Supplementary Material

6 Relation Between the Proposed VP Objective and the InfoGAN Objective

6.1 Introduction of InfoGAN

InfoGAN [10] decomposes the latent representation in GAN into two separate parts: $\boldsymbol{z} = [\boldsymbol{z}_{noise}, \boldsymbol{z}_{code}]$, where $\boldsymbol{z}_{noise}$ denotes the incompressible noise and $\boldsymbol{z}_{code}$ denotes latent code capturing salient semantic features. In order to force the latent code $\boldsymbol{z}_{code}$ to learn salient semantics, InfoGAN add a regularization term into the objective of GAN:

$$\min_G \max_D V_{in}(G, D) = V(G, D) - \lambda I(\boldsymbol{x}; \boldsymbol{z}_{code}), \tag{8}$$

where $\boldsymbol{x} = G(\boldsymbol{z}_{code}, \boldsymbol{z}_{noise})$. In this objective, the generator network G is not only guided to synthesize real images with GAN loss $V(G, D)$, but forced to maximize the mutual information between the generated image and the latent codes $\boldsymbol{z}_{code}$. This modification has been shown effective for learning disentangled representations.

6.2 Variation Predictability in InfoGAN

As shown in Eq. 8, the InfoGAN maximizes the mutual information between the generated images $\boldsymbol{x}$ and the latent codes $\boldsymbol{z}$ (we omit the *code* subscript for concision). In this section, we show that maximizing this term implicitly maximizes a variant of the variation predictability objective.

Proposition 1. *For random variables $\boldsymbol{y}$, $\boldsymbol{z}_1$, $\boldsymbol{z}_2$, $(dim(\boldsymbol{z}_1) = dim(\boldsymbol{z}_2))$, and $\Delta\boldsymbol{z} = \boldsymbol{z}_1 - \boldsymbol{z}_2$, we have:*

$$I(\boldsymbol{y}; \boldsymbol{z}_1, \boldsymbol{z}_2) = I(\boldsymbol{y}; \Delta\boldsymbol{z}, \boldsymbol{z}_1). \tag{9}$$

Proof. For the spaces $(\boldsymbol{Z}_1, \boldsymbol{Z}_2)$ and $(\Delta\boldsymbol{Z}, \boldsymbol{Z}_1)$, there is a one-to-one mapping between them:

$$\begin{pmatrix} \Delta\boldsymbol{z} \\ \boldsymbol{z}_1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{I} & -\boldsymbol{I} \\ \boldsymbol{I} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{pmatrix}, \tag{10}$$

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{0}$ is the zero matrix. Note that the transformation matrix is a full-rank matrix so the mapping between the two space is one-to-one. Therefore a probability density function $p_{(\boldsymbol{Z}_1,\boldsymbol{Z}_2)}(\boldsymbol{z}_1, \boldsymbol{z}_2)$ defined in space $(\boldsymbol{Z}_1, \boldsymbol{Z}_2)$, can also be described in the space $(\Delta\boldsymbol{Z}, \boldsymbol{Z}_1)$, which we denote as $p_{(\Delta\boldsymbol{Z},\boldsymbol{Z}_1)}(\Delta\boldsymbol{z}, \boldsymbol{z}_1)$, with equivalence holds at each point:

$$p_{(\boldsymbol{Z}_1,\boldsymbol{Z}_2)}(\boldsymbol{z}_1, \boldsymbol{z}_2) = p_{(\Delta\boldsymbol{Z},\boldsymbol{Z}_1)}(\Delta\boldsymbol{z}, \boldsymbol{z}_1). \tag{11}$$

Similarly, for spaces $(\boldsymbol{Y}, \boldsymbol{Z}_1, \boldsymbol{Z}_2)$ and $(\boldsymbol{Y}, \Delta\boldsymbol{Z}, \boldsymbol{Z}_1)$, we also have:

$$p_{(\boldsymbol{Y},\boldsymbol{Z}_1,\boldsymbol{Z}_2)}(\boldsymbol{y}, \boldsymbol{z}_1, \boldsymbol{z}_2) = p_{(\boldsymbol{Y},\Delta\boldsymbol{Z},\boldsymbol{Z}_1)}(\boldsymbol{y}, \Delta\boldsymbol{z}, \boldsymbol{z}_1), \tag{12}$$

because there is a full-rank transformation matrix between these two spaces:

$$\begin{pmatrix} \boldsymbol{y} \\ \Delta\boldsymbol{z} \\ \boldsymbol{z}_1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} & -\boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{pmatrix}. \tag{13}$$

Now we can expand the mutual information terms in Eq. 9 to show their equivalence.

$$\begin{aligned} I(\boldsymbol{y}; \boldsymbol{z}_1, \boldsymbol{z}_2) &= \int_{\boldsymbol{y}} \int_{\boldsymbol{z}_1} \int_{\boldsymbol{z}_2} p(\boldsymbol{y}, \boldsymbol{z}_1, \boldsymbol{z}_2) \log \frac{p(\boldsymbol{y}, \boldsymbol{z}_1, \boldsymbol{z}_2)}{p(\boldsymbol{y}) p(\boldsymbol{z}_1, \boldsymbol{z}_2)} d\boldsymbol{y} d\boldsymbol{z}_1 d\boldsymbol{z}_2 && (14) \\ &= \int_{\boldsymbol{y}} \int_{\Delta\boldsymbol{z}} \int_{\boldsymbol{z}_1} p_{(\boldsymbol{Y},\Delta\boldsymbol{Z},\boldsymbol{Z}_1)}(\boldsymbol{y}, \Delta\boldsymbol{z}, \boldsymbol{z}_1) && (15) \\ &\qquad \cdot \log \frac{p_{(\boldsymbol{y},\Delta\boldsymbol{z},\boldsymbol{z}_1)}(\boldsymbol{y}, \Delta\boldsymbol{z}, \boldsymbol{z}_1)}{p(\boldsymbol{y}) p_{(\Delta\boldsymbol{Z},\boldsymbol{Z}_1)}(\Delta\boldsymbol{z}, \boldsymbol{z}_1)} d\boldsymbol{y} d\Delta\boldsymbol{z} d\boldsymbol{z}_1 && (16) \\ &= I(\boldsymbol{y}; \Delta\boldsymbol{z}, \boldsymbol{z}_1), && (17) \end{aligned}$$

where the subscripts for $p_{(\boldsymbol{Z}_1,\boldsymbol{Z}_2)}(\boldsymbol{z}_1, \boldsymbol{z}_2)$ and $p_{(\boldsymbol{Y},\boldsymbol{Z}_1,\boldsymbol{Z}_2)}(\boldsymbol{y}, \boldsymbol{z}_1, \boldsymbol{z}_2)$ are omitted for concision. □

