Challenging $\beta$-VAE with $\beta < 1$ for Disentanglement Via Dynamic Learning

Huajie Shao,1∗ Haohong Lin,2∗ Qinmin Yang,2 Shuochao Yao,1 Tarek Abdelzaher1

1Department of Computer Science at UIUC
2College of Control Science and Engineering, Zhejiang University
1{hshao5, syao9,zaher}@illinois.edu
2{lhh2017,qmyang}@zju.edu.cn

Abstract

This paper challenges the common assumption that the weight of $\beta$-VAE should be larger than 1 in order to effectively disentangle latent factors. We demonstrate that $\beta$-VAE with $\beta \leq 1$ can not only obtain good disentanglement but significantly improve the reconstruction accuracy via dynamic control. The goal of this paper is to deal with the trade-off problem between reconstruction accuracy and disentanglement with unsupervised learning. The existing methods, such as $\beta$-VAE and FactorVAE, assign a large weight in the objective, leading to high reconstruction errors in order to obtain better disentanglement. To overcome this problem, ControlVAE is recently developed to dynamically tune the weight to achieve the trade-off between disentangling and reconstruction using control theory. However, ControlVAE cannot fully decouple disentanglement learning and reconstruction, because it suffers from overshoot problem of the designed controller and does not timely respond to the target KL-divergence at the beginning of model training.

In this paper, we propose a novel DynamicVAE that leverages an incremental PI controller, a variant of proportional–integral–derivative controller (PID) controller, and moving average as well as hybrid annealing method to effectively decouple the reconstruction and disentanglement learning. We then theoretically prove the stability of the proposed approach. Evaluation results on benchmark datasets demonstrate that DynamicVAE significantly improves the reconstruction accuracy for the comparable disentanglement compared to the existing methods. More importantly, we discover that our method is able to separate disentanglement learning and reconstruction without introducing any conflict between them.

1 Introduction

This paper proposes a novel unsupervised disentangled representation learning, called DynamicVAE, that turns the weight of $\beta$-VAE ($\beta > 1$) [5, 14] into a small value ($\beta \leq 1$) to achieve good disentanglement via dynamic control. The proposed method can significantly improve the reconstruction quality for the comparable disentanglement compared to the competitive models, such as FactorVAE. More importantly, we discover that DynamicVAE is able to separate disentanglement learning and reconstruction optimization without hurting the performance of each other. The goal of disentangled representation learning is to encode the sensory data into a low-dimensional vector that contains the information about the salient factors of variation, in which each dimension of the representation corresponds to a distinct factor in the dataset [3, 31, 42]. Many recent works figure out that a disentangled representation is helpful to a variety of downstream tasks [16, 25, 30, 28, 9, 32], such as abstract visual reasoning [42], zero-shot transfer learning [5, 25, 14] and image generation [34].

∗The first two authors contribute equally.

Preprint. Under review.
In recent years, there is large body of work on the learning of disentangled representations, including unsupervised learning [8, 7, 5, 21, 9, 24, 11] and supervised learning [6, 30, 39, 4, 34, 43]. The latter, however, requires supervised knowledge of data generative factors, which are difficult for initial learners to discover from complex data in most real world scenarios. Hence, we are interested in the unsupervised learning in this paper. One major challenge of unsupervised disentanglement learning is that there exists a trade-off between reconstruction quality and disentanglement. Take $\beta$-VAE and its variants [5, 7, 14] as an example. They assign a large and fixed weight in the objective to improve the disentanglement at the cost of reconstruction quality. Actually, some recent works [42, 31] demonstrate that reconstruction quality is highly correlated to the down-stream accuracy. To improve the reconstruction quality, researchers recently propose a dynamic learning approach, ControlVAE [38], to dynamically adjust the weight on KL term to balance disentanglement learning and reconstruction. However, one problem of ControlVAE is that the proposed step function annealing method may lead to overshoot at some points, causing latent factors to come out earlier such that they are entangled with each other, as illustrated in Fig. 1(a). Besides, it adopts a positional PI control algorithm to manipulate the KL-divergence, which cannot quickly respond to the desired (small) KL-divergence at the beginning of model training as shown in dashed red rectangle in Fig. 1(a). As a result, it cannot fully decouple reconstruction and disentanglement learning as the weight $\beta$ drops to a small value.

Contributions In this work, we attempt to reduce the weight $\beta$ in $\beta$-VAE to a mall value (less than 1) with the hope that it can improve reconstruction accuracy meanwhile obtaining good disentanglement. The main contributions of this paper are summarized as follows.

- We propose a new model, DynamicVAE, that leverages incremental PI controller and moving average to dynamically tune the weight on KL term to stabilize the KL-divergence to a specified value for disentanglement. Different from ControlVAE, our method is able to quickly respond to the desired KL-divergence at the beginning of model training, as shown in Fig. 1(b).
- We leverage a hybrid annealing method that combines step function with ramp function to alleviate the overshoot problem of designed PI controller, preventing multiple latent factors coming out together to be entangled with each other.
- We theoretically prove the stability of the proposed DynamicVAE and further verify it by doing experiments on benchmark datasets.
- Evaluation results demonstrate that our approach turns the weight of $\beta$-VAE ($\beta > 1$) to $\beta \leq 1$, achieving higher reconstruction quality yet comparable disentanglement compared to FactorVAE.
- We also discover that the proposed method decouples disentanglement learning and reconstruction without hurting the performance of each other.
2 Background

2.1 Variational Autoencoder (VAE)

As one of the most popular generative models, VAE [22, 36] assumes latent variable \( z \) with prior \( p(z) \), and a conditional distribution \( p_{\theta}(x|z) \), to model the observed data \( x \). The generative model, denoted by \( p_{\theta}(x) \), can be expressed as \( p_{\theta}(x) = \int p_{\theta}(x|z)p(z)dz \). However, due to intractable posterior inference, the model is trained by optimizing the following variational lower bound (ELBO):

\[
\log p_{\theta}(x) \geq \mathbb{E}_{q_{\phi}(x|z)}[\log p_{\theta}(x|z)] - D_{KL}(q_{\phi}(z|x)||p(z)) \triangleq \mathcal{L}_{vae},
\]

where \( p_{\theta}(x|z) \) is a probabilistic decoder parameterized by a neural network to generate data \( x \) given the latent variable \( z \), and the posterior distribution of latent variable \( z \) given data \( x \) is approximated by the variational posterior, \( q_{\phi}(z|x) \), which is parameterized by an encoder network.

