A Appendix

The appendix is structured into three main parts.

The first part (section A.1, A.2) provides additional details about SPS. Section A.1 focuses on implementation-related aspects, while section A.2 presents experimental results concerning the reconstruction and prediction loss.

The second part (section A.3 to A.5) introduces an extended version of SPS called SPS+. Section A.3 describes the capabilities of SPS+ in achieving content-style disentanglement along with interpretable learning. The related experiments are presented in section A.4 and section A.5.

The third part (section A.6) presents two additional complex experiments conducted separately using SPS+ and SPS, respectively.

A.1 SPS implementation details

A.1.1 Architecture details

Our models for both tasks share the following architecture. The encoder first uses a 2D-CNN with ReLU activation to shrink the input down to an 8×8 middle layer, and then a linear layer to obtain z. If the encoder is in a VAE (instead of an AE), two linear layers characterises the posterior, one for the mean and the other for the log-variance. The prior model is a vanilla RNN of one layer with 256 hidden units and one linear layer projection head. The decoder consists of a small fully-connected network followed by 2D transposed convolution layers mirroring the CNN in the encoder. Its output is then passed through a sigmoid function. We use no batch normalisation or dropout layers.

Minor variations exist between the models for the two tasks. In the audio task, we use three convolution layers in the encoder, with three linear and three 2D transposed convolution layers in the decoder. In the vision task, as the data are more complex, we use four convolution layers in the encoder, with four linear and four 2D transposed convolution layers in the decoder.

A.1.2 Training details

For both tasks, we use the Adam optimiser with learning rate $= 10^{-3}$. The training batch size is 32 across all of our experiments. For all VAE-based models, including SPS_{VAE} (ours/ablation) and β-VAE (baseline), we set β (i.e., λ_3 in Equation (1)) to 0.01, with $\lambda_1 = 1$ and $\lambda_2 = 2$. All BCE and MSE loss functions are calculated in sum instead of mean. $K = 4$ for all SPS models except for those discussed in section 5 where we analyse the influence of different K.

The RNN predicts $z_{n+1:T}$ given the first n embeddings $z_{1:n}$. We choose $n = 3$ for the audio task and $n = 5$ for the vision task. We adopt scheduled sampling [Bengio *et al.*, 2015] during the training stage, where we gradually reduce the guidance from teacher forcing. After around 50000 batch iterations, the RNN relies solely on the given $z_{1:T}$ and predicts auto-regressively.

A.2 SPS reconstruction and prior prediction results

We investigate the reconstruction and prediction capacities of our model and show that they are not harmed by adding symmetry constraints. For the music task, we compare our model, our model ablating symmetry constraints, and a β-VAE trained solely for the reconstruction of power spectrogram. Table 3 reports per-pixel BCE of the reconstructed sequences from the original input frames (Self-recon) and from the RNN predictions (Image-pred). We also include $\mathcal{L}_{\text{prior}}$, the MSE loss on the RNN-predicted $\hat{z}$ as defined in section 3.2. The results show that our models slightly surpasses the ablation and baseline models in all three metrics.

Similarly, Table 4 displays the reconstruction and prediction losses on the test set for the video task. Results show that adding symmetry constraints does not significantly hurt the prediction losses. Frame-wise self-reconstruction is significantly lower for the SPS models, but only by a small margin.

A.3 SPS+

SPS can use physical symmetry to learn interpretable factors that evolve over time. We call those factors content representation. However, many problems can not be represented by content represen-

Table 3: Reconstruction and prediction results on the audio task.

Methods	Self-recon ↓	Image-pred ↓	$\mathcal{L}_{\text{prior}}$ ↓
SPS_{VAE}, K=4 (Ours)	**0.0375±0.0012**	**0.0377±0.0007**	**0.0057±0.0015**
SPS_{AE}, K=4 (Ours)	0.0381±0.0012	0.0384±0.0009	0.0068±0.0031
SPS_{VAE}, K=0 (Ablation)	0.0384±0.0014	0.0388±0.0013	0.0134±0.0101
SPS_{AE}, K=0 (Ablation)	0.0386±0.0012	0.0391±0.0012	0.0075±0.0024
β-VAE	0.0406±0.0008	N/A	N/A

Table 4: Reconstruction and prediction losses of the video task. Two outliers are removed from the 50 runs.

Method	Self-recon ↓	Image-pred ↓	$\mathcal{L}_{\text{prior}}$ ↓
SPS_{VAE}, K=4 (Ours)	0.64382 ± 9e-05	0.6456 ± 4e-04	0.14 ± 0.05
SPS_{AE}, K=4 (Ours)	0.64386 ± 7e-05	0.6458 ± 3e-04	0.17 ± 0.07
SPS_{VAE}, K=0 (Ablation)	0.64372 ± 4e-05	0.6459 ± 2e-04	0.19 ± 0.10
SPS_{AE}, K=0 (Ablation)	0.64367 ± 5e-05	**0.6456 ± 1e-04**	**0.11 ± 0.03**
β-VAE	**0.64345 ± 5e-05**	N/A	N/A

tation alone. For example, the bouncing balls can have different colours and the pitch scales can be generated by different instruments. If the colour of a ball or the timbre of a sound scale are constant within a trajectory, those latent spaces are hard to constrain by physical symmetry. We call such invariant factors style representation. In order to deal with these problems, we combine SPS with a simple content-style disentanglement technique: SPS+, a more general framework of SPS. We use random pooling to constrain the style factors, and use physical symmetry to constrain the content representation in the same way as SPS in Section 3.1