Proposition 2. *For variables $\boldsymbol{x}_1 = G(\boldsymbol{z}_1)$ and $\boldsymbol{x}_2 = G(\boldsymbol{z}_2)$, where $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are independent variables from an identical prior distribution $p(\boldsymbol{z})$, and $\Delta\boldsymbol{z} = \boldsymbol{z}_1 - \boldsymbol{z}_2$, we have:*

$$I(\boldsymbol{x}_1; \boldsymbol{z}_1) + I(\boldsymbol{x}_2; \boldsymbol{z}_2) = I(\boldsymbol{x}_1, \boldsymbol{x}_2; \Delta\boldsymbol{z}) + H(\boldsymbol{z}_1|\Delta\boldsymbol{z}) - H(\boldsymbol{z}_1|\Delta\boldsymbol{z}, \boldsymbol{x}_1, \boldsymbol{x}_2). \tag{18}$$

Proof.

$$\begin{aligned} I(\boldsymbol{x}_1;&\boldsymbol{z}_1) + I(\boldsymbol{x}_2; \boldsymbol{z}_2) && (19) \\ &= I(\boldsymbol{x}_1, \boldsymbol{x}_2; \boldsymbol{z}_1, \boldsymbol{z}_2) && (20) \\ &= I(\boldsymbol{x}_1, \boldsymbol{x}_2; \Delta\boldsymbol{z}, \boldsymbol{z}_1) && (21) \\ &= H(\Delta\boldsymbol{z}, \boldsymbol{z}_1) - H(\Delta\boldsymbol{z}, \boldsymbol{z}_1|\boldsymbol{x}_1, \boldsymbol{x}_2) && (22) \\ &= H(\Delta\boldsymbol{z}) + H(\boldsymbol{z}_1|\Delta\boldsymbol{z}) - H(\Delta\boldsymbol{z}|\boldsymbol{x}_1, \boldsymbol{x}_2) - H(\boldsymbol{z}_1|\Delta\boldsymbol{z}, \boldsymbol{x}_1, \boldsymbol{x}_2) && (23) \\ &= I(\boldsymbol{x}_1, \boldsymbol{x}_2; \Delta\boldsymbol{z}) + H(\boldsymbol{z}_1|\Delta\boldsymbol{z}) - H(\boldsymbol{z}_1|\Delta\boldsymbol{z}, \boldsymbol{x}_1, \boldsymbol{x}_2) && (24) \end{aligned}$$

Eq. 20 is because $\boldsymbol{z}_1 \perp\!\!\!\perp \boldsymbol{z}_2$ and $\boldsymbol{x}_1 \perp\!\!\!\perp \boldsymbol{x}_2$. Eq. 21 is based on proposition 1. Eq. 23 uses the chain rule in entropy. □

Since $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are independent variables from the same distribution, maximizing the left-hand-side of Eq. 18 is equivalent to maximizing the $I(\boldsymbol{x}; \boldsymbol{z})$ in InfoGAN objective (Eq. 8). We can see based on proposition 2, optimizing the InfoGAN objective implicitly maximizes the subterm $I(\boldsymbol{x}_1, \boldsymbol{x}_2; \Delta\boldsymbol{z})$, which is very similar to our proposed VP objective $I(\boldsymbol{x}_1, \boldsymbol{x}_2; d)$. The difference between these two terms is that $I(\boldsymbol{x}_1, \boldsymbol{x}_2; \Delta\boldsymbol{z})$ tries to maximize the mutual information between all the variations in latent variables and the paired images while ours tries to emphasize on the learning of a single dimension of variation, which is an easier objective than InfoGAN and more focused on disentanglement. This may be the reason why our models can learn better disentangled representations and maintain a stabler training.

7 Proof of Lemma 1

Lemma 1. *For the mutual information between two random variables $I(\boldsymbol{x}; \boldsymbol{y})$, the following lower bound holds:*

$$I(\boldsymbol{x}; \boldsymbol{y}) \geq H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})} \log q(\boldsymbol{y}|\boldsymbol{x}), \tag{25}$$

where the bound is tight when $q(\boldsymbol{y}|\boldsymbol{x}) = p(\boldsymbol{y}|\boldsymbol{x})$.

Proof.

$$\begin{aligned}
I(\boldsymbol{x}; \boldsymbol{y}) &= H(\boldsymbol{y}) - H(\boldsymbol{y}|\boldsymbol{x}) && (26)\\
&= H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})} \log p(\boldsymbol{y}|\boldsymbol{x}) && (27)\\
&= H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x})}\mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{x})} \log \frac{p(\boldsymbol{y}|\boldsymbol{x})q(\boldsymbol{y}|\boldsymbol{x})}{q(\boldsymbol{y}|\boldsymbol{x})} && (28)\\
&= H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x})}\mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{x})} \log q(\boldsymbol{y}|\boldsymbol{x}) + \mathbb{E}_{p(\boldsymbol{x})}\mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{x})} \log \frac{p(\boldsymbol{y}|\boldsymbol{x})}{q(\boldsymbol{y}|\boldsymbol{x})} && (29)\\
&= H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})} \log q(\boldsymbol{y}|\boldsymbol{x}) + \mathbb{E}_{p(\boldsymbol{x})} D_{KL}(p(\boldsymbol{y}|\boldsymbol{x})||q(\boldsymbol{y}|\boldsymbol{x})) && (30)\\
&\geq H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})} \log q(\boldsymbol{y}|\boldsymbol{x}), && (31)
\end{aligned}$$

where Eq. 31 is because of the non-negativity of KL divergence. □

8 VP Metric: Training Set Ratio Study

We conducts experiments to show how the parameter η in our proposed VP metric influences the evaluation results in Table 6 and 7. We can see when the training ratio is large, it is hard to distinguish the entangled and the disentangled models based on the VP metric, and that is why we choose to keep the ratio low in our experiments.

9 More Quantitative Comparisons on Disentanglement

We directly compare our VPGAN with baselines of CascadeVAE and InfoGAN using FactorVAE metric and our VP metric in Table 8 and 9. For VPGAN, we train an inference network to map images to the latent space. Scores are averaged by 3 random runs except for CascadeVAE. We can see CascadeVAE still achieves

η	0.01	0.5	0.9
β-VAE	42.2	89.4	97.1
CascadeVAE	62.4	95.8	98.5

Table 6. Ratio study on Dsprites.

η	0.01	0.5	0.9
β-VAE	39.7	87.5	98.2
CascadeVAE	70.6	97.1	99.4

Table 7. Ratio study on 3DShapes.

Model	FactorVAE Score	VP Score
CascadeVAE	**91.3** (7.4)	**59.2** (4.6)
InfoGAN-0.1	62.7 (5.2)	24.3 (6.9)
VPGAN-0.1	69.5 (4.7)	38.8 (5.1)

Table 8. Comparison on Dsprites.

