EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

VQ-Style: Disentangling Style and Content in Motion with Residual Quantized Representations

Fatemeh Zargarbashi[1,2], Dhruv Agrawal[1,2], Jakob Buhmann[2], Martin Guay[2], Stelian Coros[1], Robert W. Sumner[1,2]

[1]ETH Zürich, Switzerland [2]DisneyResearch|Studios, Switzerland

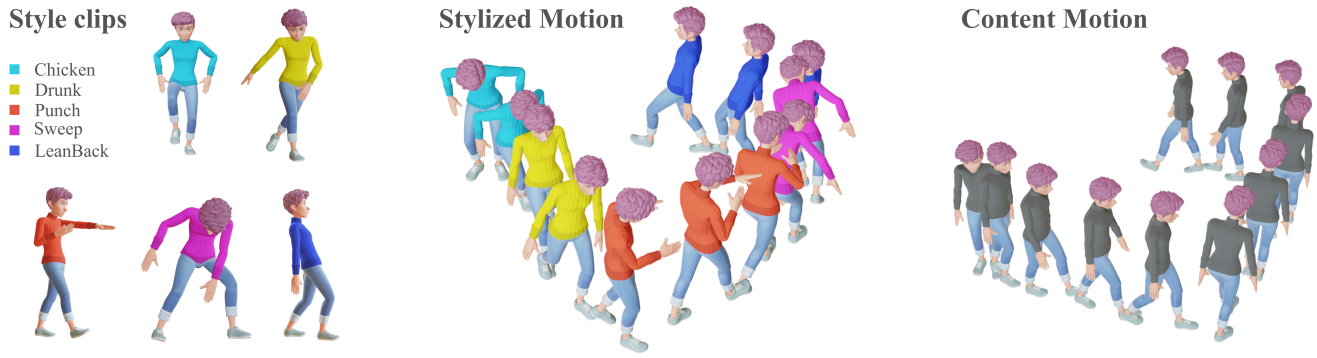


Figure 1: *On the left, we see different styles, while at the far right is a content motion that is much longer. In the middle, we transferred the style from different clips over different time windows of the content while seamlessly transitioning between them.*

Abstract

Human motion data is inherently rich and complex, containing both semantic content and subtle stylistic features that are challenging to model. We propose a novel method for effective disentanglement of the style and content in human motion data to facilitate style transfer. Our approach is guided by the insight that content corresponds to coarse motion attributes while style captures the finer, expressive details. To model this hierarchy, we employ Residual Vector Quantized Variational Autoencoders (RVQ-VAEs) to learn a coarse-to-fine representation of motion. We further enhance the disentanglement by integrating codebook learning with contrastive learning and a novel information leakage loss to organize the content and the style across different codebooks. We harness this disentangled representation using our simple and effective inference-time technique Quantized Code Swapping*, which enables motion style transfer without requiring any fine-tuning for unseen styles. Our framework demonstrates strong versatility across multiple inference applications, including style transfer, style removal, and motion blending.*

CCS Concepts

• ***Computing methodologies*** → ***Animation; Learning latent representations;***

1. Introduction

Virtual characters play a central role in media and entertainment, from animation and video games to immersive experiences. However, creating realistic and expressive character animation remains a labor-intensive process that often requires extensive manual work from artists, making it both time-consuming and costly. Recent research in character animation has explored how modern machine learning techniques can reduce repetitive work and automate aspects of the animation pipeline. One notable problem is *style transfer*, where the *style* of a motion—such as walking happily or angrily—is transferred from one motion clip to another, while preserving the content, or semantic meaning, of the latter motion, which is usually referred to as the *content* clip.

Achieving compelling motion style transfer requires disentangling style from content—a fundamental and challenging problem in motion representation. While humans naturally exhibit a wide

range of walking styles, clearly defining and separating style and content in motion data is non-trivial. Our goal is to learn a representation that effectively disentangles the content and style from motion clips and allows for transferring styles realistically. We approach this problem by interpreting content as the coarse and structural attributes of motion, and style as the finer details that introduce expressive nuances to the movement. To achieve this separation, we employ Residual Vector Quantized Variational Autoencoders (RVQ-VAEs) [LKK*22] to learn a coarse-to-fine representation. This is complemented by novel contrastive and mutual information losses that prevent style leakage into content. We demonstrate that the resulting representation effectively separates style from content and enables style transfer through *Quantized Code Swapping*. Our approach performs style transfer as an inference-only task, requiring no additional fine-tuning for unseen styles. Furthermore, unlike prior works, our model is trained without adversarial or cyclic training, enabling a more stable convergence.

We evaluate our method on multiple motion capture datasets [MSK22, AWL*20, XWCH15] and show that our representation embeds the content in the initial codebooks and the style in latter ones. Our experiments show effective style transfer across both seen and unseen stylistic motions, measured via style classifier accuracy, while maintaining the original content, measured via trajectory deviation. In addition, it enables further capabilities including smooth transitions between styles, content extraction, motion interpolation, and content-style mixing for data augmentation.

In summary, our contributions are as follows:

- We learn an interpretable coarse-to-fine representation of motion that disentangles content and style.
- We propose a novel strategy by utilizing a mutual info loss to prevent style leakage, and combining contrastive learning with non-differentiable residual codebook learning.
- Our framework supports several applications at inference, including style transfer and style transitions over arbitrary-length motions, via swapping and blending operations on the disentangled quantized codebooks.

2. Related Work

[XWCH15] were one of the first to apply style transfer to motion data using a motion database and a KNN search. They first pre-process the database to track the closest neighboring pose for each pose in the dataset. Afterwards, they can perform real-time style transfer relying on a KNN search and simple linear transformation. In the context of deep learning, [AWL*20] learn separate style and temporal content codes. They use Instance Normalization (IN) and Adaptive Instance Normalization (AdaIN) to remove the original style from the content motion and reintroduce the desired style codes, respectively. Since the model relied heavily on style labels and a discriminator, it struggles to generalize to unseen styles. Both these works also released their annotated styled motion data that have been used by many works since.

[PJL21] use a graph convolutional network (GCN) where the pooling and unpooling operations are performed according to the human skeleton. In addition, they learn a mapping network from random noise to a style code to perform style transfer from unseen styles at inference time. Motion Puzzle [JPL22] also uses a similar GCN, but trains from one source motion to multiple target motions each to stylize one or more body parts. This lets them combine different styles across body parts during inference. More recently, [TWW*24] used contact information for style transfer via learning separate GCNs to extract contact, trajectory and style information from the source motion. They can then perform style transfer at inference by swapping the style codes passed to the generator transformer. Similarly, the contact timing can also be swapped between the style and content motion.

100STYLE dataset [MSK22] is a locomotion dataset with a combination of 100 different styles and different gaits (walking, running, & strafing). In addition to introducing the dataset, [MSK22] train a mixture of expert (MoE) model with a FiLM [DPS*18, PSDV*18] generator and a pre-trained periodic autoencoder [SMK22] providing learned phase labels. Although their approach works well for seen styles, they require fine-tuning to adapt to unseen styles. [TWW*23] also use an MoE model along with the periodic autoencoder to learn a motion VAE to reconstruct the next frame conditioned on the current frame. The decoder can be conditioned on a style token to perform style transfer. [DWF*25] further extend the periodic autoencoder to learn body part-wise phase labels. They can then encode these part-wise phase labels into style labels, allowing mixing styles across body parts similar to Motion Puzzle. These methods rely on a pretrained phase manifold for motion, a MoE and a motion sampler, constructing a complex training pipeline.

Diffusion models have also been employed for motion style transfer. [RGS*24] show that for a pre-trained motion diffusion model [TRG*23], the queries and keys in the self-attention blocks correspond to motion *outline* and *motifs* respectively. Using noise inversion and swapping the queries and keys between a content and a style motion clips, they can perform zero-shot style transfer. In contrast, [ZXJ*24] learn a ControlNet [ZRA23] based style adapter and combine classifier-based and classifier-free [HS22] guidance to adapt a pre-trained text-to-motion model for stylization. [SJL*24] specifically train a diffusion model for style transfer and first learn a multi-condition extractor that can extract trajectory and remove style from the content motion, and extract the style from the style motion. A latent diffusion model is then trained on these features to stylize the content motion.

While the above-mentioned diffusion models offer promising results, their iterative generative process makes them too slow for real-time applications and for stylizing motion of arbitrary lengths. Hence, [GMZ*24] first learn a motion latent representation using a VAE [KW*19]. This representation is then disentangled using style and content encoders and a generator. Reconstruction, homo-style alignment, and cycle consistency losses are used to ensure that the new latent space is sufficiently disentangled. Although they can transfer unseen styles, their method requires training multiple models, and losses such as homo-style alignment and cycle consistency can make the training more unstable.