A.3.1 Model

Figure 9 shows the design of SPS+, which belongs to the family of disentangled sequential autoencoders [Bai *et al.*, 2021; Hsu *et al.*, 2017; Vowels *et al.*, 2021; Yingzhen and Mandt, 2018; Zhu *et al.*, 2020]. During the training process, the temporal data input $\mathbf{x}_{1:T}$ is first fed into the encoder E to obtain the corresponding representation $\mathbf{z}_{1:T}$. $\mathbf{z}_{1:T}$ is then split into two parts: the style factor $\mathbf{z}_{1:T,s}$ and the content factor $\mathbf{z}_{1:T,c}$. The style factor $\mathbf{z}_{1:T,s}$ is passed through the random-pooling module P, where one element $z_{\tau,s}$ is randomly picked. The content factor $\mathbf{z}_{1:T,c}$ is fed into *three* branches, then combined with $z_{\tau,s}$ to reconstruct. For random pooling in the training stage, one style vector is randomly selected from all time steps (i.e., 15 for the music task and 20 for the vision task) of the sequence to represent z_s. In the testing stage, only the first 5 (vision task) or 3 (music task) frames are given, and z_s will be selected from them.

A.3.2 Training objective

The following loss functions in SPS+ slightly vary from those in SPS. For SPS+, $\mathcal{L}_{\text{prior}}$ and $\mathcal{L}_{\text{sym}}$ work on the content part of latent variables only. Other loss functions are exactly the same as those defined in section 3.2

$$\mathcal{L}_{\text{prior}} = \ell_2(\hat{\mathbf{z}}_{2:T,c}, \mathbf{z}_{2:T,c}), \tag{8}$$

$$\mathcal{L}_{\text{sym}} = \ell_2(\tilde{\mathbf{z}}_{2:T,c}, \hat{\mathbf{z}}_{2:T,c}) + \ell_2(\tilde{\mathbf{z}}_{2:T,c}, \mathbf{z}_{2:T,c}). \tag{9}$$

$$\ell_2(\tilde{\mathbf{z}}_{2:T,c}, \mathbf{z}_{2:T,c}) = \frac{1}{K}\sum_{k=1}^{K} \ell_2(S_k^{-1}(R(S_k(\mathbf{z}_{1:T-1,c}))), \mathbf{z}_{2:T,c}), \tag{10}$$

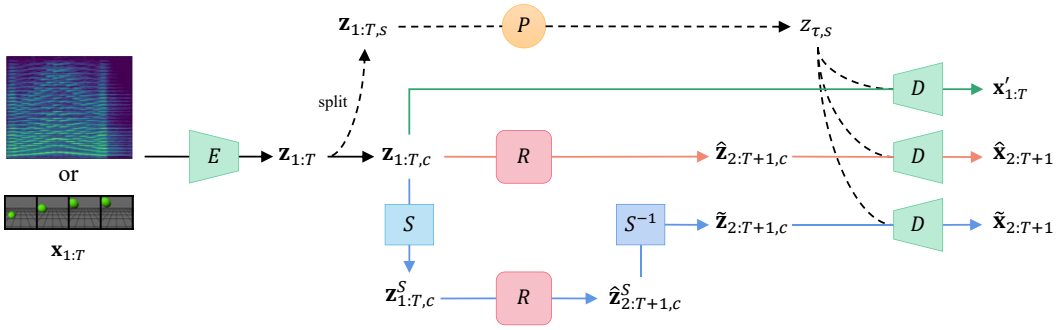

Figure 9: An overview of our model. $\mathbf{x}_{1:T}$ is fed into the encoder E to obtain the corresponding representation $\mathbf{z}_{1:T}$, which is then split into two parts: the style factor $\mathbf{z}_{1:T,s}$ and the content factor $\mathbf{z}_{1:T,c}$. The style factor is passed through the random-pooling layer P, where an element $z_{\tau,s}$ is randomly selected. The content factor is fed into three different branches and combined with $z_{\tau,s}$ to reconstruct three outputs respectively: $\mathbf{x}'_{1:T}$, $\hat{\mathbf{x}}_{2:T+1}$ and $\tilde{\mathbf{x}}_{2:T+1}$. Here, R is the prior model and S is the symmetric operation. The inductive bias of physical symmetry enforces R to be equivaraint w.r.t. to S, so $\tilde{\mathbf{z}}$ and $\hat{\mathbf{z}}$ should be close to each other and so are $\tilde{\mathbf{x}}$ and $\hat{\mathbf{x}}$.

$$\ell_2(\tilde{\mathbf{z}}_{2:T,c}, \hat{\mathbf{z}}_{2:T,c}) = \frac{1}{K}\sum_{k=1}^{K} \ell_2(S_k^{-1}(R(S_k(\mathbf{z}_{1:T-1,c}))), \hat{\mathbf{z}}_{2:T,c}), \tag{11}$$

$$\mathcal{L}_{\text{BCE}}(\tilde{\mathbf{x}}_{2:T}, \mathbf{x}_{2:T}) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{L}_{\text{BCE}}(D(S_k^{-1}(R(S_k(\mathbf{z}_{1:T-1,c}))), z_{\tau,s}), \mathbf{x}_{2:T}). \tag{12}$$

A.4 SPS+ on learning pitch & timber factors from audios of multiple instruments

A.4.1 Dataset and setups

We synthesise a dataset that contains around 2400 audio clips played by **multiple instruments**. Similar to the dataset in section 4.1.1, each clip contains 15 notes in major scales with the first 8 notes ascending and the last 8 notes descending. Each note has the same volume and duration. The interval between every two notes is equal. We vary the starting pitch such that every MIDI pitch in the range C2 to C7 is present in the dataset. For each note sequence, we synthesise it using 53 different instruments, yielding 2376 audio clips. Specifically, two soundfonts are used to render those audio clips respectively: FluidR3_GM [Wen, 2013] for the train set and GeneralUser GS v1.471 [Chris, 2017] for the test set. The pitch ranges for different instruments vary, so we limit each instrument to its common pitch range (See Table 14).

