info}	λ_{cycle}	λ_{KL}	λ_{lat}
CelebA	1.0	0.05	10.0	0.01	1.0
Monet2Photo	1.0	0.05	50.0	0.01	1.0
Cat2Dog	1.0	0.07	10.0	0.01	1.0
Summer2Winter	1.0	0.10	50.0	0.01	1.0
Edges2Shoes	1.0	0.05	10.0	0.01	1.0
Edges2Handbags	1.0	0.05	10.0	0.01	1.0

Table 3: Hyperparameters to balance between loss functions.

C Hyperparameters

We use the images with size of 128×128 and set the size of style representation to 7. For optimization, we use the Adam optimizer [1] with $(\beta_1, \beta_2) = (0.5, 0.999)$ and learning rate of 0.0002 and do not apply learning rate decay. We set the batch size to 8. Table. 3 shows hyperparameters to balance between the loss functions.

References

[1] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[2] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[3] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.