To our knowledge, we are the first work to explore VQ-VAEs [VDOV*17] in the context of motion stylization. We show that using a Residual VQ-VAE (RVQ-VAE) [LKK*22], we can learn a disentangled latent space for motion style transfer without special-

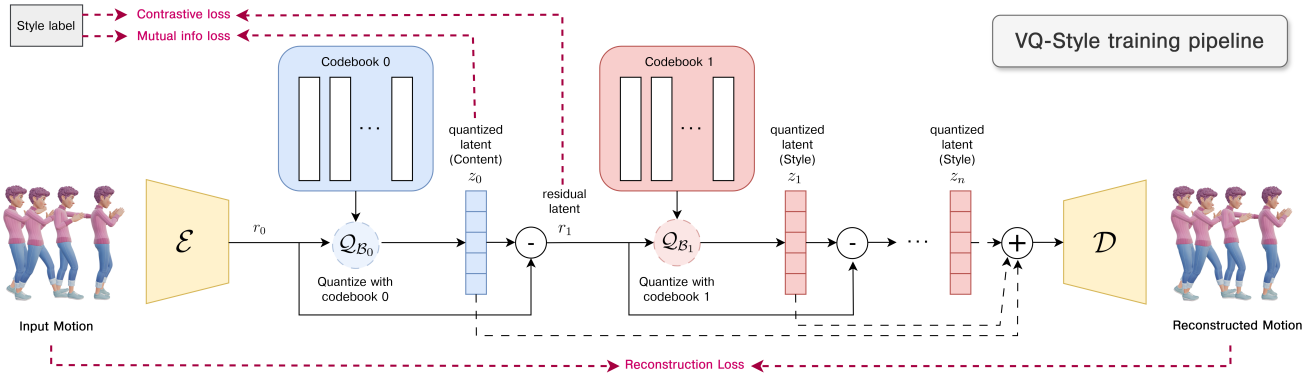


Figure 2: *Our approach encodes motion into a number of codebooks stacked in a residual manner. Our training strategy enables the content to be represented by the first (blue) codebook, while style is represented by the downstream codebooks (in red).*

ized style and content encoders and perform style transfer for arbitrary long motion sequences in real time.

3. Method

Human motion can be represented as a sequence of T states, $\mathcal{M} = \{\mathbf{s}_t\}$. We define the state of the character at timestep t as $\mathbf{s}_t = [\mathbf{p}_t, \mathbf{R}_t, \mathbf{v}_t, \omega_t, \mathbf{h}_t, \mathbf{u}]$, where $\mathbf{p}_t \in \mathbb{R}^{3\times J}, \mathbf{R}_t \in \mathbb{R}^{6\times J}, \mathbf{v}_t \in \mathbb{R}^{3\times J}, \omega_t \in \mathbb{R}^{3\times J}$ represent the position, 6D orientation, linear velocity and angular velocity of each body in local (root) frame, respectively. $\mathbf{h}_t \in \mathbb{R}^3$ is the global height of the root, and $\mathbf{u} \in \mathbb{R}^3$ denotes the global up direction in the local frame. J is the total number of joints in the skeleton.

Our goal is to learn a motion representation that effectively disentangles the stylistic characteristics of human motion, from its semantic content. We utilize an RVQ-VAE [YSZ*24, LKK*22] framework that consists of an encoder $\mathcal{E}$, a decoder $\mathcal{D}$ and the latent space is modeled by multiple categorical distributions or codebooks $\mathcal{B}$. Overview of our framework is demonstrated in Fig. 2. We propose a training strategy in which the initial codebooks capture the content of the motion, while the subsequent codebooks encode the stylistic nuances. Key components of our training strategy include contrastive learning (Sec. 3.3) and mutual information loss (Sec. 3.4). This disentangled representation can be utilized at inference time to perform style transfer via Quantized Code Swapping (Sec. 3.5).

3.1. Quantized Motion Embedding

RVQ-VAEs can be trained by sampling a random number of residual codebooks in use, e.g. for each training sample, only the first n number of codebooks $[\mathcal{B}_0, \mathcal{B}_1, ..., \mathcal{B}_{n-1}]$ are employed to reconstruct the data. This training strategy effectively enables learning a coarse-to-fine representation [YSZ*24, LKK*22], where the initial codebooks encode coarse information and the subsequent codebooks capture finer details. Intuitively, the content of a motion clip corresponds to these coarse components of the data, while stylistic variations are expressed through finer nuances. Under this perspective, RVQ-VAEs provide a natural and effective framework for disentangling style from content in motion.

We train an RVQ-VAE for reconstruction of motion data. The encoder $\mathcal{E}$ is a 1D convolutional neural network that downsamples the motion sequence and the decoder $\mathcal{D}$ is a 1D deconvolutional neural network that upsamples the motion back to the original sequence length. A motion sequence $\mathcal{M}$ with T frames is encoded into K latent embeddings $\mathbf{r}_0 = [\mathbf{r}_0^k]_K = \mathcal{E}(\mathcal{M})$. Then, the first codebook $\mathcal{B}_0$ is used to quantize each embedding in $\mathbf{r}_0$ into the nearest codes $\mathbf{z}_0 = [\mathbf{z}_0^k]_K$,

$$\mathbf{z}_0^k = \arg\min_{\mathbf{c}_i \in \mathcal{B}_0} ||\mathbf{r}_0^k - \mathbf{c}_i||_2^2 = \mathcal{Q}_{\mathcal{B}_0}(\mathbf{r}_0^k), \tag{1}$$

where $\mathcal{Q}_{\mathcal{B}}(.)$ denotes the quantization w.r.t. the codebook $\mathcal{B}$.

The quantized code is used to compute the residual to previous continuous embedding

$$\mathbf{r}_i = \mathbf{r}_{i-1} - \mathbf{z}_{i-1} = \mathbf{r}_0 - \sum_{j=0}^{i-1} \mathbf{z}_j, \tag{2}$$

which is then projected iteratively to the next codebooks,

$$\mathbf{z}_j^k = \arg\min_{\mathbf{c}_i \in \mathcal{B}_j} ||\mathbf{r}_j^k - \mathbf{c}_i||_2^2. \tag{3}$$

The input to the decoder is prepared by using the first n embedded codes, where $n \leq N$ is sampled randomly during training and N is the maximum number of codebooks. The final motion $\mathcal{M}'$ is reconstructed by

$$\mathcal{M}' = \mathcal{D}\left(\sum_{j=0}^{n-1} \mathbf{z}_j\right). \tag{4}$$

3.2. Training

To train the motion embedding, we use a weighted reconstruction loss on the data,

$$\mathcal{L}_{rec} = ||w \cdot (\mathcal{M} - \mathcal{M}')||_2^2, \tag{5}$$

where w is a weighting vector applied element-wise to the signal to emphasize specific motion features.

To prevent error accumulation along the kinematic chain, we employ a forward kinematics (FK) loss,

$$\mathcal{L}_{FK} = ||\mathbf{p}_g - \mathbf{p}'_g||_2^2, \tag{6}$$

where $\mathbf{p}_g$ are ground truth global positions and $\mathbf{p}'_g = \text{FK}(\mathcal{M}')$, where $\text{FK}(\cdot)$ refers to forward kinematic operation on joint orientations in $\mathcal{M}'$.

To ensure temporal consistency in the generated motion, we use velocity loss, $\mathcal{L}_{vel}$, defined as the mean squared error in the velocities of the generated motion as follows.

$$\mathcal{L}_{vel} = ||\dot{\mathbf{p}}_g - \dot{\mathbf{p}}'_g||_2^2, \tag{7}$$

where $\dot{\mathbf{p}}_g$ and $\dot{\mathbf{p}}'_g$ are velocities calculated via finite differences. Lastly, to regularize the motion, we penalize very high accelerations in the output motion via the acceleration loss,

$$\mathcal{L}_{acc} = ||\ddot{\mathbf{p}}'_g||_2^2, \tag{8}$$

where $\ddot{\mathbf{p}}'_g$ refers to the finite-difference acceleration of $\mathbf{p}'_g$.

To effectively train the codebooks $\mathcal{B}_i$, we use commitment loss and Exponential Moving Average (EMA) update as detailed in Appendix A. Code reset is also done similar to [ŁCS*20] for codes that are unused during training to encourage more uniform usage of all codes in the codebook.

3.3. Contrastive Learning

While the inherent coarse-to-fine structure of RVQ-VAE provides a basic separation of style and content, with initial codebooks encoding coarse content and deeper codebooks capturing the stylistic details, this separation arises purely from reconstruction objectives, and does not achieve a sufficiently robust disentanglement for style transfer applications. To enhance this separation, we incorporate a contrastive learning objective that organizes the latent motion embeddings according to style labels, pulling embeddings with the same label closer together and pushing embeddings with different styles farther apart. We apply the contrastive objective exclusively to the deeper residual codebook embeddings intended to encode the style, leaving the initial content codebooks untouched.