Model	FactorVAE Score	VP Score
CascadeVAE	**94.7** (2.1)	**62.3** (4.9)
InfoGAN-0.1	60.4 (5.1)	31.7 (7.9)
VPGAN-0.1	74.1 (3.2)	57.2 (6.5)

Table 9. Comparison on 3DShapes.

the best performance, and our VPGAN can outperform the InfoGAN baseline by an obvious margin. Note that GAN-based models are sensitive to network architectures, which are not carefully tuned in our experiments, thus the results between the VAE- and GAN-based models may not be directly comparable.

10 Network Architectures

The encoders, decoders, and recognizors for Dsprites and 3DShapes are shown in Table 10, Table 11, and Table 12. The $\beta_{high} = 20$, $\beta_{low} = 2$ and $\alpha = 100$ for Dsprites dataset, while $\beta_{high} = 40$, $\beta_{low} = 3$, and $\alpha = 10$ for 3DShapes dataset. The generators for flat and hierarchical VPGANs on CelebA 128×128 dataset are in Table 13, Table 14, Note that for CelebA we used conv-layers and modulated-conv-layers from stylegan2 code base https://github.com/NVlabs/stylegan2 as building blocks for implementation. The discriminators for all CelebA models are default discriminators in stylegan2 (with residual net connections) but with 128× 128 input size. The recognizors are the same as discriminators but with output layer modified to output dimensions same as the number of input latent codes. The InfoGAN baselines are using the same generator and discriminator as the VPGAN-hierarchical, and there is another branch at the end of its discriminator to output logits to form the InfoGAN regularization term in InfoGAN objective. The generators for VPGANs on 3DChairs are in Table 15. All models are trained with Adam optimizer with learning rate of 0.002.

11 Traversal of InfoGAN

See Fig. 7 to Fig. 14.

Encoder-64
$64 \times 64 \times$ nchannel
4×4 Conv. 32, ReLU, Stride 2
4×4 Conv. 32, ReLU, Stride 2
4×4 Conv. 64, ReLU, Stride 2
4×4 Conv. 64, ReLU, Stride 2
FC. 256
FC. $2 \times$ nconti

Table 10. Encoder architecture on Dsprites and 3DShapes datasets.

Decoder-64
latent code $\in \mathbb{R}^{\text{nconti}}$
FC. 128, ReLU
FC. $4 \times 4 \times 64$, ReLU
4×4 Deconv. 64, ReLU, Stride 2
4×4 Deconv. 32, ReLU, Stride 2
4×4 Deconv. 32, ReLU, Stride 2
4×4 Deconv. nchannel, ReLU, Stride 2

Table 11. Decoder architecture on Dsprites and 3DShapes datasets.

Encoder-64
$64 \times 64 \times$ nchannel
4×4 Conv. 32, ReLU, Stride 2
4×4 Conv. 32, ReLU, Stride 2
4×4 Conv. 64, ReLU, Stride 2
4×4 Conv. 64, ReLU, Stride 2
FC. 256
FC. nconti

Table 12. Recognizor architecture on Dsprites and 3DShapes datasets.

12 Traversal of VPGAN-flat

See Fig. 15 to Fig. 22. The traversal animation can be found in the file *VPGAN_flat_animations.gif*.

13 Traversal of VPGAN-hierarchical

See Fig. 23 to Fig. 31. The traversal animation can be found in the file *VPGAN_hier_animations.gif*.

Generator-flat-CelebA128
4 × 4 × 128 Learnable Constant
3 × 3 Deconv. ReLU
3 × 3 ModuConv. ReLU, Latents 30
3 × 3 Conv. Relu
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Deconv. ReLU
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Deconv. ReLU
3 × 3 Conv. Relu
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Deconv. ReLU
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv. Relu
3 × 3 Conv. Relu
3 × 3 Deconv. ReLU
3 × 3 Conv. Relu
128 × 128 × nchannel

Table 13. Generator of VPGAN-flat on CelebA dataset.

Generator-hierarchical-CelebA128
4 × 4 × 128 Learnable Constant
3 × 3 Deconv. ReLU
3 × 3 ModuConv. ReLU, Latents 10
3 × 3 Conv. Relu
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Deconv. ReLU
3 × 3 ModuConv. ReLU, Latents 10
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Deconv. ReLU
3 × 3 ModuConv. ReLU, Latents 5
3 × 3 Conv. Relu
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Deconv. ReLU
3 × 3 ModuConv. ReLU, Latents 5
3 × 3 Conv. Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv.+Noise Relu
3 × 3 Conv. Relu
3 × 3 Conv. Relu
3 × 3 Deconv. ReLU
3 × 3 Conv. Relu
128 × 128 × nchannel

Table 14. Generator of VPGAN-hierarchical on CelebA dataset.

Generator-3DChairs
$4 \times 4 \times 128$ Learnable Constant
3×3 Deconv. ReLU
3×3 ModuConv. ReLU, Latents 5
3×3 Deconv. ReLU
3×3 ModuConv. ReLU, Latents 5
3×3 Deconv. ReLU
3×3 ModuConv. ReLU, Latents 3
3×3 Deconv. ReLU
3×3 Conv. Relu

Table 15. Generator for 3DChairs.

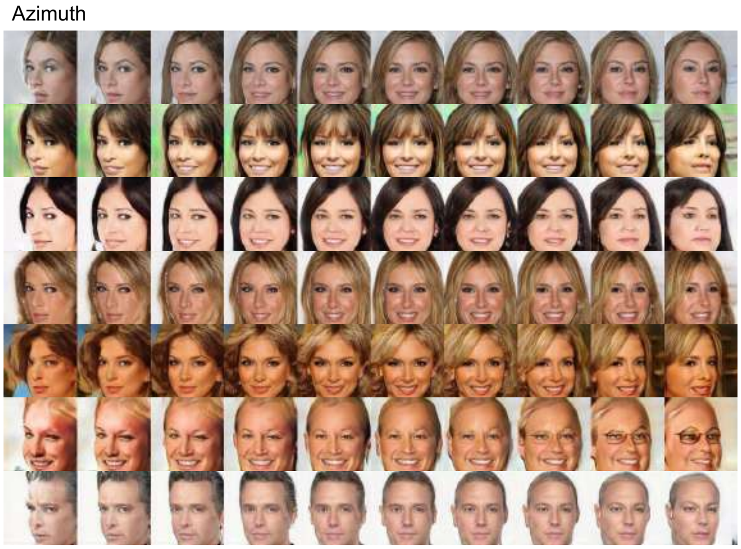

Fig. 7. Azimuth of InfoGAN.