2.2 \( \beta \)-VAE and its Variants

\( \beta \)-VAE [15, 7] is a popular unsupervised method for learning a disentangled representation of the data generative factors [3]. Compared to the original VAE, \( \beta \)-VAE adds an extra hyperparameter \( \beta (\beta > 1) \) as the weight of KL term in VAE objective:

\[
\mathcal{L}_{\beta} = \mathbb{E}_{q_{\phi}(x|z)}[\log p_{\theta}(x|z)] - \beta D_{KL}(q_{\phi}(z|x)||p(z)).
\]

In order to discover more disentangled factors, researchers further put a constraint on total information capacity, \( C \), to control the capacity of the latent channels [5] to transmit information. The constraint can be formulated as an optimization method:

\[
\mathcal{L}_{\beta} = \mathbb{E}_{q_{\phi}(x|z)}[\log p_{\theta}(x|z)] - \beta D_{KL}(q_{\phi}(z|x)||p(z)) - C,
\]

where \( \beta \) is a large and fixed hyperparameter (e.g., 100).

However, \( \beta \)-VAE sacrifices the reconstruction quality to obtain better disentanglement. When the weight \( \beta \) is large, the optimization algorithm tends to optimize the second term in (3), leading to much higher reconstruction error.

2.3 PID Control Algorithm

PID (proportional–integral–derivative) is a simple and effective dynamic control method that can stabilize the system output to a desired value via feedback control [41, 38, 1, 13]. PID algorithm calculates an error, \( e(t) \), between a set point (in this case, the desired KL-divergence) and the current value of the controlled variable (in this case, the actual KL-divergence), then apply a correction in a direction that reduces that error. The correction is applied to some intermediate directly accessible variable (in our case, \( \beta(t) \)) that influences the value of the variable we ultimately want to control (KL-divergence). In many real world applications, such as deep learning, the discrete PID controller is often used below.

\[
\beta(t) = K_p e(t) + K_i \sum_{i=0}^{t} e(i) + K_d e(t) - e(t-1),
\]

where \( \beta(t) \) is the output of the controller; \( e(t) \) is the error between the output value and the desired value at time \( t \); \( K_p, K_i \) and \( K_d \) denote the coefficients for the P term, I term and D term, respectively.

We can observe that the positional PID controller gradually accumulates the past errors to tune \( \beta(t) \), which may not quickly respond to the small desired value of KL-divergence, as shown in Fig. 1. To deal with this issue, we adopt the following incremental PID controller

\[
\beta(t) = \Delta \beta(t) + \beta(t-1),
\]

where

\[
\Delta \beta(t) = K_p [e(t) - e(t-1)] + K_i e(t) + K_d [e(t) - 2e(t-1) + e(t-2)].
\]

We can see that incremental PID can be initialized with a large value. Note that, since derivatives essentially compute the slope of a signal, when the signal is noisy, the slope often responds more to variations induced by noise. Hence, following established best practices in control of noisy systems, we do not use the derivative (D) term in our specific controller.
3 The DynamicVAE Algorithm

The goal of DynamicVAE is to maximize the log likelihood meanwhile stabilizing the KL-divergence to a target value, $C$. It can be formulated as the following constraint optimization problem.

$$\max_{\phi, \theta} \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$$

s.t. $D_{KL}(q_\phi(z|x)||p(z)) - C = 0$

($7$)

$\beta$-VAE [5] formulates it as an optimization problem using Lagrange multiplier, and then assigns a large and fixed hyperparameter $\beta$ to constrain KL-divergence in Eq. (3), which, however, results in high reconstruction error. In this paper, we try to design a controller from feedback control to dynamically adjust $\beta(t)$ in the following VAE objective to stabilize the KL-divergence to set point $C$.

$$\mathcal{L}_d = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta(t)D_{KL}(q_\phi(z|x)||p(z)).$$

($8$)

We hope to decouple the reconstruction and disentanglement learning (controlling KL-divergence) via dynamic control. To reach this goal, we need to address the following challenges:

i) $\beta(t)$ should dynamically change from a large value to small one. Specifically, at the beginning of training, $\beta(t)$ should be large enough to control the information bottleneck (KL-divergence) to disentangle latent factors. After that, $\beta(t)$ is required to gradually drop to a small value to optimize the reconstruction.

ii) $\beta(t)$ should not change too fast or oscillate too frequently. When $\beta(t)$ drops too fast or oscillates, it may cause KL-divergence to grow with a large value. Consequently, some latent factors may come out earlier so that they would be entangled with each other.

In this paper, we propose a series of methods and tricks to deal with these challenges below.

