A Appendix

A.1 Architecture

Encoder	Decoder
4×4 conv. 32 stride 2	FC.256
4×4 conv. 32 stride 2	FC. $4 \times 4 \times 64$
4×4 conv. 64 stride 2	4×4 deconv. 64 stride 2
4×4 conv. 64 stride 2	4×4 deconv. 32 stride 2
FC. 256	4×4 deconv. 32 stride 2
FC. 20	4×4 deconv. c stride 2

Table 4: The architecture details. "FC." denotes fully connected layer, "conv." denotes convolutional layer, "deconv" denotes transposed convolution layer. c is the dimension of color channel.

We use symmetric convolutional networks for encoders and decoders as shown in Table 4. $c = 1$ for dSprites, and $c = 3$ for Shapes3D. All layers are activated by ReLU. The final layer of encoder generates 10 variables for *mean* and 10 variables for the *logvar*.

A.2 Disentanglement-invariant Representations

In this section, we prove the proposed disentanglement-invariant transformation. Consider that we have a new representation by multiplying a diagonal matrix: $\boldsymbol{z}' = \boldsymbol{w}\boldsymbol{z}$, $\boldsymbol{w}$. We can calculate the Covariance between any two latent variables:

$$\begin{aligned} \mathrm{Cov}(\boldsymbol{w}_i\boldsymbol{z}_i, \boldsymbol{w}_j\boldsymbol{z}_j) &= \mathbb{E}[(\boldsymbol{w}_i\boldsymbol{z}_i - \mathbb{E}[\boldsymbol{w}_i\boldsymbol{z}_i])(\boldsymbol{w}_j\boldsymbol{z}_j - \mathbb{E}[\boldsymbol{w}_j\boldsymbol{z}_j])] \\ &= \boldsymbol{w}_i\boldsymbol{w}_j(\mathbb{E}[\boldsymbol{z}_j] - \mathbb{E}[\boldsymbol{z}_i]\mathbb{E}[\boldsymbol{z}_j]) \\ &= \boldsymbol{w}_i\boldsymbol{w}_j\mathrm{Cov}(\boldsymbol{z}_i, \boldsymbol{z}_j), \end{aligned} \tag{8}$$

where the subscript denotes the index of latent variables. Note that we use a different notion in this section to simplify the formula.

Then we can get the correlation coefficient by

$$\begin{aligned} \rho(\boldsymbol{w}_i\boldsymbol{z}_i, \boldsymbol{w}_j\boldsymbol{z}_j) &= \frac{\mathrm{Cov}(\boldsymbol{w}_i\boldsymbol{z}_i, \boldsymbol{w}_j\boldsymbol{z}_j)}{\sqrt{\mathrm{Var}[\boldsymbol{w}_i\boldsymbol{z}_i]\,\mathrm{Var}[\boldsymbol{w}_j\boldsymbol{z}_j]}} \\ &= \rho(\boldsymbol{z}_i, \boldsymbol{z}_j). \end{aligned} \tag{9}$$

Therefore, the correlation matrix will not change by multiplying a diagonal matrix $w, w \neq 0$. The proposed transformation is disentanglement-invariant.

A.3 Estimation of $I(\boldsymbol{z}_j; \boldsymbol{c}_i)$

Given an inference network $q(\boldsymbol{z}|\boldsymbol{x})$, we use the Markov chain Monte Carlo (MCMC) method to get samples from $q(\boldsymbol{z})$ by the formula $q(\boldsymbol{z}) = q(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})$. We use 10, 000 points to

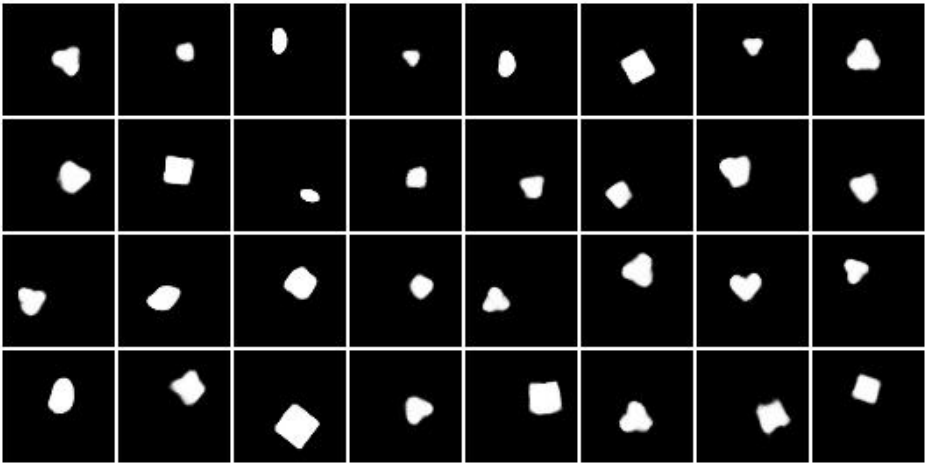

Figure 5: Reconstruction from noise.