For this purpose, we adopt the Multi-Pos contrastive loss introduced in [TFI*23] to contrast all pairs of positive and negative samples within a batch. Let $\mathbf{a}$ denote an anchor embedding sampled from the batch, and $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_x\}$ denote the remainder. The Multi-Pos contrastive loss is given by the cross entropy, H, between similarity-based distribution of how closely $\mathbf{a}$ matches each $\mathbf{b}$ and target distribution constructed from ground truth style labels.

$$\mathcal{L}_{con} = H\left(\frac{\exp(\mathbf{a}\cdot\mathbf{b}_i/\tau)}{\sum\exp(\mathbf{a}\cdot\mathbf{b}_i/\tau)}, \frac{\mathbb{1}_{match(\mathcal{S}(\mathbf{a}),\mathcal{S}(\mathbf{b}_i))}}{\sum\mathbb{1}_{match(\mathcal{S}(\mathbf{a}),\mathcal{S}(\mathbf{b}_i))}}\right), \tag{9}$$

where $\mathcal{S}(.)$ obtains the style label of a sample and τ is a tunable hyperparameter.

[HC20] introduced contrastive learning within a VQ-VAE framework, simply by applying the contrastive objective to the continuous embeddings prior to quantization. In contrast, our method applies contrastive loss directly to the residual embeddings, i.e. after quantization. We show that this choice enables backpropagation to update the codes in the latest codebook, without explicitly affecting the gradients of the earlier stages in the pipeline. This is particularly crucial for our goal, since the style-related contrasts should not affect the initial codebooks which are intended to encode content. To show this, note that gradients of the forward path are computed using straight-through method [HCAI23]:

$$\nabla_\theta(\mathbf{z}_i) := \nabla_\theta(\mathbf{r}_i), \tag{10}$$

where θ corresponds to network parameters before codebook $\mathcal{B}_i$. The gradient of the next residuals will be:

$$\nabla_\theta(\mathbf{r}_{i+1}) = \nabla_\theta(\mathbf{r}_i - \mathbf{z}_i) = \nabla_\theta(\mathbf{r}_i) - \nabla_\theta(\mathbf{z}_i) = 0, \tag{11}$$

where the last equality comes from Eq. 10. The gradient with respect to the codebook is

$$\begin{aligned}\nabla_{\mathcal{B}_i}(\mathbf{r}^k_{i+1}) &= \nabla_{\mathcal{B}_i}(\mathbf{r}^k_i) - \nabla_{\mathcal{B}_i}(\mathbf{z}^k_i) = 0 - \nabla_{\mathcal{B}_i}(\mathcal{Q}_{\mathcal{B}_i}(\mathbf{r}^k_i)) \\ &= -\mathbb{1}_{match}(\mathbf{z}^k_i, \mathbf{c}_i), \quad \forall \mathbf{c}_i \in \mathcal{B}_i\ .\end{aligned} \tag{12}$$

A second key distinction from [HC20] lies in how the codebooks are optimized. While their method uses the standard VQ objective, we adopt the more robust EMA update (see Appendix A). Because EMA updates are applied manually rather than through backpropagation, they break differentiability. Therefore, incorporating the contrastive loss into codebook learning requires careful design. Although the contrastive loss can update the codebook via backpropagation, EMA is typically performed in the forward step and prior to backpropagation, invalidating the gradients. To properly incorporate the contrastive updates, the EMA update must instead be applied after backpropagation, as a second update step.

3.4. Mutual Information Loss

The contrastive loss on style labels enables disentanglement of different styles in the residual codebooks. However, style information can still be stored in the content codebook. Incorporating a similar content contrastive loss for the content codebooks is infeasible as semantic rich content labels are not always available. Instead, we propose restricting the model from inferring style labels from the content codebooks. This can be achieved using a mutual information loss [FH61, NYC19]. Unlike prior work that typically maximize mutual information between given labels and embeddings, we minimize the mutual information between the selected content codes, z, and style labels, l. This ensures that no style can be inferred from the content embeddings. This is formulated as,

$$\mathcal{L}_{mi} = I(Z_{content}; \mathcal{S}) = \sum_{z\in Z_{content}} \sum_{l\in\mathcal{S}} p(z,l) \log \frac{p(z,l)}{p(z)p(l)}. \tag{13}$$

To estimate $p(z,l)$ we use Monte-Carlo sampling,

$$p(z = c_i, l = l_j) \approx \frac{1}{N}\sum_i^N q(z = c_i|r) \cdot \mathbb{1}_{match}(l, l_i), \tag{14}$$

where $q(z = c_i|r_i)$ denotes the assignment probability of quantization, which can be represented as a one-hot vector. However, for

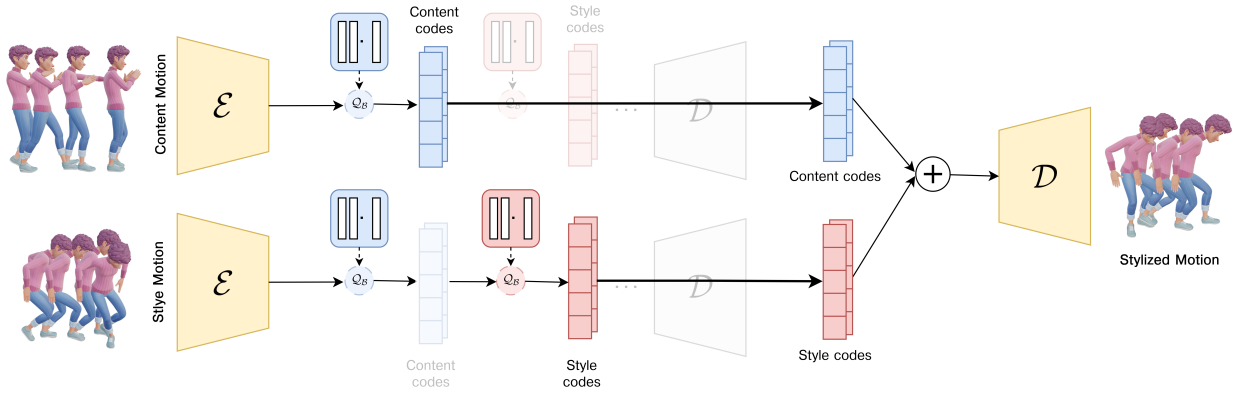


Figure 3: *Quantized Latent Code Swapping. To transfer style from one clip onto another, we encode both clips, and decode a combined embedding constructed by adding the content code (blue) from the content clip and the style code (red) from the second clip.*

better gradient propagation, we replace it with a soft assignment probability,

$$q(z=c_i|r)=\frac{\exp(-||r-c_i||^2/\tau)}{\sum_j \exp(-||r-c_j||^2/\tau)}, \tag{15}$$

where τ is a tunable hyperparameter.

3.5. Quantized Code Swapping

After learning a quantized motion representation that effectively disentangles style and content into separate codebooks, the style transfer tasks can be easily performed at inference time.

To achieve style transfer, we introduce *Quantized Code Swapping* (see Fig. 3). First, the content clip is encoded using the encoder $\mathcal{E}$ and quantization layers $\mathcal{Q}_{\mathcal{B}_{i<N}}$, yielding the corresponding codes $\mathbf{z}_{i,\text{content}}$. Similarly, the style clip is encoded to obtain $\mathbf{z}_{i,\text{style}}$. We then swap the codes after a specified residual layer s, replacing those from the content clip with the corresponding codes from the style clip. The resulting set of codes are then combined and passed through the decoder to produce the final motion,

$$\bar{\mathcal{M}}=\mathcal{D}\left(\sum_{i=0}^{s}\mathbf{z}_{i,\text{content}}+\sum_{i=s+1}^{N-1}\mathbf{z}_{i,\text{style}}\right). \tag{16}$$

The resulting motion preserves the content of the original clip while adopting the style of the reference style clip.

4. EXPERIMENTS AND RESULTS

We evaluate the effectiveness of our method in disentangling style from content in motion clips and demonstrate its applicability across several tasks, including style transfer, style transition, inverse style, motion blending and data augmentation. Finally, we present a detailed quantitative analysis of style transfer task, along with ablation studies. For a comprehensive overview of the results, including style transfer and other demonstrations, please refer to the supplementary video.

4.1. Style-Content Disentanglement

In this section, we evaluate the disentanglement between style and content in our motion embedding via low-dimensional projection of the residual vectors and by performing applications such as content extraction and style interpolation using interpretable operations.

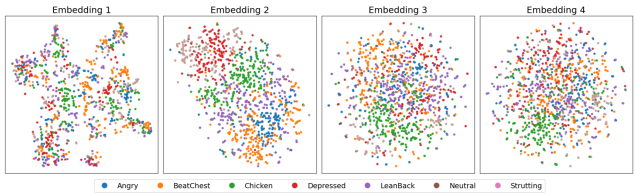
Low-dimensional projection. Fig. 4 presents a TSNE [VdMH08] visualization of the residual embeddings $\mathbf{r}_i$ for each layer, color-coded by the style labels. As shown in Fig. 4a, even without contrastive learning, the RVQ-VAE architecture already clusters motions of same style close together in the second codebook. When the contrastive loss is introduced, the separation between styles becomes more distinct (Fig. 4b). Adding the mutual information loss further strengthens the disentanglement (Fig. 4c). Notably, the model also exhibits disentanglement for styles that were unseen during training (Fig. 4d).