We assume $z_c \in \mathcal{R}$ and $z_s \in \mathcal{R}^2$, and use random $S \in G \cong (\mathbb{R}, +)$ to augment z_c with K=4.

A.4.2 Results on pitch-timbre disentanglement

We evaluate the content-style disentanglement using factor-wise data augmentation following [Yang *et al.*, 2019]. Namely, we change (i.e., augment) the instrument (i.e., style) of notes while keeping their pitch the same, and then measure the effects on the encoded z_c and z_s. We compare the normalised z_c and z_s, ensuring they have the same dynamic range. Ideally, the change of z_s should be much more significant than z_c. Here, we compare four approaches: 1) our model (SPS+), 2) our model without splitting for z_s (SPS with $z \in \mathcal{R}^3$ and $S \in G \cong (\mathbb{R}, +)$) as an ablation, 3) GMVAE [Luo *et al.*, 2019], a domain-specific pitch-timbre disentanglement model trained with *explicit pitch labels*, and 4) TS-DSAE [Luo *et al.*, 2022], a recent unsupervised pitch-timbre disentanglement model based on Disentangled Sequential Autoencoder (DSAE).

Figure 10 presents the changes in normalised z_c and z_s measured by L2 distance when we change the instrument of an anchor note whose pitch is D3 and synthesised by accordion. Table 5 provides a more quantitative version by aggregating all possible instrument combinations and all different

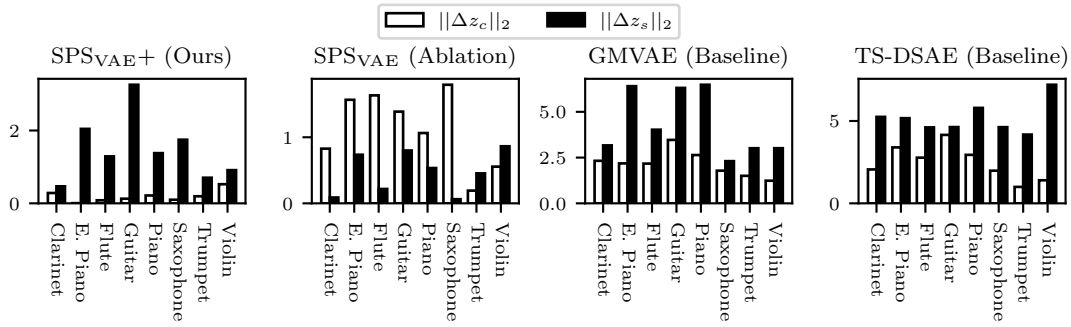

Figure 10: Comparisons for Δz_c and Δz_s for different instruments against accordion, with pitch kept constant at MIDI pitch D3. Δz_c and Δz_s are changes in normalised z_c and z_s, so that higher black bars relative to white bars means better results. All results are evaluated on the test set.

Table 5: Mean ratios of changes in normalised z_c and z_s under timbre augmentation across all possible instrument combinations under different constant pitches in the test set.

Methods	$\|\|\Delta z_c\|\|_2/\|\|\Delta z_s\|\|_2 \downarrow$
SPS$_{\text{VAE}}$+ (Ours)	**0.49**
SPS$_{\text{VAE}}$ (Ablation)	2.20
GMVAE (Baseline)	0.67
TS-DSAE (Baseline)	0.65

pitch pairs. Both results show that SPS+ produces a smaller relative change in z_c under timbre augmentation, demonstrating a successful pitch-timbre disentanglement outperforming both the ablation and baseline. Note that for the ablation model, z_c varies heavily under timbre augmentation, seemingly containing timbre information. This result indicates that the design of an invariant style factor over the temporal flow is necessary to achieve good disentanglement.

We further quantify the results in the form of augmentation-based queries following [Yang *et al.*, 2019], regarding the intended split in z as ground truth and the dimensions with the largest variances from factor-wise augmentation after normalisation as predictions. For example, under timbre augmentation under a given pitch for our model, if z_1 and z_3 are the two dimensions of z that produce the largest variances after normalisation, we count one false positive (z_1), one false negative (z_2), and one true positive (z_3). The precision would be 0.67. Tables 6 shows the precision scores of the four approaches against their corresponding random selection. The results are in line with our observation in the previous evaluation, with our model more likely to produce the largest changes in dimensions in z_c under content augmentation and that in z_s under style augmentation.

Table 6: Results on augmentation-based queries on the audio task. Precision, recall and F1 are the same since the number of predicted and ground-truth positives are identical. Note that random precisions for different approaches can be different as z_c and z_s are split differently.

