Sequential Representation Learning via Static-Dynamic Conditional Disentanglement

Mathieu Cyrille Simon[1(✉)], Pascal Frossard[2], and Christophe De Vleeschouwer[1]

[1] UCLouvain, ICTEAM, Louvain-la-Neuve, Belgium
{mathieu.simon,christophe.devleeschouwer}@uclouvain.be
[2] EPFL, LTS4 laboratory, Lausanne, Switzerland
pascal.frossard@epfl.ch

Abstract. This paper explores self-supervised disentangled representation learning within sequential data, focusing on separating time-indep- endent and time-varying factors in videos. We propose a new model that breaks the usual independence assumption between those factors by explicitly accounting for the causal relationship between the static/dynamic variables and that improves the model expressivity through additional Normalizing Flows. A formal definition of the factors is proposed. This formalism leads to the derivation of sufficient conditions for the ground truth factors to be identifiable, and to the introduction of a novel theoretically grounded disentanglement constraint that can be directly and efficiently incorporated into our new framework. The experiments show that the proposed approach outperforms previous complex state-of-the-art techniques in scenarios where the dynamics of a scene are influenced by its content.

Keywords: Sequential disentanglement · Representation learning

1 Introduction

Disentangled Representation Learning (DRL) focuses on embedding high dimensional complex data into a low dimensional space which factorizes the hidden underlying factors of variations. This disentanglement is expected to facilitate numerous downstream generation or classification tasks and enhance model explainability [5,14,31,34,51]. The present work in particular is concerned with unsupervised sequential data DRL [44,46,48] which, compared to disentanglement on static data [7,20,27,33,39,51,59], exhibits specific temporal structure that could be leveraged in the learning process. More precisely, the objective is to learn a representation of video data that factorizes the time independent factors (i.e., what is *static*; the identities, the backgrounds) and time varying

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-73226-3_7.

A. Leonardis et al. (Eds.): ECCV 2024, LNCS 15133, pp. 110–126, 2025.
https://doi.org/10.1007/978-3-031-73226-3_7

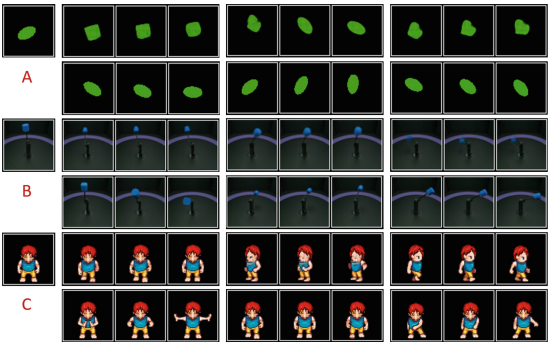

Fig. 1. Dynamic generation results for the *c-dSprites* (A), *MPI3D* (B) and *LPCSprites* (C) datasets. The sequences are generated by fixing the static code to the value given by the left image and sampling the dynamic variables from the prior. For each dataset, the first row corresponds to samples from the competing CDSVAE model [4] and the second row to samples from our proposed model.

Fig. 2. Static/dynamic swap results for the MUG dataset. Odd rows are test input sequences while even rows are sequences generated by swapping the static and dynamic codes of the test sequences using our model.

factors (i.e., what is *dynamic*; the poses, the motions) in two separate feature vectors. In Fig. 2, we show how this disentanglement allows for example to swap the facial expressions (dynamic factors) between faces (static factors).

As the above description might leave room to some ambiguity into what constitutes a static/dynamic variable, we introduce in this work an explicit and novel formal definition of those factors. Through this formalism, we notably unveil that the factors might not be independent. This paper proposes to investigate the effect of that dependency on the disentanglement models and shows that it translates in difficulties for SotA methods to accurately capture a representation of static and dynamic features. Their extracted codes remain entangled, highligting the importance of developing a robust disentanglement framework that also generalizes to non independent factors. To solve those issues, we propose a new generative model that explicitly incorporates the causal relationship between static and dynamic factors. Diverging from the classical Variational AutoEncoder (VAE) framework, it leverages Conditional Normalizing Flow (cNF) [1,40,53] to offer improved disentanglement and model expressivity. By exploiting our new formalism, we uncover conditions under which the ground truth factors can be provably inferred and show that our model directly satisfies those conditions using a simple shuffle operation. This leads to a novel evidence lower bound that naturally encourages disentanglement and makes the learned representation provably disentangled.

Among the other works that have addressed this objective under the VAE framework [3,6,12,22], DSVAE [29] is one of the first models that proposed unsupervised disentanglement of the static/dynamic factors through a factorized sequential latent variable model. However, it was proven to fail in properly disentangling the factors, and to result in representations that heavily depend on the

architecture and hyperparameters [35]. Haga *et al.*[18] propose to alleviate this problem using action labels as supervision. However, this requires costly annotations that are rarely available. Subsequent works have therefore extended the DSVAE framework using self-supervised methods aimed at constraining the disentanglement [4,19,35,41,45,60]. Although these approaches demonstrate improvements over DSVAE, they rely on increasingly complex and cumbersome constraints, besides limiting themselves to cases with independent static/dynamic factors. Comparatively, our approach solves both these problems with our shuffle constraint that does not require any additional loss hence making our method highly simple and non dataset/domain specific.

2 Background

Let $\mathcal{G} = \{\mathbf{x}^j_{1:T_j}\}_{j=1}^N$ denote a dataset of N video sequences of length T_j where each $\mathbf{x}^j_t$ corresponds to a video frame at time step t in sequence j. The main hypothesis underlying this work is that the frames $\mathbf{x}_t$ can be uniquely represented into two parts, $\mathbf{s}$ and $\mathbf{d}_t$ with:

- $\mathbf{s}$: the *static* factors, common and shared by all frames, representing the *time invariant* information;
- $\mathbf{d}_t$: the *dynamic* factors, changing between frames, corresponding to the *time varying* information.

Due to temporal coherence, consecutive frames have common factors that are shared between them. Therefore, inspired by Pattern Transformation Manifolds (PTM) representation [50], we consider that each frame resides on a common manifold that corresponds to that shared content. The code $\mathbf{s}$ represents the manifold while $\mathbf{d}_{1:T}$ captures the varying parts specific to each frame. For example, for a video of a person walking, all frames share the same background and person identity thus $\mathbf{s}$ represents those static factors, while $\mathbf{d}_{1:T}$ encodes the person specific pose at each frame i.e., the motion.

2.1 Disentangled Sequential VAE

With the above representation, the work in [29] proposed the following probabilistic model where each frame is generated from its corresponding static and dynamic codes and the dynamic codes are obtained given previous time steps:

$$p(\mathbf{x}_{1:T}, \mathbf{s}, \mathbf{d}_{1:T}) = p(\mathbf{s}) \prod_{t=1}^{T} p(\mathbf{d}_t|\mathbf{d}_{<t}) p(\mathbf{x}_t|\mathbf{s}, \mathbf{d}_t). \tag{1}$$

The objective is to extract the codes $\mathbf{s}$ and $\mathbf{d}_{1:T}$ given the observed data i.e., to learn the factorized posterior distribution $p(\mathbf{s}, \mathbf{d}_{1:T}|\mathbf{x}_{1:T})$. Variational inference can be used to learn an approximation of that posterior:

$$q(\mathbf{s}, \mathbf{d}_{1:T}|\mathbf{x}_{1:T}) = q(\mathbf{s}|\mathbf{x}_{1:T}) \prod_{t=1}^{T} q(\mathbf{d}_t|\mathbf{s}, \mathbf{d}_{<t}, \mathbf{x}_{\leq t}) \tag{2}$$

with the two terms obtained through separate sequential networks [29]. The model is trained via the VAE algorithm. This encoder network, introduced in

[29], is named 'full q' and is the building block of all following sequential disentanglement works. Unfortunately, despite the disentanglement bias imposed by the factorized latent representation, this model might not be sufficient to achieve a well-disentangled representation. Indeed, as formalized by [32], unsupervised disentanglement is fundamentally impossible to achieve without inductive biases since there exists a potentially infinite number of entangled $\mathbf{s}$ and $\mathbf{d}_{1:T}$ having the same marginal $\mathbf{x}_{1:T}$, making the generative model non identifiable. Additionally, VAEs are known to suffer from the so-called 'information preference property' [11,58]. A model with a complex conditional likelihood distribution tends to model a maximum of information in the decoder and ignores the latent variables that become non-informative and trivially match the prior.

DSVAE [29] tries to mitigate that problem using low dimensional dynamic codes $\mathbf{d}_{1:T}$ to promote information in the static variable. However, this requires careful adaptation of the hyperparameters making this method highly impractical without any guarantees on the resulting representation. Following works have therefore extended the DSVAE framework in order to constrain the disentanglement using self-supervised methods that augment the model loss with mutual information (MI) terms. More specifically, the aim is to maximize $I(\mathbf{x}_{1:T}, \mathbf{s})$ and $I(\mathbf{x}_{1:T}, \mathbf{d}_{1:T})$ while simultaneously minimizing $I(\mathbf{d}_{1:T}, \mathbf{s})$. Estimating those MI terms is not straightforward and is done either through domain specific methods (e.g., data augmentation, adversarial training or external models [4,19,60]), or through the model itself (e.g., costly encoding-decoding counterfactual losses [45] or contrastive MI estimation with VAE-based sampling [41]). Additionally, noting an apparent improvement on disentanglement, these methods modify Eq.(2) in order to model both codes independently by ablating the conditional link.

The above models however have several limitations:

1. While the proposed constraints have shown to improve over DSVAE, these methods are either modality-based i.e., data/task-dependent, or they rely on the inherent ability of their model architecture to naturally disentangle. Thus, might end up promoting an improperly disentangled representation.
2. Even for simple datasets, semantically meaningful factors are generally not independent [54]. Hence, the conditional link between the two codes should *not* be ablated. We argue that the improvement observed from ablation is due to a lack of expressivity in the conditioning model and MI constraints that are contradicting under potential conditionality.
3. The definition of the static and dynamic factors is not formal, leaving room to some ambiguity into what constitutes a static/dynamic factor.

2.2 Other Related Works

Differently than the VAE-based methods derived from the DSVAE model [3,4,18,19,29,35,41,45,60], Berman *et al.*[6] recently developed structured Koopman autoencoders that try to achieve multifactor disentanglement of the sequential data with more than two disentangled components. However, their results still lie behind recent DSVAE-based models [41]. In parallel, the work FHVAE [21,22] also proposed unsupervised disentanglement through an LSTM-based

model but is limited to audio data. Then, within the works that try to tackle disentanglement learning for static data [10,20,26], we can mention the field of style/content disentanglement [7,15,49,57], with a definition of the static variables related to ours (Sec.3). Interestingly, the effectiveness of those methods can thus be explained through the prism of our formalism in Sect. 4.1. This further demonstrates our propositions and justifies our approach for sequential disentanglement, which comparatively does not require additional losses [15,49] or model settings that depend on the dataset [57]. Finally, several GAN-based models have been successfully applied for disentangling style/content [24,25] or appearance/motion for video generation [44,46,52]. However, compared to VAE approaches, these methods do not allow to easily encode the input frames and are thus typically less applicable in the context of representation learning.

For image generation, several works have also explored improving VAE models expressivity using NFs to augment either the approximated posterior [28,47], the prior [11] or the conditional likelihood [1,53]. For videos, Marino *et al.*[37] tried to augment the conditional likelihood of sequential latent variable models based on Autoregressive flows [23]. However, none of these approaches were investigated for disentanglement. Closer to our work, Ma *et al.*[36] decouple the global and local representation of images by embedding a NF in the VAE framework to model the decoder. This shows that cNF naturally tends to disentangle, hence justifying the approach taken in this paper.

Following the work of Locatello *et al.*[32], a handful of methods have recently tried to propose weak-supervision methods based on pairs of images [33] to ensure the representation from a VAE is identifiable. This was extended to causal variables in [8] and to sequences in [30]. However, those methods rely either on knowing the number of changing variables or on labels.

Contributions: Our work seeks to solve the limitations of previous works by first proposing to formalize the definition of the factors and then building on this formalism to introduce a new model that accounts for the potential dependency between the variables. Our model extends the classical VAE approach through the use of additional Normalizing Flows. In stark contrast to previous works, we show that, by leveraging the data structure through a simple shuffle operation, we can obtain a provably disentangled representation without any additional loss or supervision (no data augmentation [4], contrastive loss [41], external models [60] or adversarial training [19]), making our method undemanding and modality free, while having theoretical guarantees. This is detailed in the following sections.

3 Problem Formulation

Before trying to disentangle the static and dynamic factors, we propose a new definition of those concepts. This is done through formalizing the video generative process as a latent variable model that comprises two disjoint codes representing the time varying and time invariant factors. We further provide the assumptions underlying this partition.

Disentangled Video Generative Model: We propose a model where the generative process of each frame consists in first drawing a latent code $\mathbf{z}_t$ from its associated probability density $p(\mathbf{z}_t)$ and then obtaining the observations $\mathbf{x}_t$ by sampling from $p(\mathbf{x}_t|\mathbf{z}_t)$. More formally, the frame generative process is:

$$\mathbf{z}_t \sim p(\mathbf{z}_t), \qquad \mathbf{x}_t = \mathbf{g}(\mathbf{z}_t) \tag{3}$$

where $p(\mathbf{z}_t)$ is a smooth, continuous density on $\mathcal{Z}$ with full support and $\mathbf{g} : \mathcal{Z} \rightarrow \mathcal{X}$ is a smooth and invertible function with smooth inverse (diffeomorphism) mapping the frame representation space $\mathcal{Z} \subseteq \mathbb{R}^k$ to the observation space $\mathcal{X} \subseteq \mathbb{R}^n$, with usually $n \gg k$.

Based on the static/dynamic disentanglement assumption presented in Sect. 2, the video sequence frame codes $\mathbf{z}_{1:T}$ can be partitioned into a single invariant static code $\mathbf{s}$ and temporally varying dynamic codes $\mathbf{d}_{1:T}$, i.e., $\mathcal{Z} = \mathcal{S} \times \mathcal{D}$ with $\mathcal{S} \subseteq \mathbb{R}^{n_s}$ and $\mathcal{D} \subseteq \mathbb{R}^{n_d}$, $k = n_s + n_d$:

$$\mathbf{z}_{1:T} = (\mathbf{s}, \mathbf{d}_{1:T}) \sim p(\mathbf{s})p(\mathbf{d}_1|\mathbf{s}) \prod_{t=2}^{T} p(\mathbf{d}_t|\mathbf{s}, \mathbf{d}_{<t}), \tag{4}$$

$$\mathbf{x}_t = \mathbf{g}(\mathbf{s}, \mathbf{d}_t), \tag{5}$$

with $\mathbf{s}$ corresponding to the first n_s dimensions of $\mathbf{z}_t$, $\mathbf{s} = \mathbf{z}_{t,1:n_s}$. Each frame is generated from its corresponding static and dynamic factors with the dynamic factors obtained given previous time steps **and** the static factors $\mathbf{s}$. From Eq. 4, it is directly given that $\forall \mathbf{x}_{1:T}$ and $t, t' \in \{1, \cdots, T\}$, $\mathbf{g}^{-1}(\mathbf{x}_t)_{1:n_s} = \mathbf{g}^{-1}(\mathbf{x}_{t'})_{1:n_s} = \mathbf{s}$. That is the static factors are invariant for each sequence. As a result, it can be noted that this expression actually matches the formalism presented in the context of style/content disentanglement [49] for static images but it is generalized here to multiple temporally varying views. It ensues that the dynamic factors can be defined based on the same assumptions than the ones adopted for the 'style' factors in [49]:

1. $p(\mathbf{d}_t|\mathbf{s}, \mathbf{d}_{<t}) = \delta(\mathbf{d}_{t,A_t^c} - \mathbf{d}_{t-1,A_t^c})p(\mathbf{d}_{t,A_t}|\mathbf{s}, \mathbf{d}_{<t})$ with $A_t \subseteq \{1, ...n_d\}$ denoting the subset of varying dynamic components at time step $t \geq 2$ with associated $p(A_t)$, and A_t^c the complement of A_t.
2. $\forall i \in \{1, ...n_d\}$, $\exists t \in \{2, ...T\}$, $A_t \subseteq \{1, ...n_d\}$ s.t. $i \in A_t$; $p_{A_t}(A_t) > 0$ and for any $(\mathbf{s}, \mathbf{d}_{<t})$, $p(\mathbf{d}_{t,A_t}|\mathbf{s}, \mathbf{d}_{<t})$ is smooth and fully supported in some open non empty subset containing $\mathbf{d}_{t-1,A_t}$.

The second assumption means that every dynamic variable varies with time. However, as stated by the first assumption, it does not force all dynamic components to change at each time step or for each sequence, echoing the fact that only a subset of the dynamic factors might be fluctuating in a given sequence.

This video generative model does not make any assumption on the relationship between the factors and allows for arbitrary marginals $p(\mathbf{s}, \mathbf{d}_{1:T})$ with potentially complex non trivial statistical relationships between (and within) $\mathbf{s}$ and $\mathbf{d}_{1:T}$. That is $I(\mathbf{s}, \mathbf{d}_{1:T}) \geq 0$. Overall, this illustrates that, while the static and dynamic factors constitute two separate semantic concepts, they might not be independent. As a result, similarly to [33,49] for pairs of images, we propose to further extend our description using a Structural Causal Model (SCM) [55] to express the relation between factors.

Causal Model: From the assumption that $\mathbf{s}$ is invariant throughout the sequences while the $\mathbf{d}_{1:T}$ varies, it results that the static factors may causally influence the dynamic ones but not the opposite. Indeed, if $\mathbf{s}$ would depend on some dynamic variable, following the definition of $\mathbf{d}$ above, this dynamic variable varies and therefore $\mathbf{s}$ would vary as well, contradicting the static invariance. Assuming the dynamic variables undergo temporally coherent perturbations, videos can thus be interpreted as temporally intervened sequences where successive frames result from the causal generative model under soft intervention on the dynamic factors $\mathbf{d}$. This can be formally written as :

$$\mathbf{s} = \mathbf{h}_s(\epsilon_s), \qquad \mathbf{d}_1 = \mathbf{h}_d(\mathbf{s}, \epsilon_{d,1}), \qquad \mathbf{d}_t = \mathbf{h}'_d(\mathbf{s}, \epsilon_{d,<t}, \epsilon_{d,t}) \tag{6}$$

with ϵ_s and $\epsilon_{d,1:T}$ independent exogenous variables and $\mathbf{h}_s$,$\mathbf{h}_d$,$\mathbf{h}'_d$ functions describing the causal mechanism. Equation 6 shows that $\mathbf{s}$ might causally influence the specific values of the dynamic factors but also their possible variations.