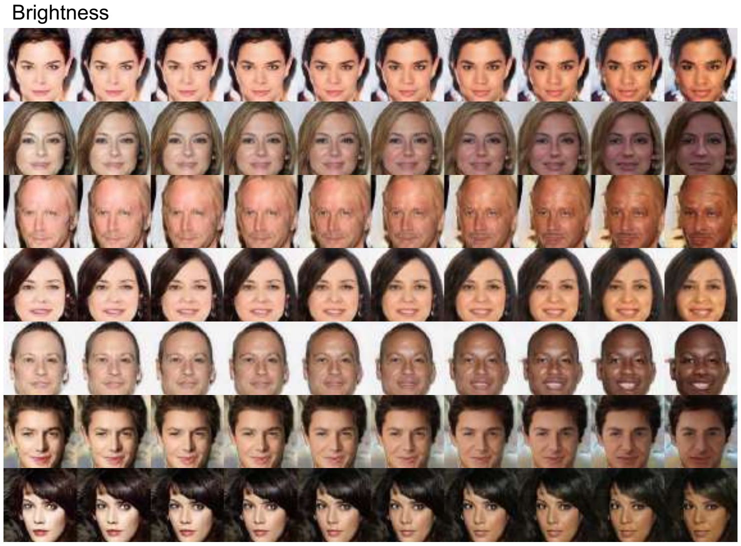

Fig. 8. Brightness of InfoGAN.

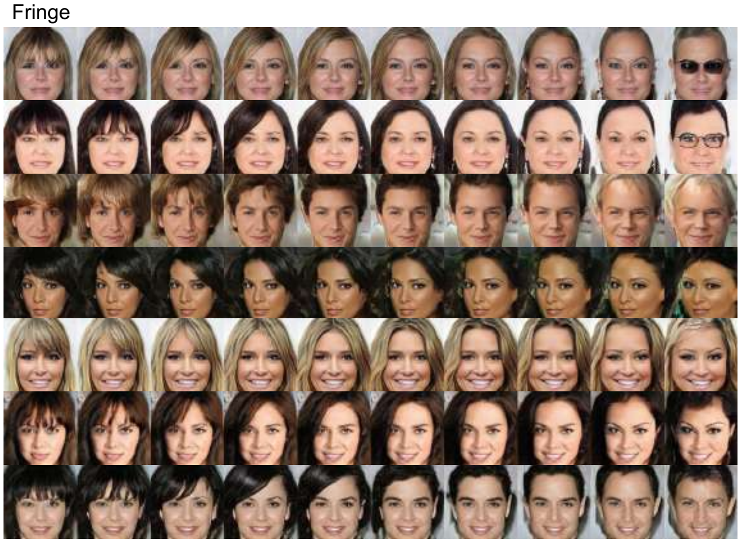

Fig. 9. Fringe of InfoGAN.

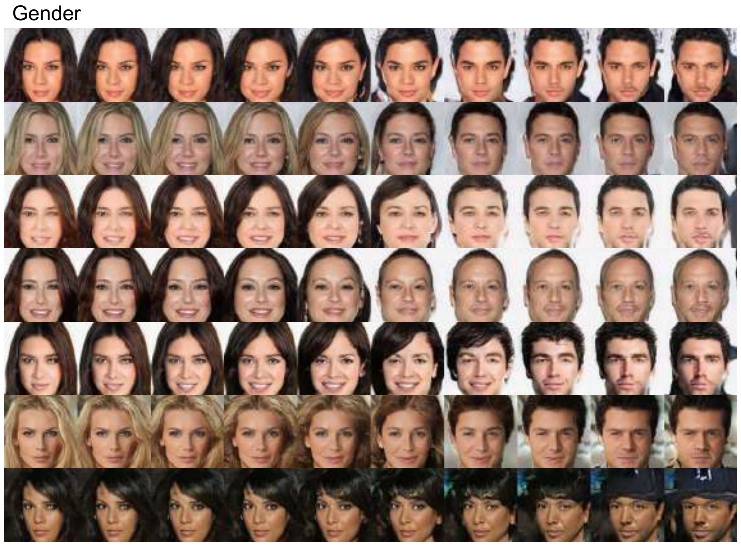

Fig. 10. Gender of InfoGAN.

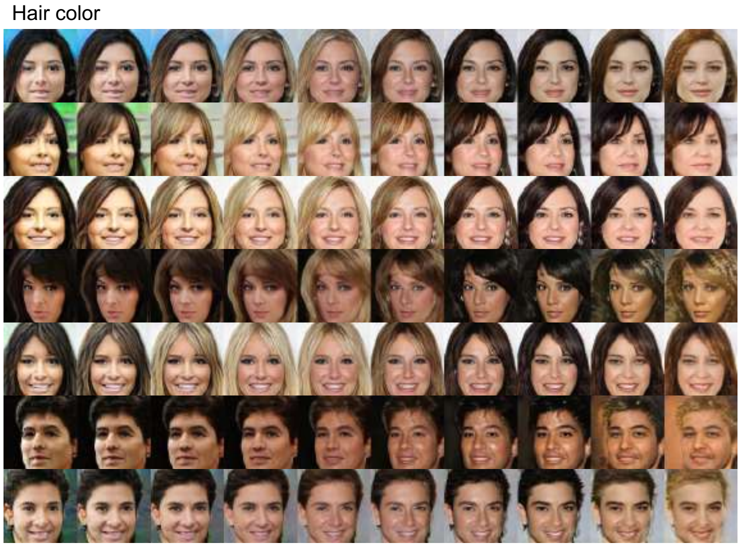

Fig. 11. Hair Color of InfoGAN.

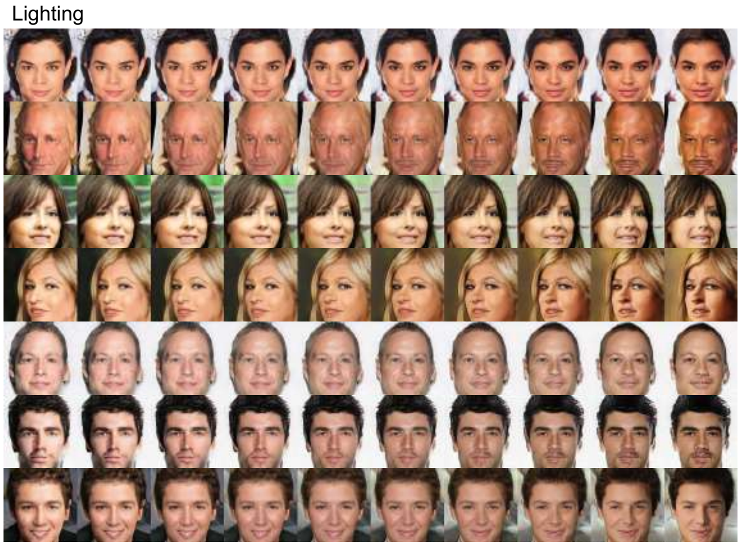

Fig. 12. Lighting of InfoGAN.