Methods	Timbre augmentation		Pitch augmentation	
	Precision ↑	Random	Precision ↑	Random
SPS$_{\text{VAE}}$+ (Ours)	**0.98**	0.67	0.82	0.33
SPS$_{\text{VAE}}$ (Ablation)	0.50	0.67	0.02	0.33
GMVAE (Baseline)	0.93	0.50	**0.83**	0.50
TS-DSAE (Baseline)	0.81	0.50	0.68	0.50

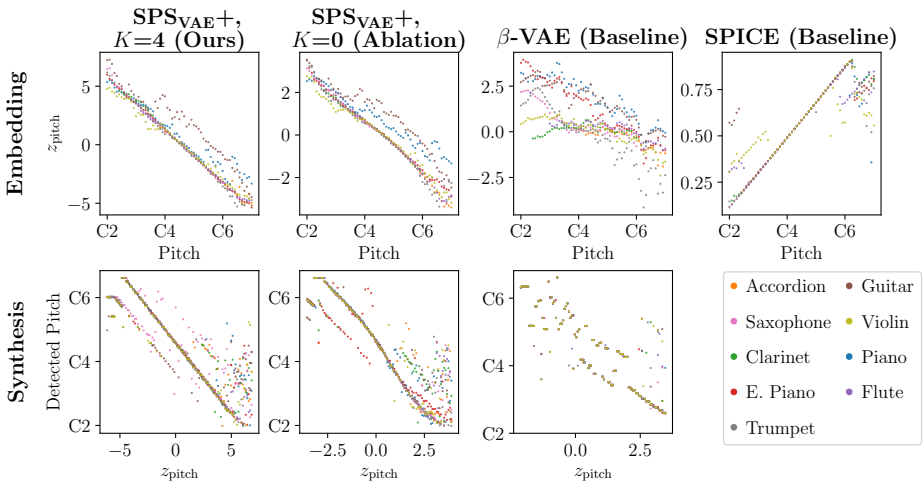

Figure 11: A visualisation of the mapping between the 1D content factor and the true pitch. In the upper row, models encode notes in the test set to z_{pitch}. The x axis shows the true pitch and the y axis shows the learned pitch factor. In the lower row, the x axis traverses the z_{pitch} space. The models decode z_{pitch} to audio clips. We apply YIN to the audio clips to detect the pitch, which is shown by the y axis. In both rows, a linear, noiseless mapping is ideal, and our method performs the best.

Table 7: Reconstruction and prediction results on the audio task.

Methods	Self-recon ↓	Image-pred ↓	$\mathcal{L}_{\text{prior}}$ ↓
SPS$_{\text{VAE}}$+, K=4 (Ours)	**0.0356**	**0.0359**	**0.0418**
SPS$_{\text{VAE}}$+, K=0 (Ablation)	0.0360	0.0363	0.0486
β-VAE (Baseline)	0.0359	N/A	N/A

A.4.3 Results on interpretable pitch space

Figure 11 shows that the pitch factor learned by SPS+ has a linear relation with the true pitch. Here, we use z_{pitch} as the synonym of z_c to denote the content factor. The plot shows the mappings of two tasks and four models. In the embedding task (the first row), x-axis is the true pitch and y-axis is embedded z_{pitch}. In the synthesis task (the second row), x-axis is z_{pitch} and y-axis is the detected pitch (by YIN algorithm, a standard pitch-estimation method by [De Cheveigné and Kawahara, 2002]) of decoded (synthesised) notes. The fours models involved are: 1) our model, 2) our model without symmetry (K=0), 3) a β-VAE trained to encode single-note spectrograms from a single instrument (banjo) to 1D embeddings, and 4) SPICE [Gfeller *et al.*, 2020], a SOTA unsupervised pitch estimator *with strong domain knowledge on how pitch linearity is reflected in log-frequency spectrograms*. As the figure shows, without explicit knowledge of pitch, our model learns a more interpretable pitch factor than β-VAE, and the result is comparable to SPICE.

Figure 12 shows a more quantitative analysis, using R^2 as the metric to evaluate the linearity of the pitch against z_{pitch} mapping. Although SPICE produces rather linear mappings in Figure 11, it suffers from octave errors towards extreme pitches, hurting its R^2 performance.

A.4.4 Reconstruction and prior prediction

We investigate the reconstruction and prediction capacities of our model and show that they are not harmed by adding symmetry constraints. We compare our model, our model ablating symmetry constraints, and a β-VAE trained solely for only image reconstruction. Table 7 reports per-pixel BCE of the reconstructed sequences from the original input frames (Self-recon) and from the RNN predictions (Image-pred). We also include $\mathcal{L}_{\text{prior}}$, the MSE loss on the RNN-predicted $\hat{z}$ as redefined in section A.3.2. The results show that our model surpasses the ablation and baseline models in all three indexes.

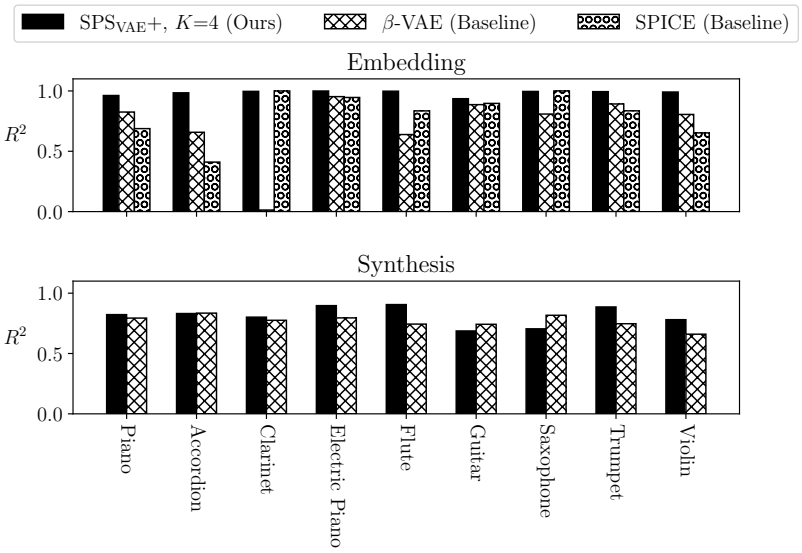

Figure 12: We use R^2 to evaluate mapping linearity. A larger R^2 indicates a more interpretable latent space. Results are evaluated on the test set.