By formalizing the video generative process based on static/dynamic factors disentanglement, the link with data augmentation generative process [49] becomes apparent. In the particular case where T=2, the two problems are equivalent with the exception that, compared to style/content, the augmentation is not human-designed but is a result of temporal variations in the sequence. This analogy provides an interpretation of videos as augmented views of a scene through temporal augmentation between each frame. Moreover, in [49] an approach is proposed to provably identify the content for style/content disentanglement. The similarities between the two problems thus also hint at the possibility of proposing a similar approach that would provably isolate the static **and** dynamic factors for sequential disentanglement. We formulate now this idea of provably identifying the different factors.

Definition 1. *The ground truth factors* $\mathbf{s}$ *and* $\epsilon_{d,1:T}$ *are said to be identified by learned vectors* $\mathbf{f}$ *and* $\lambda_{1:T}$ *if there exist invertible transformations* $\mathbf{m}$ *and* $\mathbf{a}$ *s.t.* $\mathbf{f} = \mathbf{m}(\mathbf{s})$, $\lambda_{1:T} = \mathbf{a}(\epsilon_{d,1:T})$ *i.e.*, $\mathbf{f}$ and $\lambda_{1:T}$ *are reparametrizations of* $\mathbf{s}$, $\epsilon_{d,1:T}$ *respectively.* This means that the learned codes $\mathbf{f}$ should capture exactly the static factors $\mathbf{s}$, while $\lambda_{1:T}$ should encode the dynamic conditionally to $\mathbf{s}$. This is termed as 'conditional disentanglement'. The use of separate notations serves to differentiate the learned codes $\mathbf{f}$ and $\lambda_{1:T}$ (extracted by the designed network and constraints) from the ground truth factors $\mathbf{s}$ and $\epsilon_{d,1:T}$ (imposed by the video generative process) which might not be equal in practice. The potential mismatch between the objective and the generative process is avoided by representing the exogeneous dynamic variables $\epsilon_{d,1:T}$ instead of $\mathbf{d}_{1:T}$. If the static/dynamic factors are independent, $\epsilon_{d,1:T}$ and $\mathbf{d}_{1:T}$ are equivalent and our formulation becomes therefore a generalization of the objective of Sect. 2 that also accounts for causal dependencies between factors.

4 Method

We seek to learn from the video sequences $\mathbf{x}_{1:T}$, two latent codes, a static code $\mathbf{f}$ and dynamic codes $\lambda_{1:T}$ such that these codes identify the ground-truth factors

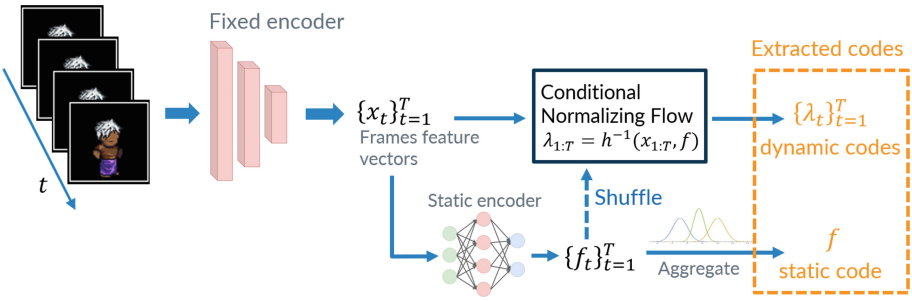

Fig. 3. Schematic view of the proposed model. A pretrained Convolutional encoder embeds each frame separately into the frame latent space $\mathbf{x}_{1:T}$. These vectors are used as input to the static encoder which estimates the static codes from each individual frame. The estimations are then aggregated giving the static code $\mathbf{f}$. The static vectors serve to condition a Conditional Normalizing Flow that models the likelihood of the frame feature vectors $\mathbf{x}_{1:T}$. The transformed vectors $\lambda_{1:T}$ correspond to the dynamic codes. The model is trained using the loss $\mathcal{L}$ in Eq. 12.

as presented in Definition 1. We start by deriving from our formalism sufficient conditions to provably extract the factors. We then propose a practical model and show how, strictly by leveraging the data structure, this method does achieve the disentanglement conditions. The complete model is summarized in Fig. 3.

4.1 Sufficient Conditions for Disentanglement

Static Disentanglement: The objective is to learn an encoder posterior distribution $q(\mathbf{f}|\mathbf{x}_{1:T})$ mapping the sequences to their static code such that the static codes $\mathbf{f}$ sampled from $q(\mathbf{f}|\mathbf{x}_{1:T})$ solely capture the static factors $\mathbf{s}$.

Proposition 1. *Consider the generative process and conditions presented in Sect. 3. Further assume that* $dim(\mathbf{f}) = n_f \geq n_s$. *Let* $\mathbf{x}$ *and* $\mathbf{x}^*$ *denote two frames (or disjoint sets of frames) belonging to the same sequence* $\mathbf{x}_{1:T}$ *and* $\mathbb{D}$ *be a divergence between two distributions. Given unlimited data from* $p(\mathbf{x}_{1:T})$, *any candidate posterior distribution* $q(\mathbf{f}|\mathbf{x}_{1:T})$ *that minimizes*

$$L_{prop1} = \mathbb{E}_{p(\mathbf{x}_{1:T})}\mathbb{D}(q(\mathbf{f}|\mathbf{x}), q(\mathbf{f}|\mathbf{x}^*)) - I(\mathbf{f}, \mathbf{s}) + I(\mathbf{s}, \mathbf{s}), \tag{7}$$

is disentangled in the sense that $q(\mathbf{f}) = \int q(\mathbf{f}|\mathbf{x}_{1:T})p(\mathbf{x}_{1:T})d\mathbf{x}_{1:T}$, *the aggregated posterior, is a reparametrization of* $p(\mathbf{s})$.

The proof is provided in App.A.2. Intuitively, by imposing $\mathbf{f}$ to be both invariant to the dynamic factors and informative of the static ones, the code $\mathbf{f}$ will capture all and only $\mathbf{s}$. Proposition 1 extends Thm.4.4 of [49] to distributions and cope with an unknown number of static variables n_s. Proposition 1 shows that simply maximizing $I(\mathbf{x}_{1:T}, \mathbf{f})$ as proposed in Sect. 2 is not sufficient to extract $\mathbf{s}$. Indeed, as shown in [16,56], a global aggregated code $\mathbf{f}$ could encode the dynamic information and should therefore additionally be made invariant.

Dynamic Disentanglement: As explained in Sect. 3, because the static and dynamic factors might not be independent, the goal for the dynamic codes $\lambda_{1:T}$ is to capture $\epsilon_{d,1:T}$. Unfortunately, following an approach similar to Proposition 1 is not possible in this case. Indeed, this approach would require to have access to two samples $\mathbf{x}$ and $\mathbf{x}^*$ that share the same motion but have different static factors. However, by the definition of the dynamic factors (Sect. 3) there are no easily obtainable such samples since $\mathbf{d}_t$ is specific to each frame and varies at unknown time steps. A solution proposed in [4] consists in generating positive samples using static data augmentation. However, designing those augmentations might be non trivial and dataset specific, with the risk of artificially and indirectly defining what is static/dynamic through the chosen augmentations. Moreover, only the static part should be affected, limiting the amount of available augmentations, possibly favoring static information in the dynamic codes. Instead, an alternative approach is proposed here that does not require positive samples.

Proposition 2. *Consider the same framework as in Proposition 1 and the static code* $\mathbf{f}$ *to be disentangled as described above. Further assume that* $dim(\lambda) = n_\lambda \geq n_d$. *Any smooth and invertible candidate function* $\mathbf{h}$ *s.t.* $\mathbf{x}_{1:T} = \mathbf{h}(\lambda_{1:T}, \mathbf{f})$ *which minimizes*

$$L_{prop2} = I(\mathbf{f}, \lambda_{1:T}), \tag{8}$$

is disentangled in the sense that $q(\lambda_{1:T})$ *is a reparametrization of* $p(\epsilon_{d,1:T})$.

The proof is provided in App.A.3. Intuitively, assuming that $\mathbf{f}$ represents exactly $\mathbf{s}$ thanks to Proposition 1 and that both $\mathbf{f}$ and $\lambda_{1:T}$ are not redundant, because $\mathbf{h}$ is invertible, the dynamic codes $\lambda_{1:T}$ will have to be informative and to capture all and only $\epsilon_{d,1:T}$.

Overall, by using the formalism in Sect. 3, we are able to discover sufficient conditions such that the learned codes are reparametrizations of the ground truth factors. We will now show how we learn $q(\mathbf{f}|\mathbf{x}_{1:T})$ and $\mathbf{h}$ in practice so that the conditions are enforced from the observed data only, without requiring explicit formulations of L_{prop1} and L_{prop2}.

4.2 Practical Implementation

Compared to previous methods that solely rely on VAEs, we propose instead to improve sequence modeling based on additional Normalizing Flows (NF) [43]. A NF is a transformation of a random variable through a sequence of differentiable invertible mappings. As a result, it is a natural candidate to learn the invertible function $\mathbf{h}$. More specifically, assuming the common static code $\mathbf{f}$ has been extracted from the frames, we model the frames likelihood $p(\mathbf{x}_{1:T}|\mathbf{f})$ using a cNF [53] as

$$p(\mathbf{x}_{1:T}|\mathbf{f}) = p(\lambda_{1:T}|\mathbf{f}) \left| \det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T}, \mathbf{f})}{\delta \mathbf{x}_{1:T}} \right| \tag{9}$$

with $\lambda_{1:T} = \mathbf{h}^{-1}(\mathbf{x}_{1:T}, \mathbf{f})$ the dynamic codes. This term can be evaluated through exact likelihood evaluation. Because a NF is a bijective transformation and $\mathbf{f}$ is

common to all frames, the transformed latents $\lambda_{1:T}$ will retain and encode the dynamics of the sequence. Since the NF is conditioned on the static code $\mathbf{f}$, the dynamics are encoded conditionally to the static part. The NF is based on affine coupling [13] and LSTM layers that model the temporal behavior of the sequence, similarly to [37], and provide a bias towards encoding the dynamics. Because, for a disentangled model, the transformation $\mathbf{h}$ needs to fully model the conditional dependence and causal mechanism, the final distribution $p(\lambda_{1:T}|\mathbf{f})$ is replaced with $p(\lambda_{1:T})$. Details on the architecture are provided in App.B.3.

It remains to show how to extract the static code $\mathbf{f}$ a priori before solving Eq.(9). Variational inference is used in order to learn an approximate posterior $q(\mathbf{f}|\mathbf{x}_{1:T})$. Following the standard VAE framework, the marginal data log-likelihood is lower estimated by the Evidence Lower BOund (ELBO):

$$\log p(\mathbf{x}_{1:T}) \geq \mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}) - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{10}$$

$$\geq \mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} \log p(\lambda_{1:T}) + \log \left| \det \frac{\delta \mathbf{h}^{-1}}{\delta \mathbf{x}_{1:T}} \right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{11}$$

with the two first terms corresponding to the cNF likelihood. The detailed derivation is shown in App.A.1. Overall, the model can be described as a VAE that extracts a single latent code $\mathbf{f}$ from the observations and has a cNF decoder to model the likelihood of the frames. By maximizing the ELBO, we learn both an approximation of the encoder posterior $q(\mathbf{f}|\mathbf{x}_{1:T})$ and the cNF likelihood i.e., the transformation $\mathbf{h}$. This model does not make any assumption on the independence between the codes and encompasses the potential conditional dependencies through the cNF. In comparison, the 'full q' model (Sect. 2) is equivalent to model $p(\mathbf{x}_{1:T}|\mathbf{f})$ using a VAE with a latent vector for each time step and a simple concatenation of the codes as inputs to the generative network. The use of a Normalizing flow acts as a more flexible and expressive 'mix-up' of the two extracted codes.

The prior distribution $p(\mathbf{f})$ is further modeled using a NF. This is in contrast with most works that propose to use NF to generate more flexible posteriors instead of priors [28,47]. As demonstrated in [11], if these two approaches are equivalent during training, at test time, for sampling, a NF prior allows to have a longer decoder path and produces better samples. To model the inference posterior distribution, the static code encoder $q(\mathbf{f}|\mathbf{x}_{1:T})$ needs to output a single latent code from multiple frames. This is achieved by means of an aggregated posterior distribution [7]. The static code distribution $q(\mathbf{f}|\mathbf{x}_{1:T})$ is constructed as the product of individual distributions estimated from each frame by a static encoder network: $q(\mathbf{f}|\mathbf{x}_{1:T}) = \prod_{t=1}^{T} q(\mathbf{f}_t|\mathbf{x}_t)$. Compared to previous works where $\mathbf{f}$ is inferred through recurrent layers, the aggregated posterior is inherently permutation invariant, which provides a bias towards our disentanglement objective.

Shuffle Constraint: Unfortunately, the numerous disentanglement biases included in this proposed architecture might not be sufficient to achieve a well-disentangled model. In order to achieve a clean disentanglement, it is therefore proposed to leverage the sequential nature of video data directly into the framework. Indeed, thanks to the aggregated posterior, the model directly provides

the static code estimation for individual frames $q(\mathbf{f}_t|\mathbf{x}_t)$, without any additional computation. The idea is to condition the cNF, during training, on the shuffled $\mathbf{f}_{1:T}$ before aggregation, instead of conditioning the cNF on the aggregated static code $\mathbf{f}$. This is justified by the fact that, for a disentangled model, the $\mathbf{f}_{1:T}$ need to be invariant i.e., $p(\mathbf{x}_{1:T}|\mathbf{f}) = p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi(T)})$ with $\pi(T)$ denoting a random permutation. Using this shuffled operation, the loss becomes:

$$\boxed{\mathcal{L} = \mathbb{E}_{p(\mathbf{x}_{1:T})}\mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} - \alpha \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi(T)}) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f}))} \quad (12)$$

which is the negative dataset ELBO where $p(\mathbf{x}_{1:T}|\mathbf{f})$ has been replaced by $p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi(T)})$. Following the method of β-VAE [20], the different terms in $\mathcal{L}$ have additionaly been weighted using weight coefficients α and β.

Proposition 3. *Consider any* $\mathbf{h}$ *and* $q(\mathbf{f}|\mathbf{x}_{1:T})$ *as described above which minimize* $\mathcal{L}$ *and further assume* $\alpha \gg \beta$. *Minimizing* $\mathcal{L}$ *is equivalent to minimizing both* L_{prop1} *and* L_{propr2}. *Propositions 1 and 2 are satisfied and the learned representation* $(\mathbf{f}, \lambda_{1:T})$ *is disentangled in the sense of Definition 1.*

The proof is given in App.A.4. Intuitively, shuffling the static codes $\mathbf{f}_{1:T}$ and using them to reconstruct the frames, enforces the static code to be time invariant, while simultaneously being informative due to the reconstruction objective (favored by imposing $\alpha \gg \beta$). Meanwhile, because we minimize $-\log p(\lambda_{1:T}) \approx H(\lambda_{1:T})$ through the cNF, the dynamic code is encouraged to be non informative. In turn, this leads to the minimization of both L_{prop1} and L_{prop2}, resulting in a disentangled model. Further details are provided in App.C.2.

Summary: we showed that, through a simple shuffle of the static codes, the proposed model is provably disentangled without requiring any additional complex loss, unlike previous methods, with all disentanglement conditions directly satisfied by our novel evidence lower bound. This makes our method modality free and highly efficient by avoiding any unnecessary constraint. In the case where the factors are independent, the model tries to capture both $\mathbf{s}$ and $\mathbf{d}_{1:T}$. However, the proposed model also generalizes to factorize the static and dynamic factors without assuming independence. It does so by directly modeling the causality and only capturing $\epsilon_{d,1:T}$.

Using Pretrained Autoencoders: Similarly to [30], to facilitate disentanglement, instead of directly extracting the two separate codes, it is proposed to first learn an Autoencoder trained to encode/decode the frames without disentanglement. Gaussian noise is added on the frame feature vectors to prevent collapse. After convergence, the autoencoder parameters are frozen and we learn the static encoder and conditional Normalizing Flow which provide the disentangled representation from the entangled one. Because a Normalizing Flow is invertible, it ensures no information is lost and the frozen decoder can be used to recover the frames without fine-tuning. This permits to fully factorize the disentanglement problem from the task of mapping high-dimensional complex images into a low dimensional space.

5 Experiments

In order to validate the effectiveness of the proposed model, it is compared qualitatively and quantitatively to state-of-the-art sequence disentanglement learning methods including MoCoGAN [46], DSVAE [29], R-WAE [19], S3VAE [60], SKD [6], CDSVAE [4] and SPYL [41].

Datasets: Similarly to the baseline works, the model is evaluated using the standard *LPCSprites* [42] and *MUG* [2] databases, which contain videos of animated cartoon characters and facial expressions respectively. As all methods manage to provide high quality results on the original *LPCSprites* dataset [41], it is modified in order to include more complex characters with static factors other than colors. Additionally, the *MPI3D* database [17] is also considered in the experiments. It consists of a real-world robot arm carrying various objects in different positions with the arm motion made dependent on the background light, and the speed of the arm dependent on the object size. We further evaluate on the human action recognition *MHAD* dataset [9] comprising videos of humans performing diverse actions. Finally, based on the *dSprites* dataset [38], sequences of moving 2D shapes are generated. Two variants are proposed, one with fixed rotation and all factors independent (*s-dSprites*) and the other with rotating shapes and objects that bounce on the edges according to their size (*c-dSprites*).

Disentanglement Metrics: Quantitative evaluation is provided following the common benchmark for sequential disentanglement [4,29,60]. After encoding a sequence with known factors, either the static or dynamic code is replaced by a sample from its prior before reconstruction. A classifier, pretrained with full supervision, is then used to measure how well the factors associated with the fixed code are preserved for the reconstructed sequence. Four metrics are extracted: the accuracy for the fixed factors (Acc), the intra-entropy ($H(y|x)$) that measures the classifier confidence for the predictions, the inter-entropy ($H(y)$) that estimates the diversity for the sampled factors and the inception score (IS) that measures the generator performance.

Details on the model architecture and hyperparameters, datasets, disentanglement metrics as well as additional results can be found in App.B and C.