estimate $q(\boldsymbol{z})$. Then, we discretize these latent variables by a histogram with 20 bins. After discretizing one latent variable, we call a discrete mutual information estimation algorithm to calculate $I(\boldsymbol{w}_j \boldsymbol{z}_j; \boldsymbol{c}_i)$ by a 2D histogram.

A.4 Visualization

Latent Traversal. We compare DeVAE to others with latent traversals on Shapes3D and dSprites. Each column shows the generated images by traversing one latent variable from -2 to 2. From Figure 6 and Figure 7, we can see that DeVAE has a lower entanglement level. Note that only DeVAE disentangles object size isolated on Shapes3D.

Random Sampling. We random sample noise from Guassian distribution $\mathcal{N}(0, 1)$ and generate images from our disentanglement model trained on dSprites. As shown in Figure 5, our model, generating heart, has a high reconstruction fidelity

A.5 CelebA

We further conduct experiments on a real dataset CelebA [15].

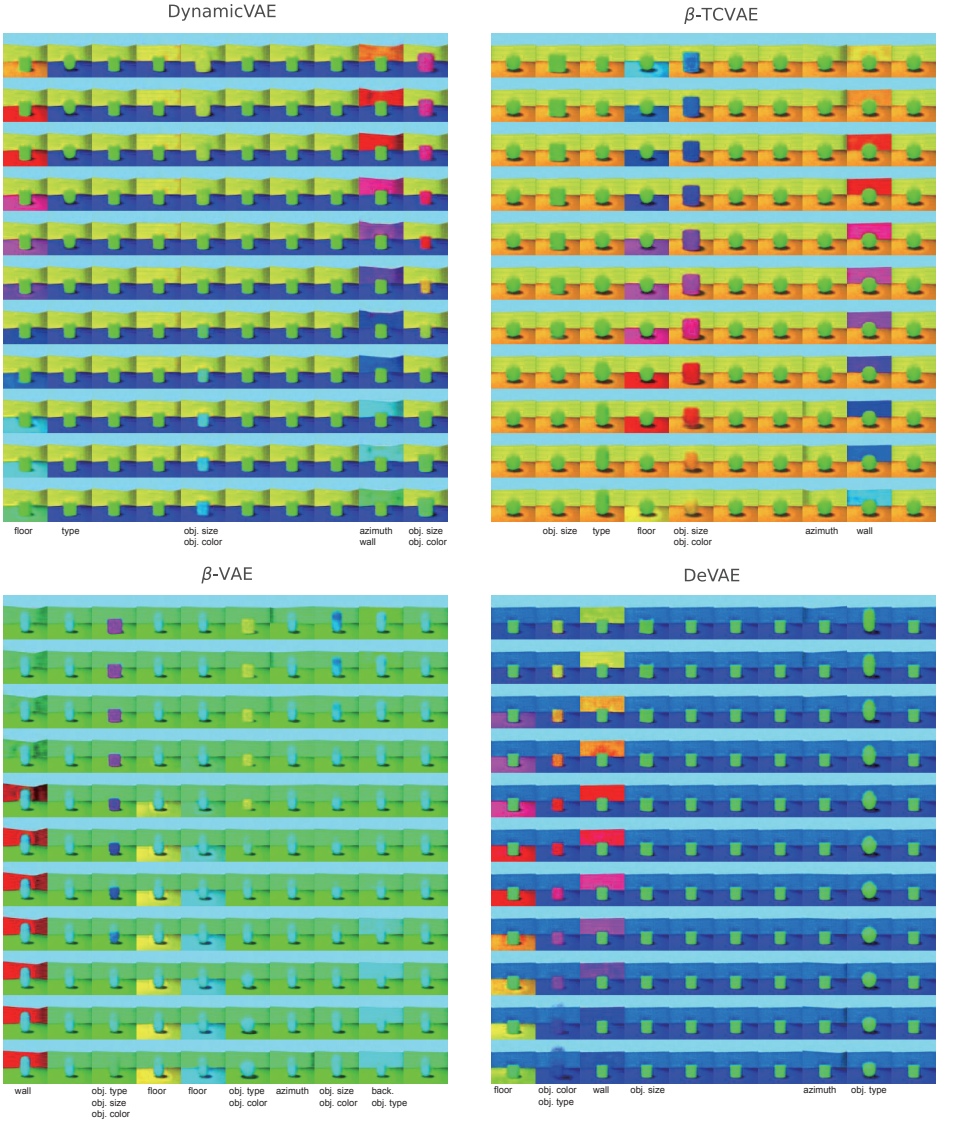

Figure 6: Latent traversal on Shapes3D. ”back.“ denotes background color, “floor” denotes floor color, “obj.” denotes object, and “wall” denotes wall color.

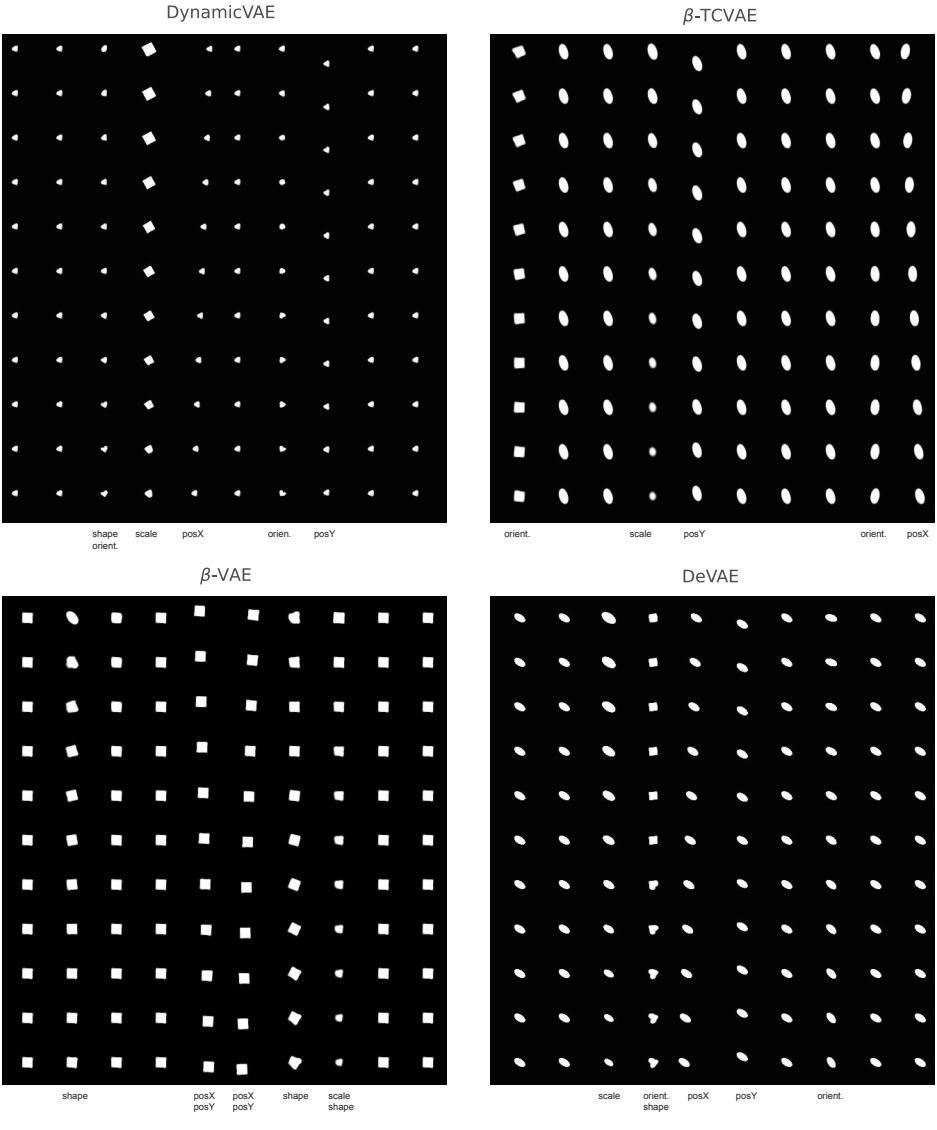

Figure 7: Latent traversal on dSprites.