Extracting content. Given the residual structure of the embeddings, the content of a motion clip can be readily extracted by decoding using only the content codebooks while discarding the subsequent residual layers. Fig. 5 shows a comparison between full motion reconstruction and a reconstruction using only the content codes. As can be seen, the style is effectively removed while the global trajectory, feet placement and semantics remain close to the original motion. Interestingly, since the *100STYLE* dataset is a highly stylized dataset, the neutral motion extracted via our disentanglement keeps its hands extended to the side instead of more closely following the *Neutral* style.

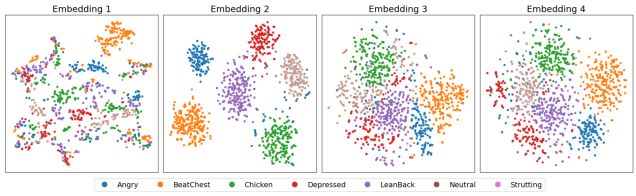
Style Interpolation. Our representation enables interpolating between content motion and stylized motion by scaling the style codes prior to decoding.

$$\bar{\mathcal{M}}=\mathcal{D}\left(\sum_{i=0}^{s}\mathbf{z}_i+\alpha\sum_{i=s+1}^{N-1}\mathbf{z}_i\right) \tag{17}$$

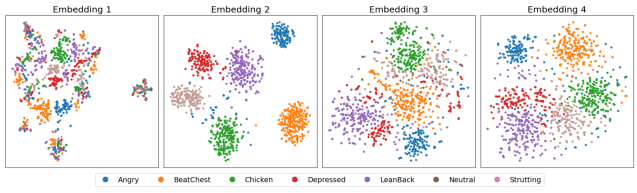
where $0<\alpha<1$ is the interpolation factor. As shown in Fig. 6, intermediate values of α determine the strength of the stylization applied to the content motion. In particular, when $\alpha=0$, the result corresponds to *content extraction*.



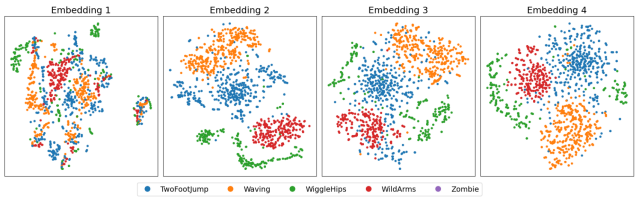
(a) RVQ trained for reconstruction.



(b) With contrastive learning



(c) With contrastive learning and mutual info loss.



(d) With contrastive learning and mutual info loss (Unseen styles).

Figure 4: *TSNE plots of residual embeddings illustrating style-content disentanglement in latent space. The colors indicated different style labels.* ***(a)*** *Without contrastive learning, styles begin to weakly cluster from the second codebook onward. Embedding 1 remains unclustered, as intended.* ***(b)*** *With contrastive learning, the separation between styles becomes more pronounced.* ***(c)*** *With mutual information loss the disentanglement is further enhanced.* ***(d)*** *The model also disentangles styles never seen during training, demonstrating its generalization ability.*

4.2. Style Transfer

Style transfer from a style reference clip onto a content motion clip can be achieved using the *quantized code swapping* technique described in Sec. 3.5. In Fig. 7, we present results of transferring the styles *Old* and *LeanBack* from *100STYLE* dataset onto two different content clips. The generated motion preserves the trajectory of the content motion (shown in blue) while incorporating the stylistic characteristics of the reference clip (shown in pink). This inference-time style transfer confirms that coarse information in the motion encodes the content while finer details correspond to the style.

Since our learned latent space is both interpretable and disentangled, it enables zero-shot style transfer to new unseen styles. As illustrated in Fig. 8, two novel styles, *WildLegs* and *Zombie*, which were not seen during training are successfully applied to a content clip.

Training our model on *Aberman* dataset which consists of fewer styles (16 styles) on a more general contents further than locomotion, we can also perform style transfer for these motions. Fig. 9 illustrates several examples of style transfer for *Depressed*, *Strutting* and *Zombie* styles.

4.3. Style Transition

Given a content motion, we can transition between various styles by concatenating style codes from different style clips along the time dimension before decoding. This enables generating long motion sequences and switching between multiple styles. As seen in Fig. 10, our method can transition between very different styles such as *Zombie*, *WideLegs* and *WhirlArms*, none of which were seen during training.

The *100STYLE* dataset contains only individual clips for each style, so transitions between different styles are never observed during training. Thanks to the power of the learned representation, we can reliably perform style transitions at inference time.

4.4. Style Inversion

In contrast to adding the style codes $z_{i>s}$ during reconstruction, one can subtract them to effectively invert the style of the input motion. Fig. 11 demonstrates the results of this operation, revealing that certain styles can be interpreted as the inverse of one another. For instance, inverting the *ArmsFolded* style results in the arms being spread apart. Similarly, inverting *PigeonToed* and *WideLegs* results in *DuckFeet* and *NarrowLegs* respectively.

Beside being an interesting application, this reflects that our training strategy produces an interpretable latent space that matches our natural intuition of style.

4.5. Motion Blending

Effectively stitching and blending between two motion clips is a key challenge to enable faster animation creation. Naive approaches to stitching of two clips often produce discontinuities in the transition. Using our RVQ-VAE, we can smoothly stitch two motion clips by concatenating the latent codes of the two clips and decoding the result. The final output seamlessly blends the motions without discontinuities and eliminating the need for manually designed blend functions.

4.6. Data Augmentation

As demonstrated in previous works [AKB*24, MU22], data augmentation and bias reduction can significantly enhance motion generation with deep learning models. Our interpretable latent space also enables new data augmentation methods.

Content interpolation. Interpolating content codes, $z_{i\leq s}$ between two different motion clips results in a valid motion with

(a) *Style: March*

(b) *Style: On the phone right*

Figure 5: *Content extraction by decoding motion using only the first codebook.* ***Left (blue):*** *ground truth,* ***Middle (pink):*** *full reconstruction,* ***Right (texture):*** *extracted content. Best viewed in color.*

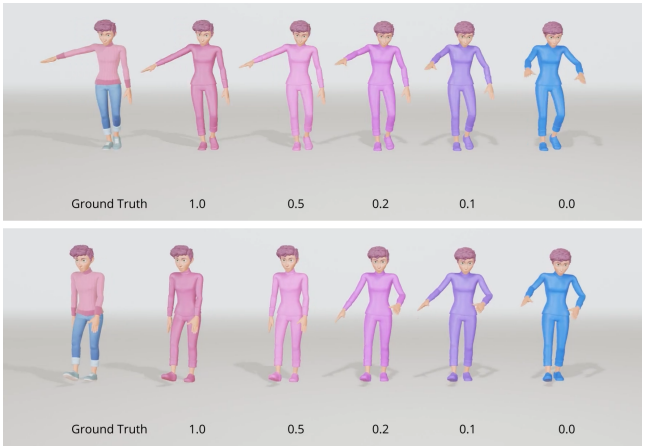
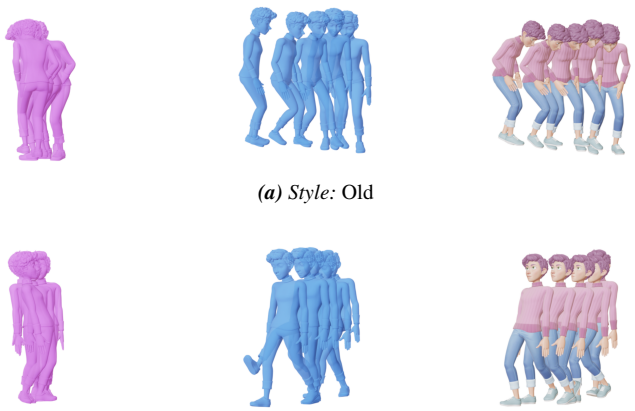


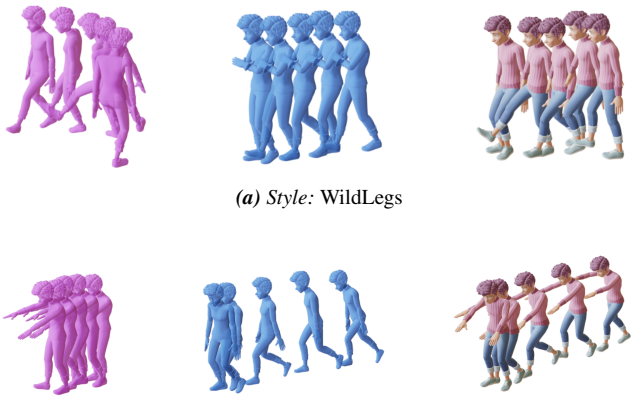
Figure 6: *Interpolation between content and stylized motion. Going from left to right, more of the style code is removed. The resulting motion smoothly approaches* Neutral *motion.*



(a) *Style:* Old

(b) *Style:* Leanback

Figure 7: *Style transfer results on* 100STYLES. ***Left:*** *Style clip,* ***Middle:*** *Content clip,* ***Right:*** *Style transfer result.*