A.5 SPS+ on learning space & colour factors from videos of colourful bouncing balls

A.5.1 Dataset and setups

We run physical simulations of a bouncing ball in a 3D space and generate 4096 trajectories, yielding a dataset of videos. Similar to the dataset in section 4.2.1, the simulated ball is affected by gravity and bouncing force (elastic force). A fixed camera records a 20-frame video of each 4-second simulation to obtain one trajectory (see Figure 13). The ball's size, gravity, and proportion of energy loss per bounce are constant across all trajectories. In this dataset, the color of the ball varies by trajectory, rather than a single color. For each trajectory, the ball's colours are uniformly randomly sampled from a continuous colour space.

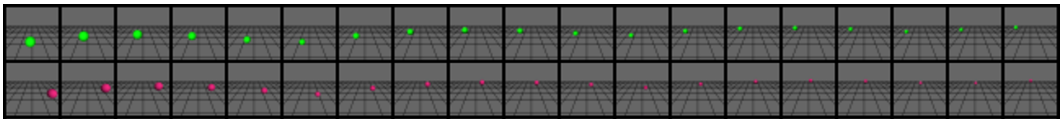

Figure 13: Two example trajectories from the bouncing ball dataset.

We set $z_c \in \mathcal{R}^3$ with the same representation augmentation as in section 4.2 ($S \in G \cong (\mathbb{R}^2, +) \times \mathrm{SO}(2)$, K=4). Two of its dimensions are intended to span the horizontal plane and the third unaugmented latent dimension is intended to encode the vertical height. We set $z_s \in \mathcal{R}^2$ which is intended to represent the ball's colour space.

A.5.2 Result on space-colour disentanglement

Similar to section A.4.2, we evaluate the space-colour disentanglement by augmenting the colour (i.e., style) of the bouncing balls while keeping their locations, and then measure the effects on the normalised z_c and z_s. Again, a good disentanglement should lead to a change in z_s much more significant than z_c. Here, we compare two approaches: 1) our model (SPS+) and 2) our model ablating splitting for z_s (SPS with $z \in \mathcal{R}^5$ and $S \in G \cong (\mathbb{R}^2, +) \times \mathrm{SO}(2)$). Note that the ablation model does not differently constrain z_2 (corresponding to the y-axis) than z_s. To ensure a meaningful comparison, under colour augmentation, we consider z_2 to be a part of z_s of the ablation model and a part of z_c of the complete model.

Figure 14 presents the changes in normalised z_c and z_s measured by L2 distance when we change the colour of an anchor ball whose location is (0, 1, 5) and rendered using white colour. Table 9

Table 8: Results on augmentation-based queries on the visual task. Since the ablation model does not differently constrain z_2 (corresponding to the y-axis) than z_s, we consider z_c and z_s differently for the two approaches. Under colour augmentation, we consider z_2 to be a part of z_s for the ablation model and a part of z_c for the complete model. Under location augmentation, we consider z_2 to be a part of z_c for both models.

Methods	Colour augmentation		Location augmentation	
	Precision ↑	Random	Precision ↑	Random
SPS_{VAE}+ (Ours)	**0.99**	0.40	**0.88**	0.40
SPS_{VAE} (Ablation)	0.64	0.60	0.36	0.40

Table 9: Mean ratios of changes in normalised z_c and z_s under colour augmentation across sampled colour combinations keeping locations constant. Results are evaluated on the test set.

Methods	$\|\|\Delta z_c\|\|_2 / \|\|\Delta z_s\|\|_2 \downarrow$
SPS_{VAE}+ (Ours)	**0.54**
SPS_{VAE} (Ablation)	1.62

provides a more quantitative version by aggregating sampled colour combinations and location pairs. Both results show that our model produces a smaller relative change in z_c under timbre augmentation, demonstrating a successful pitch-timbre disentanglement outperforming the ablation model. Note that for the ablation model, z_c varies heavily under colour augmentation. Table 8 shows the precision scores of the SPS+ and its ablation against their corresponding random selection for the ball task. These results agree with section A.4.2 and again indicate that the design of an invariant style factor helps with disentanglement.

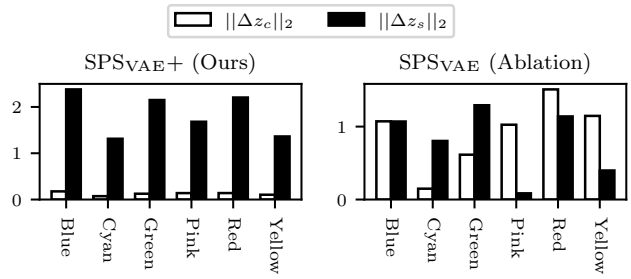

Figure 14: Comparisons of normalised Δz_c and Δz_s for different colours against white, with the ball's location kept constant at (0, 1, 5). Higher black bars (relative white bars) means better a result. (Results are evaluated on the test set.)

Figure 15 evaluates the learned colour factor of our model. Each pixel shows the colour of the ball synthesised by the decoder using different z coordinates. The ball colour is detected using naive saturation maxima. In the central subplot, the location factor $z_{1:3}$ stays at zeros while the colour factor $z_{4:5}$ is controlled by the subplot's x, y axes. As shown in the central subplot, our model (a) learns a natural 2D colour space. The surrounding subplots keep the colour factor $z_{4:5}$ unchanged, and the location factor $z_{1,3}$ is controlled by the subplot's x, y axes. A black cross marks the point where the entire $z_{1:5}$ is equal to the corresponding black cross in the central subplot. As is shown by the surrounding subplots, varying the location factor does not affect the colour produced by our model (a), so the disentanglement is successful. The luminosity changes because the scene is lit by a point light source, making the ball location affect the surface shadow. On the other hand, β-VAE (b) learns an uninterpretable colour factor.