Results: The *MUG*, *MHAD* and *s-dSprites* datasets have independent static-dynamic factors. Their corresponding results are shown in Table 1. As expected, when the factors are independent all recent methods manage to provide a disentangled representation. In particular, while using a simple shuffle operation, our proposed model performs on par with the complex state-of-the-art methods on all datasets. Despite the dynamic codes being extracted conditionally to $\mathbf{f}$, the learned representation correctly factorizes the independent factors. Only the SPYL method [41] gives slightly better results on *MUG* and *MHAD*. However, we show in App.C.1 that this can be mainly explained by the tendency of SPYL to encode too much information in its dynamic codes. In Fig. 2, qualitative results for the MUG dataset are displayed with sequences generated by swapping the dynamic and static codes of test samples. The resulting videos accurately retain

Table 1. Disentanglement metrics on *s-dSprites*, *MUG* and *MHAD* datasets with independent factors. The dynamic factors are highligted in red.

MUG	Acc↑ (%) action	IS↑	$H(y\|x)$↓	$H(y)$↑
MoCoGAN	63.12	4.332	0.183	1.721
DSVAE	54.29	3.608	0.374	1.657
R-WAE	71.25	5.149	0.131	1.771
S3VAE	70.51	5.136	0.135	1.760
SKD	77.45	**5.569**	**0.052**	1.769
C-DSVAE	81.16	5.341	0.092	1.775
SPYL	**85.71**	5.548	0.066	**1.779**
Ours	84.32	5.329	0.068	1.778

s-dSprites	Acc↑ (%) color	shape	size	IS↑
DSVAE	79.38	42.10	43.61	3.035
CDSVAE	96.18	98.22	98.71	3.995
SPYL	97.76	98.72	98.12	3.685
Ours	**98.91**	**98.98**	**98.76**	**4.263**

MHAD	Acc↑ (%) action	IS↑	$H(y\|x)$↓	$H(y)$↑
C-DSVAE	65.59	1.749	0.420	1.282
SPYL	**68.63**	1.780	**0.362**	1.299
Ours	67.65	**1.896**	0.371	**1.325**

Table 2. Disentanglement metrics on *c-dSprites*, *MPI3D* and *LPCSprites* with dependent factors. In red and blue are highligted respectively the dynamic factors and the static factors that condition the motion.

c-dSprites	Acc↑ color	(%) shape	size	IS↑	$H(y\|x)$↓	$H(y)$↑
DSVAE	72.09	32.66	79.38	4.128	0.046	1.41
CDSVAE	96.33	74.32	93.17	4.141	0.028	1.39
SPYL	68.41	33.28	75.10	**4.392**	0.027	1.43
Ours	**98.92**	**98.98**	**98.59**	4.371	**0.008**	**1.44**

MPI3D	Acc↑ shape	(%) size	camera	light	IS↑	$H(y\|x)$↓	$H(y)$↑
DSVAE	28.29	65.57	98.81	87.71	3.505	0.056	1.19
CDSVAE	43.81	79.17	**99.99**	**99.94**	3.487	0.138	1.26
SPYL	16.78	50.29	33.32	86.54	3.819	0.043	1.28
Ours	**90.15**	**97.56**	99.97	99.93	**3.849**	**0.029**	**1.29**

LPCSprites	Acc↑ body	(%) hair	orient.	action	IS↑	$H(y\|x)$↓	$H(y)$↑
CDSVAE	54.94	85.75	33.32	67.71	2.737	1.6e-3	1.00
SPYL	70.89	99.89	33.32	**99.86**	2.735	2.0e-3	**1.01**
Ours	**99.97**	**99.91**	**99.94**	99.83	**2.768**	**3.3e-4**	**1.01**

the person identities while exchanging the face motions with high quality non blurry outputs.

For the *LPCSprites*, *c-dSprites* and *MPI3D* datasets the static-dynamic factors are not independent. As an example, the *c-dSprites* dataset displays 2D objects that move and rotate. Because the 2D shapes are either squares, ellipses or hearts, even if the rotation is generated independently, it is still causally influenced by the shape due to the different order of rotational symmetry between

the shapes. Overall, this example highlights the fact that, even in simple cases, the static content of a scene may influence the dynamic of the sequence by conditioning the possible variations between frames. In Table 2 the results for the three aforementioned datasets are given. Compared to the case with independent factors, the performance of the baseline methods significantly drop and those models fail to provide a disentangled representation. As explained in Sect. 2, these methods extract the static and dynamic variables separately and enforce independence between the codes. This conflicts with the inherent causality of the ground truth data generative process (Sect. 3) and results in ambiguity into where the shared information between the dynamic and static factors should lie. If the dynamic code is extracted independently, it will inevitably have to retain some static information, translating in entangled codes and potentially encouraging static factors into the dynamic variables. In contrast, our method manages to provide a disentangled representation and generalizes to settings with complex statistical relationships between the factors. The conditionality is properly captured and high accuracy scores are maintained. These results are illustrated in Fig. 1 which shows sequences generated by fixing the static code and sampling the dynamic one. Compared to the competing CDSVAE model [4], the designed method presents improved sample quality that maintains the object shapes and identity. The samples display diverse motions that abide by the causal mechanism (i.e., only vertical motion for the *MPI3D* sample and fixed orientation for the *LPCSprites* one, see App.B.1 for details on the datasets).

Limitations: The proposed formalism (Sect. 3) defines the factors. But, by doing so, it also draws the intrinsic limitations of the static/dynamic disentanglement problem as it provides what could be disentangled at best through DSVAE based methods. First, it is the dataset itself that defines what is static/dynamic, which might not match the human intuition or expectations, e.g., for the *LPCSprites* dataset in Fig. 1, from a human perspective the orientation of a character would probably be labelled as a motion of that specific character identity. However, from the data perspective the orientation is static since it is time invariant. Second, the static code can only capture what is always invariant, i.e., factors that are static in every sequence of the dataset. A factor that can be either static or dynamic, depending on the sequence, is encompassed in $\mathbf{d}_{1:T}$. Hence, future work may investigate whether the proposed method can be extended to adaptively extract the static code for each sequence and further disentangle the factors at the sequence level.

6 Conclusion

In this work, we introduce a novel formal approach toward sequential data disentanglement. Diverging from previous works, it extends the VAE framework with a conditional Normalizing Flow that directly models the relationships between the static and dynamic factors, making it closer to the data generative process. This constitutes a crucial change compared to the baseline methods by lifting

their most detrimental assumption. It translates into large improvements over the state-of-the-art models, which fail to disentangle dependent factors even in simple cases. To further enforce disentanglement, a simple shuffle constraint is proposed which leads to a novel ELBO formulation and our model to be provably disentangled. Hence, compared to the baselines, our method is simple, non dataset/domain specific and theoretically justified. It proved its ability to properly disentangle the static/dynamic factors on multiple datasets and to generalize to less restrictive cases with dependent variables. Interestingly, while our method has been only evaluated for videos, the presented approach directly extend to other domains such as audio or biology which could be explored in future works.

Acknowledgements. Mathieu Cyrille Simon is a Research Fellow of the Fonds de la Recherche Scientifique - FNRS of Belgium. Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region

References

1. Agrawal, S., Dukkipati, A.: Deep variational inference without pixel-wise reconstruction. arXiv preprint arXiv:1611.05209 (2016)
2. Aifanti, N., Papachristou, C., Delopoulos, A.: The MUG facial expression database. In: International Workshop on Image Analysis for Multimedia Interactive Services (2010)
3. Albarracin, J.F.H., Rivera, A.R.: Video reenactment as inductive bias for content-motion disentanglement. In: IEEE TIP (2022)
4. Bai, J., Wang, W., Gomes, C.P.: Contrastively disentangled sequential variational autoencoder. In: NeurIPS (2021)
5. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. In: IEEE TPAMI (2013)
6. Berman, N., Naiman, I., Azencot, O.: Multifactor sequential disentanglement via structured Koopman autoencoders. arXiv preprint arXiv:2303.17264 (2023)
7. Bouchacourt, D., Tomioka, R., Nowozin, S.: Multi-level variational autoencoder: learning disentangled representations from grouped observations. In: AAAI (2018)
8. Brehmer, J., De Haan, P., Lippe, P., Cohen, T.S.: Weakly supervised causal representation learning. In: NeurIPS (2022)
9. Chen, C., Jafari, R., Kehtarnavaz, N.: UTD-MHAD: a multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In: ICIP (2015)
10. Chen, R.T., Li, X., Grosse, R.B., Duvenaud, D.K.: Isolating sources of disentanglement in variational autoencoders. In: NeurIPS (2018)
11. Chen, X., et al.: Variational lossy autoencoder. arXiv preprint arXiv:1611.02731 (2016)
12. Denton, E.L., et al.: Unsupervised learning of disentangled representations from video. In: NeurIPS (2017)
13. Dinh, L., Krueger, D., Bengio, Y.: NICE: Non-linear independent components estimation. arXiv preprint arXiv:1410.8516 (2014)

14. Fragemann, J., Ardizzone, L., Egger, J., Kleesiek, J.: Review of disentanglement approaches for medical applications–towards solving the Gordian knot of generative models in healthcare. arXiv preprint arXiv:2203.11132 (2022)
15. Gabbay, A., Hoshen, Y.: Demystifying inter-class disentanglement. arXiv preprint arXiv:1906.11796 (2019)
16. Garnelo, M., et al.: Neural processes. arXiv preprint arXiv:1807.01622 (2018)
17. Gondal, M.W., et al.: On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In: NeurIPS (2019)
18. Haga, T., Kera, H., Kawamoto, K.: Sequential variational autoencoder with adversarial classifier for video disentanglement. Sensors **23**(5), 2515 (2023)
19. Han, J., Min, M.R., Han, L., Li, L.E., Zhang, X.: Disentangled recurrent wasserstein autoencoder. arXiv preprint arXiv:2101.07496 (2021)
20. Higgins, I., et al.: beta-VAE: learning basic visual concepts with a constrained variational framework. In: ICLR (2016)
21. Hsu, W.N., Glass, J.: Scalable factorized hierarchical variational autoencoder training. arXiv preprint arXiv:1804.03201 (2018)
22. Hsu, W.N., Zhang, Y., Glass, J.: Unsupervised learning of disentangled and interpretable representations from sequential data. In: NeurIPS (2017)
23. Huang, C.W., Krueger, D., Lacoste, A., Courville, A.: Neural autoregressive flows. In: ICML (2018)
24. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: CVPR (2019)
25. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of styleGAN. In: CVPR (2020)
26. Kim, H., Mnih, A.: Disentangling by factorising. In: ICML (2018)
27. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
28. Kingma, D.P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., Welling, M.: Improved variational inference with inverse autoregressive flow. In: NeurIPS (2016)
29. Li, Y., Mandt, S.: Disentangled sequential autoencoder. arXiv preprint arXiv:1803.02991 (2018)
30. Lippe, P., Magliacane, S., Löwe, S., Asano, Y.M., Cohen, T., Gavves, S.: CITRIS: causal identifiability from temporal intervened sequences. In: ICML (2022)
31. Liu, X., Sanchez, P., Thermos, S., O'Neil, A.Q., Tsaftaris, S.A.: Learning disentangled representations in the imaging domain. Med. Image Anal. **80**, 102516 (2022)
32. Locatello, F., et al.: Challenging common assumptions in the unsupervised learning of disentangled representations. In: ICML (2019)
33. Locatello, F., Poole, B., Rätsch, G., Schölkopf, B., Bachem, O., Tschannen, M.: Weakly-supervised disentanglement without compromises. In: ICML (2020)
34. Locatello, F., Tschannen, M., Bauer, S., Rätsch, G., Schölkopf, B., Bachem, O.: Disentangling factors of variation using few labels. arXiv preprint arXiv:1905.01258 (2019)
35. Luo, Y.J., Ewert, S., Dixon, S.: Towards robust unsupervised disentanglement of sequential data–a case study using music audio. arXiv preprint arXiv:2205.05871 (2022)
36. Ma, X., Kong, X., Zhang, S., Hovy, E.: Decoupling global and local representations via invertible generative flows. arXiv preprint arXiv:2004.11820 (2020)
37. Marino, J., Chen, L., He, J., Mandt, S.: Improving sequential latent variable models with autoregressive flows. In: Symposium on Advances in Approximate Bayesian Inference (2020)

38. Matthey, L., Higgins, I., Hassabis, D., Lerchner, A.: dSprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/ (2017)
39. Mita, G., Filippone, M., Michiardi, P.: An identifiable double VAE for disentangled representations. In: ICML (2021)
40. Morrow, R., Chiu, W.C.: Variational autoencoders with normalizing flow decoders. arXiv preprint arXiv:2004.05617 (2020)
41. Naiman, I., Berman, N., Azencot, O.: Sample and predict your latent: Modality-free sequential disentanglement via contrastive estimation. arXiv preprint arXiv:2305.15924 (2023)
42. Reed, S.E., Zhang, Y., Zhang, Y., Lee, H.: Deep visual analogy-making. NeurIPS (2015)
43. Rezende, D., Mohamed, S.: Variational inference with normalizing flows. In: ICML (2015)
44. Tian, Y., et al.: A good image generator is what you need for high-resolution video synthesis. arXiv preprint arXiv:2104.15069 (2021)
45. Tonekaboni, S., Li, C.L., Arik, S.O., Goldenberg, A., Pfister, T.: Decoupling local and global representations of time series. In: International Conference on Artificial Intelligence and Statistics (2022)
46. Tulyakov, S., Liu, M.Y., Yang, X., Kautz, J.: MoCoGAN: decomposing motion and content for video generation. In: CVPR (2018)
47. Vahdat, A., Kautz, J.: NVAE: a deep hierarchical variational autoencoder. In: NeurIPS (2020)
48. Villegas, R., Yang, J., Hong, S., Lin, X., Lee, H.: Decomposing motion and content for natural video sequence prediction. arXiv preprint arXiv:1706.08033 (2017)
49. Von Kügelgen, J., et al.: Self-supervised learning with data augmentations provably isolates content from style. In: NeurIPS (2021)
50. Vural, E., Frossard, P.: Learning pattern transformation manifolds for classification. In: ICIP (2012)
51. Wang, X., Chen, H., Tang, S., Wu, Z., Zhu, W.: Disentangled representation learning. arXiv preprint arXiv:2211.11695 (2022)
52. Wang, Y., Bilinski, P., Bremond, F., Dantcheva, A.: G3AN: disentangling appearance and motion for video generation. In: CVPR (2020)
53. Winkler, C., Worrall, D., Hoogeboom, E., Welling, M.: Learning likelihoods with conditional normalizing flows. arXiv preprint arXiv:1912.00042 (2019)
54. Yang, M., Liu, F., Chen, Z., Shen, X., Hao, J., Wang, J.: CausalVAE: Structured causal disentanglement in variational autoencoder. arXiv preprint arXiv:2004.08697 (2020)
55. Yang, M., Liu, F., Chen, Z., Shen, X., Hao, J., Wang, J.: CausalVAE: disentangled representation learning via neural structural causal models. In: CVPR (2021)
56. Ye, X., Bilodeau, G.A.: A unified model for continuous conditional video prediction. In: CVPR (2023)
57. Yin, D., Ren, X., Luo, C., Wang, Y., Xiong, Z., Zeng, W.: Retriever: Learning content-style representation as a token-level bipartite graph. arXiv preprint arXiv:2202.12307 (2022)
58. Zhao, S., Song, J., Ermon, S.: Towards deeper understanding of variational autoencoding models. arXiv preprint arXiv:1702.08658 (2017)
59. Zhu, X., Xu, C., Tao, D.: Commutative lie group VAE for disentanglement learning. ICML (2021)
60. Zhu, Y., Min, M.R., Kadav, A., Graf, H.P.: S3VAE: self-supervised sequential VAE for representation disentanglement and data generation. In: CVPR (2020)

Sequential Representation Learning via Static-Dynamic Conditional Disentanglement

Supplementary material

A. Proofs and Detailed Derivations

A.1. Derivation of the Per-Sequence and Dataset ELBO

The Evidence lower bound (ELBO) provides a lower estimate of the marginal data log-likelihood $\log p(\mathbf{x}_{1:T})$. Given the joint distribution $p(\mathbf{x}_{1:T}, \mathbf{f})$ and the approximate posterior distribution $q(\mathbf{f}|\mathbf{x}_{1:T})$, the ELBO can be computed as:

$$\log p(\mathbf{x}_{1:T}) \tag{1}$$

$$\geq \mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} \log \frac{p(\mathbf{x}_{1:T}, \mathbf{f})}{q(\mathbf{f}|\mathbf{x}_{1:T})} \tag{2}$$

$$= \mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} \log \frac{p(\mathbf{f})p(\mathbf{x}_{1:T}|\mathbf{f})}{q(\mathbf{f}|\mathbf{x}_{1:T})} \tag{3}$$

$$= \mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}) - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{4}$$

$$= \mathbb{E}_{q(\mathbf{f}|\mathbf{x}_{1:T})} \log p(\lambda_{1:T}|\mathbf{f}) + \log \left| \det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T}; \mathbf{f})}{\delta \mathbf{x}_{1:T}} \right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{5}$$

where in the last line we model $p(\mathbf{x}_{1:T}|\mathbf{f})$ using a NF, $\mathbf{x}_{1:T} = \mathbf{h}(\lambda_{1:T}; \mathbf{f})$. This proof is adapted from the standard VAE framework [2, 9]. This expression constitutes the per-sequence ELBO.

However, in our paper, we consider two important modifications from that original ELBO. We replace $p(\lambda_{1:T}|\mathbf{f})$ with $p(\lambda_{1:T})$ and condition the NF on the shuffled static vectors instead of the aggregated one. We show how these modifications still lead to a valid lower bound. We start with the dataset ELBO and further note that, because the frame static vectors are extracted independently, $q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T}) = \prod_t q(\mathbf{f}_t|\mathbf{x}_t) = q(\mathbf{f}|\mathbf{x}_{1:T})$:

$$\mathbb{E}_{p_{Data}} \log p(\mathbf{x}_{1:T}) \tag{6}$$

$$\geq \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} \log p(\lambda_{1:T}|\mathbf{f}_{1:T}) + \log \left| \det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T}; \mathbf{f}_{1:T})}{\delta \mathbf{x}_{1:T}} \right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{7}$$

$$= \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{8}$$

Let us denote $\mathbf{f}_{\pi_T}$, the randomly shuffled $\mathbf{f}_{1:T} \sim q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T})$. The ELBO can extended as:

$$\text{Eq.8} = \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}) + \underbrace{\log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})}}_{(*1)} - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{9}$$

We can note from Eq.8 that, to maximize the ELBO (with q fixed), the optimum value of the likelihood $\mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})$ is achieved for $KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}))$ minimized i.e., $p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) = q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})$.