(a) *Style:* WildLegs

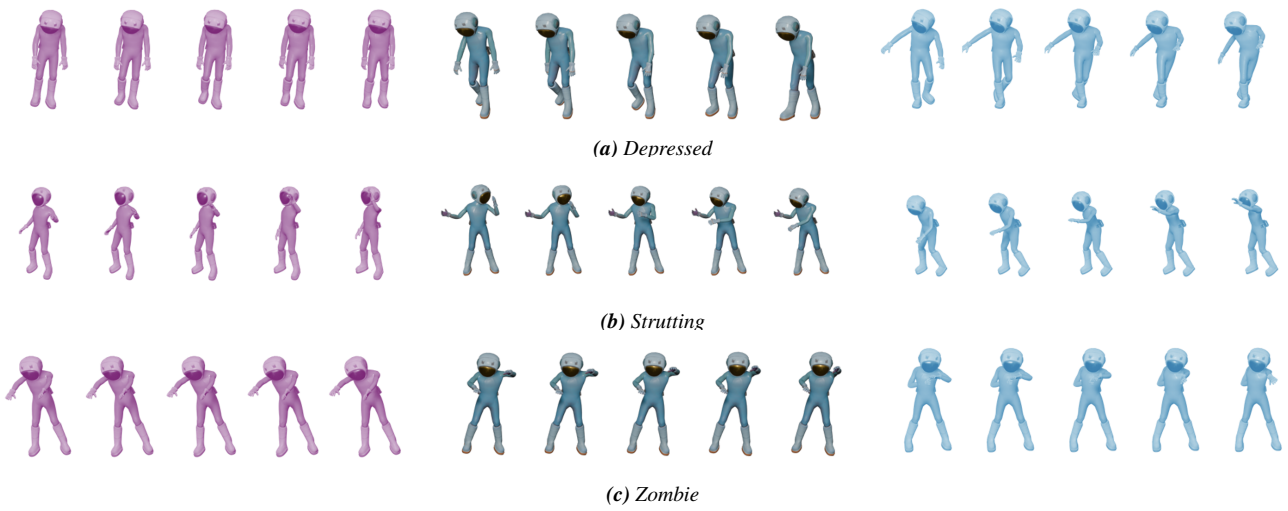
(b) *Style:* Zombie

Figure 8: *Style transfer for* ***Unseen*** *styles.* ***Left:*** *Style clip,* ***Middle:*** *Content clip,* ***Right:*** *Style transfer result.*

novel content. This technique can be used to enhance the diversity of motion trajectories in the dataset. As shown in the supplementary video, interpolating between two content codes results in a trajectory that effectively averages the original trajectories. This approach can reliably introduce new turns and sequences of actions that were not present in the original dataset.

Random style selection As discussed in Sec. 4.3, our model can generate smooth transitions between styles of different reference clips. Even in the absence of a style clip, we can apply multiple styles to a content clip by replacing its style codes, $z_{i>s}$, with different codes randomly selected from our residual codebooks. The resulting motion remians smooth while exhibiting multiple styles, enabling natural style transitions that are absent in the original dataset. As shown in Fig. 12, the generated motion follows the content from the input clip (on the left), while sequentially exhibiting multiple distinct styles.

These two techniques are orthogonal and can be easily combined to increase both trajectory diversity and style transition variety in the dataset.



(a) *Depressed*

(b) *Strutting*

(c) *Zombie*

Figure 9: *Style transfer results on* Aberman *dataset.* ***Left:*** *style clips,* ***Middle:*** *transfer results,* ***Right:*** *content clips.*

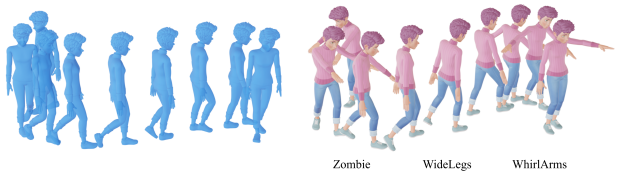


Figure 10: *Our method enables switching between multiple styles while following a longer content sequence. The styles used in this motion were unseen during training.*

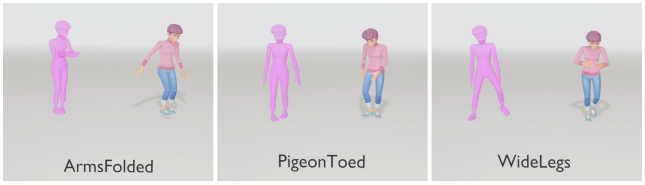


Figure 11: *Style inversion. By subtracting the style codes from motion content we obtain a new motion with* inverted *style.*

Figure 12: *Our pipelines enables data augmentation by following a content motion and sampling random style codes.* ***Left:*** *content motion,* ***Right:*** *result motion.*

Table 1: *Baseline comparison on* 100STYLE *dataset with LPN-Style [MSK22]. Numbers in parenthesis show top-5 accuracy. LPN-Style* denotes the experiment excluding 8 broken styles (those with accuracy lower than 10%).*

	Style-acc (%) ↑		
Method	Test	Unseen style	Fine-tuned
LPN-Style	73.24(87.50)	✗	86.92(**100.0**)
LPN-Style*	77.53(91.70)	✗	86.92(**100.0**)
Ours	**83.20**(**95.12**)	**68.95**(**98.83**)	**96.88**(99.02)
	Content-err (m) ↓		
Ours	0.075 ± 0.056	0.091 ± 0.073	0.079 ± 0.065
	Cross-classification (%) ↓		
Ours	0.78	0.00	0.00

4.7. Quantitative analysis

In this section, we quantitatively compare our method with the baselines [MSK22] and [GMZ*24], in terms of *style preservation accuracy* and *content trajectory deviation*. We compare against the motion-based supervised setting for [GMZ*24] as it is closest to our inference configuration. We define *style preservation accuracy*, A_S as the classification accuracy of a pretrained style classifier over the generated motion. *Content trajectory deviation*, D_C measures the deviation of the generated motion's root trajectory from that of the content motion. For more details on the metric definition and dataset splits, please refer to Appendix C.

4.7.1. Style Accuracy

In Table 1, we report style accuracy, A_S, for the test and the unseen style subsets of the *100STYLE* [MSK22] dataset. As a baseline, we use the locomotion controller LPN-Style provided by [MSK22] to

Table 2: *Comparison to [GMZ*24] on* Aberman *and* Xia *datasets. Numbers in parenthesis show top-3 accuracy.*

Method	Style-acc (%) ↑			Content-err (m) ↓			Cross-cls (%) ↓	
	Train	Test	*Xia* styles	Train	Test	*Xia* styles	Train	Test
GenMoStyle	73.47(93.36)	76.02(92.34)	30.77 (76.92)	$\mathbf{0.018 \pm 0.012}$	$\mathbf{0.019 \pm 0.014}$	$\mathbf{0.024 \pm 0.018}$	7.14	8.16
Ours	**83.38**(**96.88**)	**80.91**(**92.27**)	**53.85**(**76.92**)	0.028 ± 0.028	0.029 ± 0.027	0.047 ± 0.026	**3.06**	**5.10**

generate stylized motion to measure their test A_S. In practice, we observed that LPN-Style produced severely broken or incorrectly stylized motion for certain styles, with accuracies falling below 10%. To ensure a fair evaluation, we report their performance both including and excluding these problematic styles in the test set. Our method scores top-1 A_S of 83.20% on the test set, outperforming both versions of LPN-Style.

For unseen styles, our method achieves 68.95% top-1 and 98.83% top-5 A_S, whereas LPN-Style is incapable of zero-shot generalization. Since LPN-Style cannot generate motion for unseen styles, they must fine-tune their FiLM module [PSDV*18] on the new styles. This fine-tuning improves their performance to 86.92% A_S, while our similarly fine-tuned model can achieve a score of 96.88%, significantly outperforming LPN-Style.

We evaluate our method on the *Aberman* [AWL*20] and *Xia* [XWCH15] datasets which contain more generic stylized motions beyond locomotion. For training, we used the retargeted motions from *Aberman* provided by GenMoStyle [GMZ*24]. Our model is trained solely on the *Aberman* training set, but we evaluate it both on the *Aberman* and *Xia* datasets. Since these datasets contain fewer styles (16 for *Aberman* and 8 for *Xia*), we report top-3 A_S rather than top-5 A_S used for *100STYLE*.

As reported in Table 2, our method outperforms GenMoStyle for style accuracy across all subsets and both top-1 and top-3 A_S. Notably, while GenMoStyle do not use the *Xia* dataset to train their style and content encoders, their general motion autoencoder is pre-trained on all *CMU* [CMU], *Aberman* and *Xia* datasets. In contrast, we use the *Xia* dataset exclusively for evaluation.

4.7.2. Content Error

We evaluate the content deviation D_C of our method across the three datasets. Since the *100STYLE* dataset is a locomotion dataset, we observe a higher test deviation of 7.5 cm (see Table 1) compared to 2.9 cm and 4.7 cm on *Aberman* and *Xia* datasets (see Table 2), respectively.