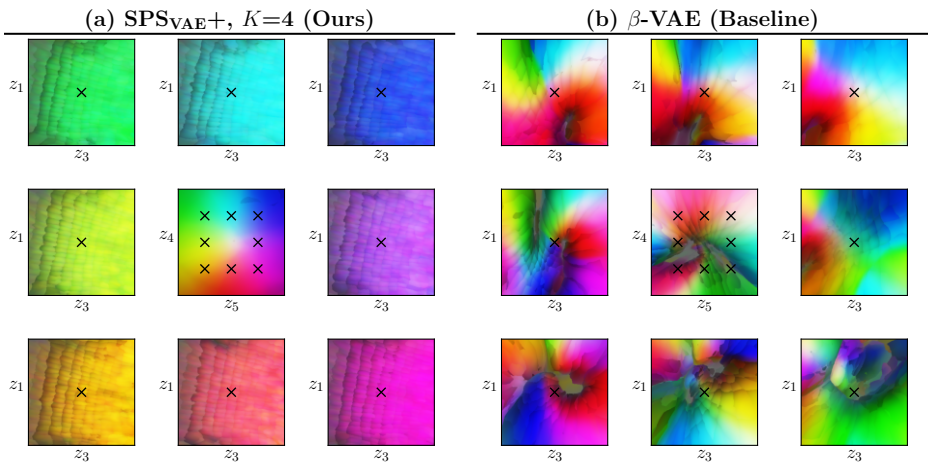

Figure 15: The colour map of the synthesised ball experiment through latent space traversal. Each pixel represents the detected colour from one synthesised image of the ball. Each subplot varies two dimensions of z, showing how the synthesised colour responds to the controlled z.

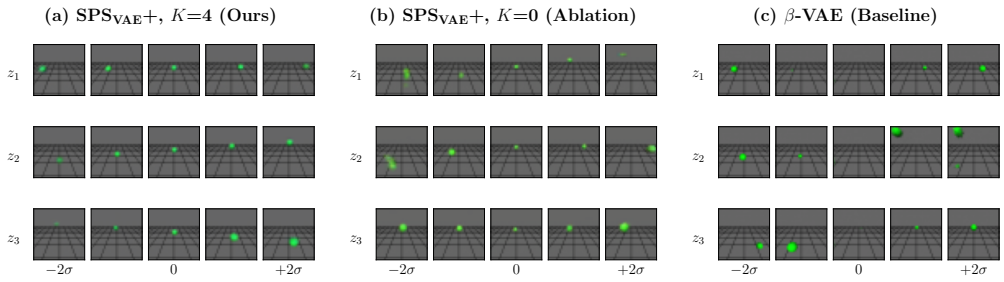

Figure 16: Row i shows the generated images when changing z_i and keeping $z_{\neq i} = 0$, where the x axis varies z_i from -2σ to $+2\sigma$. In (a), changing z_2 controls the ball's height, and changing z_1, z_3 moves the ball parallel to the ground plane.

A.5.3 Results on interpretable 3D representation

Figure 16 illustrates the interpretability of learned content factor using latent space traversal. Each row varies only one dimension of the learned 3D content factor, keeping the other two dimensions at zero. Figure 16(a) shows the results of our model. We clearly observe that: i) *increasing z_1 (the first dimension of z_c) mostly moves the ball from left to right, increasing z_2 moves the ball from bottom to top, and increasing z_3 mostly moves the ball from far to near.* Figure 16(b) is the ablation model without physical symmetry, and (c) shows the result of our baseline model β-VAE, which is trained to reconstruct static images of a single colour (green). Neither (b) nor (c) learns an interpretable latent space.

Table 10: Linear fits between the true location and the learned location factor. We run the encoder on the test set to obtain data pairs in the form of (location factor, true coordinates). We then run a linear fit on the data pairs to evaluate factor interpretability.

Method	x axis MSE ↓	y axis MSE ↓	z axis MSE ↓	MSE ↓
SPS$_{\text{VAE}}$+, K=4 (Ours)	**0.11**	**0.06**	**0.09**	**0.09**
SPS$_{\text{VAE}}$+, K=0 (Ablation)	0.35	0.72	0.68	0.58
β-VAE (Baseline)	0.37	0.76	0.73	0.62

Table 11: Reconstruction and prediction results on the video task with variable colours.

Method	Self-recon ↓	Image-pred ↓	$\mathcal{L}_{\text{prior}}$ ↓
SPS_{VAE}+, K=4 (Ours)	0.6457	**0.6464**	**0.0957**
SPS_{VAE}+, K=0 (Ablation)	0.6456	**0.6464**	0.1320
β-VAE (Baseline)	**0.6455**	N/A	N/A

Table 12: R^2 aggregated across all instruments in the test set. A larger R^2 indicates a more interpretable latent space.

Method	Self-recon ↓	Image-pred ↓	$\mathcal{L}_{\text{prior}}$ ↓	Embedding R^2 ↑	Synthesis R^2 ↑
SPS_{VAE}+, K=4 (Ours)	0.0384	**0.0396**	**0.7828**	**0.89**	**0.47**
SPS_{VAE}+, K=0 (Ablation)	0.0388	0.0400	0.9909	0.83	0.25
β-VAE (Baseline)	**0.0324**	N/A	N/A	0.19	0.29

Table 10 quantitatively evaluates the linearity of the learned location factor. We fit a linear regression from z_c to the true 3D location over the test set and then compute the Mean Square Errors (MSEs). A smaller MSE indicates a better fit. All three methods (as used in Figure 16) are evaluated on a single-colour (green) test set. Results show that our model achieves the best linearity in the learned latent factors, which aligns with our observations in Figure 16.