As a result we have:

$$\mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) = \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) - KL(\log q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})|| \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})) \tag{10}$$

$$\Leftrightarrow \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) \geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) \tag{11}$$

$$\Leftrightarrow \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}) + \log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})} \geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) \tag{12}$$

$$\Leftrightarrow \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})} \geq \mathbb{E}_{q(\mathbf{f}_{1:T})} KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})) \tag{13}$$

with equality when $p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) = q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})$ and thus

$$\boxed{(*1) = \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})} \geq \mathbb{E}_{q(\mathbf{f}_{1:T})} KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||q(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})) \geq 0} \tag{14}$$

Injecting this into the ELBO gives:

$$\text{Eq.9} = \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}) + \log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})} - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{15}$$

$$\geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}) - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{16}$$

$$= \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\lambda_{1:T}|\mathbf{f}_{\pi_T}) + \log \left|\det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T};\mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{17}$$

$$= \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\lambda_{1:T}) + \underbrace{\log \frac{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})}{p(\lambda_{1:T})}}_{(*2)} + \log \left|\det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T};\mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{18}$$

As done previously, Eq.16 is maximized (with q fixed) for $KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}))$ minimized i.e., $q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) = p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})$. We thus have:

$$\mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}) = \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) - KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})) \tag{19}$$

$$\Leftrightarrow \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\lambda_{1:T}) + \log \frac{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})}{p(\lambda_{1:T})} + \log \left|\det \frac{\delta \mathbf{h}^{-1}}{\delta \mathbf{x}_{1:T}}\right| \geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) \tag{20}$$

$$\Leftrightarrow \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log \frac{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})}{p(\lambda_{1:T})} \geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} - \log \left|\det \frac{\delta \mathbf{h}^{-1}}{\delta \mathbf{x}_{1:T}}\right| + KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||p(\lambda_{1:T})) \tag{21}$$

with equality when $q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) = p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})$ meaning

$$\mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log \frac{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})}{p(\lambda_{1:T})} \geq \mathbb{E}_{q(\mathbf{f}_{1:T}),p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})} - \log \left|\det \frac{\delta \mathbf{h}^{-1}}{\delta \mathbf{x}_{1:T}}\right| + KL(p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T})) \tag{22}$$

$$\Rightarrow \boxed{(*2) = \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log \frac{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})}{p(\lambda_{1:T})} \geq \mathbb{E}_{q(\mathbf{f}_{1:T})} KL(p(\lambda_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T})) \geq 0} \tag{23}$$

As a result the ELBO becomes:

$$\text{Eq.18} = \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\lambda_{1:T}) + \log \frac{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})}{p(\lambda_{1:T})} + \log \left|\det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T};\mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{24}$$

$$\geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\lambda_{1:T}) + \log \left|\det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T};\mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right| - KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{25}$$

which is the loss optimized by our proposed model. **Overall, we demonstrated that the two proposed modifications still lead to a valid lower bound on the marginal data log-likelihood. In App.A.4, we show how that novel modified ELBO does naturally encourage disentanglement.**

A.2. Proof of Proposition 1

Proposition 1. *Consider the generative process and conditions presented in Sec.3. Further assume that dim*$(\mathbf{f}) = n_f \geq n_s$. *Let* $\mathbf{x}$ *and* $\mathbf{x}^*$ *denote two frames (or disjoint sets of frames) belonging to the same sequence* $\mathbf{x}_{1:T}$ *and* $\mathbb{D}$ *be a divergence between two distributions. Given unlimited data from* $p(\mathbf{x}_{1:T})$*, any candidate posterior distribution* $q(\mathbf{f}|\mathbf{x}_{1:T})$ *that minimizes*

$$L_{prop1} = \mathbb{E}_{p(\mathbf{x}_{1:T})}\mathbb{D}(q(\mathbf{f}|\mathbf{x}), q(\mathbf{f}|\mathbf{x}^*)) - I(\mathbf{f}, \mathbf{s}) + I(\mathbf{s}, \mathbf{s}), \tag{26}$$

is disentangled in the sense that $q(\mathbf{f}) = \int q(\mathbf{f}|\mathbf{x}_{1:T})p(\mathbf{x}_{1:T})d\mathbf{x}_{1:T}$*, the aggregated posterior, is a reparametrization of* $p(\mathbf{s})$.

Based on the generative process, the true data conditional likelihood $p(\mathbf{x}_t|\mathbf{z}_t)$ is fully specified by the smooth and invertible function $\mathbf{g}$. Consequently, the true posterior $p(\mathbf{s}|\mathbf{x}_t)$ is determined by $\mathbf{g}^{-1}_{1:n_s}$, i.e., the inverse of $\mathbf{g}$ restricted to the first n_s dimensions, for any t. Hence, It is assumed that the model family for the candidate posterior distribution $q(\mathbf{f}|\mathbf{x})$ is specified by the set of smooth functions $\mathbf{g}' : \mathcal{X} \to \mathcal{F}$ with $\mathcal{F} \subseteq \mathbb{R}^{n_f}$ and $\mathbf{x}$ a frame or set of frames from a sequence $\mathbf{x}_{1:T}$, $\mathbf{f} = \mathbf{g}'(\mathbf{x})$. To prove identifiability, we have to show that, for the $\mathbf{g}'$ associated to $q(\mathbf{f}|\mathbf{x})$, $\mathbf{g}'(\mathbf{x}) = \mathbf{m}(\mathbf{s})$ with $\mathbf{m} : \mathcal{S} \to \mathcal{F}$ a smooth and invertible function with smooth inverse.

One could note that Prop.1 is similar to what has been proposed to provably extract the content for content/style disentanglement in Thm.4.4 of [30]. This is not surprising since, as uncovered in the main paper, static/disentanglement is actually a content/style problem generalized to multiple temporally varying styles. As a result, both problems are equivalent and the codes are defined similarly. In fact, since the global code $\mathbf{s}$ is in both cases extracted as the common part, Thm.4.4 in [30] is also directly valid to extract the static factors for static/dynamic disentanglement. However, compared to the aforementioned theorem, Prop.1 extends Thm.4.4 of [30] to stochastic representations and, by leveraging mutual information, does not require the number of static variables n_s to be known apriori or impose a uniform distribution to the variables. Nonetheless, as it directly extends the work of Von Kügelgen et al. [30], the proof can be structured similarly.

The proof is constructed as follows : First we show that any candidate function that attains the global minimum of $\mathcal{L}_{\text{prop1}}$ outputs a representation that has maximum mutual information with $\mathbf{s}$ and is invariant across inputs of the same sequence i.e., $\mathbf{x}$ and $\mathbf{x}^*$. Second, leveraging the similarities of static/dynamic disentanglement with content/style disentanglement [4, 30], we use results from [30] to show that the obtained representation does not depend on the dynamic variables $\mathbf{d}_{1:T}$ and relate to $\mathbf{s}$ through a smooth function. Finally, we show that the latter function is invertible using the mutual information properties.

Step 1. As $\mathbb{D}$ is a divergence its minimum value is $\mathbb{D}(q(\mathbf{f}|\mathbf{x}), q(\mathbf{f}|\mathbf{x}^*)) = 0$ when both distributions are equal i.e., $\mathbf{g}'(\mathbf{x}) = \mathbf{g}'(\mathbf{x}^*)$. Moreover, from the properties of mutual information $I(\mathbf{f}, \mathbf{s}) \leq I(\mathbf{s}, \mathbf{s})$ with equality when $\mathbf{f}$ capture all information in $\mathbf{s}$ ($n_f \geq n_s$). The global minimum of $\mathcal{L}_{\text{prop1}} = 0$ is achieved when the mutual information is maximized.

Consider the function $\mathbf{b} = \mathbf{v} \circ \mathbf{g}^{-1}_{1:n_s} : \mathcal{X} \to \mathcal{F}$, the composition of $\mathbf{g}^{-1}_{1:n_s}$ described above and $\mathbf{v} : \mathcal{S} \to \mathcal{F}$ a smooth and invertible function with smooth inverse. $\mathbf{b}$, being a composition of two smooth invertible functions, is smooth and invertible. We can show that the function $\mathbf{b}$ attains the global minimum of $\mathcal{L}_{\text{prop1}}$:

$$\mathbf{b}(\mathbf{x}) = \mathbf{v}(\mathbf{s}) = \mathbf{b}(\mathbf{x}^*) \Rightarrow \mathbb{D}(q(\mathbf{f}|\mathbf{x}), q(\mathbf{f}|\mathbf{x}^*)) = 0, \tag{27}$$

$$I(\mathbf{b}(\mathbf{x}), \mathbf{s}) = I(\mathbf{v}(\mathbf{s}), \mathbf{s}) = I(\mathbf{s}, \mathbf{s}) \tag{28}$$

with Eq.27 resulting from the fact that $\mathbf{x}$ and $\mathbf{x}^*$ are frames from the same sequence and Eq.28 resulting from the invariance of MI to invertible transformations. This ensures the existence of a smooth function that attains the global minimum.

Let us now consider **any** smooth function $\mathbf{g}' : \mathcal{X} \to \mathcal{F}$ that achieves the global minimum of $\mathcal{L}_{\text{prop1}}$:

$$\mathcal{L}_{\text{prop1}} = \mathbb{E}_{p(\mathbf{x}_{1:T})}\mathbb{D}(q(\mathbf{f}|\mathbf{x}), q(\mathbf{f}|\mathbf{x}^*)) - I(\mathbf{f}, \mathbf{s}) + I(\mathbf{s}, \mathbf{s}) = 0 \tag{29}$$

All candidate functions can be written as $\mathbf{g}' = \mathbf{m} \circ \mathbf{g}^{-1}$ with $\mathbf{m} : \mathcal{Z} \to \mathcal{F}$ a smooth function that maps the true frame code $\mathbf{z}$ to $p(\mathbf{f})$. Because $\mathbf{g}(\mathbf{z}) = \mathbf{x}$ with $\mathbf{z} = (\mathbf{s}, \mathbf{d})$, Eq.29 results in:

$$\mathcal{L}_{\text{prop1}} = 0 \iff \begin{cases} \mathbf{m}(\mathbf{s}, \mathbf{d}) = \mathbf{m}(\mathbf{s}, \mathbf{d}^*) \\ I(\mathbf{m}(\mathbf{s}, \mathbf{d}), \mathbf{s}) = I(\mathbf{s}, \mathbf{s}). \end{cases} \tag{30}$$

This means that the representation $\mathbf{f} = \mathbf{m}(\mathbf{s}, \mathbf{d})$ must be invariant between frames belonging to the same sequence and have maximum mutual information with $\mathbf{s}$.

Step 2. We now show that $\mathbf{f} = \mathbf{g}'(\mathbf{x}) = \mathbf{m}(\mathbf{s}, \mathbf{d})$ does not depend on the true dynamic factors $\mathbf{d}$. As explained above, the static and dynamic codes are defined similarly to content/style in [30] and in both cases the same invariance condition is required i.e., $\mathbf{m}(\mathbf{s}, \mathbf{d}) = \mathbf{m}(\mathbf{s}, \mathbf{d}^*)$. We thus directly refer to Step 2 of Thm 4.2 in [30] that proved that, under the invariance condition and the generative process defined in Sec.3 of the main paper, $\mathbf{f}$ can only depend on the true static factors $\mathbf{s}$.

Step 3. Following previous steps, we have that $\mathbf{f} = \mathbf{m}(\mathbf{s})$ with $\mathbf{m}$ smooth. It remains to show that the function $\mathbf{m}$ is also invertible. From Eq.30, the representation must follow:

$$I(\mathbf{f}, \mathbf{s}) = I(\mathbf{m}(\mathbf{s}), \mathbf{s}) = I(\mathbf{s}, \mathbf{s}). \tag{31}$$

Mutual information is invariant under non-linear homeomorphisms [15] and $I(\mathbf{m}(\mathbf{s}), \mathbf{s}) = I(\mathbf{s}, \mathbf{s})$ iff $\mathbf{m}(\mathbf{s})$ is invertible (otherwise $I(\mathbf{m}(\mathbf{s}), \mathbf{s}) < I(\mathbf{s}, \mathbf{s})$). Hence, we finally obtain $\mathbf{f} = \mathbf{g}'(\mathbf{x}) = \mathbf{m}(\mathbf{s})$ with $\mathbf{m} : \mathcal{S} \to \mathcal{F}$ a smooth and invertible function. By the change of variables formula, we have:

$$q(\mathbf{f}) = p(\mathbf{m}^{-1}(\mathbf{f})) \left| \det \frac{\delta \mathbf{m}^{-1}(\mathbf{f})}{\delta \mathbf{f}} \right|, \tag{32}$$

showing that $q(\mathbf{f})$ is a reparametrization of $p(\mathbf{s})$, concluding the proof. The obtained representation $\mathbf{f}$ captures all and only the static factors $\mathbf{s}$ and is thus disentangled.

A.3. Proof of Proposition 2

Proposition 2. *Consider the same framework as in Prop.1 and the static code $\mathbf{f}$ to be disentangled as described above. Further assume that dim$(\lambda) = n_\lambda \geq n_d$. Any smooth and invertible candidate function $\mathbf{h}$ s.t. $\mathbf{x}_{1:T} = \mathbf{h}(\mathbf{f}, \lambda_{1:T})$ which minimizes*

$$L_{prop2} = I(\mathbf{f}, \lambda_{1:T}), \tag{33}$$

is disentangled in the sense that $q(\lambda_{1:T})$ is a reparametrization of $p(\epsilon_{d,1:T})$.

In order to prove that $\lambda_{1:T}$ identifies the true dynamic exogeneous variables $\epsilon_{d,1:T}$, we start by noting that, following the assumptions and generative process, the frame codes $\mathbf{x}_{1:T}$ can be written as

$$\mathbf{x}_t = \mathbf{g}(\mathbf{s}, \mathbf{h}'_d(\mathbf{s}, \epsilon_{d,<t}, \epsilon_{d,t})) = \mathbf{w}(\mathbf{s}, \epsilon_{d,\leq t}), \tag{34}$$

where $\mathbf{w}$ denote an invertible and smooth function with smooth inverse that represents the composition of the causal process and the successive invertible mappings. Hence, by denoting $\mathbf{w}' = \mathbf{w}^{-1} \circ \mathbf{h}$ and considering the fact that $\mathbf{h}$ generates the frames auto-regressively, we obtain from proposition 2:

$$\mathbf{x}_t = \mathbf{h}(\mathbf{f}, \lambda_{\leq t}) \tag{35}$$

$$\Leftrightarrow \quad (\mathbf{s}, \epsilon_{d,\leq t}) = \mathbf{w}'(\mathbf{f}, \lambda_{\leq t}) \tag{36}$$

with $\mathbf{w}'$ a smooth and invertible function. There is an invertible mapping between the codes and the ground-truth factors. By assumption, $\mathbf{f}$ is disentangled and thus, following the development in Sec.A.2, $\mathbf{f} = \mathbf{m}(\mathbf{s})$ with $\mathbf{m}$ smooth and invertible. Hence, as claimed in the main paper, by using an invertible mapping $\mathbf{h}$ to model the frames, if $\mathbf{f}$ is disentangled, the dynamic codes $\lambda_{1:T}$ have to be informative and encompass at least $\epsilon_{d,1:T}$ (since $\mathbf{w}'$ is invertible). It remains to show that the dynamic codes $\lambda_{1:T}$ are not dependent on the static factors $\mathbf{s}$.

By the definition of exogeneous variables we have :

$$I(\mathbf{s}, \epsilon_{d,1:T}) = KL(p(\mathbf{s}, \epsilon_{d,1:T}) || p(\mathbf{s})p(\epsilon_{d,1:T})) = 0. \tag{37}$$

Since $\mathbf{w}'$ and $\mathbf{m}$ are invertible, we can make the change of variables :

$$p(\mathbf{s}) = q(\mathbf{f}) \left| \det \frac{\delta \mathbf{m}(\mathbf{s})}{\delta \mathbf{s}} \right|, \tag{38}$$

$$p(\mathbf{s}, \epsilon_{d,1:T}) = q(\mathbf{f}, \lambda_{1:T}) \left| \det \frac{\delta \mathbf{w}'^{-1}(\mathbf{s}, \epsilon_{d,1:T})}{\delta \mathbf{s}, \epsilon_{d,1:T}} \right|. \tag{39}$$

Eq.39 can be further simplified. Indeed, because $\mathbf{f}$ is assumed disentangled, it only depends on the static factors $\mathbf{s}$ and thus $\mathbf{w}'^{-1}(\mathbf{s}, \epsilon_{d,1:T})$ restricted to the first n_f output dimensions must be equal to $\mathbf{m}$:

$$\mathbf{w}'^{-1}(\mathbf{s}, \epsilon_{d,1:T})_{1:n_f} = \mathbf{m}(\mathbf{s}) = \mathbf{f}. \tag{40}$$

This means the Jacobian term in Eq.39 can be factorized as :

$$\left|\det \frac{\delta \mathbf{w}'^{-1}(\mathbf{s}, \epsilon_{d,1:T})}{\delta \mathbf{s}, \epsilon_{d,1:T}}\right| = \left|\det \frac{\delta \mathbf{m}(\mathbf{s})}{\delta \mathbf{s}}\right| \left|\det \frac{\delta \mathbf{a}(\mathbf{s}, \epsilon_{d,1:T})}{\delta \epsilon_{d,1:T}}\right| \tag{41}$$

with $\mathbf{a}$ denoting $\mathbf{w}'^{-1}$ restricted the last output dimensions corresponding to $\lambda_{1:T}$.