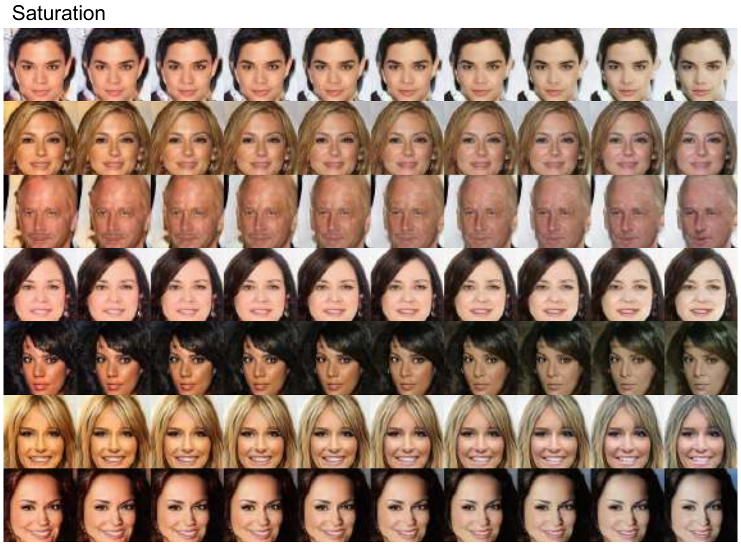

Fig. 13. Saturation of InfoGAN.

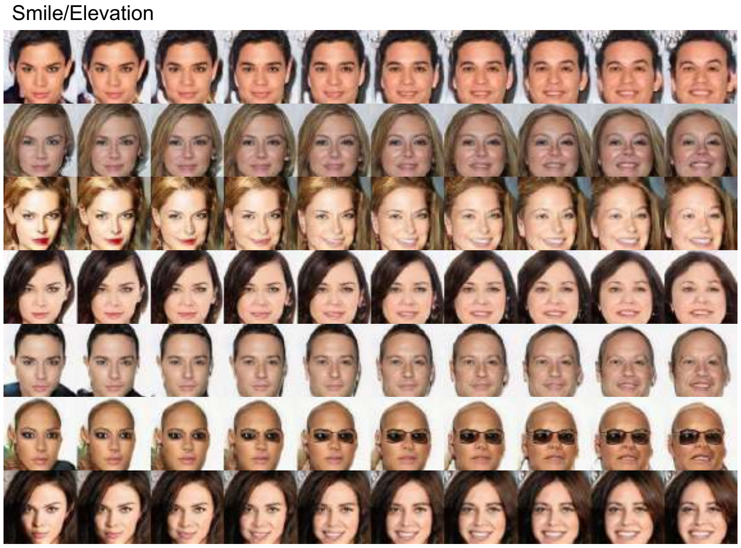

Fig. 14. Smile of InfoGAN.

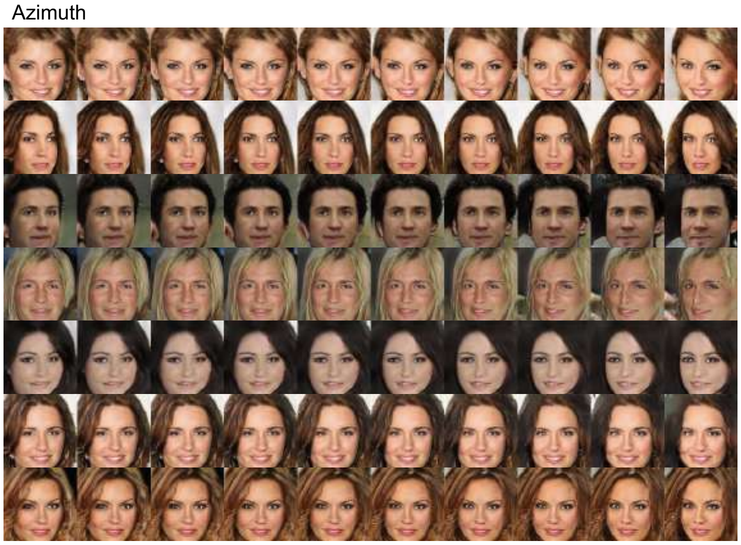

Fig. 15. Azimuth of VPGAN-flat.

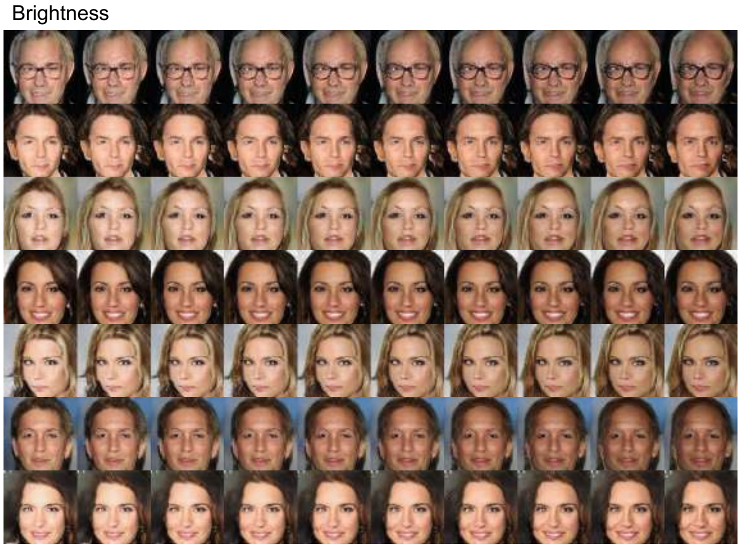

Fig. 16. Brightness of VPGAN-flat.

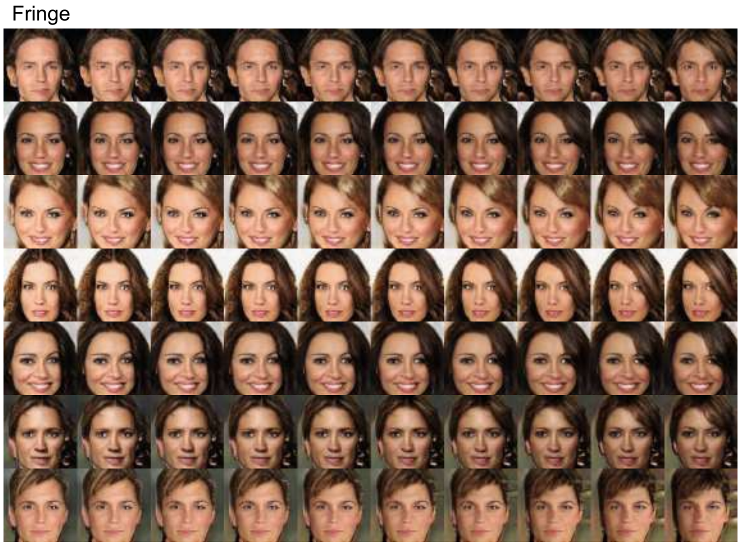

Fig. 17. Fringe of VPGAN-flat.