Incremental PI controller. In order to dynamically tune the weight $\beta(t)$ on the KL term, we design a non-linear PI controller, a variant of PID, based on the output KL-divergence during training, as illustrated in Fig. 2 (a). It samples the output KL-divergence, denoted by $y_{kl}(t)$, at training step $t$. Then we use the difference $e(t)$ between the sampled KL-divergence at time $t$ with the set point, $C$, as the feedback to PI controller to tune $\beta(t)$. The basic idea of designed controller is that when KL-divergence drops below the set point, the controller counteracts this change by reducing the weight $\beta(t)$ (to reduce penalty for KL-divergence in the objective (8)). The reduced weight, $\beta(t)$, allows KL-divergence to grow, thus approaching the set point again. Conversely, when KL-divergence rises above the set point, the controller increases $\beta(t)$ (up to a certain value), thereby increasing the penalty for KL-divergence and forcing it to drop. This effect is achieved by computing $\beta(t)$ using the following nonlinear PI control algorithm:

$$\beta(t) = K_p\sigma(e(t)) - K_i \sum_{j=0}^{t} e(j),$$

($9$)

where $e(t)$ is the difference between desired KL-divergence, $C$, and the actual output KL-divergence; $K_p$ and $K_i$ are positive coefficients for the P term and I term respectively. In addition,

$$\sigma(e(t)) = \frac{1}{1 + \exp(e(t))},$$

($10$)

is a sigmoid function that ranges between 0 and 1. Here the motivation of using sigmoid is to mitigate the overshoot problem of PI controller [37] in order to prevent multiple latent factors coming out together to be entangled.

As mentioned above, we need a large $\beta(t)$ in the beginning to control KL-divergence from a small value to a large target value so that the information can be transmitted through the latent channels per data sample. Accordingly, we propose an incremental PI controller that can be initialized with a large value as follows:

$$\beta(t) = \Delta \beta(t) + \beta(t-1),$$

($11$)

where

$$\Delta \beta(t) = K_p[\sigma(e(t)) - \sigma(e(t-1))] - K_i e(t).$$

($12$)

Different from ControlVAE, the designed incremental PI controller can be initialized with a large value $\beta(0)$ so that it can quickly respond to the desired (small) KL-divergence at the beginning of model training, as illustrated in Fig. 1 (b).
(b) KL varies with training steps

Algorithm 1 Incremental PI Control algorithm.

1: **Input:** desired KL $C$, coefficients $K_p$, $K_i$, $\beta_{\text{min}}$.
2: **Output:** weight $\beta(t)$ at training step $t$.
3: **Initialization:** $\beta(0) = 150$ (100), $y_{\text{kl}}(0) = 0$.
4: for $t = 1$ to $N$ do
5: Sample KL-divergence, $y_{\text{kl}}(t)$
6: $y(t) = \sum_{i=t-T}^{t} \alpha_i y_{\text{kl}}(i)$
7: $e(t) \leftarrow C - y(t)$
8: $dP(t) \leftarrow K_p[\sigma(e(t)) - \sigma(e(t - 1))]$
9: $dI(t) \leftarrow K_i e(t)$
10: if $\beta(t - 1) < \beta_{\text{min}}$ then
11: \hspace{1cm} $dI(t) \leftarrow 0$ // wind up
12: end if
13: $\beta(t) \leftarrow \beta(t) + \beta(t - 1)$
14: if $\beta(t) < \beta_{\text{min}}$ then
15: \hspace{1cm} $\beta(t) \leftarrow \beta_{\text{min}}$
16: end if
17: $\beta(t) \leftarrow \beta(t)$
18: Return $\beta(t)$
19: end for

**Moving average.** Since our model is trained with mini-batch data, it often produces noise to make $\beta(t)$ oscillate. This may lead to multiple latent factors coming out together to be entangled. To mitigate this issue, we adopt moving average to smooth the output KL-divergence as the feedback of PI controller below.

$$y(t) = \alpha_t y_{\text{kl}}(t) + \alpha_{t-1} y_{\text{kl}}(t-1) + \cdots + \alpha_{t-T} y_{\text{kl}}(t-T) = \sum_{i=t-T}^{t} \alpha_i y_{\text{kl}}(i), \quad (13)$$

where $\alpha_i$ denotes weight and $T$ denotes the window size of past training steps.

**Hybrid annealing.** One drawback of ControlVAE is that it leverages step (input) function as annealing method, which leads to overshoot problem. In other words, the actual KL-divergence significantly exceeds the desired value (set point) so that it leads to some latent factors coming out earlier than expected to be entangled. To address this problem, we develop a hybrid annealing method that combines step function with ramp function to smoothly increase the target KL-divergence. Please refer to the orange curve in Fig. 1 (b).

With these three methods working together, our DynamicVAE can effectively separate disentanglement learning and reconstruction optimization, turning the weight of $\beta$-VAE into a small value (less than 1) for disentanglement learning. We summarize the proposed incremental PI algorithm in Algorithm 1.

### 3.1 Stability Analysis of DynamicVAE

We theoretically analyze the stability of the proposed DynamicVAE. Our first step is to build the state space model for our control system. In this paper, the state variable at training step $t$ is defined as

$$x(t) = \beta(t). \quad (14)$$

Accordingly, the model of incremental PI controller can be written as:

$$x(t + 1) - x(t) = K_p[\sigma(e(t)) - \sigma(e(t - 1))] - K_i e(t), \quad (15)$$

where error $e(t)$, as shown in Fig. 2(a), is given by

$$e(t) = C - y(t - 1). \quad (16)$$
Here $y(t)$ is a dynamic model about the time response of the output KL divergence, $y_{kl}(t)$. Referring to [27], stochastic gradient descent (SGD) for optimizing objective function can be described by a first-order dynamic model, and also our experiment, as illustrated in Fig. 2(b), shows that $y(t)$ in the open loop system approximately meets a negative exponential function. We hence use the first-order dynamic model to describe it below.