As [MSK22] train a locomotion controller rather than a style transfer model, their content deviation error cannot be measured. On the *100STYLE* dataset, we find that much of the deviation arises from errors in predicted velocities during sharp turns in the content motion. Since the final motion is calculated by integrating the velocities over time, these velocity errors accumulate and lead to larger deviations toward the end of the sequence. Operating in a global reference space could potentially improve content preservation; however, we leave this investigation for future work.

As content and style are more closely coupled for non-locomotion data, we see a stronger correlation between better style removal and worse content deviation for such data. This is corroborated when comparing content deviation and cross-classification rate between our method and GenMoStyle. The cross-classification rate measures the percentage of times the style classifier classified the generated motion as the style of the content clip. On *Aberman* and *Xia* datasets, GenMoStyle provides lower content deviation compared to our method. However, we see that GenMoStyle's generated motion is misclassified as the content style 8% over test data compared to 5% for our method. This suggests that our method is better able to remove the style of the content motion. As a result, our method removes more of the content clip motion. By adjusting the number of content codebooks s, one can fine-time the model performance for their preferred balance of style transfer and content preservation.

4.8. Ablation Studies

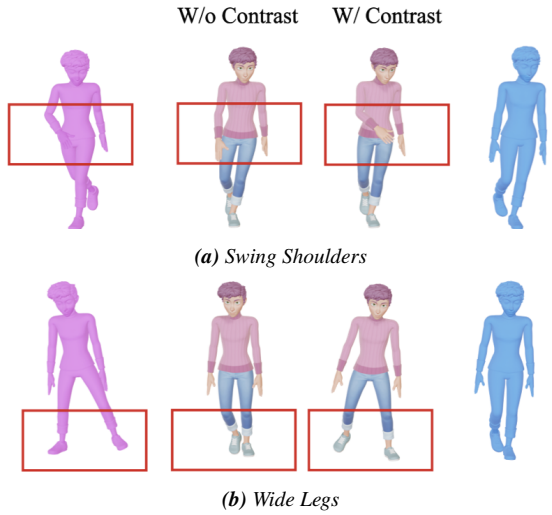
4.8.1. Effect of contrastive learning

In this section, we evaluate the effect of contrastive loss, $\mathcal{L}_{cl}$ and mutual information loss, $\mathcal{L}_{mi}$. As discussed in Sec. 4.1, adding contrastive loss to the training improves the style-content separation. This is also confirmed by our experiments on style transfer via quantized code swapping. As shown in Fig. 13, the style transfer result from the model trained with contrastive loss adheres more to the nuances of style clip and shows less leakage of content, compared to the one obtained without contrastive loss. For instance, in the *WideLegs* style, the result with contrast opens the legs wide while the one without it keeps the legs close to each other similar to the content clip. A similar phenomena can be seen in the *SwingShoulders* style when comparing the arms movement.

Table 3: *Ablation results for contrastive and mutual info losses.*

Model	Style-acc (%) ↑	Content-err (m)↓	Rec-err (m) ↓
RVQ	55.86	0.076 ± 0.063	$\mathbf{0.029 \pm 0.016}$
+ $\mathcal{L}_{con}$	68.95	$\mathbf{0.074 \pm 0.059}$	0.029 ± 0.017
+ $\mathcal{L}_{mi}$	73.24	0.081 ± 0.065	0.031 ± 0.020
+ $\mathcal{L}_{con}$ + $\mathcal{L}_{mi}$	**87.50**	0.077 ± 0.059	0.031 ± 0.019

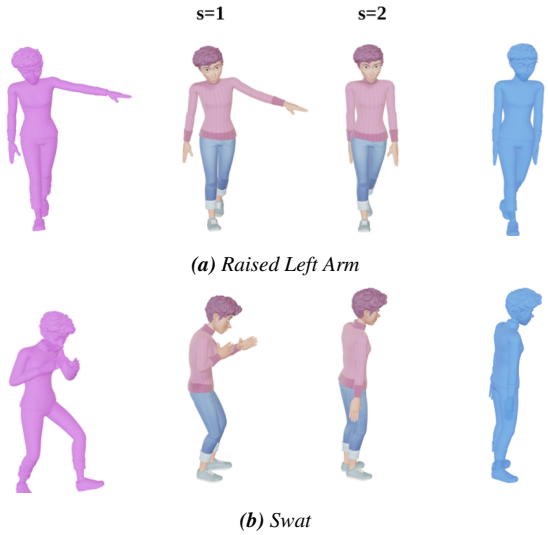
Our quantitative measures are consistent with the qualitative results mentioned above. In Table 3, we measure A_S, D_C and reconstruction error *L2P* for a base RVQ-VAE, adding each losses individually and training with both losses simultaneously. Across all four models, we measure similar reconstruction error with only training with RVQ-VAE with the lowest *L2P*. The difference between our full model and the base RVQ-VAE lies within the standard deviation range, hence, our disentanglement of the learnt space does not result in notably worse reconstruction.



(a) Swing Shoulders

(b) Wide Legs

Figure 13: *Contrastive learning improves style-content separation and results in better style transfer.* ***Left (pink):*** *style clip,* ***Middle (textured):*** *stylized result of a model trained without (middle left) and with (middle right) contrastive loss,* ***Right (blue):*** *content clip.*



(a) Raised Left Arm

(b) Swat

Figure 14: *Ablation on different cut-offs for style-content codebooks in style transfer application.* ***Left (pink)****: style clip,* ***Middle****: style transfer result,* ***Right (blue)****: content clip.*

We see that adding each individual loss improves style classification accuracy compared to the base RVQ-VAE. As we measure D_C after swapping codes, we see that adding $\mathcal{L}_{con}$ also improves the content error, as it encourages the initial codebooks to contain most of the content information.

In contrast, since $\mathcal{L}_{mi}$ forces the content codes to have no style information, it is moved into later codebooks, resulting in a greater improvement in A_S compared to $\mathcal{L}_{con}$. However, we notice that adding $\mathcal{L}_{mi}$ results in higher D_C error. This could be since the MI loss prevents style information leakage to the first codebook, it may also remove trajectory information that is partially style-dependent from the first codebook, resulting in a higher content error. Lastly, the addition of both losses compounds their effect and results in the best style accuracy and a reasonable content trajectory error.

4.8.2. Choice of swapping cut-off

In our experiments with basic training of RVQ-VAE without contrastive and mutual info losses, the first codebook captures the content and the second codebook onwards contain information about the motion style. This is inline with the intuition that the style is the second greatest feature after content that is represented in the data. Therefore we found $s = 1$ is a good choice for the cut-off between content and style codebooks. Fig. 14 shows style transfer results for $s = 1$ and $s = 2$, where the latter almost replicates the content motion, indicating that the second codebook already captures most of the stylistic features. Although one can use only the first and second codebooks for decoding the motion in case of style transfer, we find addition of more residual style codes helps with overall motion quality and reducing artifacts such as foot sliding.

5. Conclusion and future work

In this work, we frame content-style separation and motion style transfer as a representation-learning problem. Exploiting the coarse-to-fine hierarchy of a Residual VQ-VAE, combined with novel disentanglement losses, we obtain a latent space that effectively separates the style from the content. This representation unlocks a broad range of downstream capabilities such as style transfer, style interpolation, and data-driven augmentation without per-style fine-tuning.

Qualitatively, our method maintains content motion's trajectory and timing, while adopting the style of the reference clip, even for styles never observed during training. Our quantitative analysis further corroborates our qualitative finding that our method successfully transfers styles to a different content motion with a high style accuracy. Furthermore, our method is able to perform zero-shot style transfer for unseen styles.

Our method builds on the intuitive interpretation of content and style as coarse features versus finer details. However, formally defining and isolating these two components remains challenging. Even across existing public datasets, the criteria for what is considered style are inconsistent. For instance, *kicking* is annotated as content in the *Aberman* dataset, but is classified as a style in the *100STYLE* dataset. Prior work has adopted various ways to define the style, e.g. via statistical features such as Gram matrices [HSK16], by shift and amplitude of instance normalization in latent space [AWL*20], or by separating the non-temporal features through attention masks [TZCvdP22]. In our work, we view the content as coarse global motion structures, and style as the finer-grained local details. For a locomotion-focused dataset such as *100STYLES*, global root trajectory and speed align more naturally with the content, and foot patterns, arm motions and other nuanced features align more closely with style. Similar to prior stud-

ies the boundary between content and style remains ambigious, and in practice the annotated style labels majorly influence the style-content distinction.

Furthermore, developing better quantitative metrics for style transfer remains an open task. For example, root trajectory error, a measure of content preservation, can be misleading. When applying a style like *Drunk*, the ideal output should intentionally alter the original motion's root trajectory, making a low error score a poor indicator of a successful transfer. One can find similar counter examples for other metrics provided in the literature such as geodesic distance [GMZ*24]. This is specifically more challenging for non-locomotion movements in which the expected result of style transfer might be ambiguous.

