A. Proof of Proposition 3.1

We offer the proof of Proposition 3.1 that the KL-divergence of ControlVAE is an upper bound of the mutual information (MI) between input data and the encoded latent factors, $\mathcal{I}(\mathbf{x},\mathbf{z})$. The detailed proof is presented below.

Proof. Let the distribution of the observed data $\mathbf{x}$ be $p(\mathbf{x})$ and the prior of latent variables be $p(\mathbf{z})$. The expectation of KL-divergence is given by

$$\begin{aligned}&\mathbb{E}_{p(\mathbf{x})}\left[D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\right]\\&=\mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\right]\right]\end{aligned}\tag{21}$$

Define the marginal distribution of the latent variable $\mathbf{z}$ from the encoder as

$$q_\phi(\mathbf{z})=\mathbb{E}_{p(\mathbf{x})}[q_\phi(\mathbf{z}\mid\mathbf{x})],$$

leading to

$$\begin{aligned}&\mathbb{E}_{p(\mathbf{x})}\left[D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\right]\\&=\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})}\left[\log\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\right]\\&=\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})}\left[\log\frac{q_\phi(\mathbf{z}|\mathbf{x})q_\phi(\mathbf{z})}{p(\mathbf{z})q_\phi(\mathbf{z})}\right]\\&=\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})}\left[\log\frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z})}\right]+\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})}\left[\log\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})}\right]\\&=\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})}\left[\log\frac{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{q_\phi(\mathbf{z})p(\mathbf{x})}\right]+\mathbb{E}_{q_\phi(\mathbf{z})}\left[\log\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})}\right]\\&=\mathcal{I}(\mathbf{x},\mathbf{z})+D_{KL}(q_\phi(\mathbf{z})||p(\mathbf{z}))\\&\geq\mathcal{I}(\mathbf{x},\mathbf{z}),\end{aligned}$$

completing the proof. □

B. Comparison of KL-Divergence for ControlVAE and DynamicVAE

Fig 7 shows the dimwise KL-divergence and total KL-divergence for ControlVAE and DynamicVAE. We can observe from Fig 7 (a) that ControlVAE suffers from overshoot problem, since there is a jump in the proposed step function annealing method. When KL-divergence is unstable, it will cause some latent factors to come out earlier (around steps $280K$) so they are entangled with each other. Conversely, our DynamicVAE in Fig 7 (b) can smoothly control the KL-divergence, mitigating the overshoot problem, hence disentangles different factors.

C. Proof of Stability in Theorem 5.1

In this section, we provide the omitted proof in the main paper about Theorem 5.1. For convenience purpose, we first restate Theorem 5.1 below.

Theorem 5.1. *Let $a>0$ and assume $g'(x)<0,\forall x>0$. Then DynamicVAE is stable at the equilibrium point C if and only if the parameters of the PI controller, K_i and K_p, satisfy the following conditions*

$$\begin{cases}K_p+K_i<-\dfrac{4(1+a)}{ag'(x_1^*)}\\0.5K_p^2ag'(x_1^*)^2+2[K_p-8K_i(1+a)]g'(x_1^*)-8(1+a)<0\\K_i>0,K_p>0\end{cases}\tag{20}$$

Proof. At a high level, the proof goes by showing the spectral norm of the Jacobian matrix A to be strictly less than 1 under the given condition, which is both sufficient and necessary for stability. To start with, recall that the Jacobian matrix A at equilibrium point x^* is defined by

$$A=\begin{bmatrix}\frac{\partial f_1}{\partial x_1}&\frac{\partial f_1}{\partial x_2}&\frac{\partial f_1}{\partial x_3}\\\frac{\partial f_2}{\partial x_1}&\frac{\partial f_2}{\partial x_2}&\frac{\partial f_2}{\partial x_3}\\\frac{\partial f_3}{\partial x_1}&\frac{\partial f_3}{\partial x_2}&\frac{\partial f_3}{\partial x_3}\end{bmatrix}_{|x=x^*}=\begin{bmatrix}K_1&K_2&K_3\\K_4&K_5&0\\0&1&0\end{bmatrix},\tag{22}$$