$$\frac{dy}{dt} + ay = ag(x), \quad (17)$$

where $a$ is a positive hyperparameter to describe the dynamic property, and $g(x)$ a mapping function between the actual KL-divergence and $\beta(t)$. Since DynamicVAE is a discrete control system with sampling period $T_s = 1$, the above first-order dynamic model can be reformulated as

$$y(t) - y(t - 1) + ay(t) = ag(x(t)) \implies y(t) = \frac{1}{1 + a} y(t - 1) + \frac{a}{1 + a} g(x(t)). \quad (18)$$

Now let $x_1(t) = x(t), x_2(t) = y(t - 1), x_3(t) = y(t - 2)$, then Eqs. (15) and (18) can be rewritten as the following state space equations.

$$\begin{cases} x_1(t + 1) = x_1(t) - K_i[C - x_2(t)] + K_p[\sigma(C - x_2(t)) - \sigma(C - x_3(t))] \triangleq f_1(x_1(t), x_2(t), x_3(t)) \\ x_2(t + 1) = \frac{a}{1 + a} g(x_1(t)) + \frac{1}{1 + a} x_2(t) \triangleq f_2(x_1(t), x_2(t), x_3(t)) \\ x_3(t + 1) = x_2(t) \triangleq f_3(x_1(t), x_2(t), x_3(t)). \end{cases} \quad (19)$$

In order to analyze the stability of the above non-linear state space model, one commonly used method is to adopt “linearization” at an equilibrium point [18]. In this paper, we use the following equilibrium point:

$$x^* = (x_1^*, x_2^*, x_3^*) = (g^{-1}(C), C, C), \quad (20)$$

where $g^{-1}(.)$ denotes the inverse function and $x_2^* = x_3^*$.

Then we apply first-order Taylor expansion to the above Eq. (19), yielding

$$X(t + 1) = AX(t), \quad (21)$$

where

$$X(t) = [x_1(t) - x_1^*, x_2(t) - x_2^*, x_3(t) - x_3^*]^T, \quad (22)$$

and $A$ is the Jacobian matrix at equilibrium point $x^*$, as defined in Eq. (23) in Appendix B. After linearization, we can prove the stability of the proposed method as the modulus of eigenvalue $\lambda$ of $A$ is smaller than 1, as introduced in the following theorem.

**Theorem 1.** When the parameters of PI controller, $K_i$ and $K_p$, meet the conditions in Eq. (34), our DynamicVAE can be stable at the equilibrium point $C$.

Please see the proof in Appendix B.

We further verify the above theorem by doing experiments on multiple benchmark datasets in Appendix B.1.

### 3.2 PI Parameter Tuning and Set Point Guidelines

We can tune PI parameters by following the conditions about the stability of the proposed method in Eq.(34) in Appendix B. In order to reduce noise during model training, we also follow ControlVAE to set $K_p$ and $K_i$ to a small value, 0.01 and 0.005, respectively. In addition, $\beta(0)$ is initialized to a sufficiently large value in order to guarantee the KL-divergence is closed to zero at the beginning of model training. On the other hand, the choice of desired value of KL-divergence (set point) is very simple. Since our method achieves good disentanglement when $\beta = 1$, i.e. VAE. We hence can set its desired value to equal or lower than the KL-divergence of the basic VAE as it converges.

### 4 Experiments

We evaluate the performance of DynamicVAE on three benchmark datasets: dSprites [5], MNIST [8] and 3D Chairs [2]. The detailed model configurations and hyperparameter settings are presented in Appendix A. Source code will be publicly available upon publication.
4.1 Baseline Methods

We compare the performance of DynamicVAE with the following baselines.

- **ControlVAE** [38]: this method proposes a positional PI controller with step function annealing method to dynamically tune the weight on KL term to disentangle the generative factors.
- **β-VAE_H** [15]: it assigns a large weight on the KL term to disentangle latent factors.
- **β-VAE_B** [5]: it forces the value of KL-divergence to a target value with a constraint.
- **FactorVAE** [21]: it adds a total correlation term to the VAE objective to encourage disentanglement.
- **VAE** [22]: the original VAE model that is composed of reconstruction and KL term.

4.2 Evaluation on Dsprites Dataset

We first evaluate the performance of DynamicVAE on the learning of disentangled representations using dSprites data. Fig. 3 (a) and (b) illustrate the comparison of reconstruction error and the hyperparameter \(\beta(t)\) (using 5 random seeds) for different approaches. We can observe from Fig. 3 (a) that DynamicVAE (KL=20) has much lower reconstruction error (about 11.8) than \(\beta\)-VAE and FactorVAE, and comparable to the basic VAE and ControlVAE. This is because DynamicVAE dynamically adjusts the weight, \(\beta(t)\), to balance the disentanglement and reconstruction. Specifically, DynamicVAE automatically assigns a large \(\beta(t)\) at the beginning of training in order to obtain good disentanglement, and then its weight gradually drops to less than 1 at the end of optimization, as shown in Fig. 3 (b). In other words, DynamicVAE turns the weight of \(\beta\)-VAE into a small value (less than 1), which significantly improves the reconstruction quality. In contrast, \(\beta\)-VAE and FactorVAE have a large and fixed weight in the objective so that their optimization algorithms tend to optimize the KL-divergence term (total correlation term for FactorVAE), leading to higher reconstruction error. For ControlVAE, it can also dynamically tune \(\beta(t)\) to control the value of KL-divergence, but its disentanglement performance degrades with the increase of KL-divergence (i.e., decrease of weight) as illustrated in the Table 1 below. In addition, Fig. 3(c) illustrates an example of KL-divergence per factor in the latent code as training progresses and the total information capacity (KL-divergence) increases from 0.5 until to 20. We can see that DynamicVAE disentangles all the five data generative factors, starting from position \((x, y)\) to scale, followed by orientation and then shape.