Such ambigious cases also exist in locomotion, particularly in motions with pronounced foot or hip movements. As a result, our methods exhibits reduced performance on such examples. Furthermore, as we integrate local root velocity to obtain the global character trajectory, the generated motion can drift from the content motion over longer motion sequences. Although our current formulation achieves strong results, it relies on datasets annotated with style labels. To disentangle unannotated data, one could perform style discovery through unsupervised clustering, and further combine it with our proposed pipeline. We leave the investigation of these challenges to future research.

Our investigation demonstrates the capabilities of RVQ-VAEs for interpretable motion representation and enabling flexible, inference-time manipulation of the latent space. Moreover, the simplicity and generality of our framework open up exciting directions for future research, positioning residual quantization as a promising framework for motion reuse, augmentation and transfer.

Acknowledgment

The authors would like to thank Dominik Borer for his technical support and Violaine Fayolle and Doriano van Essen for their artistic support during this project.

References

[AKB*24] AGRAWAL D., KÖNIG M., BUHMANN J., SUMNER R., GUAY M.: Trajectory augmentation for robust neural locomotion controllers. In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications-Volume 1: GRAPP, HUCAPP and IVAPP* (2024), SciTePress, pp. 23–33. 6

[AWL*20] ABERMAN K., WENG Y., LISCHINSKI D., COHEN-OR D., CHEN B.: Unpaired motion style transfer from video to animation. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 64–1. 2, 9, 10, 13

[CMU] CMU Mocap Dataset. Carnegie Mellon University https://mocap.cs.cmu.edu/. 9

[DPS*18] DUMOULIN V., PEREZ E., SCHUCHER N., STRUB F., VRIES H. D., COURVILLE A., BENGIO Y.: Feature-wise transformations. *Distill 3*, 7 (2018), e11. 2

[DWF*25] DAI M., WANG J., FAN K., JI B., ZHAO H., DONG J., DAI B.: Towards synthesized and editable motion in-betweening through part-wise phase representation. *arXiv preprint arXiv:2503.08180* (2025). 2

[FH61] FANO R. M., HAWKINS D.: Transmission of information: A statistical theory of communications. *American Journal of Physics 29*, 11 (1961), 793–794. 4

[GMZ*24] GUO C., MU Y., ZUO X., DAI P., YAN Y., LU J., CHENG L.: Generative human motion stylization in latent space. *CoRR* (2024). 2, 8, 9, 11, 13, 14

[GZWC22] GUO C., ZUO X., WANG S., CHENG L.: Tm2t: Stochastic and tokenized modeling for the reciprocal generation of 3d human motions and texts. In *European Conference on Computer Vision* (2022), Springer, pp. 580–597. 13

[HC20] HADJERES G., CRESTEL L.: Vector quantized contrastive predictive coding for template-based music generation. *arXiv preprint arXiv:2004.10120* (2020). 4

[HCAI23] HUH M., CHEUNG B., AGRAWAL P., ISOLA P.: Straightening out the straight-through estimator: Overcoming optimization challenges in vector quantized networks. In *International Conference on Machine Learning* (2023), PMLR, pp. 14096–14113. 4

[HS22] HO J., SALIMANS T.: Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022). 2

[HSK16] HOLDEN D., SAITO J., KOMURA T.: A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (ToG) 35*, 4 (2016), 1–11. 10

[JPL22] JANG D.-K., PARK S., LEE S.-H.: Motion puzzle: Arbitrary motion style transfer by body part. *ACM Transactions on Graphics (TOG) 41*, 3 (2022), 1–16. 2

[KW*19] KINGMA D. P., WELLING M., ET AL.: An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning 12*, 4 (2019), 307–392. 2

[ŁCS*20] ŁAŃCUCKI A., CHOROWSKI J., SANCHEZ G., MARXER R., CHEN N., DOLFING H. J., KHURANA S., ALUMÄE T., LAURENT A.: Robust training of vector quantized bottleneck models. In *2020 International Joint Conference on Neural Networks (IJCNN)* (2020), IEEE, pp. 1–7. 4, 13

[LKK*22] LEE D., KIM C., KIM S., CHO M., HAN W.-S.: Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 11523–11532. 2, 3, 13

[MSK22] MASON I., STARKE S., KOMURA T.: Real-time style modelling of human locomotion via feature-wise transformations and local motion phases. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 5*, 1 (2022), 1–18. 2, 8, 9, 13, 14

[MU22] MAEDA T., UKITA N.: Motionaug: Augmentation with physical correction for human motion prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2022), pp. 6427–6436. 6

[NYC19] NA S., YOO S., CHOO J.: Miso: Mutual information loss with stochastic style representations for multimodal image-to-image translation. *arXiv preprint arXiv:1902.03938* (2019). 4

[PJL21] PARK S., JANG D.-K., LEE S.-H.: Diverse motion stylization for multiple style domains via spatial-temporal graph-based generative model. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 4*, 3 (2021), 1–17. 2

[PSDV*18] PEREZ E., STRUB F., DE VRIES H., DUMOULIN V., COURVILLE A.: Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence* (2018), vol. 32. 2, 9

[RGS*24] RAAB S., GAT I., SALA N., TEVET G., SHALEV-ARKUSHIN R., FRIED O., BERMANO A. H., COHEN-OR D.: Monkey see, monkey do: Harnessing self-attention in motion diffusion for zero-shot motion transfer. In *SIGGRAPH Asia 2024 Conference Papers* (2024), pp. 1–13. 2

[S*59] SHANNON C. E., ET AL.: Coding theorems for a discrete source with a fidelity criterion. *IRE Nat. Conv. Rec 4*, 142-163 (1959), 1. 13

[SJL*24] SONG W., JIN X., LI S., CHEN C., HAO A., HOU X., LI N., QIN H.: Arbitrary motion style transfer with multi-condition motion latent diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 821–830. 2

[SMK22] STARKE S., MASON I., KOMURA T.: Deepphase: Periodic autoencoders for learning motion phase manifolds. *ACM Transactions on Graphics (ToG) 41*, 4 (2022), 1–13. 2

[TFI*23] TIAN Y., FAN L., ISOLA P., CHANG H., KRISHNAN D.: Stablerep: Synthetic images from text-to-image models make strong visual representation learners. In *Advances in Neural Information Processing Systems* (2023), Oh A., Naumann T., Globerson A., Saenko K., Hardt M., Levine S., (Eds.), vol. 36, Curran Associates, Inc., pp. 48382–48402. 4

[TRG*23] TEVET G., RAAB S., GORDON B., SHAFIR Y., COHEN-OR D., BERMANO A. H.: Human motion diffusion model. In *The Eleventh International Conference on Learning Representations* (2023). 2

[TWW*23] TANG X., WU L., WANG H., HU B., GONG X., LIAO Y., LI S., KOU Q., JIN X.: Rsmt: Real-time stylized motion transition for characters. In *ACM SIGGRAPH 2023 Conference Proceedings* (2023), pp. 1–10. 2

[TWW*24] TANG X., WU L., WANG H., WU Y., HU B., LI S., GONG X., LIAO Y., KOU Q., JIN X.: Decoupling contact for fine-grained motion style transfer. In *SIGGRAPH Asia 2024 Conference Papers* (2024), pp. 1–11. 2

[TZCvdP22] TAO T., ZHAN X., CHEN Z., VAN DE PANNE M.: Style-erd: Responsive and coherent online motion style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 6593–6603. 10

[VdMH08] VAN DER MAATEN L., HINTON G.: Visualizing data using t-sne. *Journal of machine learning research 9*, 11 (2008). 5

[VDOV*17] VAN DEN OORD A., VINYALS O., ET AL.: Neural discrete representation learning. *Advances in neural information processing systems 30* (2017). 2, 13

[XWCH15] XIA S., WANG C., CHAI J., HODGINS J.: Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 1–10. 2, 9

[YSZ*24] YAO H., SONG Z., ZHOU Y., AO T., CHEN B., LIU L.: Moconvq: Unified physics-based motion control via scalable discrete representations. *ACM Transactions on Graphics (TOG) 43*, 4 (2024), 1–21. 3

[ZRA23] ZHANG L., RAO A., AGRAWALA M.: Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision* (2023), pp. 3836–3847. 2

[ZXJ*24] ZHONG L., XIE Y., JAMPANI V., SUN D., JIANG H.: Smoodi: Stylized motion diffusion model. In *European Conference on Computer Vision* (2024), Springer, pp. 405–421. 2

Appendix A: Background

VQ-VAEs [VDOV*17] are powerful and compact representation models that have demonstrated success in encoding complex data, including human motion [GZWC22]. However, preserving high quality reconstruction when encoding a large and diverse dataset typically requires the codebook size to grow exponentially, posing challenges for scalability [S*59, LKK*22]. Residual VQ-VAE (RVQ-VAE) addresses this limitation by employing a hierarchical set of codebooks that encode residual errors at multiple levels [LKK*22]. Rather than mapping the entire continuous latent vector to a single code drawn from a large codebook, the RVQ mechanism performs sequential refinement using N codebooks $[\mathcal{B}_0, \mathcal{B}_1, \cdots, \mathcal{B}_{N-1}]$. The first codebook quantizes the initial continuous latent vector, after which a residual is computed by subtracting the quantized output from the original latent. The second codebook then quantizes this residual, producing an additional refinement. This hierarchical residual quantization process continues through all N codebooks, and the final latent representation is obtained by summing the quantized outputs from all stages.