where

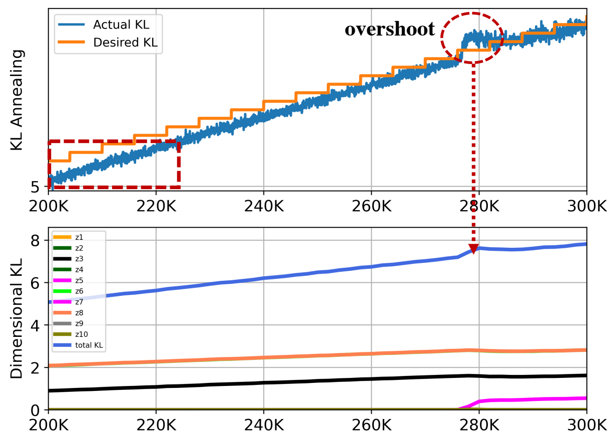
$$\begin{aligned}K_1&=\frac{\partial f_1}{\partial x_1}|_{x_1=x_1^*}=1\\K_2&=\frac{\partial f_1}{\partial x_2}|_{x_2=x_2^*}=K_i+K_p\sigma(x_2^*-C)[1-\sigma(x_2^*-C)]\\&=\frac{1}{4}K_p+K_i\\K_3&=\frac{\partial f_1}{\partial x_3}|_{x_3=x_3^*}=-K_p\sigma(x_3^*-C)[1-\sigma(x_3^*-C)]\\&=-\frac{1}{4}K_p\\K_4&=\frac{\partial f_2}{\partial x_1}|_{x_1=x_1^*}=\frac{a}{1+a}g'(x_1^*)\\K_5&=\frac{\partial f_2}{\partial x_2}|_{x_2=x_2^*}=\frac{1}{1+a}\\&\frac{\partial f_2}{\partial x_3}|_{x_3=x_3^*}=0\\&\frac{\partial f_3}{\partial x_1}|_{x_1=x_1^*}=0,\ \frac{\partial f_3}{\partial x_2}|_{x_2=x_2^*}=1,\ \frac{\partial f_3}{\partial x_3}|_{x_3=x_3^*}=0\end{aligned}\tag{23}$$

In order to guarantee the stability of our state space model, the modulus of eigenvalue λ of A should be smaller than 1, i.e., $|\lambda|<1$. By definition, the eigenvalues of A can be obtained by computing the roots of the following characteristic polynomial:

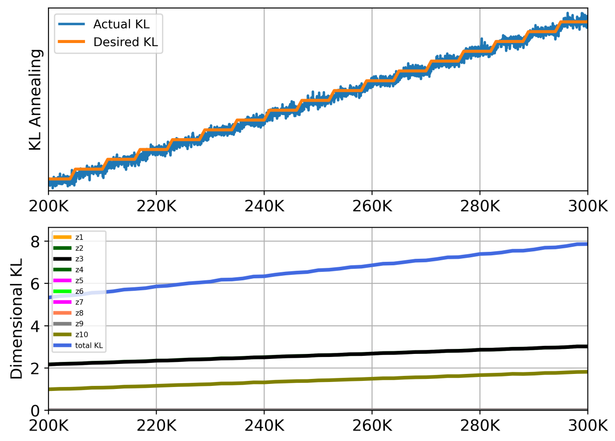
$$\begin{aligned}det(\lambda I-A)&=\begin{bmatrix}\lambda-K_1&-K_2&-K_3\\-K_4&\lambda-K_5&0\\0&-1&\lambda\end{bmatrix}\\&=\lambda^3-(K_1+K_5)\lambda^2+(K_1K_5-K_2K_4)\lambda-K_3K_4=0\end{aligned}\tag{24}$$

Instead of computing the (complex) roots of the above cubic polynomial analytically, we use the following bilinear transformation [16] to map the unit circle $|\lambda|<1$ to the left half plane such that its real root is less than 0 [14]:

$$\xi=\frac{\lambda-1}{\lambda+1}\iff\lambda=-\frac{\xi+1}{\xi-1}.\tag{25}$$



(a) ControlVAE.



(b) DynamicVAE

Figure 7. Comparison of ControlVAE and DynamicVAE.