![Figure 3](image)

(a) Reconstruction loss  (b) \(\beta(t)\)  (c) Disentangled factors

Figure 3: (a) shows the comparison of reconstruction error and \(\beta(t)\) using dSprites data over 5 random seeds. DynamicVAE (KL=20) has comparable reconstruction errors as the basic VAE. (b) shows that our DynamicVAE can turn the weight of \(\beta\)-VAE into a small value less than 1. (c) shows an example about the disentangled factors in the latent variable as the total KL-divergence increases from 0.5 to 20 for DynamicVAE.

Next, we use a robust disentanglement metric, robust mutual information gap (RMIG) [10], to evaluate the disentanglement of different methods. It can be seen from Table 1 that DynamicVAE has a comparable RMIG score to the FactorVAE, but it has much lower reconstruction error as illustrated in Fig. 3. Moreover, DynamicVAE has higher RMIG score but lower reconstruction error than \(\beta\)-VAE models. We also find that our method achieves better disentanglement than ControlVAE for the comparable reconstruction accuracy. Hence, we can conclude that DynamicVAE is able to improve the reconstruction quality yet obtain good disentanglement.

Since there does not exist a very accurate metric to fully measure disentanglement, we also show the qualitative results of different models in Fig. 4. It can be observed that DynamicVAE disentangles all the five generative factors on dSprites data. However, ControlVAE is not very effective to disentangle
Table 1: RMIG for different methods averaged over 5 random seeds. The higher is better. Dynamic-VAE has a comparable RMIG score as FactorVAE and higher score than the other methods.

| Models/Metric     | pos. x | pos. y | Shape  | Scale  | Orientation | RMIG       |
|-------------------|--------|--------|--------|--------|-------------|------------|
| DynamicVAE (KL=20)| 0.7166 | 0.7179 | 0.2004 | 0.6530 | 0.1024      | **0.4781 ± 0.0172** |
| ControlVAE (KL=20)| 0.6802 | 0.6597 | 0.0956 | 0.6040 | 0.1081      | 0.4295 ± 0.0865 |
| FactorVAE (γ = 10)| 0.7482 | 0.7276 | 0.1383 | 0.6262 | 0.1412      | 0.4763 ± 0.0513 |
| β-VAE_B (γ = 100)| 0.5666 | 0.5763 | 0.4353 | 0.3814 | 0.0631      | 0.4045 ± 0.0345 |
| β-VAE_H (β = 4)  | 0.1635 | 0.1047 | 0.1391 | 0.3958 | 0.0127      | 0.1632 ± 0.0626 |
| VAE               | 0.0359 | 0.0243 | 0.0116 | 0.1507 | 0.0039      | 0.0452 ± 0.0326 |

Figure 4: Rows: latent traversals ordered by the value of KL-divergence with the prior in a descending order. We initialize the latent representation from a seed image, and then traverse a single latent code in a range of $[-3, 3]$, while keeping the remaining latent code fixed. We can observe that DynamicVAE ($\beta < 1$) can disentangle all the five latent factors for dSprites data.

all the factors when its KL-divergence is set to a large value, such as 20. What is more, β-VAE_B ($\gamma = 100$) disentangles four generative factors except for entangling the scale and shape together (in the third row) while the other methods not perform well for disentanglement.

4.3 Evaluation on MNIST and 3D Chair Datasets

We also evaluate the proposed method on the other two datasets: MNIST and 3D Chairs. Fig. 5 illustrates some samples of disentangled factors for DynamicVAE on MNIST. It can been seen that our method disentangles more latent factors compared with the other methods in Appendix C. In addition, our method with $\beta < 1$ can significantly improve the reconstruction accuracy than $\beta$-VAE as illustrated in Fig. 10 in Appendix C.

We also demonstrate that DynamicVAE can learn many different data generative factors on another challenging dataset, 3D Chairs. We can observe from Fig. 6 that our method disentangles six different latent factors, such as wheels, and leg height and azimuth, same as FactorVAE in [21].

4.4 Decoupled Reconstruction and Disentanglement

Additionally, we show that the proposed DynamicVAE is able to decouple the reconstruction and disentanglement learning into two phases, overcoming the problem of balancing the trade-off between reconstruction and disentanglement. Fig. 7 illustrates the RMIG score and reconstruction loss with the increase of training steps after all the factors are disentangled (before 800,000). It can be seen that RMIG score of our method remains stable as the reconstruction loss drops. Therefore, the proposed method does not introduce any conflict between reconstruction and disentanglement learning.
Figure 5: Latent traversals on MNIST for DynamicVAE. It can be seen that our method can disentangle four different factors: rotation, thickness, size (width) and writing style.

Figure 6: Sample traversals for the six latent factors in our model on 3D Chairs.

Figure 7: Averaged RMIG score and reconstruction loss vary with training steps.

4.5 Ablation Studies

Finally, we perform ablation studies to compare the performance of DynamicVAE and its variants below:

- DynamicVAE-P: it uses positional PI controller instead of incremental PI to tune the weight on KL term in the VAE objective.
- DynamicVAE-step: it solely adopts step function without ramp function for our annealing method.
- DynamicVAE-t: this model directly uses the output KL-divergence at time \( t \) as a feedback of PI controller without using moving average to smooth it.

Table 2 shows the comparison of RMIG score for DynamicVAE and its variants. It can be observed that DynamicVAE outperforms the other methods in terms of overall RMIG score. We also find that DynamicVAE-step does not perform well because the ramp function is removed from our annealing method, leading to overshoot of PI controller. As a result, it makes the other factors come out earlier and entangled to each other. Thus, we can conclude the importance of adding ramp function for our annealing method. In addition, we can see that the proposed moving average and incremental PI control algorithm also play a critical role to improve the disentanglement.