A.5.4 Reconstruction and prior prediction

Similar to section A.4.4, we show that our model suffers little decrease in reconstruction and prediction performance while surpassing the ablation model in terms of $\mathcal{L}_{\text{prior}}$ by table 11.

A.6 More complicated tasks

The main part of this paper focuses on simple, straight-forward experiments. Still, we supplement our findings by reporting our current implementation's performance on more complicated tasks involving natural melody and real-world video data.

A.6.1 Learning interpretable pitch factors from natural melodies

We report the performance of SPS+ on learning interpretable pitch factors from monophonic melodies under a more realistic setup. We utilize the melodies from the Nottingham Dataset [Foxley, 2011], a collection of 1200 American and British folk songs. For simplicity, we quantise the MIDI melodies by eighth notes, replace rests with sustains and break down sustains into individual notes. We synthesise each non-overlapping 4-bar segment with the accordion soundfonts in FluidR3 GM [Wen, 2013], resulting in around 5000 audio clips, each of 64 steps.

This task is more realistic than the audio task described in A.4 since we use a large set of natural melodies instead of one specified melody line. The task is also more challenging as the prior model has to predict long and more complex melodies. To account for this challenge, we use a GRU [Cho *et al.*, 2014] with 2 layers of 512 hidden units as the prior model. We perform early-stopping after around 9000 iterations based on spectrogram reconstruction loss on the training set. The model and training setup is otherwise the same as in A.4.

Following A.4.3, We evaluate our approach on notes synthesised with all instruments in GeneralUser GS v1.471 [Chris, 2017] in the MIDI pitch range of C4 to C6, where most of the melodies in Foxley [2011] take place. Note that this is a challenging zero-shot scenario since the model is trained on only one instrument. We compare our model, our model ablating the symmetry loss and a β-VAE baseline. We visualise the embedded z_{pitch} and synthesised pitches for different instruments in Figure 17. Following 12, R^2 results are shown in Figure 18 and Table 12. Even when tested on unseen timbres, our model can learn linear and interpretable pitch factors and demonstrates better embedding and synthesis performance compared with the ablation model, which outperforms the β-VAE baseline.

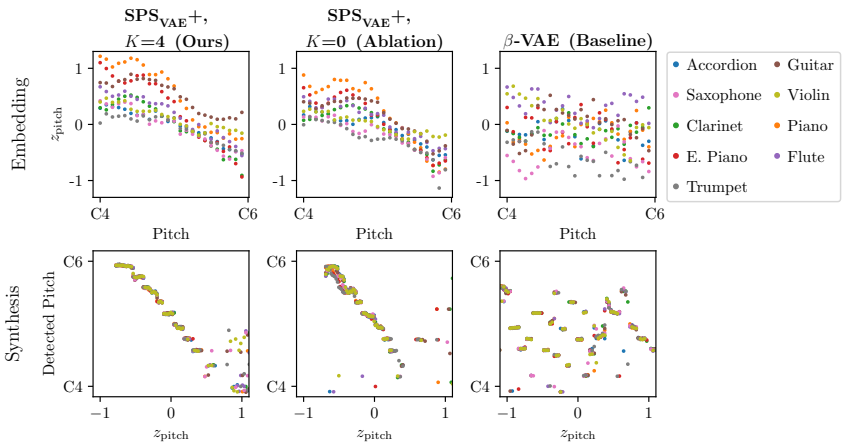

Figure 17: A visualisation of the mapping between the embedded 1D content factor and the true pitch for the model trained on Nottingham dataset.

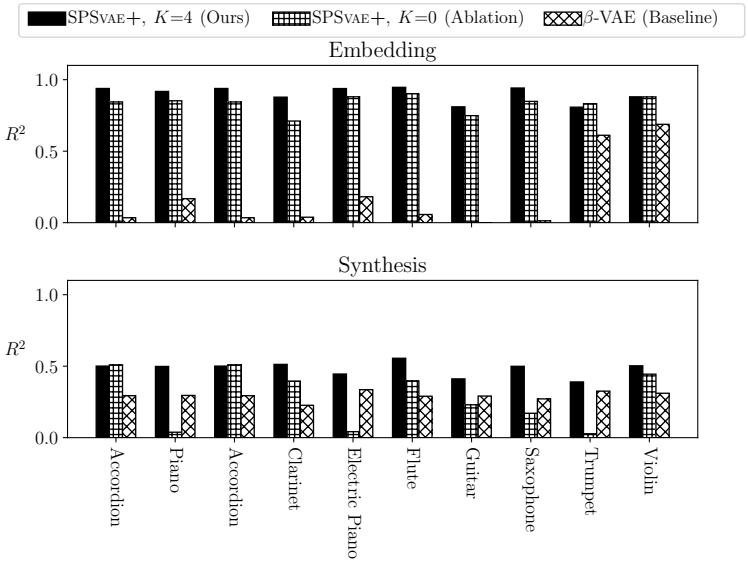

Figure 18: R^2 for select instruments in the test set. A larger R^2 indicates a more linear and interpretable latent space.