Substituting λ in Eq.(24) with (25) , we have

$$b_3\xi^3 + b_2\xi^2 + b_1\xi + b_0 = 0, \tag{26}$$

where

$$\begin{cases} b_3 = K_1 + K_5 + K_1K_5 - K_2K_4 + K_3K_4 + 1 \\ b_2 = K_1 + K_5 - K_1K_5 + K_2K_4 - 3K_3K_4 + 3 \\ b_1 = -K_1 - K_5 - K_1K_5 + K_2K_4 + 3K_3K_4 + 3 \\ b_0 = -K_1 - K_5 + K_1K_5 - K_2K_4 - K_3K_4 + 1 \end{cases} \tag{27}$$

Clearly, using the above transformation, we know that $|\lambda| < 1$ iff the real part of λ is less than 0, i.e., $Re\{\xi\} < 0$. In order to ensure $Re\{\xi\} < 0$, based on Routh–Hurwitz stability criterion [47], b_0, b_1, b_2, b_3 should satisfy the following sufficient and necessary condition.

$$\begin{cases} b_0 > 0 \\ b_1 > 0 \\ b_2 > 0 \\ b_1b_2 > b_0b_3 \end{cases} \tag{28}$$

Substitute Eqs. (23), (26) and (28) into the above system of inequalities, yielding the following formula (29) (next page)

To complete the proof, recall that $a > 0$ and we assume $g'(x) < 0, \forall x$. Hence the coefficients of PI controller, K_p and K_i in Eq.(29), need to satisfy the following conditions.

$$\begin{cases} K_p + 2K_i > \dfrac{4(2+a)}{ag'(x_1^*)} \\ K_p + K_i < -\dfrac{4(1+a)}{ag'(x_1^*)} \\ 0.5K_p^2 ag'(x_1^*)^2 + 2[K_p - 8K_i(1+a)]g'(x_1^*) - 8(1+a) < 0 \\ K_i > 0 \end{cases} \tag{30}$$

Since $K_p > 0, K_i > 0$ in our designed PI control algorithm and $g'(x_1^*) < 0$, we can further simplify it as

$$\begin{cases} K_p + K_i < -\dfrac{4(1+a)}{ag'(x_1^*)} \\ 0.5K_p^2 ag'(x_1^*)^2 + 2[K_p - 8K_i(1+a)]g'(x_1^*) - 8(1+a) < 0 \\ K_i > 0, K_p > 0 \end{cases} \tag{31}$$

Therefore, as K_p and K_i meet the above conditions (31), our DynamicVAE would be stable at the set point, which is verified by the following experiments on different datasets. □

C.1. Verification on Benchmark Datasets

We first verify the validity of our assumption that $g'(x) < 0$ in Theorem 5.1. Fig. 8 illustrates the relationship between $\beta(t)$ and the actual KL when model training converges on dSprites and MNIST datasets. We can observe that the actual output KL-divergence and $\beta(t)$ have a highly negative correlation, which means $g'(x) < 0$.

Next, we are going to verify the stability of the proposed DynamicVAE on MNIST and dSprites datasets. On MNIST dataset, its mapping function $g(x)$ in Fig. 8 (a) can be approximately obtained by curve fitting with the following negative exponential function:

$$g(x(t)) = 26.38 \exp(-0.0476x(t)). \tag{32}$$

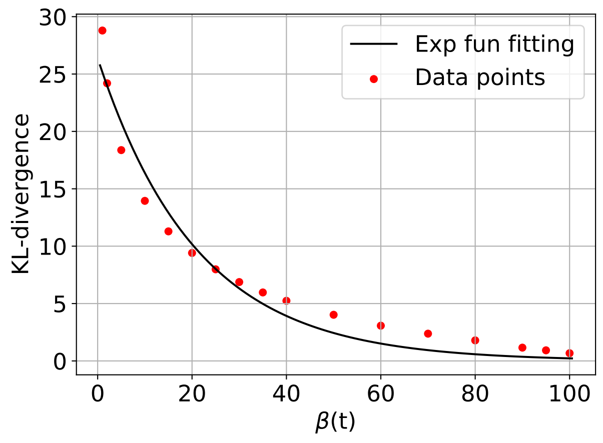
And the corresponding derivative is

$$g'(x(t)) = -1.26 \exp(-0.0476x(t)) \le -1.26. \tag{33}$$

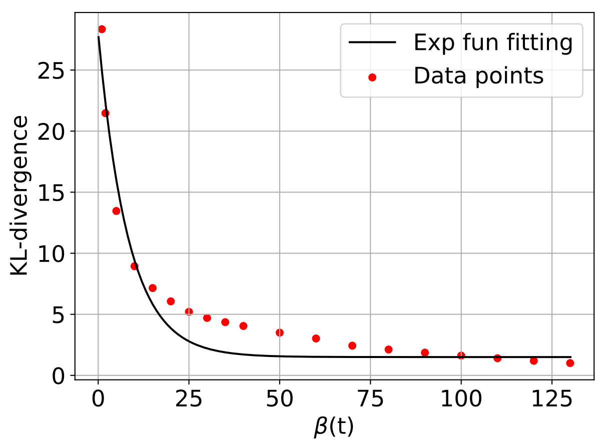
In addition, we introduce how to obtain the hyperparameter a in our dynamic model in Eq. (15). Assume that KL-divergence converges to a certain value C' in the open loop control system during model training, then the dynamic model in Eq. (15) can be rewritten as

$$y(t) - y(t-1) + ay(t) = aC'. \tag{34}$$

$$
\begin{cases}
b_3 = \dfrac{4a+8-(K_p+2K_i)ag'(x_1^*)}{2(1+a)} > 0 \\
b_2 = \dfrac{4(1+a)+(K_p+K_i)ag'(x_1^*)}{(1+a)} > 0 \\
b_1b_2 - b_3b_0 = \dfrac{-0.5K_p^2a^2g'(x_1^*)^2 - 2a[K_p - 8K_i(1+a)]g'(x_1^*) + 8a(1+a)}{(1+a)^2} > 0 \\
b_0 = \dfrac{-K_i a}{1+a} g'(x_1^*) > 0
\end{cases} \tag{29}
$$



(a) MNIST dataset



(b) dSprites dataset

Figure 8. (a) $g(x(t))$ on MNIST dataset. (b) $g(x(t))$ on dSprites dataset.

When $y(0) = 0$, and the sampling period of our system is $T_s = 1$, the corresponding solution is given by

$$y(t) = C'(1 - \exp(-at)). \tag{35}$$

In order to obtain the value of a, one commonly used method in control theory is to set $a = \frac{1}{t^*}$ as we have $y(t^*) = C'(1 - \exp(-1)) \approx 0.632C'$. In this way, we can derive a based on the training steps t^* as KL-divergence reaches 63.2% of its final value C' [15] in the experiments, as shown in Fig. 9. For MNIST dataset, we can get the hyperparameter $a = \frac{1}{5000}$ around based on the time response of KL-divergence in the open loop system, as shown in Fig. 9 (a).

Similarly, the derivative of mapping function on dSprites can be approximately expressed by

$$g'(x(t)) = -3.2\exp(-0.121x(t)) \leq -3.2. \tag{36}$$

In addition, we can get the hyperparameter $a = \frac{1}{2500}$ around based on the time response of KL-divergence in the open loop system, as shown in Fig. 9 (b).

We summarize the parameters a and $g'(x(t))$ for different datasets in the following Table 3.

In this paper, we choose $K_p = 0.01$ and $K_i = 0.005$ with the parameters in Table 3 to validate our model meets the conditions in Eq. (31). In addition, our experimental results in Section 6 further demonstrate that our method can stabilize the KL-divergence to the set points.

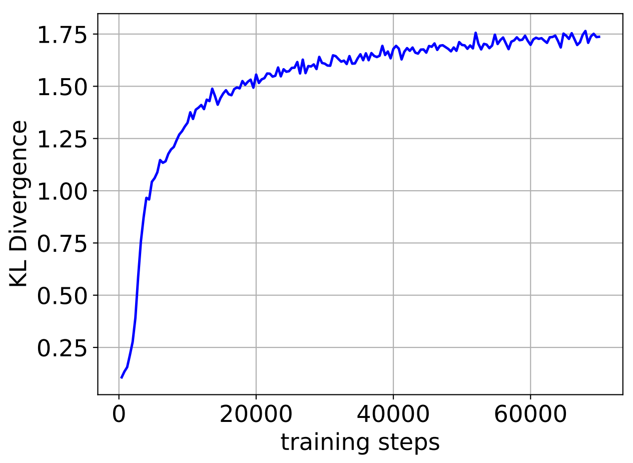
Table 3. Parameters summary for different datasets

Dataset	a	$g'(x(t))_{min}$
MNIST	$\frac{1}{5000}$	-1.26
dSprites	$\frac{1}{2500}$	-3.2

D. Model Configurations and Hyperparameter Settings

We summarize the detailed model configurations and hyperparameter settings for DynamicVAE below.