5 Related Work

Disentangled representation learning can be divided into two main categories: unsupervised learning and supervised learning.
Supervised disentanglement learning requires the prior knowledge of some data generative factors from human annotation to train the model. The earliest works [33, 40, 23, 35] try to disentangle some observed factors of variations from the other entangled latent variables. In recent years, some works [28, 29] figure out that it is hard to achieve reliable and good disentanglement without supervision. For instance, researchers [30] discover that a simple supervised learning with few labels outperforms unsupervised learning method, because the limited labeling information can help ensure a latent space of the VAE with desirable structure w.r.t to the ground-truth latent factors. More recently, there are some studies on weakly supervised learning [4, 17, 31] in order to reduce human annotations. Nevertheless, they require explicit human labeling or assume the change of the two observations is small. In practice, it is unrealistic for initial learners to discover the data generative factors in most real world scenarios. Therefore, we are more interested in unsupervised learning of disentangled representations in this paper.

For unsupervised learning methods, the recent approaches mainly build on Variational Autoencoders (VAEs) [22] and Generative Adversarial Networks (GAN) [12]. InfoGAN [26] is the first scalable unsupervised learning method for disentangled representations. It, however, suffers from training instabilities and does not perform well in disentanglement learning [16], so most recent works are largely based on VAEs models. The VAE models, such as $\beta$-VAE ($\beta > 1$) [14, 5], FactorVAE and $\beta$-TCVAE [7] often suffer from a large reconstruction loss in order to obtain better disentanglement that, since they add a large weight to terms in the objective. To address this issue, Shao et al [38] develop a controllable variational autoencoder, ControlVAE, to dynamically tune the weight $\beta$ to achieve the trade-off between reconstruction quality and disentanglement. Though ControlVAE improves reconstruction accuracy, its disentanglement performance would be weakened with the increase of target KL-divergence (i.e., the decrease of weight $\beta$). In this paper, we propose a DynamicVAE that significantly improves the reconstruction quality yet obtains a good disentanglement.

## 6 Conclusion

In this paper, we proposed a novel dynamic learning method, DynamicVAE, to address the trade-off problem between reconstruction and disentanglement. Our method is able to turn the weight of $\beta$-VAE to a small value ($\beta \leq 1$) to achieve good disentanglement against the previous default consumption, $\beta > 1$. Specifically, we design an incremental PI controller with a hybrid annealing (combining step function and ramp function) for better stabilizing the KL-divergence to a specified value to disentangle latent factors. We also theoretically prove the stability of the proposed DynamicVAE and verify it on different benchmark datasets. The evaluation results demonstrate our method can significantly improve the reconstruction quality meanwhile obtaining good disentanglement. More importantly, it decouples disentanglement learning and reconstruction without introducing any conflict between them. For future work, we plan to apply our method to semi-supervised and weakly supervised learning of disentangled representations.
References

[1] Karl Johan Åström, Tore Hägglund, and Karl J Astrom. Advanced PID control, volume 461. ISA-The Instrumentation, Systems, and Automation Society Research Triangle . . . , 2006.

[2] Mathieu Aubry, Daniel Maturana, Alexei A Efros, Bryan C Russell, and Josef Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3762–3769, 2014.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence, 35(8): 1798–1828, 2013.

[4] Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[5] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in beta-vae. arXiv preprint arXiv:1804.03599, 2018.

[6] Junxiang Chen and Kayhan Batmanghelich. Weakly supervised disentanglement by pairwise similarities. arXiv preprint arXiv:1906.01044, 2019.

[7] Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In Advances in Neural Information Processing Systems, pages 2610–2620, 2018.

[8] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Advances in neural information processing systems, pages 2172–2180, 2016.

[9] Emily L Denton et al. Unsupervised learning of disentangled representations from video. In Advances in neural information processing systems, pages 4414–4423, 2017.

[10] Kien Do and Truyen Tran. Theory and evaluation metrics for learning disentangled representations. In Proceedings of ICLR, 2020.

[11] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In Advances in Neural Information Processing Systems, pages 3601–3610, 2017.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

[13] Joseph L Hellerstein, Yixin Diao, Sujay Parekh, and Dawn M Tilbury. Feedback control of computing systems. John Wiley & Sons, 2004.

[14] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. Proceedings of ICLR, 2017.

[15] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. ICLR, 2(5):6, 2017.

[16] Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. arXiv preprint arXiv:1812.02230, 2018.

[17] Haruo Hosoya. Group-based learning of disentangled representations with generalizability for novel contents. In IJCAI, pages 2506–2513, 2019.

[18] Jonathan L Hughes. Applications of stability analysis to nonlinear discrete dynamical systems modeling interactions. 2015.

[19] IS Isa, BCC Meng, Z Saad, and NA Fauzi. Comparative study of pid controlled modes on automatic water level measurement system. In 2011 IEEE 7th International Colloquium on Signal Processing and its Applications, pages 237–242. IEEE, 2011.

[20] Eliahu Ibrahim Jury. Theory and application of the z-transform method. 1964.
[21] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, pages 2654–2663, 2018.

[22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[23] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[24] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. In *International Conference on Learning Representations*, 2018.

[25] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

[26] Zinan Lin, Kiran K Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans. In *International conference on machine learning*, 2020.

[27] Guan-Horng Liu and Evangelos A Theodorou. Deep learning theory review: An optimal control and dynamical systems perspective. *arXiv preprint arXiv:1908.10920*, 2019.

[28] Francesco Locatello, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem. On the fairness of disentangled representations. In *Advances in Neural Information Processing Systems*, pages 14611–14624, 2019.