Next, we use the fact that proposition 2 imposes the mutual information $I(\mathbf{f}, \lambda_{1:T})$ to be minimized. The global minimum is achieved when $I(\mathbf{f}, \lambda_{1:T}) = 0$ meaning $q(\mathbf{f}, \lambda_{1:T}) = q(\mathbf{f})q(\lambda_{1:T})$. Overall, Eq.39 can thus be rewritten as :

$$p(\mathbf{s}, \epsilon_{d,1:T}) = q(\mathbf{f})q(\lambda_{1:T}) \left|\det \frac{\delta \mathbf{m}(\mathbf{s})}{\delta \mathbf{s}}\right| \left|\det \frac{\delta \mathbf{a}(\mathbf{s}, \epsilon_{d,1:T})}{\delta \epsilon_{d,1:T}}\right| \tag{42}$$

Finally, by injecting Eq.38 and 42 in Eq.37 and then simplifying the common terms, we obtain

$$KL(p(\mathbf{s}, \epsilon_{d,1:T})||p(\mathbf{s})p(\epsilon_{d,1:T})) = 0 \tag{43}$$

$$\Leftrightarrow \quad KL(q(\lambda_{1:T}) \left|\det \frac{\delta \mathbf{a}(\mathbf{s}, \epsilon_{d,1:T})}{\delta \epsilon_{d,1:T}}\right| ||p(\epsilon_{d,1:T})) = 0 \tag{44}$$

$$\Leftrightarrow \quad q(\lambda_{1:T}) \left|\det \frac{\delta \mathbf{a}(\mathbf{s}, \epsilon_{d,1:T})}{\delta \epsilon_{d,1:T}}\right| = p(\epsilon_{d,1:T}) \tag{45}$$

with the last equation being the change of variable formula. $\lambda_{1:T}$ does not encompass the static factors $\mathbf{s}$ and we obtain that $\lambda_{1:T} = \mathbf{a}^{-1}(\epsilon_{d,1:T})$:

$$q(\lambda_{1:T}) = p(\epsilon_{d,1:T}) \left|\det \frac{\delta \mathbf{a}^{-1}(\lambda_{1:T})}{\delta \lambda_{1:T}}\right|, \tag{46}$$

showing that $q(\lambda_{1:T})$ is a reparametrization of $p(\epsilon_{d,1:T})$, concluding the proof. If the static code $\mathbf{f}$ is disentangled, by imposing the independence between the codes, we enforce the Jacobian of $\mathbf{w}'$ to be block-diagonal with blocks matching the codes and the true factors.

A.4. Proof of Proposition 3

Proposition 3. *Consider the model proposed in Sec.4.2 of the main paper and further assume $\alpha \gg \beta$. Minimizing*

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} - \alpha \left(\log p(\lambda_{1:T}) + \log \left|det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T}; \mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right|\right) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{47}$$

is equivalent to minimize both L_{prop1} and L_{propr2}. The model satisfies both Prop.1 and 2.

We start by noting that $\mathcal{L}$ is the negative dataset ELBO obtained in Eq.25 where, following the approach in [13] as it is common for VAE approaches, the likelihood and divergence terms of the VAE have been weighted using different coefficients. We propose to further develop $\mathcal{L}$ as :

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} - \alpha \left(\log p(\lambda_{1:T}) + \log \left|\det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T}; \mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right|\right) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{48}$$

$$= \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} - \alpha \left(\log p(\lambda_{1:T}|\mathbf{f}_{\pi_T}) + \log \frac{p(\lambda_{1:T})}{p(\lambda_{1:T}|\mathbf{f}_{\pi_T})} + \log \left|\det \frac{\delta \mathbf{h}^{-1}(\mathbf{x}_{1:T}; \mathbf{f}_{\pi_T})}{\delta \mathbf{x}_{1:T}}\right|\right) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{49}$$

$$\geq \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} - \alpha \log p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}) + \alpha KL(p(\lambda_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T})) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{50}$$

$$= \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} - \alpha \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) + \alpha \log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})} + \alpha KL(p(\lambda_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T})) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{51}$$

$$\geq \mathbb{E}_{q(\mathbf{f}_{1:T}, \mathbf{x}_{1:T})} - \alpha \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) + \alpha KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||q(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})) + \alpha KL(p(\lambda_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T})) + \beta KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) \tag{52}$$

The two inequalities result from Eq.23 and 14 subsequently. It can be observed that, compared to the classical VAE loss, the two modifications made i.e, replace $p(\lambda_{1:T}|\mathbf{f}_{\pi_T})$ with $p(\lambda_{1:T})$ and condition on the shuffled static vectors $\mathbf{f}_{\pi_T}$, cause the apparition of two extra divergence terms $KL(p(\lambda_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T}))$ and $KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||q(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T}))$ respectively. We will now develop the different terms in Eq.52. We note that, because $q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T}) = \prod_t q(\mathbf{f}_t|\mathbf{x}_t)$ and $\mathbf{f}_{\pi_T}$ are the shuffled $\mathbf{f}_{1:T}$, we have $q(\mathbf{f}_{\pi_T}|\mathbf{x}_{1:T}) = q(\mathbf{f}_{1:T}|\mathbf{x}_{\pi_T})$ and $q(\mathbf{f}_{\pi_T}) = q(\mathbf{f}_{1:T})$.

1.
$$\begin{aligned}\mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T}) &= \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}) + \log \frac{p(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})}{p(\mathbf{x}_{1:T})} \\ &\geq \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}) + KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||q(\mathbf{x}_{1:T})) \\ &= \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} \log p(\mathbf{x}_{1:T}) + I(\mathbf{f}, \mathbf{x}_{1:T}) \end{aligned} \tag{53}$$

2.
$$\begin{aligned}\mathbb{E}_{q(\mathbf{f}_{1:T})} KL(q(\mathbf{x}_{1:T}|\mathbf{f}_{1:T})||q(\mathbf{x}_{1:T}|\mathbf{f}_{\pi_T})) &= \mathbb{E}_{q(\mathbf{x}_{1:T})} KL(q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T})||q(\mathbf{f}_{\pi_T}|\mathbf{x}_{1:T})) \\ &= \mathbb{E}_{q(\mathbf{x}_{1:T})} KL(q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T})||q(\mathbf{f}_{1:T}|\mathbf{x}_{\pi_T})) \end{aligned} \tag{54}$$

3.
$$\mathbb{E}_{q(\mathbf{f}_{1:T})} KL(p(\lambda_{1:T}|\mathbf{f}_{\pi_T})||p(\lambda_{1:T})) \approx I(\lambda_{1:T}, \mathbf{f}_{\pi_T}) \tag{55}$$

4.
$$\mathbb{E}_{q(\mathbf{x}_{1:T})} KL(q(\mathbf{f}|\mathbf{x}_{1:T})||p(\mathbf{f})) = KL(q(\mathbf{f})||p(\mathbf{f})) + I(\mathbf{f}, \mathbf{x}_{1:T}) \tag{56}$$

By substituting those expressions in Eq.52, we get:

$$\begin{aligned}\text{Eq.52} \gtrsim \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} &- \alpha \log p(\mathbf{x}_{1:T}) - (\alpha - \beta) I(\mathbf{f}, \mathbf{x}_{1:T}) + \alpha KL(q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T})||q(\mathbf{f}_{1:T}|\mathbf{x}_{\pi_T})) \\ &+ \alpha I(\lambda_{1:T}, \mathbf{f}_{\pi_T}) + \beta KL(q(\mathbf{f})||p(\mathbf{f})) \end{aligned} \tag{57}$$

If we additionally consider that between the frames $\mathbf{x}_{1:T}$ only the dynamic variables $\epsilon_{1:T}$ vary, we have for $\mathbb{E}_{q(\mathbf{x}_{1:T})} KL(q(\mathbf{f}_{1:T}|\mathbf{x}_{1:T})||q(\mathbf{f}_{1:T}|\mathbf{x}_{\pi_T})) \approx I(\mathbf{f}, \epsilon_{1:T}|\mathbf{s}) = I(\mathbf{f}, \epsilon_{1:T})$. Henceforth, we finally have:

$$\begin{aligned}\mathcal{L} \gtrsim \mathbb{E}_{q(\mathbf{f}_{1:T},\mathbf{x}_{1:T})} &- \alpha \log p(\mathbf{x}_{1:T}) \underbrace{- (\alpha - \beta) I(\mathbf{f}, \mathbf{s}) + \beta \sum_t KL(q(\mathbf{f}_t|\mathbf{x}_t)||q(\mathbf{f}_t|\mathbf{x}_{\pi(t)}))}_{L_{prop1}} \\ &\underbrace{+ \alpha I(\lambda_{1:T}, \mathbf{f}_{\pi_T})}_{L_{prop2}} + \beta KL(q(\mathbf{f})||p(\mathbf{f})) \end{aligned} \tag{58}$$

which incorporates L_{prop1} and L_{prop2} if $\alpha > \beta$.

By imposing, during training, $p(\lambda_{1:T}|\mathbf{f}_{1:T}) = p(\lambda_{1:T})$, the model enforces $\lambda_{1:T}$ to be non informative and thus L_{prop2} while conditioning the cNF using the shuffled static vectors results in L_{prop1}.

Overall, by minimizing $\mathcal{L}$, in addition to learn a representation of the data and the prior as in a classical VAE, we also directly minimize an upper bound on L_{prop1} and L_{prop2}. Because the model parametrizes both the posterior $q(\mathbf{f}|\mathbf{x}_{1:T})$ and the invertible transformation h, the model satisfies Prop.1 and 2 and is provably disentangled with the factors being identified. This is achieved through a simple shuffle operation on the static vectors and by imposing $p(\lambda_{1:T})$ after the cNF. This concludes our proof.

B. Experimental Setup

B.1. Datasets

In this section, we describe the different datasets. A summary of the factors of variations for each dataset is provided in Tab.1 and examples of sequences are provided in Fig.1.

LPCSprites [25] is a dataset made up of image sequences of animated cartoon characters. Each video is composed of 8 RGB frames at a resolution of 64×64. The dynamic factors include three possible motions: walking, casting spells and slashing. The static factors consist of each character's type and color of body, shirt, pants and hair. This differs from the original dataset as the character identity is not entirely defined by the colors. In addition, compared to previous sequential disentanglement works [2, 3, 12, 16, 23, 35], because it is constant throughout the sequences, the orientation of the characters is also considered to be a static factor. The orientation conditions the motion due to the fact that the action does not appear the same way depending on the side the character is facing (e.g. slashing is made with different hands). We use in total 20000 sequences for training and 5000 for testing.

Dataset	Static factors	Dynamic factors
LPCSprites	orientation, type+color of body, shirt, pants and hair	action (walking, slashing or casting)
MUG	subject identity	facial expression
MHAD	subject identity	person's pose
MPI3D	camera position, background light, object size/shape/color	vertical and horizontal positions
s-dSprites	object color, shape and size	X-Y position
c-dSprites	object color, shape and size	X-Y position and rotation

Table 1. Summary of the factors of variations for the datasets. In blue are highligted the static factors that condition the dynamic ones.

MUG [1] is a facial expression dataset that comprises video sequences of 52 subjects performing six different expressions: anger, disgust, fear, happiness, sadness and surprise. Each sequence is made up of 50 to 160 frames. Following [2], we randomly sample lenght-15 clips from the original videos and crop the face region using Haar Cascades face detection before resizing to 64×64. There are in total 3429 samples, 25% of which are kept for the test set.

MHAD [8] UTD Multimodal Human Action Dataset is a real world action recognition dataset that comprises 861 video sequences, captured using a Microsoft Kinect camera at resolution of 640×480 pixels, of 8 subjects (4 females and 4 males) performing 27 different actions including waving, sitting, throwing, walking etc. Each subject repeated each action 4 times. Similarly to MUG, we randomly sample length-15 clips from the original videos and crop at the subject position before resizing to 96×96.

MPI3D [11] is a real-world dataset made up of images of a robot arm carrying different 3D printed objects at different positions. The dataset consists of 1036800 RGB images at resolution 64×64, corresponding to all the possible combinations of the following factors: camera position, background light, horizontal/vertical position, object color/shape/size. Length-16 clips are created by sampling subsequent frames from the dataset. The static factors are the camera position, the background light and the object color/shape/size. The dynamic factors (i.e., the motion) is the horizontal/vertical position that varies with time. The motion is made conditional to some of the static factors. If the arm is carrying a large object, it is allowed to have a smaller maximum speed. If the background light is purple, sea green or salmon, the robot arm is only allowed to move vertically, horizontally or both. Finally, the camera position also influences how the motion appears on the frames. We use in total 20000 sequences for training and 5000 for testing.

dSprites [21] is a synthetic dataset of 2D shapes that comprises 737280 64×64 images, corresponding to all the possible combinations of the following factors: shape, size, orientation, position X, position Y. We also add the color as a supplementary factor of variation. Video clips of length 16 are created by sampling subsequent frames. The static factors encompass the object color, shape and size. For the motion, we use two different variants. For the first one, *s-dSprites*, the orientation is fixed and the X/Y-motion of the objects is sampled independently from the static factors. For the second one, *c-dSprites*, the objects are also allowed to rotate. The rotation is conditioned on the shape due to the different order of rotational symmetry between the different shapes. Moreover, larger objects have lower maximum speeds and bounce on the edges according to their size. Each of the two variants consists of 20000 sequences for training and 5000 for testing.

B.2. Disentanglement Metrics

Accuracy (Acc) measures how well the factors associated with a fixed latent code are preserved when sampling the other codes. A classifier, pre-trained with full-supervision on the same train and test sets as the model, predicts the factors based on the reconstructed frames after sampling. For each categorical factor of the dataset, it outputs the probability measures and the predicted labels which are compared to the labels that should be obtained for a perfectly disentangled model.

Inception Score (IS) measures the quality of images created by the generative model. Based on the conditional predicted label distribution $p(y|\mathbf{x}_{1:T})$ from the classifier and the marginal predicted label distribution $p(y)$, the inception score is defined as : $IS = \exp(\mathbb{E}_{p(\mathbf{x}_{1:T})}[KL(p(y|\mathbf{x}_{1:T})||p(y))])$.

Inter-Entropy ($H(y)$) measures the entropy of the marginal predicted label of the generated sequences. Hence, it estimates the diversity for the sampled factors in the generated sequences. It is computed using $p(y)$ obtained from the classifier outputs.

Intra-Entropy ($H(y|x)$) measures the entropy of the conditional predicted label of the generated sequences. Hence, it

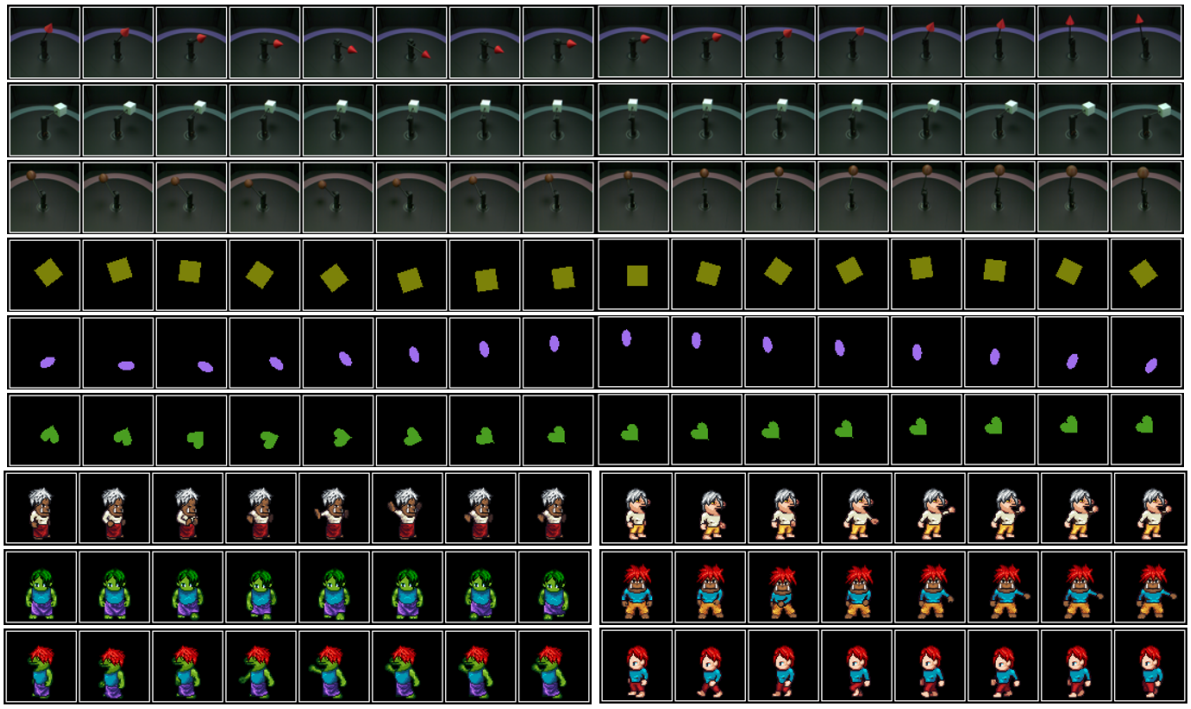

Figure 1. Examples of test sequences from *MPI3D* (row 1-3), *c-dSprites* (row 4-6) and *LPCSprites* (row 7-9 in two columns).

estimates the confidence of the classifier on the predictions. It is computed using $p(y|\mathbf{x}_{1:T})$ obtained from the classifier outputs.

In the case where the dynamic variables are causally dependent on the static ones, the accuracies for the dynamic factors cannot be computed directly. Indeed, due to the causal relationship, the dynamic factors might change when sampling a new static code. Following the causal model, we can only ensure the dynamic factors are preserved if the conditional static factors are kept fixed. As an example, for the *MPI3D* dataset, if the background is green or purple, the robot arm is only able to move horizontally or vertically respectively. Therefore, if the original sequence has a purple background (move vertically) and the new sampled static factors impose a green background, then the new generated frames should have a different motion and move according to the causal mechanism i.e., horizontally. The model should capture the causal mechanism. Following this remark, to compute the disentanglement metrics for the dynamic factors we thus restrict the samples of the static codes to the ones with the same conditional static factors (i.e., the blue ones in Tab.1) and check that the dynamic factors are preserved when sampling the independent static factors.

B.3. Architecture and Hyperparameters

The proposed model is implemented in PyTorch [24] and is trained using the Adam optimizer [14] with a learning rate of 0.001. We use a batch size of 64 for all datasets except *MHAD* and *MUG* where we use batches of size 16. The model is trained until convergence for up to 500 epochs.

As it is common practice for VAEs [13], we add coefficients to the different loss terms in order to weight their influence. As an example, by weighting the disentanglement constraints or the KL-divergence terms, we can improve disentanglement and control the information bottleneck of the VAE. This principle is what leads to our provably disentangled model as shown in Sec.A.4. In previous static/dynamic sequential disentanglement works [2, 3, 12, 16, 23, 35], these hyperparameters are chosen through grid search and the models that provide the best validation accuracies are used for testing. However, by doing so, the loss coefficients are chosen using the full supervision and the labels of all factors. This is in contradiction with the fact that the method should be self-supervised and labels should only be used for testing. This ultimately results in a bias for the final performances, especially since disentanglement models have been found to be particularly sensitive to the choice of hyperparameters [18]. This problematic goes beyond the subject of this paper and is left for future work. Nonetheless, to alleviate this issue, instead of using dataset specific loss coefficients, we propose here to choose coefficients that work well

on all datasets. For α and β, we use the following coefficients respectively : 10.0, 0.5.