Following the same model architecture of β-VAE, we adopt a convolutional layer and deconvolutional layer for our experiments. We use Adam optimizer with $\beta_1 = 0.90$, $\beta_2 = 0.99$ and a learning rate tuned from 10^{-4}. We set K_p and K_i for PI algorithm to 0.01 and 0.005, respectively. The weight $\beta(t)$ for incremental PI controller is initialized with 150, 100 and 50 for dSprites, MNIST and 3D Chairs, respectively. The batch size is set to 128. Using the simi-


(a) MNIST when $\beta = 80$

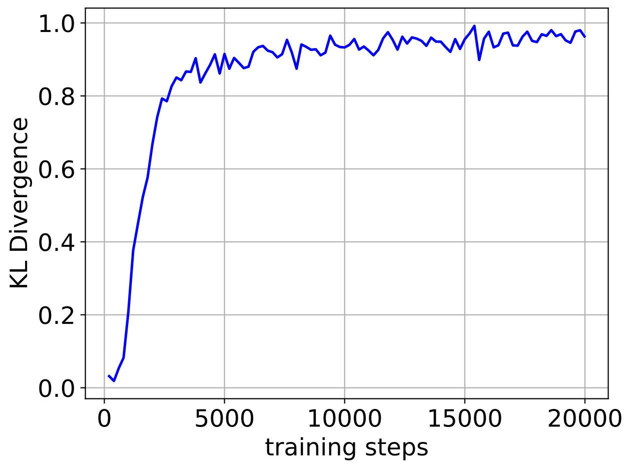

(b) dSprites when $\beta = 130$

Figure 9. Time response of KL-divergence under different β on MNIST and dSprites datasets respectively

lar methodology in [5], we train a single model by gradually increasing KL-divergence from 0.5 to a desired value C with a step function s and ramp function for every M training steps, as shown in Fig. 7(b). In the experiment, we set the step, s, to 0.15 per $M = 6,000$ training steps (including $5,000$ in step function and $1,000$ in ramp function) as the information capacity (desired KL- divergence) increases from 0.5 until to 20, 26 and 18 for dSprites, MNIST and 3D Chairs datasets respectively. In addition, the window size of moving average is $T = 5$ with equal weight α. Our model adopts the same encoder and decoder architecture as β-VAE$_H$ and ControlVAE except for plugging in PI control algorithm, as illustrated in Table 4 and Table 5.

Table 4. Encoder and decoder architecture for disentangled representation learning on dSprites and MNIST.

Encoder	Decoder
Input 64×64 binary image	Input $\in \mathbb{R}^{10}$
4×4 conv. 32 ReLU. stride 2	FC. 256 ReLU.
4×4 conv. 32 ReLU. stride 2	4×4 upconv. 256 ReLU. stride 2
4×4 conv. 64 ReLU. stride 2	4×4 upconv. 64 ReLU. stride 2.
4×4 conv. 64 ReLU. stride 2	4×4 upconv. 64 ReLU. stride 2
4×4 conv. 256 ReLU. stride 1	4×4 upconv. 32 ReLU. stride 2
FC 256. FC. 2×10	4×4 upconv. 32 ReLU. stride 2

E. Extra Experiments on Other Datasets

Fig 10 shows an example of latent traversals for the proposed DynamicVAE on smallNORB dataset. We can observe from it that it disentangles three different latent factors: lighting, elevation, and azimuth. Moreover, we present some samples of latent traversals for our method and the baselines on MNIST dataset. We can see DynamicVAE outperforms ControlVAE in term of rotation factor as illustrated in Fig. 11 and 12, though they have comparable disentanglement score. Moreover, DynamicVAE has a better disentanglement than β-VAE, LM, and FactorVAE in the following figures.

Table 5. Encoder and decoder architecture for disentangled representation learning on 3D Chairs.