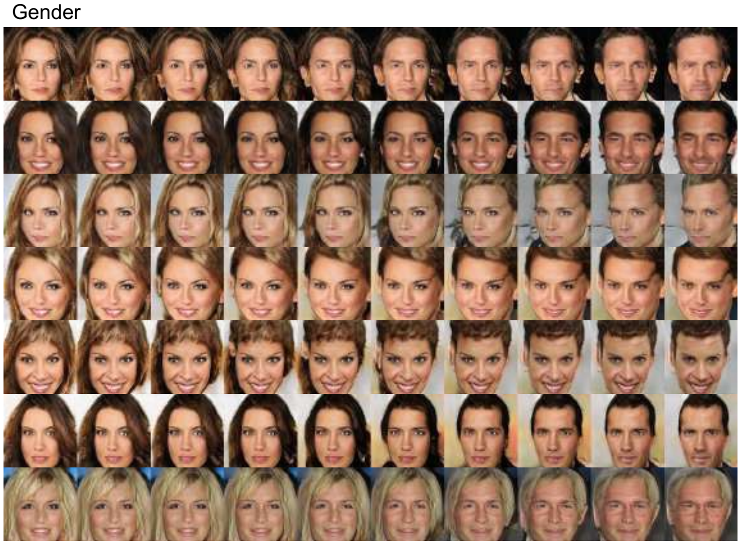

Fig. 18. Gender of VPGAN-flat.

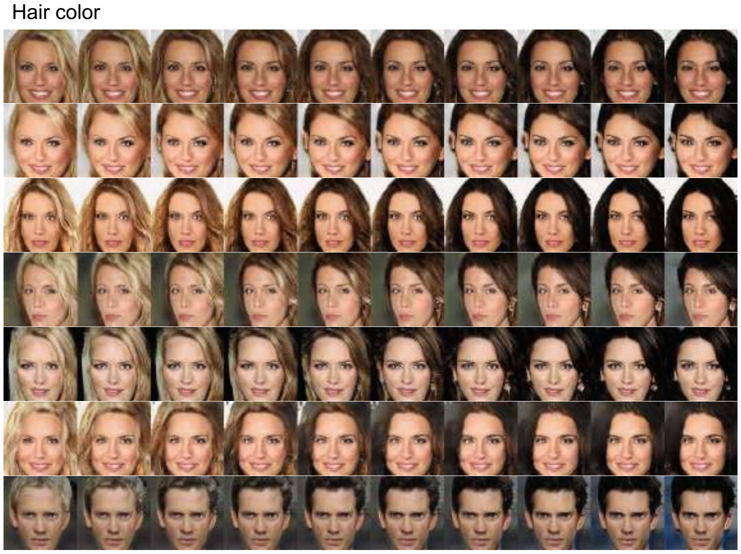

Fig. 19. Hair Color of VPGAN-flat.

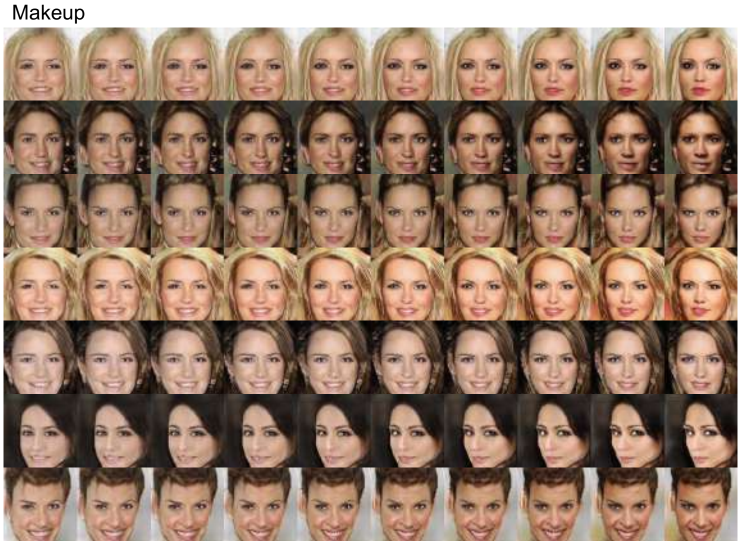

Fig. 20. Makeup of VPGAN-flat.

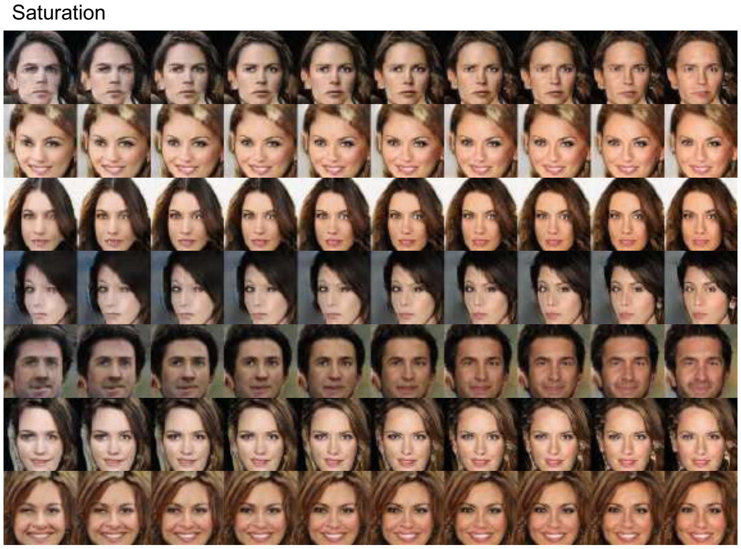

Fig. 21. Saturation of VPGAN-flat.