For the model architecture, the network is composed of the following blocks : Image encoder-decoder, Static encoder and conditional Normalizing Flow.

- **Image encoder-decoder** is in charge of encoding independently each frame into a low-dimensional space $\mathbf{x}_{1:T}$ and then reconstructing the videos. The encoder is a convolutional neural network with 5 layers of channels $[64, 128, 256, 512, 128]$. Each layer consists of a convolution with kernel of size 4 followed by a GroupNorm (8 groups) and SiLU activation function. A feature vector of size 128 is outputted for each frame. The decoder is also a convolutional neural network with 5 layers of channels $[512, 256, 128, 64, 3]$. Each layer encompasses a transposed convolution with kernel of size 4 followed by a GroupNorm (8 groups) and SiLu activation function except for the last layer with a Sigmoid. Following [17], this autoencoder is pretrained to encode/decode the frames without disentanglement and Gaussian noise is added on the frame feature vectors $\mathbf{x}_{1:T}$ to prevent collapse.
- **Static encoder** takes the frame codes $\mathbf{x}_{1:T}$ as input and outputs the static code $\mathbf{f}$. It is made of a MLP with 2 linear layers each followed by a SiLU activation function. Two final linear layers predict the mean and standard deviation of the static code. The MLP processes each frame independently and the predictions $\mathbf{f}_{1:T}$ are then aggregated following the approach in [4]. All layers are of dimension 128 and $n_f = 128$. The prior $p(\mathbf{f})$ for the static code $\mathbf{f}$ is learned using a Normalizing Flow which consists of 4 Affine Coupling layers [10] each made up of 3 linear layers with hidden size of 256. The final distribution of the NF is set to a standard Gaussian distribution with diagonal covariance matrix.
- **Conditional Normalizing Flow** models the frame latent codes likelihood $p(\mathbf{x}_{1:T}|\mathbf{f})$. It is composed of 5 Affine coupling layers similar to the ones used for the static prior except for the last layer which is conditioned on the previous frames in order to model the transitions auto-regressively in the same fashion as the AF layer in [20]. The static code is also concatenated to the input of each Affine Coupling layer and thus conditions the transformation. All layers have a hidden size of 256. The transformed latent codes correspond to the dynamic codes $\lambda_{1:T}$ and are of size $n_\lambda = 128$. The final distribution $p(\lambda_{1:T}) = p(\lambda_{1:T}|\mathbf{f})$ is parameterized by an LSTM similar to [2]. As uncovered in our paper, to achieve a well disentangled model, during training, instead of using the aggregated static code $\mathbf{f}$ we additionally make use of the static estimations before aggregation i.e., $\mathbf{f}_{1:T}$. We condition the cNF using the randomly shuffled $\mathbf{f}_{1:T}$. As such, even if $\mathbf{f}_t$ still encompasses dynamic information, due to the random shuffling, that information does not match the frame it conditions and the dynamic information is modeled by the cNF with $\lambda_{1:T}$ which leads to a disentangled model as proved in Sec.A.4.

C. Additional Results

C.1. Complete Tables of Results

Due to the limited space available, some categorical factors accuracies have not been reported into the tables of the main paper and are reported instead in Tab.2. Additionally, in order to also assess if continuous factors are preserved after sampling, we build a regression model based on the frames and then report the R2 score (coefficient of determination) following the same procedure than for the accuracies (Sec.B.2). A R2-score of 1. means that the continuous factors have been perfectly preserved after sampling. A model is disentangled if it achieves a perfect score for **all** factors.

In Tab.2 it can be observed that, compared to the baseline methods, our proposed model manages to properly disentangle all factors and learn the causal mechanisms. Some competing methods achieve higher scores for some factors, however this can be explained by the decrease in performance regarding the other factors. A model that encodes everything in its dynamic codes will trivially maintain the dynamic when sampling the static code since the latter is uninformative. As an example, for *c-dSprites*, the SPYL method [23] obtains a larger score for the rotation. However, this can be explained by the fact that the shape is wrongly encoded into the dynamic codes and the causal mechanism is discarded by the network; meaning the rotation is trivially preserved after sampling the static code. A similar justification can be made for the *MPI3D* and *LPCSprites* datasets. Overall, the SPYL method seems to be biased towards encoding information in its dynamic codes at the expense of its static code, which might explain why it achieves better performances on *MUG* and *MHAD* when comparing for the actions.

Discussion from the formalism perspective : Using our proposed definition of the factors it is possible to gain further understanding on the results of the baseline methods. These methods enforce simultaneously the extracted dynamic and static codes to be informative and mutually independent. As a result, when the factors are truly independent these constraints are equivalent to our proposed approach and they obtain a disentangled representation, albeit with much more complicated losses e.g. contrastive learning, adversarial training etc. However, by construction, they cannot disentangle when the factors

s-dSprites	Acc↑ (%) color	shape	size	R^2↑ x-pos	y-pos	IS↑	$H(y\|x)\downarrow$	$H(y)\uparrow$
DSVAE	79.38	42.10	43.61	0.852	0.920	3.035	0.038	1.17
CDSVAE	96.18	98.22	98.71	0.992	0.989	3.995	0.010	1.37
SPYL	97.76	98.72	98.12	0.998	0.985	3.685	9.5e-3	1.29
Ours	**98.91**	**98.98**	**98.76**	**0.999**	**0.990**	**4.263**	**5.9e-4**	**1.41**

c-dSprites	Acc↑ (%) color	shape	size	R^2↑ rot.	x-pos	y-pos	IS↑	$H(y\|x)\downarrow$	$H(y)\uparrow$
DSVAE	72.09	32.66	79.38	0.593	0.986	0.985	4.128	0.046	1.41
CDSVAE	96.33	74.32	93.17	0.127	0.986	0.985	4.141	0.028	1.39
SPYL	68.41	33.28	75.10	**0.956**	**0.995**	**0.993**	**4.392**	0.027	1.43
Ours	**98.92**	**98.98**	**98.59**	0.895	0.972	0.972	4.371	**0.008**	**1.44**

MPI3D	Acc↑ (%) shape	color	size	camera	R^2↑ light	vert.-pos	hori.-pos	IS↑	$H(y\|x)\downarrow$	$H(y)\uparrow$
DSVAE	28.29	85.51	65.57	98.81	87.71	0.307	0.312	3.505	0.056	1.19
CDSVAE	43.81	94.09	79.17	**99.99**	**99.94**	0.901	0.707	3.487	0.138	1.26
SPYL	16.78	59.19	50.29	33.32	86.54	**0.994**	**0.977**	3.819	0.043	1.28
Ours	**90.15**	**99.61**	**97.56**	99.97	99.93	0.892	0.878	**3.849**	**0.029**	**1.29**

LPCSprites	Acc↑ (%) body	hair	shirt	pants	orient.	action	IS↑	$H(y\|x)\downarrow$	$H(y)\uparrow$
CDSVAE	54.94	85.75	99.96	99.54	33.32	67.71	2.737	1.6e-3	1.00
SPYL	70.89	99.89	**100**	**99.74**	33.32	**99.86**	2.735	2.0e-3	**1.01**
Ours	**99.97**	**99.91**	99.99	99.63	**99.94**	99.83	**2.768**	**3.3e-4**	**1.01**

Table 2. Disentanglement metrics on *s-dSprites*, *c-dSprites*, *MPI3D* and *LPCSprites*, in red and blue are highligted respectively the dynamic factors and the static factors that condition the motion.

are dependent. The reason is that they generate the sequences by first sampling the codes from a **factorized** prior $p(\mathbf{f}, \lambda_{1:T}) = p(\mathbf{f})p(\lambda_{1:T})$ and then obtain the frames as $\mathbf{x}_t = h(\mathbf{f}, \lambda_t)$. Because the transitions between dynamic codes are not conditioned on $\mathbf{f}$ and $\mathbf{x}_t = h(\mathbf{f}, \lambda_t)$, in the best case scenario we can only have $I(\mathbf{f}, \lambda_t) = 0\ \forall t$ and $I(\mathbf{f}, \lambda_{1:T}) \geq 0$ meaning information is encoded in both codes and the model is not disentangled. Comparatively our method directly imposes $I(\mathbf{f}, \lambda_{1:T}) = 0$ and can thus disentangle the factors.

C.2. Shuffle Ablation and Varying Coeffecients

In order to confirm what has been demonstrated in Sec.A.4, we propose here to compare the performances of the model when ablating the shuffle operation and when varying the β coefficient. The results are provided in Tab.3 for the *c-dSprites* dataset.

By looking at Eq.58, the β coefficient controls the amount of information enforced in the static code $I(\mathbf{f}, \mathbf{s})$ and how strongly the static invariance is enforced $KL(q(\mathbf{f}_t|\mathbf{x}_t)||q(\mathbf{f}_t|\mathbf{x}_{\pi(t)}))$. On the other hand, the α coefficient controls how much the codes are independent $I(\lambda_{1:T}, \mathbf{f}_{\pi_T})$. Intuitively, at fixed α, if the β coefficient is increased (decreased), less (more) information is encouraged in the static code $\mathbf{f}$ and the invariance is more (less) enforced resulting in a representation where potentially no (all) information is encoded in $\mathbf{f}$ with $I(\lambda_{1:T}, \mathbf{f}_{\pi_T})$ trivially satisfied. These results are coherent with what can be observed in Tab.3 when varying β, confirming Prop.3 and Eq.58. As stated by Prop.3, β needs to be $\ll \alpha$ to avoid all information to go in $\lambda_{1:T}$ but past a certain point it becomes a trade off between having improved performances on the static or dynamic factors. The comparison between our proposal and β-VAE [13] constitutes also a very interesting case. In β-VAE, it is proposed to impose $\beta > \alpha$ to improve disentanglement, at the expense of reconstruction, by more strongly encouraging the

c-dSprites		Acc↑ (%) color	shape	size	R^2↑ rot.	x-pos	y-pos
Full model	$\beta = 0.1$	99.92	98.74	99.60	0.716	0.953	0.957
	$\beta = 0.5$	98.92	98.98	98.59	0.895	0.972	0.972
	$\beta = 1.0$	93.92	87.13	91.93	0.966	0.984	0.985
	$\beta = 2.0$	0.08	0.01	0.03	0.998	0.987	0.986
Ablated shuffle		99.94	94.01	96.36	0.0505	0.0561	0.278

Table 3. Disentanglement metrics on *c-dSprites* for varying β and with ablated shuffle operation. In red and blue are highligted respectively the dynamic factors and the static factors that condition the motion. The results boxed are from the model used on all datasets.

posterior to follow a factorized distribution. Conversely, our method does not rely on imposing a factorized distribution for disentanglement but instead makes use directly of the reconstruction term through a cNF. As a result, the disentanglement is instead favored by imposing $\beta < \alpha$ and we do not suffer from the same reconstruction vs disentanglement trade off.

When ablating the shuffle operation, Eq.58 is no longer valid. Instead of encouraging the static factors in $\mathbf{f}$ ($I(\mathbf{f}, \mathbf{s})$) we now encourage $\mathbf{f}$ to encode everything ($I(\mathbf{f}, \mathbf{x}_{1:T})$) with no invariance across frames while still enforcing the codes to be independent. As a result the static code $\mathbf{f}$ will encode dynamic information and $\lambda_{1:T}$ is encouraged to be non informative. The representation is not properly disentangled and constrained which translates into lower performances for the disentanglement metrics.

C.3. Representation Swapping

In Fig.2, 4, 5 and 6, we present several qualitative results when swapping the dynamic and static codes for the *MUG*, *LPCSprites*, *MPI3D* and *c-dSprites* datasets. In all examples, clean swaps are obtained and the static factors are correctly preserved. As discussed in Sec.B.2, the motion can only be perfectly preserved if the static factors that causally influence the motion are fixed. For the proposed model, it can be observed that when the conditional static factors are shared between the two test sequences, it results in swapped video clips that correctly preserve the motion. Conversely, when the conditional static factors are different, the swapped video clips exhibit motions that are close to the original dynamic but abide by the causal mechanism. This demonstrates that the model manages to learn the causal mechanism. As an example, for *c-dSprites* in Fig.6, when both test sequences share the same shape or size the swapped clip maintains the same rotations and positions respectively. If the objects have different sizes, the swapped clip shows positions that are close to the original driving dynamic code but bounce on the edges according to the new object size.

In Fig.3, we also provide a comparison between our proposed method and the competing CDSVAE model when swapping the dynamic and static codes for the *MHAD* dataset. *MHAD* is a much more challenging dataset with more complex motions at higher resolutions. Comparatively, our model achieves cleaner swap motion transfers and provides higher quality non blurry outputs while also better maintaining the identities. As expected however, the results still fall behind what can be achieved by recent human video reenactment SotA methods [5–7, 26–28, 32, 33]. Indeed those models rely on high-dimensional structured representations and external human specific signals such as skeletons. In contrast, disentanglement learning aims at obtaining a low-dimensional general representation useful for diverse downstream tasks where the feature dimensions capture the individual semantic concepts, which is different from high dimensional motion transfer tasks. As a result, it cannot preserve detailed spatial information so accurately and general purpose disentanglement of real, in-the-wild, data is still an unsolved problem. As our method manages to properly swap the motion, it shows that the bottleneck is in the decoder and future work may investigates better architectures to reconciliate low dimensional representations with high dimensional reconstructions. This limitation is intrinsic and shared by all the literature in the disentanglement domain [4, 13, 19, 22, 31, 34] and is out of the scope of this paper.

C.4. Representation Sampling

In Fig.8, 9 and 10, we present several qualitative results when sampling either the dynamic or static codes for the *LPCSprites*, *MPI3D* and *c-dSprites* datasets. The discussion is similar to Sec.C.3. On all datasets, the results demonstrate diverse generated factors while preserving the dynamic/static variables following the data causal generative process.

To highlight a potential advantage of the static/dynamic disentanglement for unconditional generation, we also compare in Fig.11 our proposed model with the same model but without a global static code (i.e., no disentanglement). It can be

Figure 2. Static/dynamic swap results for the MUG dataset. Odd rows are test input sequences while even rows are video clips generated by swapping the static and dynamic codes of the test sequences.

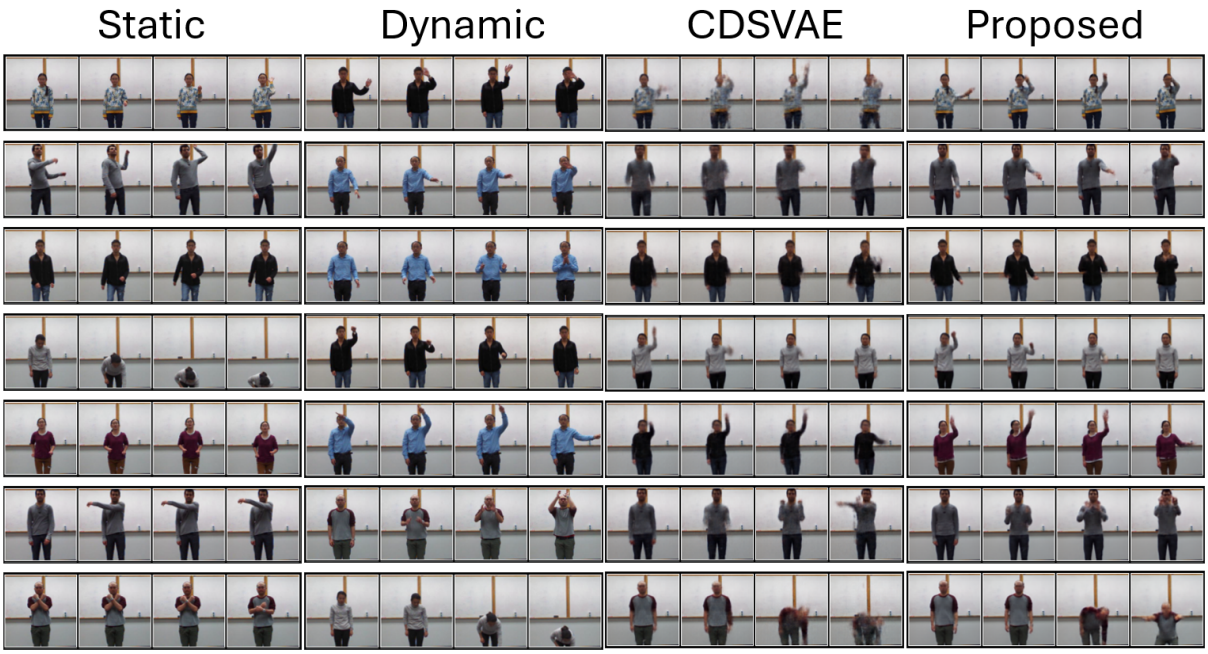

Figure 3. Static/dynamic swap results for the MHAD dataset. The first two columns are test input sequences while the last two columns are the video clips generated by swapping the static and dynamic codes of the test sequences using the competing CDSVAE model and our proposed method.

observed that by using a global static code for generating new sequences, the aspects of the shapes are better preserved across time steps. The colors, shapes and sizes are provided to the conditional Normalizing Flow via $\mathbf{f}$ to reconstruct each frame allowing to better maintain these factors and avoid flickering. This implies that adding a global code could help video generation problems in better maintaining the temporal coherence.

Figure 4. Static/dynamic swap results for the LPCSprites dataset. For each group of three sequences, the two first rows are test input sequences while the last row is the video clip generated by swapping the static and dynamic codes of the test sequences.