Two main losses are involved in training a VQ-VAE. The first is the commitment loss, which encourages the encoder to generate continuous embeddings $\mathbf{r}_i$ that are closer to their corresponding quantized codes $\mathbf{z}_i$.

$$\mathcal{L}_{commit} = ||\mathbf{r}_i - \text{sg}(\mathbf{z}_i)||_2^2, \tag{18}$$

where $\text{sg}(.)$ is the stop gradient operator.

The second component is a VQ-objective loss that updates the codebook to align with the continuous embeddings assigned to it,

$$\mathcal{L}_{VQ} = ||\text{sg}(\mathbf{r}_i) - \mathbf{z}_i||_2^2. \tag{19}$$

Rather than optimizing this objective directly, we employ an Exponential Moving Average (EMA) update which has been shown to improve the robustness and stability of codebook training [ŁCS*20],

$$N_i \leftarrow \gamma N_i + (1-\gamma) \sum_{batch} \mathbb{1}_{match}(\mathcal{Q}_{\mathcal{B}_i}(\mathbf{r}_i), \mathbf{c}_i), \tag{20}$$

$$\mu_i \leftarrow \gamma \mu_i + (1-\gamma) \sum_{batch} (\mathbf{r}_i \cdot \mathbb{1}_{match}(\mathcal{Q}_{\mathcal{B}_i}(\mathbf{r}_i), \mathbf{c}_i)), \tag{21}$$

$$\mathbf{c}_i = \frac{\mu_i}{N_i}, \quad \forall \mathbf{c}_i \in \mathcal{B}_i. \tag{22}$$

Here, $\mathbf{c}_i$ corresponds to a codeword in codebook i and $\mathbb{1}_{match}(\mathbf{x}, \mathbf{y}) = 1$ if $\mathbf{x} = \mathbf{y}$ and 0 otherwise, showing if the specific code was selected for that data. N_i tracks the usage and μ_i tracks the mean code, while γ is a discount factor.

Appendix B: Training hyperparameters

The hyperparameters used for training our RVQ-VAE over different datasets are detailed in Table 4.

Appendix C: Metrics

We define style accuracy, A_S, as the accuracy of a style classifier on motion generated after applying style transfer. We train one classifier with identical architecture as [GMZ*24]. This consists of four 1D convolutional layers followed by three deconvolutional layers and one linear layer. The classifiers are trained on all styles in their respective datasets and have test accuracies of 96.87% over the *Aberman* [AWL*20] dataset and 98.57% over the *100STYLE* dataset [MSK22].

Table 4: *Training hyperparameters values for reconstruction on 100Styles [MSK22] dataset. Values in parenthesis represent configuration used for training on* Aberman *[AWL*20] dataset.*

Hyperparameter	Value
Num residual layers	8 (4)
Num codes per codebook	512 (256)
Latent size	256
Conv feature size	512
Learning rate	1.0e-4
Max grad norm clip	1.0
$\mathcal{L}_{rec}$ coefficient	1.0
$\mathcal{L}_{FK}$ coefficient	0.01
$\mathcal{L}_{vel}$ coefficient	0.1
$\mathcal{L}_{acc}$ coefficient	0.05
$\mathcal{L}_{commit}$ coefficient	0.05
$\mathcal{L}_{con}$ coefficient	0.005 (0.05)
$\mathcal{L}_{mi}$ coefficient	0.02 (0.12)

To quantify how well the content of the original motion is preserved, we measure the deviation of the root trajectory of the generated motion from the root trajectory of the content clip. This is computed as,

$$D_C = \frac{1}{T} \sum_{t=0}^{T} |{p^r}_t - \hat{p^r}_t|, \tag{23}$$

where ${p^r}_t$ is the position of the content clip root at frame t and $\hat{p^r}_t$ is the generated root position at the same frame.

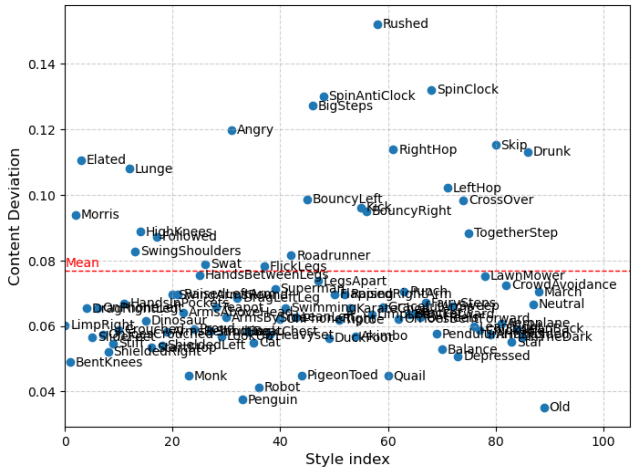
Appendix D: Dataset details

Although the *100STYLE* dataset with its 100 unique styles categories is a large enough dataset to train a model for style transfer, Aberman and Xia datasets are much smaller. In particular, the Aberman dataset has only 193 minutes of motion data with 16 style categories while Xia dataset has 25 minutes of data with 8 style categories.

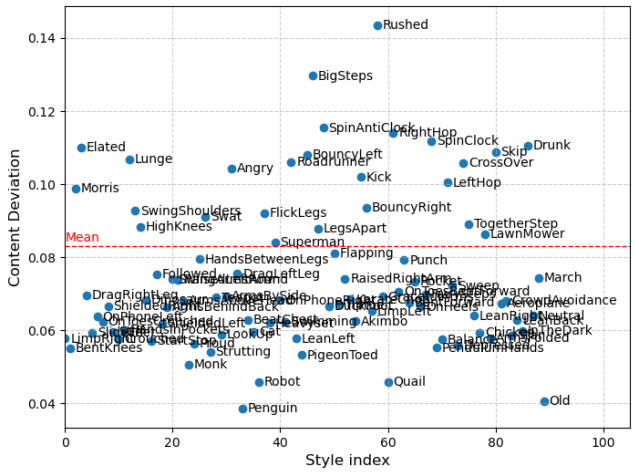
We report A_S and D_C for train, test, and unseen styles subsets of the datasets. The train subset contains the same clips as the model was trained on. The test subset contains clips withheld during training but with the same styles seen during training.

For the *100STYLE* dataset, the unseen dataset contains clips with styles that have never been seen by the model before. These styles are selected by choosing the last 10 styles in the alphabetical order. For comparison with LPN-Style [MSK22], we retrain our model on the same data split as theirs, where the unseen styles include *HandsInPockets*, *Roadrunner*, *Skip*, *Star*, *WildArms* and the model is trained on the remaining 95 styles.

For the Aberman dataset, we split the Aberman into train and test subsets similar to [GMZ*24]. However, we withhold the complete Xia dataset as the unseen subset. Note that some of the styles



(a) Train



(b) Test

Figure 15: *Plotting per class content deviation for train and test subsets of the 100Styles [MSK22] dataset.*

in the Xia dataset are also present in the Aberman dataset. We choose to split this the datasets as such to for consistent comparison with [GMZ*24]. However, it's worth noting that the original GenMoStyle only uses Xia dataset as the content, never testing on unseen styles.

Appendix E: Per-style class content deviation

Figure 15 plots the content deviation, D_C, for the train and test subsets of the *100STYLE* [MSK22] dataset. We observe that the distribution of deviations is skewed, with most styles demonstrating a content deviation well below the mean deviation of their respective subsets. Specifically, of the 90 styles used for training, only 23 styles have an average deviation that exceeds the mean of the training subset. For the test subset, 25 styles have an average deviation greater than the test subset's mean. In both training and testing, more energetic styles such as *Rushed*, *Spin*, *Bounce*, and *Hop* have worse content deviation.

Table 5: *Rec-err (L2P) (m) vs. different number of codebooks used for reconstruction (#s).*

#s	RVQ-8-512	RVQ-8-256	RVQ-5-512	RVQ-3-512
1	0.100	0.101	0.107	0.097
2	0.061	0.065	0.068	0.074
3	0.052	0.056	0.056	**0.067**
4	0.046	0.051	0.051	–
5	0.042	0.048	**0.048**	–
6	0.039	0.045	–	–
7	0.037	0.043	–	–
8	**0.036**	**0.041**	–	–

Appendix F: Varying number of residual layers

Using the residual VQ-VAE architecture, one can use different number of codebooks for reconstructing a motion. Increasing the number of codebooks gradually improves the reconstruction error, as shown in Table 5. However, the improvement gets less and less significant as the number of codebooks increase. This aligns with the intuition of coarse-to-fine representation where the later codebooks only add finer details to the constructed motion. Models in Table 5 follow the naming convention of *RVQ-N-X* where *N* is the total number of codebooks and *X* refers to the size of each codebook. Ablation on number of codes in the codebook shows small degrade in the performance when reducing the codebook size.