[29] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International conference on machine learning*, pages 4114–4124, 2019.

[30] Francesco Locatello, Michael Tschannen, Stefan Bauer, Gunnar Rätsch, Bernhard Schölkopf, and Olivier Bachem. Disentangling factors of variations using few labels. In *International Conference on Learning Representations*, 2019.

[31] Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. *arXiv preprint arXiv:2002.02886*, 2020.

[32] Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In *International Conference on Machine Learning*, pages 4402–4412, 2019.

[33] Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in neural information processing systems*, pages 5040–5048, 2016.

[34] Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *International Conference on Machine Learning*, pages 1431–1439, 2014.

[35] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[36] Homayoun Seraji. A new class of nonlinear pid controllers. *IFAC Proceedings Volumes*, 30(20):65–71, 1997.

[37] Huajie Shao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Dongxin Liu, Jun Wang, and Tarek Abdelzaher. Controlvae: Controllable variational autoencoder. *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

[38] Rui Shu, Yining Chen, Abhishek Kumar, Stefano Ermon, and Ben Poole. Weakly supervised disentanglement with guarantees. *arXiv preprint arXiv:1910.09772*, 2019.
[40] Narayanaswamy Siddharth, Brooks Paige, Jan-Willem Van de Meent, Alban Desmaison, Noah Goodman, Pushmeet Kohli, Frank Wood, and Philip Torr. Learning disentangled representations with semi-supervised deep generative models. In Advances in Neural Information Processing Systems, pages 5925–5935, 2017.

[41] Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. arXiv preprint arXiv:2007.03964, 2020.

[42] Sjoerd van Steenkiste, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem. Are disentangled representations helpful for abstract visual reasoning? In Advances in Neural Information Processing Systems, pages 14245–14258, 2019.

[43] Jimei Yang, Scott E Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In Advances in Neural Information Processing Systems, pages 1099–1107, 2015.

[44] Jerzy Zabczyk. Mathematical control theory. An Introduction, 1992.
A Model Configurations and Hyperparameter Settings

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

Following the same model architecture of ControlVAE [38], 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 similar 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. 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 $\alpha$. Our model adopts the same encoder and decoder architecture as $\beta$-VAE$_H$ and ControlVAE except for plugging in PI control algorithm, as illustrated in Table 3 and Table 4.

Table 3: 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 |

Table 4: Encoder and decoder architecture for disentangled representation learning on 3D Chairs.

| Encoder | Decoder |
|---------|---------|
| Input 64 × 64 × 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 |

B Proof of Stability in Theorem 1

The Jacobian matrix $A$ at equilibrium point $x^*$ is defined by

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

(23)
where

\[ K_1 = \frac{\partial f_1}{\partial x_1}|_{x_1 = x_1^*} = 1 \]  \hspace{1cm} (24a)

\[ K_2 = \frac{\partial f_1}{\partial x_2}|_{x_2 = x_2^*} = K_i + K_p \sigma(C - x_2^*)[1 - \sigma(C - x_2^*)] = \frac{1}{4} K_p + K_i \]  \hspace{1cm} (24b)

\[ K_3 = \frac{\partial f_1}{\partial x_3}|_{x_3 = x_3^*} = -K_p \sigma(C - x_3^*)[1 - \sigma(C - x_3^*)] = -\frac{1}{4} K_p \]  \hspace{1cm} (24c)

\[ K_4 = \frac{\partial f_2}{\partial x_1}|_{x_1 = x_1^*} = \frac{a}{1 + a} g'(x_1^*) \]  \hspace{1cm} (24d)

\[ K_5 = \frac{\partial f_2}{\partial x_2}|_{x_2 = x_2^*} = \frac{1}{1 + a} \]  \hspace{1cm} (24e)

\[ \frac{\partial f_2}{\partial x_3}|_{x_3 = x_3^*} = 0 \]  \hspace{1cm} (24f)

\[ \frac{\partial f_3}{\partial x_1}|_{x_1 = x_1^*} = 0, \quad \frac{\partial f_3}{\partial x_2}|_{x_2 = x_2^*} = 1, \quad \frac{\partial f_3}{\partial x_3}|_{x_3 = x_3^*} = 0. \]  \hspace{1cm} (24g)

In order to guarantee the stability of our state space model, the modulus of eigenvalue \( \lambda \) of \( A \) should be smaller than 1, i.e., \( |\lambda| < 1 \). The eigenvalue of \( A \) can be obtained by the following formula.

\[ \text{det}(\lambda I - A) = \begin{vmatrix} \lambda - K_1 & -K_2 & -K_3 \\ -K_4 & \lambda - K_5 & 0 \\ 0 & -1 & \lambda \end{vmatrix} = \lambda^3 - (K_1 + K_5)\lambda^2 + (K_1 K_5 - K_2 K_4)\lambda - K_3 K_4 = 0 \]  \hspace{1cm} (25)

Since it is very hard to obtain the real root and complex root of the above equation directly, we utilize the following bilinear transformation [20] to map the unit circle \(|\lambda| < 1\) into the left half plane such that its real root is less than 0 [18].

\[ \xi = \frac{\lambda - 1}{\lambda + 1} \rightarrow \lambda = -\xi + 1. \]  \hspace{1cm} (26)

Substituting \( \lambda \) in Eq.(25) with (26), we have

\[ b_3 \xi^3 + b_2 \xi^2 + b_1 \xi + b_0 = 0, \]  \hspace{1cm} (27)

where

\[
\begin{cases}
    b_3 = K_1 + K_5 + K_1 K_5 - K_2 K_4 + K_3 K_4 + 1 \\
    b_2 = K_1 + K_5 - K_1 K_5 + K_2 K_4 - 3 K_3 K_4 + 3 \\
    b_1 = -K_1 - K_5 - K_1 K_5 + K_2 K_4 + 3 K_3 K_4 + 3 \\
    b_0 = -K_1 - K_5 + K_1 K_5 - K_2 K_4 - 3 K_3 K_4 + 1
\end{cases}
\]  \hspace{1cm} (28)