A.6.2 Learning an intepretable location factor from KITTI-Masks

In this task, we evaluate our method's capability on a real-world dataset, KITTI-Masks [Klindt *et al.*, 2021]. The dataset provides three labels for each image: X and Y for the mask's 2D coordinate, and AR for the pixel-wise area of the mask. Based on the provided labels, we use simple geometrical relation to estimate the person-to-camera distance d, computed as $d = 1/\tan(\alpha\sqrt{AR})$, where α is a constant describing the typical camera's Field of View (FoV).

We use a 3-dimensional latent code for all models. For SPS, all 3 dimensions are content factors z_c and no style factor z_s is used. We apply group assumption $(\mathbb{R}^3, +)$ to augment representations with $K = 1$. To measure the interpretability, we fit a linear regression from z_c to the ground truth labels and calculate MSEs in the same way as in section A.5.3. The results are shown in Table 13. Linear proj. MSE 1 measures the errors of linear regression from z_c to the original dataset labels. Linear proj. MSE 2 measures the errors of linear regression from z_c to the person's 3-D location, estimated from the labels.

Table 13: Results of KITTI-Masks task, averaging on 30 random initialisations for each method.

Methods	Self-recon ↓	Image-pred ↓	Linear proj. MSE 1 ↓	Linear proj. MSE 2 ↓
SPS_{VAE}, K=4 (Ours)	0.030±0.001	**0.084±0.006**	**0.215±0.067**	**0.203±0.065**
SPS_{VAE}, K=0 (Ablation)	0.030±0.001	0.093±0.010	0.235±0.077	0.243±0.088
β-VAE (Baseline)	**0.028±0.001**	N/A	0.403±0.194	0.399±0.204

As is shown in Table 13, MSE 2 is smaller than MSE 1 for SPS, indicating that SPS learns more fundamental factors (person's location) rather than superficial features (pixel-wise location and area). For the baseline methods, MSE 2 is almost equal to MSE 1, and both of them are higher than those of SPS. In summary, our experiment shows that SPS learns more interpretable representations than the baseline (as well as the ablation method) on KITTI-Masks dataset.

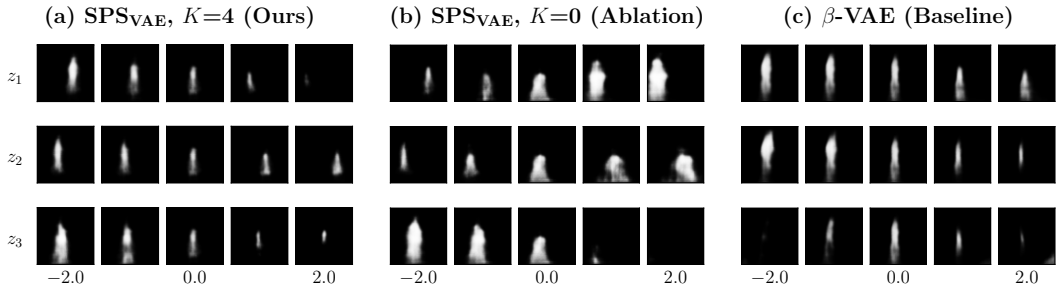

Figure 19: Latent space traversal on different models. Row i shows the generated images when changing z_i and keeping $z_{\neq i} = 0$. For our model, the range of z_1 from -2 to 2 corresponds to the human location from near-right to far-left, z_2 from near-left to far-right, and z_3 from near to far. We can see that other methods produce more non-linear trajectories, for example in (c), the human location hardly changes when $z_1 < 0$, but it changes dramatically when $z_1 > 0$.

Figure 19 shows the generated images, which illustrates that the factors learned by SPS are more linear than those learned by other methods in the human location attribute. For the experiment, We choose all sequences with length ≥ 12 from KITTI-Masks as our dataset; we use 1058 sequences for training and 320 sequences for evaluation; In the inference stage, only the first 4 frames are given. All three methods are trained 30 times with different random initializations. Table 13 shows the average results evaluated on the same test set with 30 different seeds.

A.7 Reproducibility statement and compute requirement

The source code for training and testing the models, as well as generating the figures and tables, is publicly available at https://github.com/double-blind-75098/Learning-basic-interpretable-factors-from-temporal-signals-via-physical-symmetry.

The GTX1080Ti graphics card is capable of effectively executing all the tasks presented in this paper.

Instrument	MIDI Note (from)	MIDI Note (to)
Accordion	58	96
Acoustic Bass	48	96
Banjo	36	96
Baritone Saxophone	36	72
Bassoon	36	84
Celesta	36	96
Church Bells	36	96
Clarinet	41	84
Clavichord	36	84
Dulcimer	36	84
Electric Bass	40	84
Electric Guitar	36	96
Electric Organ	36	96
Electric Piano	36	96
English Horn	36	85
Flute	48	96
Fretless Bass	36	84
Glockenspiel	36	96
Guitar	36	96
Harmonica	36	96
Harp	36	96
Harpsichord	36	96
Horn	36	96
Kalimba	36	96
Koto	36	96
Mandolin	36	96
Marimba	36	96
Oboe	36	96
Ocarina	36	96
Organ	36	96
Pan Flute	36	96
Piano	36	96
Piccolo	48	96
Recorder	36	96
Reed Organ	36	96
Sampler	36	96
Saxophone	36	84
Shakuhachi	36	96
Shamisen	36	96
Shehnai	36	96
Sitar	36	96
Soprano Saxophone	36	96
Steel Drum	36	96
Timpani	36	96
Trombone	36	96
Trumpet	36	96
Tuba	36	72
Vibraphone	36	96
Viola	36	96
Violin	36	96
Violoncello	36	96
Whistle	48	96
Xylophone	36	96

Table 14: Pitch range (in MIDI note) for each instrument in our dataset.