Encoder	Decoder
Input $64 \times 64 \times 3$	Input $\in \mathbb{R}^{16}$
4×4 conv. 32 ReLU. stride 2	FC. 256 ReLU.
4×4 conv. 32 ReLU. stride 2	4×4 upconv. 256 ReLU. stride 2
4×4 conv. 64 ReLU. stride 2	4×4 upconv. 64 ReLU. stride 2.
4×4 conv. 64 ReLU. stride 2	4×4 upconv. 64 ReLU. stride 2
4×4 conv. 256 ReLU. stride 1	4×4 upconv. 32 ReLU. stride 2
FC 256. FC. 2×10	4×4 upconv. 32 ReLU. stride 2

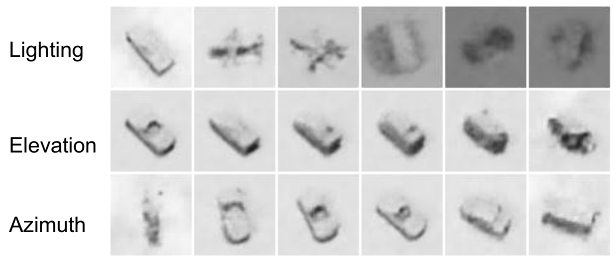
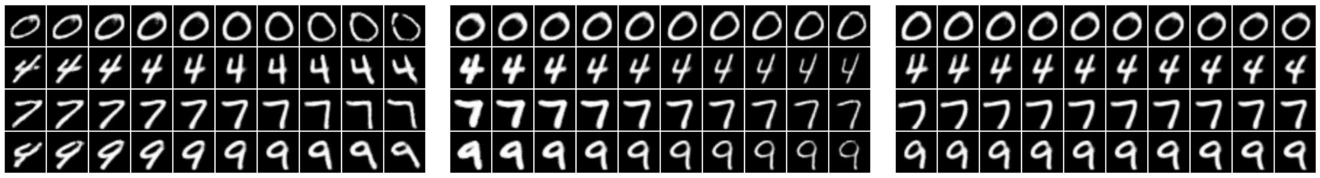
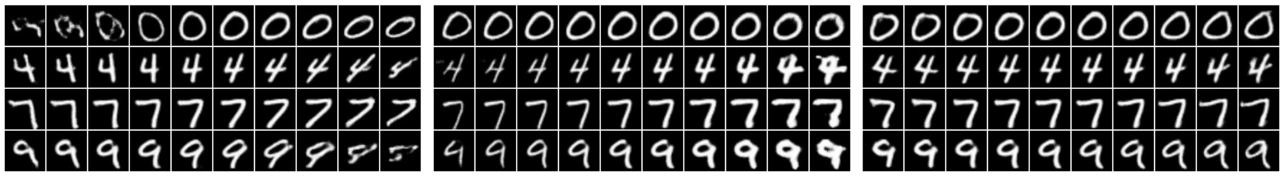


Figure 10. Latent traversals on smallNORB dataset for DynamicVAE.



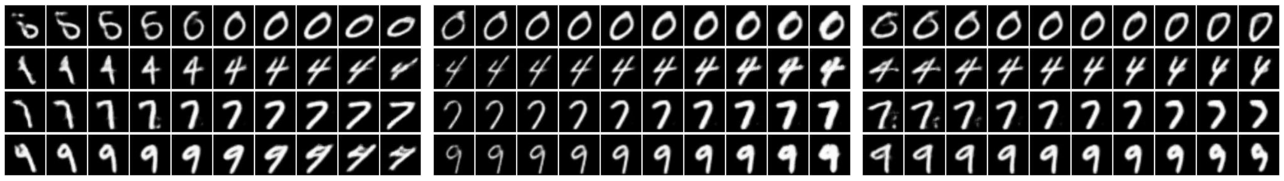
(a) Rotation (b) Thickness (c) Style from left to right

Figure 11. Latent traversals on MNIST for DynamicVAE. We can see our method can disentangle four different factors: rotation, thickness, size(width) and writing style.



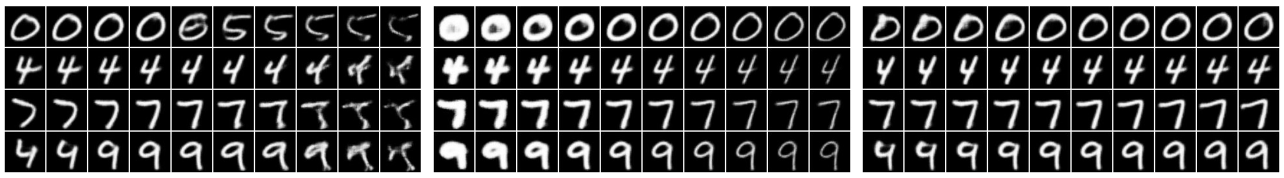
(a) Rotation (b) Thickness (c) Style from left to right

Figure 12. Latent traversals on MNIST for ControlVAE.