Referring to [18], the condition \(|\lambda| < 1\) is equivalent to the real root of the Eq. (27) should be less than 0, i.e., \( \text{Re}\{\xi\} < 0 \). In order to make \( \text{Re}\{\xi\} < 0 \), based on Routh–Hurwitz stability criterion [44], \( b_0, b_1, b_2, b_3 \) should satisfy the following sufficient and necessary condition.

\[
\begin{cases}
    b_0 > 0 \\
    b_1 > 0 \\
    b_2 > 0 \\
    b_1 b_2 > b_0 b_3
\end{cases}
\]  \hspace{1cm} (29)

Based on Eqs. (24), (27) and (29), we can have

\[
\begin{cases}
    b_3 = \frac{4a + 8 - (K_p + 2K_i)ag'(x_1^*)}{2(1 + a)} > 0 \\
    b_2 = \frac{4(1 + a) + (K_p + K_i)ag'(x_1^*)}{(1 + a)} > 0 \\
    b_1 b_2 - b_3 b_0 = \frac{-0.5K_p^2a^2g'(x_1^*)^2 - 2a[K_p - 8K_i(1 + a)]g'(x_1^*) + 8a(1 + a)}{(1 + a)^2} > 0 \\
    b_0 = \frac{-K_i a g'(x_1^*)}{1 + a} > 0
\end{cases}
\]  \hspace{1cm} (30)
In order to derive the range of $K_p$ and $K_i$ to stabilize our system, we first take a look at the property of $g(x(t))$, which is a mapping function between $\beta(t)$ and the actual output KL-divergence. Fig. 8 illustrate the relationship between $\beta(t)$ and the actual KL when model training converges on dSprites and MNIST datasets. We can observe that the actual KL-divergence and $\beta(t)$ have a highly negative correlation. Thus, the derivative of mapping function $g(x(t))$ is negative, $g'(x(t)) < 0$.

Next, we discuss the hyperparameter $a$ in our dynamic model in Eq. (18). 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. (18) can be rewritten as

$$y(t) - y(t-1) + ay(t) = aC'.$$

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

$$y(t) = C'(1 - \exp(-at)).$$

In order to get $a$, one commonly used method in control theory is to set $a = \frac{1}{T}$ so that we have $y(t) = C'(1 - \exp(-1)) \approx 0.632C'$. By doing this, we can derive $a$ based on the training steps $t$ as KL-divergence reaches 63.2% of its final value $C'$ [19] in our experiments. In general, it takes many training steps to reach that goal, so $a$ is a small and positive value.

Based on the above discussion about $a$ and $g'(x(t))$, the coefficients of PI controller, $K_p$ and $K_i$ in Eq.(30), need to satisfy the following conditions.

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

(33)

Since $K_p > 0$ in our designed PI control algorithm, we can further simplify it as

$$
\begin{align*}
K_p + K_i &< -\frac{4(1 + a)}{ag'(x_1^*)} \\
&- 0.5K_p^2ag'(x_1^*)^2 - 2[K_p - 8K_i(1 + a)]g'(x_1^*) + 8(1 + a) > 0. \\
K_i &> 0 \\
K_p &> 0
\end{align*}
$$

(34)

Therefore, as $K_p$ and $K_i$ meet the above conditions (34), our DynamicVAE would be stable at the set point, which is verified by the following experiments on different datasets.
Figure 9: Time response of KL-divergence under different $\beta$ on MNIST and dSprites datasets respectively

### B.1 Verification on Benchmark Datasets

We are going to verify the stability of our method 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:

$$g(x(t)) = 26.38\exp(-0.0476x(t)). \tag{35}$$

And the corresponding derivative is

$$g'(x(t)) = -1.26\exp(-0.0476x(t)) \leq -1.26. \tag{36}$$

Besides, we can also 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

$$g'(x(t)) = -3.2\exp(-0.121x(t)) \leq -3.2. \tag{37}$$

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 5.

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

In this paper, we choose $K_p = 0.01$ and $K_i = 0.005$ with the parameters in Table 5 to validate our model meets the conditions in Eq. (34). In addition, our experimental results in Section 4 demonstrate that our method can stabilize the KL-divergence to the set points.

### C Extra Experiments on MNIST

#### C.1 Evaluation on Reconstruction Quality

Fig. 10 shows the comparison of reconstruction loss and weight $\beta(t)$ for different methods. It can be observed that DynamicVAE and ControlVAE have comparable reconstruction accuracy to the basic VAE, but they have better disentanglement than it, as shown in Fig. 5 and 15. In addition, we can see that DynamicVAE has better reconstruction quality than the two $\beta$-VAE models.
C.2 MNIST Latent Traversals for Baselines

We present some samples of latent traversals for the baseline methods. We find that DynamicVAE outperforms ControlVAE in term of rotation factor as illustrated in Fig. 5 and 11, though they have comparable disentanglement score. Moreover, DynamicVAE has a better disentanglement than $\beta$-VAE and FactorVAE in the following figures.

Figure 11: Latent traversals on MNIST for ControlVAE.
Figure 12: Latent traversals on MNIST for FactorVAE.

Figure 13: Latent traversals on MNIST for $\beta$-VAE$_H$ ($\beta = 10$).

Figure 14: Latent traversals on MNIST for $\beta$-VAE$_B$ ($\gamma = 100$).
Figure 15: Latent traversals on MNIST for the basic VAE.