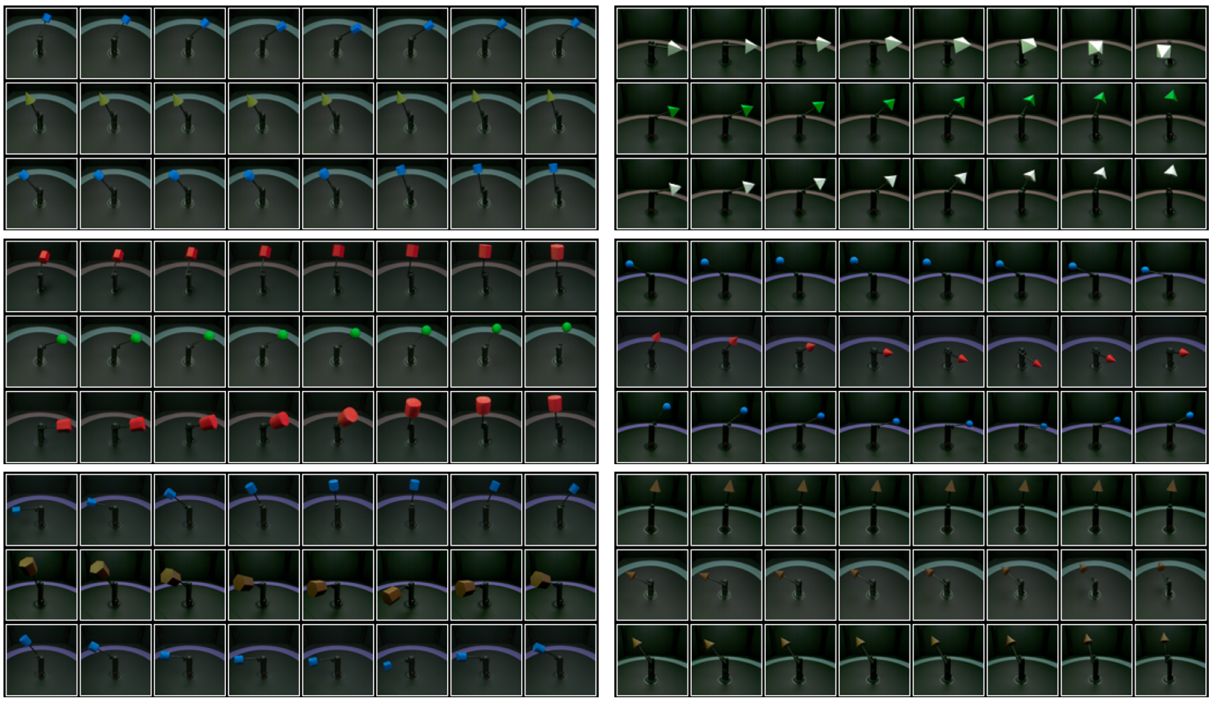

Figure 5. Static/dynamic swap results for the MPI3D dataset. For each group of three sequences, the two first rows are test input sequences while the last row is the video clip generated by swapping the static and dynamic codes of the test sequences.

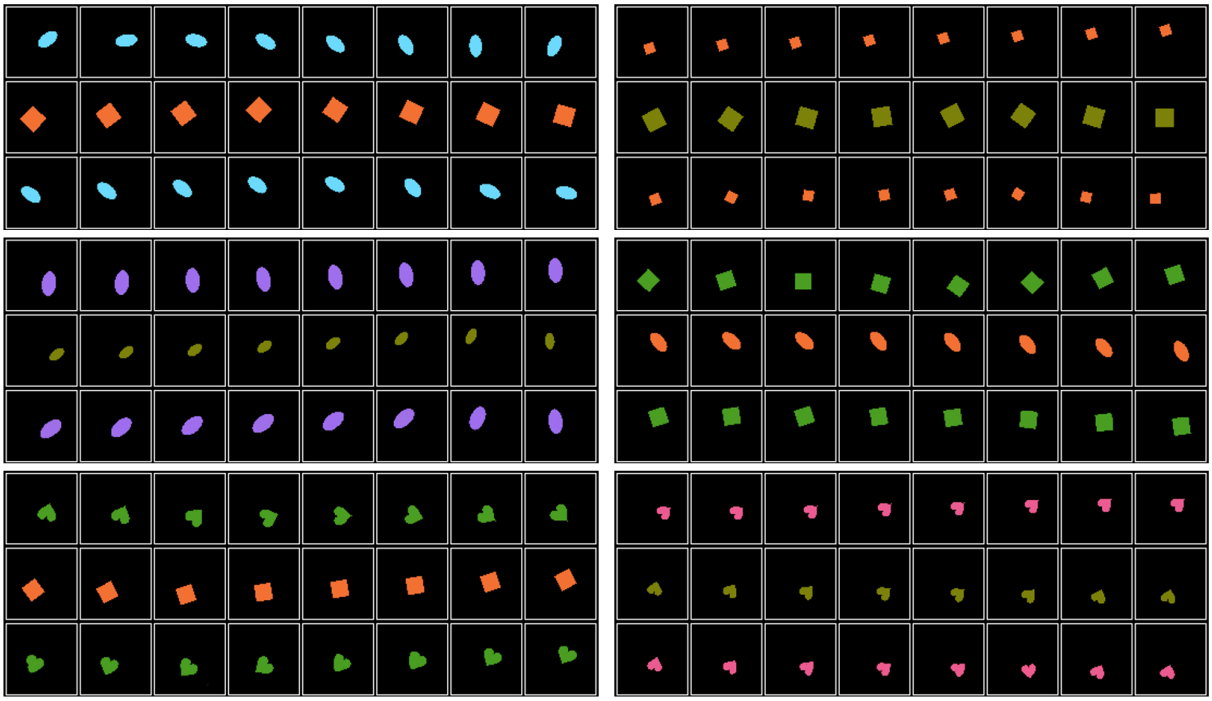

Figure 6. Static/dynamic swap results for the c-dSprites dataset. For each group of three sequences, the two first rows are test input sequences while the last row is the video clip generated by swapping the static and dynamic codes of the test sequences.

C.5. Interpolation in Latent Space

In Fig.12 and 13, we show results when interpolating between sequences in both the static and dynamic latent spaces. These figures demonstrate that the learned latent spaces are smooth and meaningful. In all examples, we can observe a smooth transition between the interpolated factors while the fixed factors are maintained, further demonstrating proper disentanglement.

C.6. Latent embedding visualization

In order to visualize the latent spaces, we plot in Fig.14 the 2D embedding of the static and dynamic codes, computed using t-SNE [29] on all *s-dSprites* test samples. For both the static and dynamic codes, we color the embedded points based on the static factors $\mathbf{s}$ i.e., the objects colors, shapes and sizes. We can observe that the static code $\mathbf{f}$ has a clear structure with clusters that correspond to all unique combinations of static factors. Conversely, the dynamic codes do not show any structure with the static factors and appear independent to $\mathbf{s}$. In Fig.15, the same experiment is repeated but with the embedded points colored based on the dynamic factors $\mathbf{d}_{1:T}$ i.e., the positions of the moving shapes and it is now only the dynamic codes that display a clear structure with respect to the factors. We thus conclude that the method subspaces are disentangled.

D. Details on the Limitations

In Sec.5 of the main paper, several limitations have been highlighted. These limits are not specific to our proposed model and are intrinsic to the two-factor static/dynamic disentanglement approach. Following the problem formulation in Sec.3, the static variables are defined as invariant in each sequence while the dynamic variables are defined as varying almost surely in the dataset (but not necessarily in every sequence). This has two consequences:

1. The static factors only capture the factors that are invariant in **all** sequences.
2. The static/dynamic factors are defined by the dataset itself.

For the first consequence, it means that by having a single static code $\mathbf{f}$, the network will only ever be able to capture in that code at maximum the factors that are static in all sequences. A factor static for a specific sequence but that sometimes may vary in others will be encoded as a dynamic factor. The factors are 'labeled' by the network as static or dynamic at the dataset

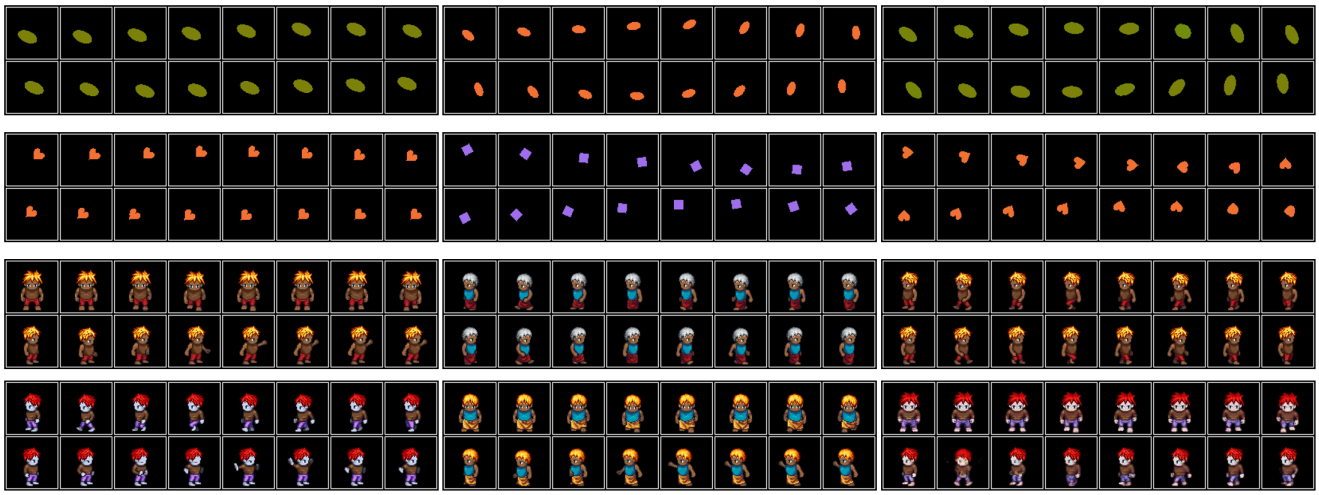

Figure 7. Illustration of the limitations of the two-factor static/dynamic disentanglement. The left and middle columns are test input sequences and the right column are video clips generated by swapping the static and dynamic codes of the test sequences.

level and not at the sequence level. As an illustrative example, in Fig.7 for *c-dSprites*, we take sequences with a fixed rotation and swap the dynamic codes. The resulting sequences have different rotations, meaning the shape angle is encoded in the dynamic code even if it was static for the specific sequence of origin.

For the second consequence, while it might be beneficial to have the network uncovering the factors and the causal relationships without supervision, it means that if one wants to specify a particular factor as dynamic instead of static then it restricts the datasets that can be used. As an illustrative example, in Fig.7, the network is trained with *LPCSprites* sequences where the cartoon characters are turning and only then the orientation of the characters is encoded as a dynamic factor. Overall it means the network **discovers** the static/dynamic factors **depending on the dataset**. This has also other major implications. For evaluation, the common benchmark involves the use of labels associated with the videos [2, 16, 35]. However, the notion of static/dynamic captured is intrinsically tied to the dataset and not to the labels. These labels are used for the metrics but might not provide the full picture of the possible dependencies and motions that are present in the dataset. This might explain the gap observed for the quantitative results when assessing on real-world datasets where the relation between the ground-truth generation process and labels is not tightly controlled unlike for the synthetic datasets. This highlights the need for future work to provide novel more robust evaluation processes. Finally, it also shows that altering the dataset e.g., with data augmentation like CDSVAE [2], actually defines what is static/dynamic, the augmentations affect the dataset and thus change what will be captured. Instead of disentangling the satic/dynamic of the data, the model will disentangle the augmentations. Similarly,using supervised learning with labels will define the disentanglement instead of discovering the true factors.

References

[1] Niki Aifanti, Christos Papachristou, and Anastasios Delopoulos. The mug facial expression database. *Int. Worksh. Image Analysis for Multimedia Interactive Services*, 2010. 7

[2] Junwen Bai, Weiran Wang, and Carla P Gomes. Contrastively disentangled sequential variational autoencoder. *Adv. Neural Inform. Process. Syst.*, 2021. 1, 6, 7, 8, 9, 15

[3] Nimrod Berman, Ilan Naiman, and Omri Azencot. Multifactor sequential disentanglement via structured koopman autoencoders. *arXiv preprint arXiv:2303.17264*, 2023. 6, 8

[4] Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. *AAAI*, 2018. 3, 9, 11

[5] Stella Bounareli, Christos Tzelepis, Vasileios Argyriou, Ioannis Patras, and Georgios Tzimiropoulos. One-shot neural face reenactment via finding directions in gan's latent space. *arXiv preprint arXiv:2402.03553*, 2024. 11

[6] Menglei Chai, Jian Ren, Aliaksandr Siarohin, Sergey Tulyakov, and Oliver Woodford. Motion representations for articulated animation, 2023.

[7] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A Efros. Everybody dance now. *Int. Conf. Comput. Vis.*, 2019. 11

[8] Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. *IEEE Int. Conf. Image Process.*, 2015. 7

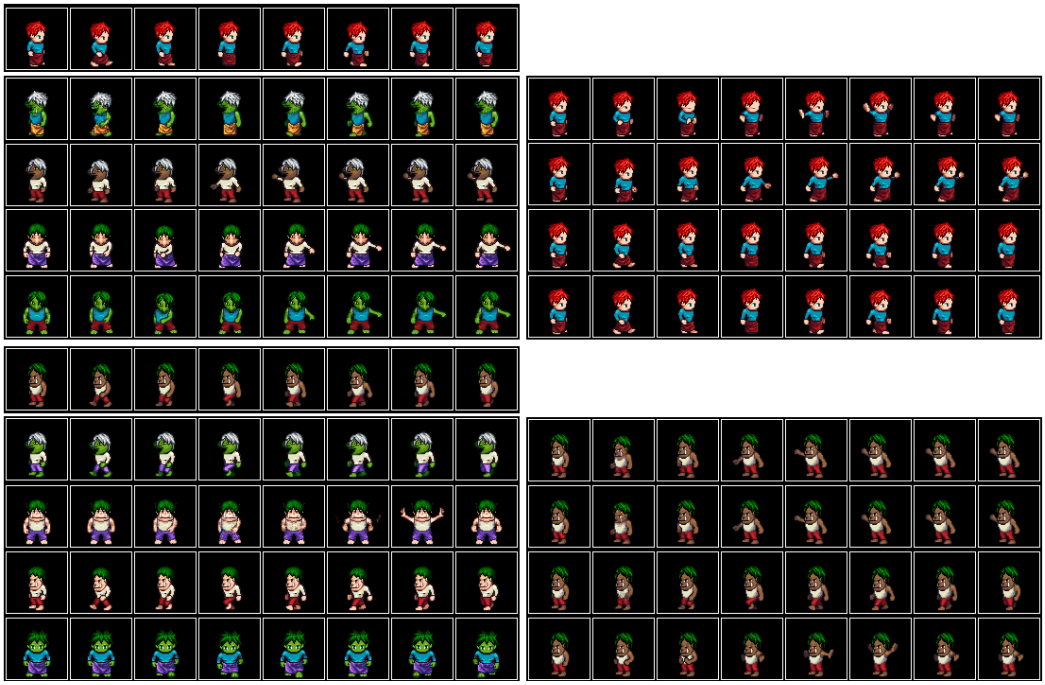

Figure 8. Static/Dynamic generation results for the LPCSprites dataset. Row 1 and 6 are test sequences. For the other rows, the left (right) column comprises video clips generated by fixing the dynamic (static) codes to the value given by the test sequence and sampling the static (dynamic) variables from the prior.

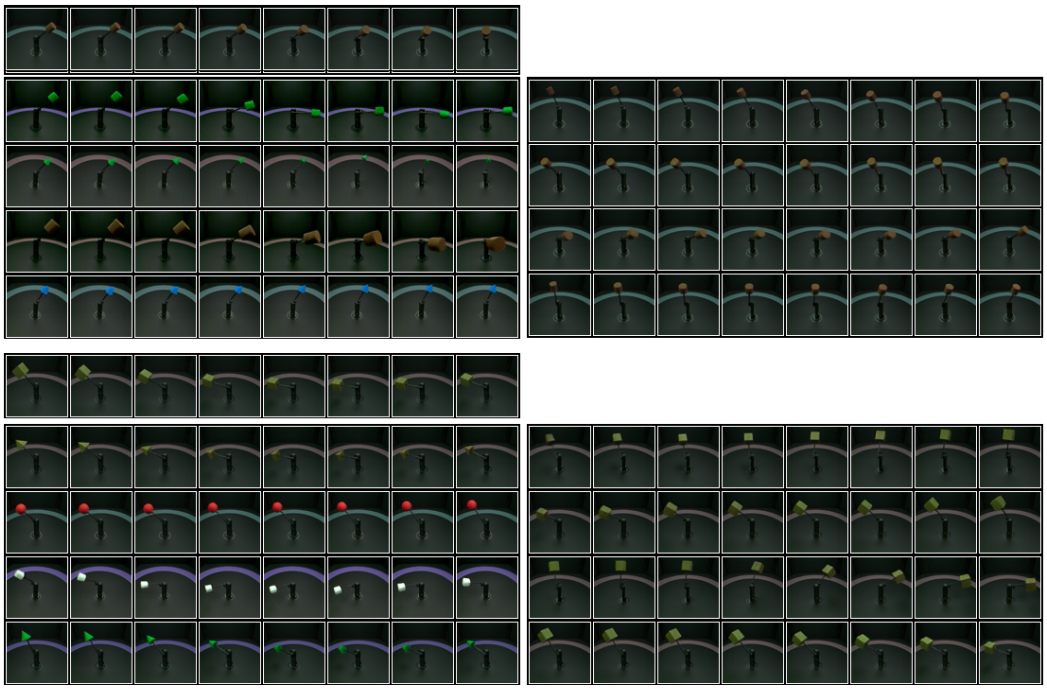

Figure 9. Static/Dynamic generation results for the MPI3D dataset. Row 1 and 6 are test sequences. For the other rows, the left (right) column comprises video clips generated by fixing the dynamic (static) codes to the value given by the test sequence and sampling the static (dynamic) variables from the prior.