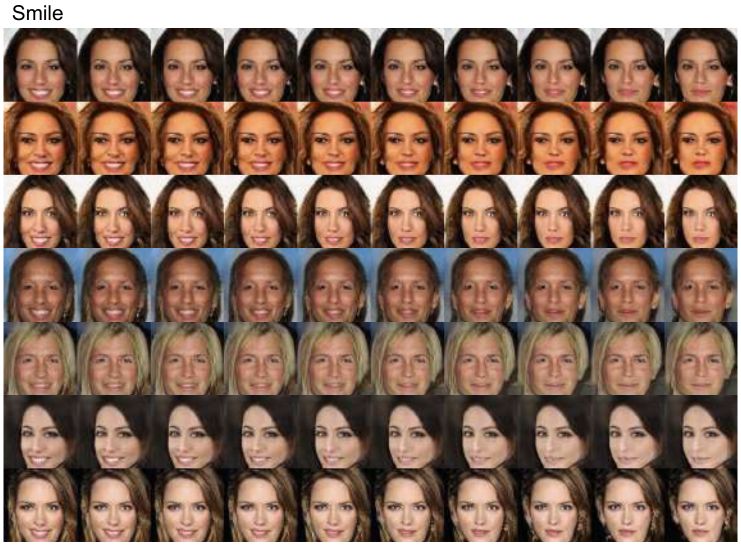

Fig. 22. Smile of VPGAN-flat.

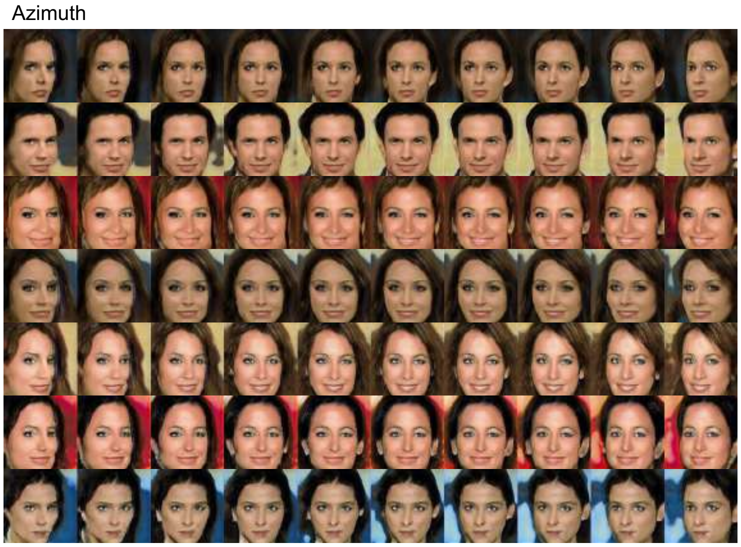

Fig. 23. Azimuth of VPGAN-hierarchical.

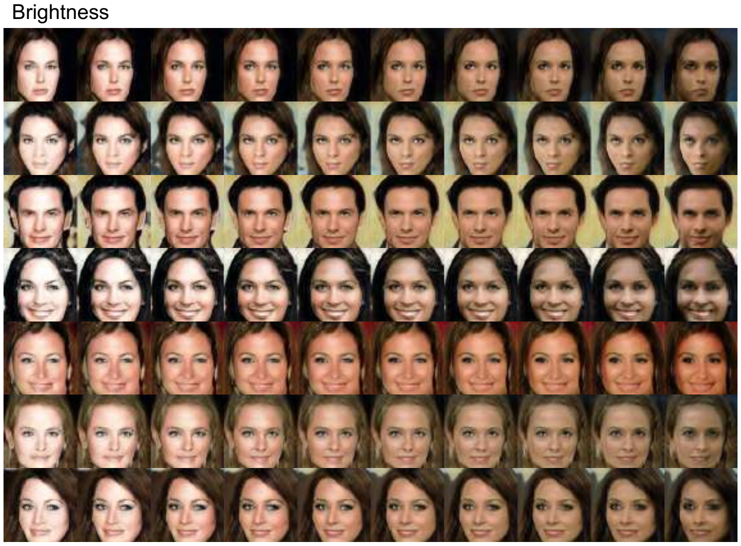

Fig. 24. Brightness of VPGAN-hierarchical.

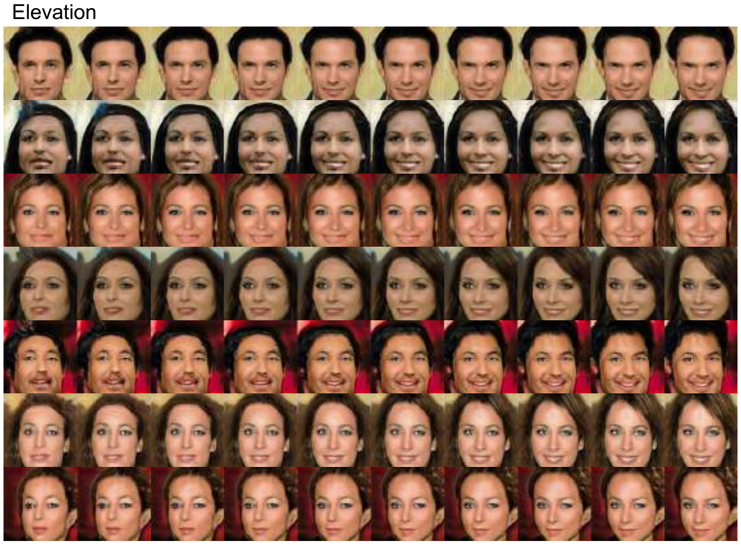

Fig. 25. Elevation of VPGAN-hierarchical.

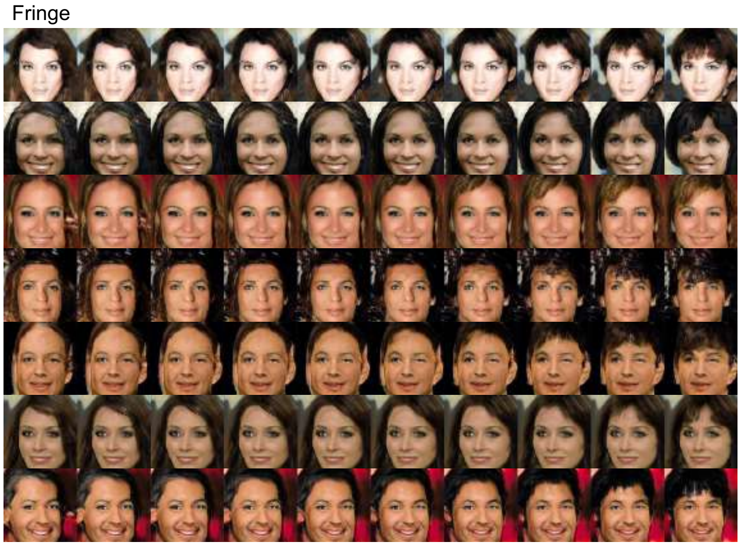

Fig. 26. Fringe of VPGAN-hierarchical.