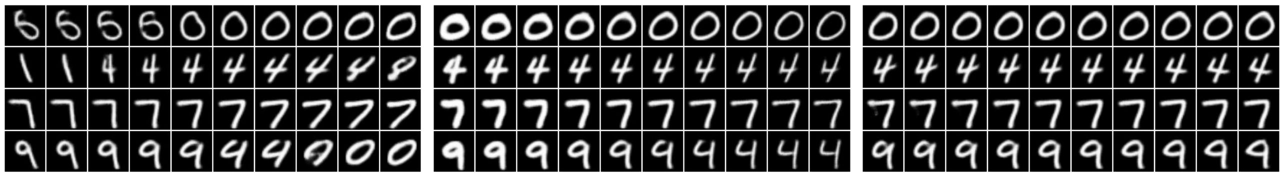
(a) Rotation (b) Thickness (c) Style from left to right

Figure 13. Latent traversals on MNIST for LM.



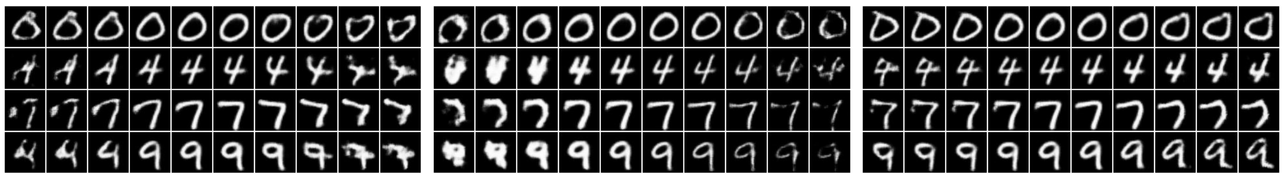
(a) Rotation (b) Thickness (c) Style from left to right

Figure 14. Latent traversals on MNIST for FactorVAE.



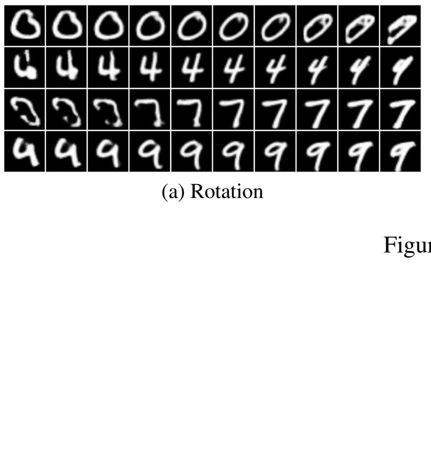
(a) Rotation (b) Thickness (c) Style from left to right

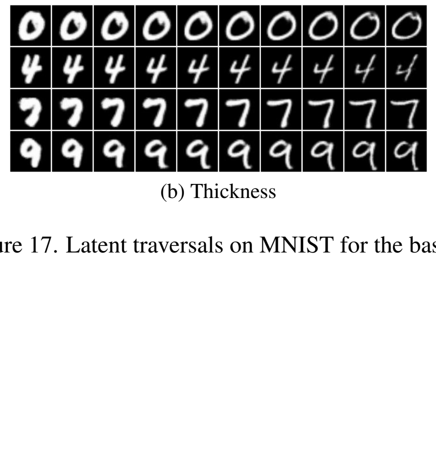
Figure 15. Latent traversals on MNIST for β-VAE$_H$ ($\beta = 10$).



(a) Rotation (b) Thickness (c) Style from left to right

Figure 16. Latent traversals on MNIST for β-VAE$_B$ ($\gamma = 100$).


(a) Rotation


(b) Thickness

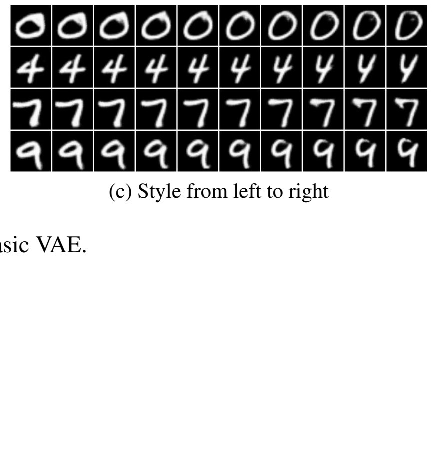

(c) Style from left to right

Figure 17. Latent traversals on MNIST for the basic VAE.