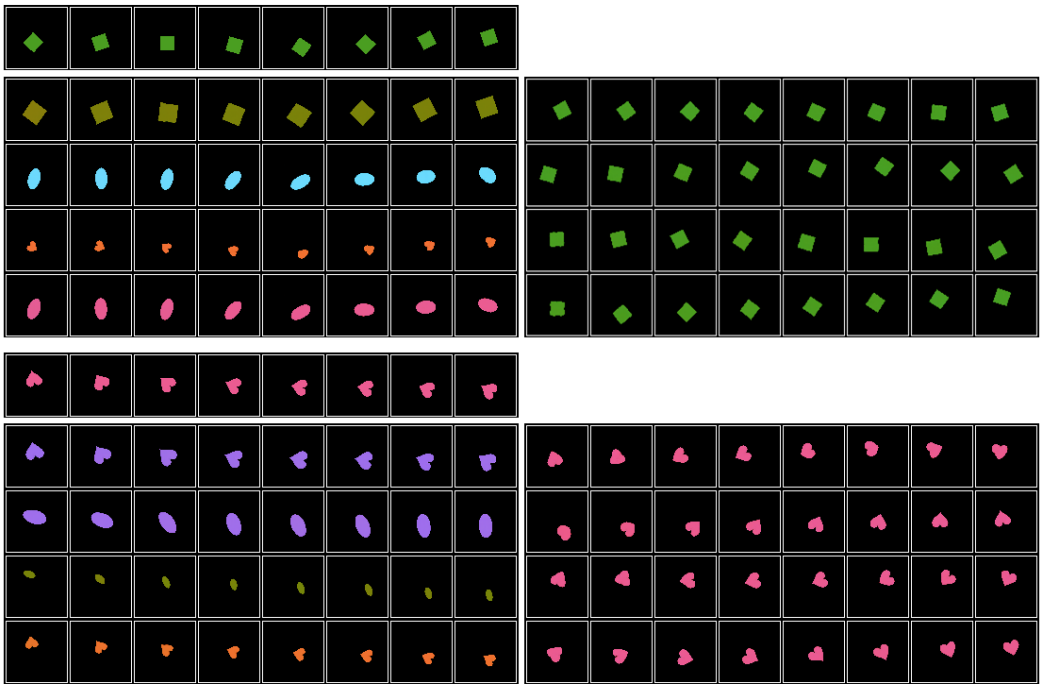

Figure 10. Static/Dynamic generation results for the c-dSprites dataset. Row 1 and 6 are test sequences. For the other rows, the left (right) column comprises video clips generated by fixing the dynamic (static) codes to the value given by the test sequence and sampling the static (dynamic) variables from the prior.

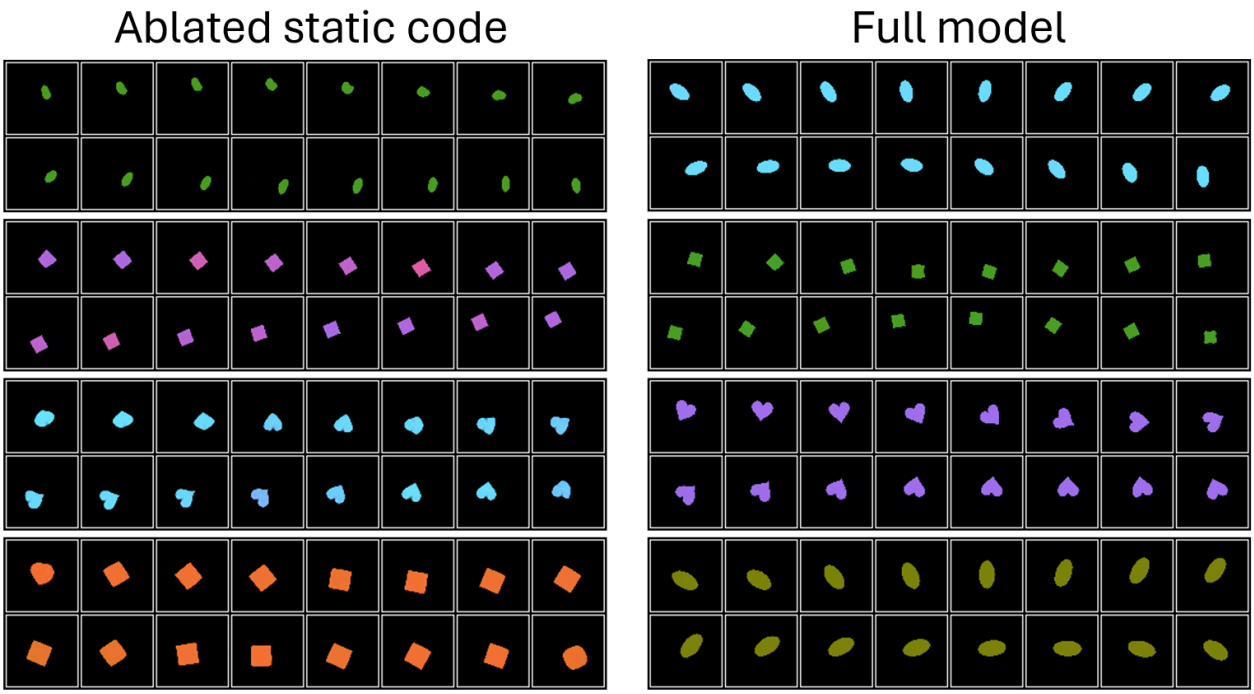

Figure 11. Unconditional generation results for the c-dSprites dataset. On the left are presented samples from the model without a global static code and on the right are shown results for the full model. Sequences generated with a global static code better maintain the shapes aspect across time steps.

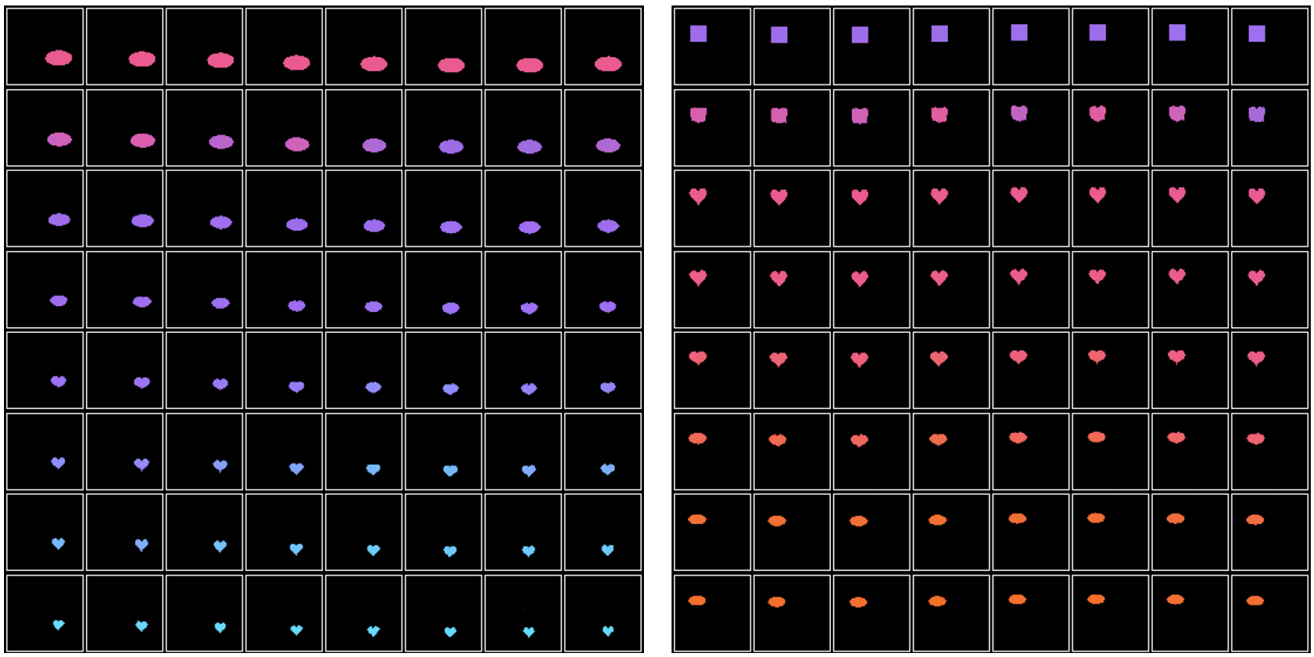

Figure 12. Interpolation in the latent space for s-dSprites. In each of the two columns, the first and last rows share the same dynamic codes $\lambda_{1:T}$ but have different static codes $\mathbf{f}$. The rows in between are generated by fixing the dynamic codes and interpolating the static one using cosine interpolation.

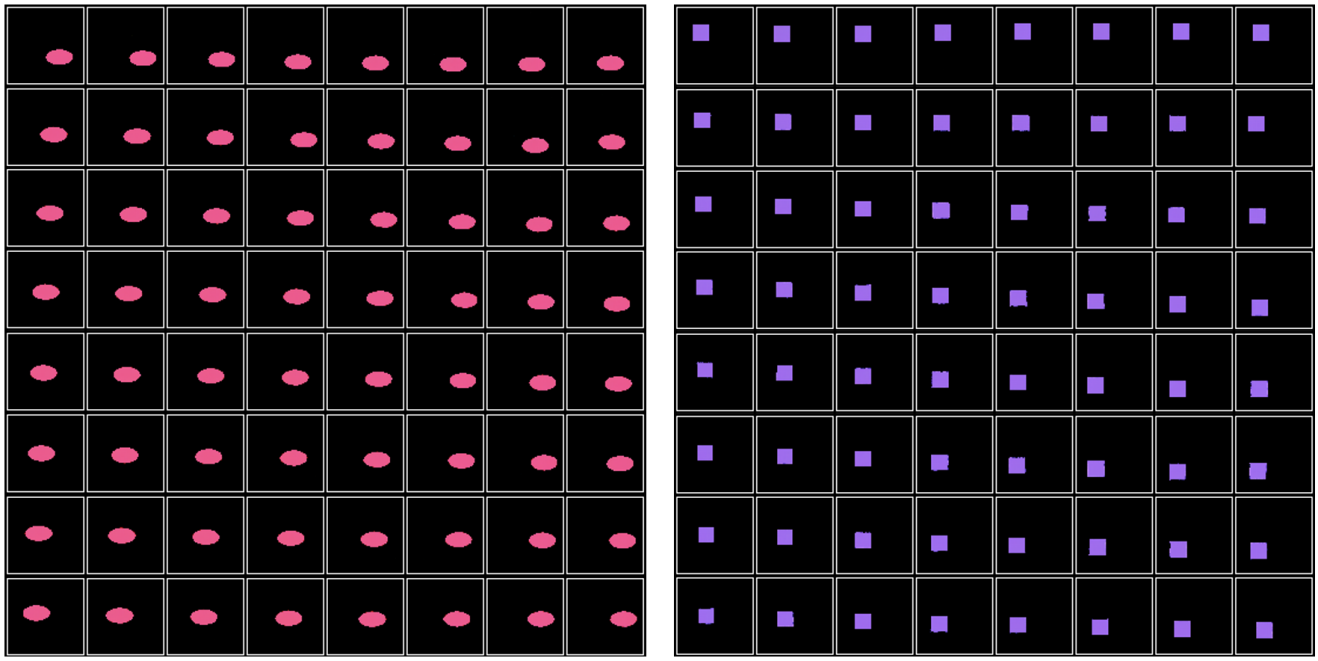

Figure 13. Interpolation in the latent space for s-dSprites. In each of the two columns, the first and last rows share the same static code $\mathbf{f}$, but have different dynamic codes $\lambda_{1:T}$. The rows in between are generated by fixing the static code and interpolating the dynamic ones using cosine interpolation.

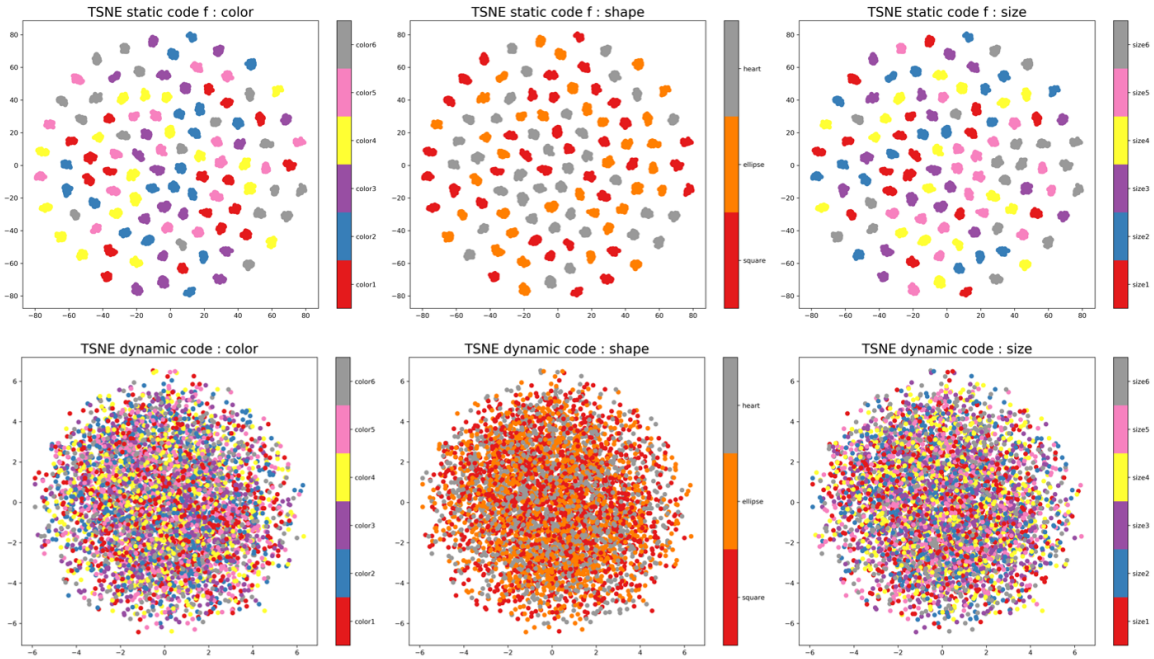

Figure 14. 2D embedding of the static (top row) and dynamic (bottom row) codes computed using t-SNE on all *s-dSprites* test samples. In each row we display the embedding with points colored according to the different static factors of the dataset. The static code embedding perfectly clusters all static factors combinations while the dynamic codes appear invariant to the static variables.

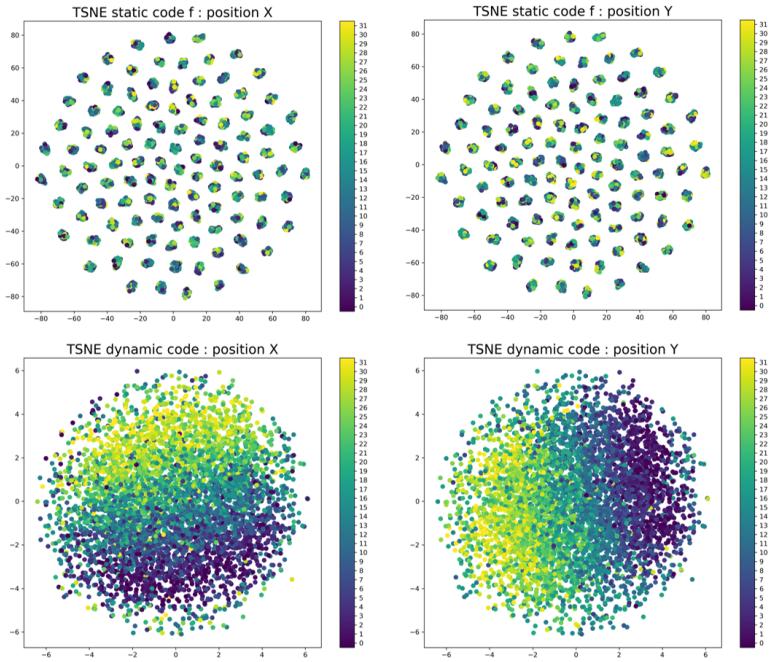

Figure 15. 2D embedding of the static (top row) and dynamic (bottom row) codes computed using t-SNE on all *s-dSprites* test samples. In each row we display the embedding with points colored according to the different dynamic factors of the dataset i.e., the 32 possible positions of the shapes. The dynamic code embedding displays a structure with respect to the dynamic factors while the static codes appear invariant to the dynamic variables.

[9] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016. 1

[10] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 9

[11] Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. *Adv. Neural Inform. Process. Syst.*, 2019. 7

[12] Jun Han, Martin Renqiang Min, Ligong Han, Li Erran Li, and Xuan Zhang. Disentangled recurrent wasserstein autoencoder. *arXiv preprint arXiv:2101.07496*, 2021. 6, 8

[13] 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. *Int. Conf. Learn. Represent.*, 2016. 5, 8, 10, 11

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 8

[15] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 2004. 4

[16] Yingzhen Li and Stephan Mandt. Disentangled sequential autoencoder. *arXiv preprint arXiv:1803.02991*, 2018. 6, 8, 15

[17] Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M Asano, Taco Cohen, and Stratis Gavves. Citris: Causal identifiability from temporal intervened sequences. *Int. Conf. Mach. Learn.*, 2022. 9

[18] 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. *Int. Conf. Mach. Learn.*, 2019. 8

[19] Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. *Int. Conf. Mach. Learn.*, 2020. 11

[20] Joseph Marino, Lei Chen, Jiawei He, and Stephan Mandt. Improving sequential latent variable models with autoregressive flows. *Symposium on Advances in Approximate Bayesian Inference*, 2020. 9

[21] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017. 7

[22] Graziano Mita, Maurizio Filippone, and Pietro Michiardi. An identifiable double vae for disentangled representations. *Int. Conf. Mach. Learn.*, 2021. 11

[23] Ilan Naiman, Nimrod Berman, and Omri Azencot. Sample and predict your latent: Modality-free sequential disentanglement via contrastive estimation. *arXiv preprint arXiv:2305.15924*, 2023. 6, 8, 9

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. 2019. 8

[25] Scott E Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. *Adv. Neural Inform. Process. Syst.*, 2015. 6

[26] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. Animating arbitrary objects via deep motion transfer. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 11

[27] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. *Adv. Neural Inform. Process. Syst.*, 2019.

[28] Aliaksandr Siarohin, Oliver J Woodford, Jian Ren, Menglei Chai, and Sergey Tulyakov. Motion representations for articulated animation. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 11

[29] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008. 14

[30] Julius Von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. *Adv. Neural Inform. Process. Syst.*, 2021. 3, 4

[31] Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled representation learning. *arXiv preprint arXiv:2211.11695*, 2022. 11

[32] Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Motion-based generator model: Unsupervised disentanglement of appearance, trackable and intrackable motions in dynamic patterns. *AAAI*, 2020. 11

[33] Zhuoqian Yang, Wentao Zhu, Wayne Wu, Chen Qian, Qiang Zhou, Bolei Zhou, and Chen Change Loy. Transmomo: Invariance-driven unsupervised video motion retargeting. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 11

[34] Xinqi Zhu, Chang Xu, and Dacheng Tao. Commutative lie group vae for disentanglement learning. *Int. Conf. Mach. Learn.*, 2021. 11

[35] Yizhe Zhu, Martin Renqiang Min, Asim Kadav, and Hans Peter Graf. S3vae: Self-supervised sequential vae for representation disentanglement and data generation. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 6, 8, 15