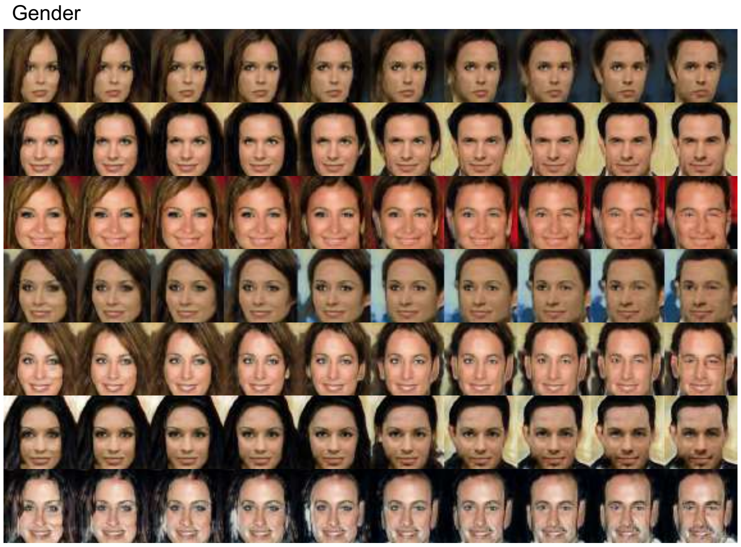

Fig. 27. Gender of VPGAN-hierarchical.

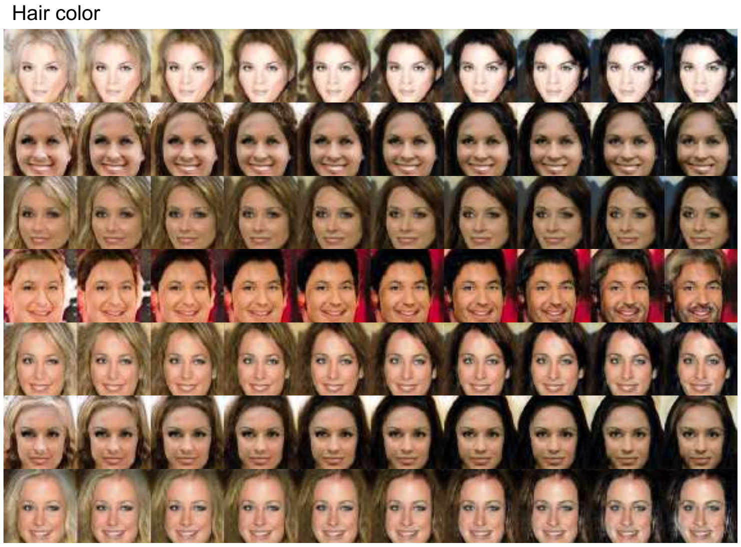

Fig. 28. Hair Color of VPGAN-hierarchical.

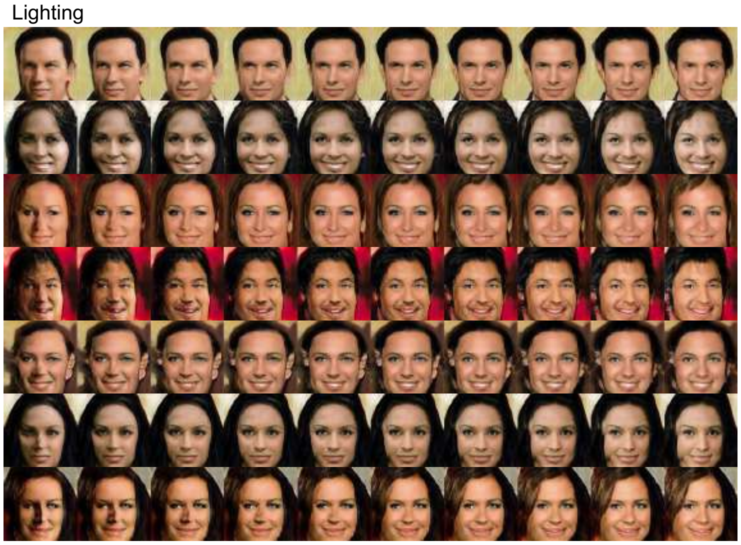

Fig. 29. Lighting of VPGAN-hierarchical.

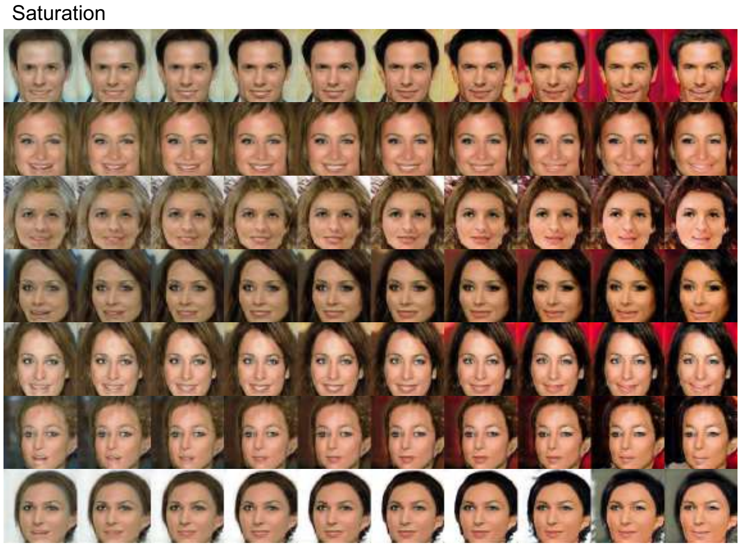

Fig. 30. Saturation of VPGAN-hierarchical.

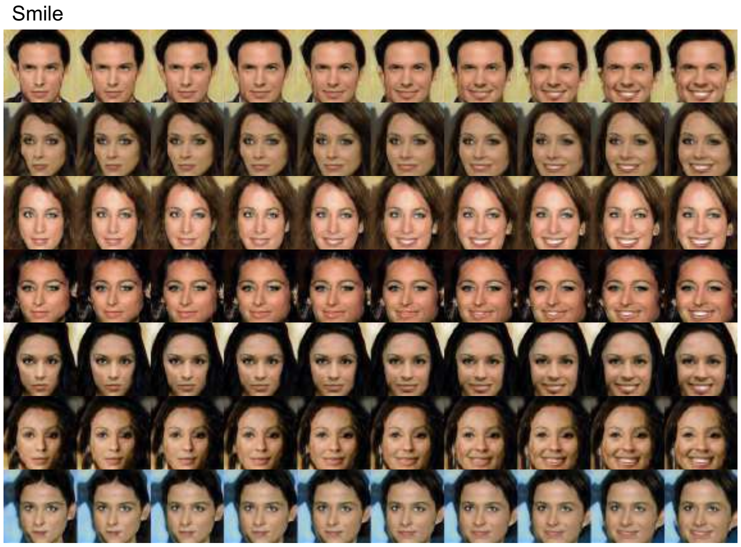

Fig. 31. Smile of VPGAN-hierarchical.