JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering, Vol. 18, No. 4, December 2024, pp. 181-195

Generative but Controllable Motion Generation through Key Poses and Context Vectors

Jaeyeong Ryu, Soungsill Park, and Youngho Chai*

Graduate School of Advanced Imaging Science, Chung-Ang University, Seoul, South Korea

puls36@cau.ac.kr, pssil12@cau.ac.kr, yhchai@cau.ac.kr

Abstract

We investigate a motion generation model capable of producing desired motions using minimal pose data. Although similar to conventional motion interpolation models in terms of motion data input, the key difference lies in our model's ability to generate diverse motions tailored to user intentions. To differentiate it from motion interpolation models, we establish motion recognition and controllable motion generation systems utilizing pretrained generative models. We develop the motion recognition system using a latent vector derived from the pretrained model's encoder, which encodes substantial contextual information and can be identified by a simple linear support vector machine. The controllable motion generation system employs the recognized latent vector and input poses, based on the pretrained model's decoder. In experiments, our model demonstrates superior generated motion accuracy compared to text-based motion generation models. We also compare our model with motion interpolation models, showing comparable performance. Furthermore, we validate the efficacy of skip connections through qualitative evaluations. Finally, we confirm that our system can generate various types of motion utilizing latent vectors.

Category: Smart and Intelligent Computing

Keywords: Motion generation; Action recognition; Generative pre-trained transformer; Metaverse applications

I. INTRODUCTION

Research on generating human motion profoundly affects various sectors, including animation, gaming, and Metaverse environments [1]. Realistic avatar motions are crucial for crafting immersive experiences in these fields. For instance, in animation, non-realistic character movements can reduce viewer immersion. Traditionally, obtaining accurate motion data has involved labor-intensive techniques such as manual motion creation or motion capture equipment. These methods are time-consuming and arduous [2]. To overcome these challenges, researchers have developed automated motion generation models employing various constraints, such as action-class labels, text, and music, to generate motion. Utilizing action-class labels facilitates the rapid generation of various motion types associated with the specified labels [3-5]. However, producing motions that incorporate the desired poses can be difficult. Studies [6-8] employing text as input data can describe desired motion in greater detail than action-class labels, thus generating relatively specific motion. Nevertheless, accurately describing all joint movements to generate the precise motion requested by the user is complex. Our generation model employs pose data as input to overcome these challenges, enabling the creation of precise motion and allowing users to develop customized motion as desired.

Open Access http://dx.doi.org/10.5626/JCSE.2024.18.4.181 http://jcse.kiise.org

Received 09 September 24; **Accepted** 10 December 24

*Corresponding Author

pISSN: 1976-4677 eISSN: 2093-8020

We selected the conditional variational autoencoder (CVAE) structure for our generative model due to its inherent advantages. In variational autoencoders (VAEs), the latent vector, condensed by the encoder, holds significant information and can be utilized across various application systems. Previous studies [3, 9] have employed latent vectors for motion recognition. Another study [4] appended the latent vectors with action-class labels and constrained the generated motions using these labels. This paper also utilizes latent vectors for tasks such as motion recognition and controllable motion generation. By adopting the CVAE structure, we can utilize two encoders and train them concurrently. The proposed model includes a full-motion encoder that learns the entire sequence of actions, and a key-motion encoder that processes only a limited set of pose data. By aligning the predicted mu (μ) and sigma (σ) values from both encoders, the key-motion encoder and full-motion encoder share their latent space. Consequently, the key-motion encoder can reliably extract latent vectors based on key poses.

The primary novelty of the proposed model is the incorporation of a skip connection [10] into the CVAE structure to address the shortcomings of previous motion generation models. Prior studies have often faced issues with training models on diverse datasets, typically converging to intermediate values [11]. Similarly, training motion generation models with extensive datasets often results in convergence to average motion. Moreover, there are instances where the generated motion does not correlate with the input data due to posterior collapse [12]. We theorize that these issues arise because the decoder depends exclusively on the latent vector to generate motion. To counter this, we introduce a skip connection that transmits the input motion data directly to the decoder. This allows the decoder to consider both the latent vector and the input motion data, significantly improving its motion generation capabilities. In the experimental section, we qualitatively compare motions generated with and without the skip connection. Training the model with extensive motion data demonstrates that the skip connection effectively preserves the input pose data, thereby enhancing motion generation accuracy.

The pretrained generative model can serve as the foundation for a motion recognition system and a controllable motion generation system. We develop a system that, using this pretrained motion decoder, can determine the class of motion to generate based on the latent vector and create motion from the input poses. We hypothesize that the latent vector captures contextual information essential for differentiating various motions. To substantiate this, we implement a motion recognition system that employs the pretrained motion encoder. Even with limited pose data, the motion recognition system achieves recognition rates nearing 80%. Subsequently, distinct latent vectors specific to each motion class are produced. The controllable motion generation system can then generate motions in any desired class using these categorized latent vectors. Input poses can be selected from the existing motion dataset to produce the desired motion.

Our motion generation model differs from motion interpolation models [13, 14]. While both models employ pose data, our proposed motion generation model can create diverse motions using pose data and latent vectors. To demonstrate this, we constructed a controllable motion generation system as described earlier. This system can identify the class of motion to be generated based on a latent vector. Even with identical pose data input, the generated motions can vary depending on the latent vector. This characteristic is unique to our motion generation model and distinguishes it from traditional motion interpolation models. The experiments showcase the differences between motions generated by the interpolation and our proposed systems using the same input pose data. The experimental results show that the interpolation model linearly connects input motions, while our system generates a variety of motion types based on the latent vector.

The primary contributions of our research are as follows. First, we introduce a motion generation model using key pose data. Second, to address limitations in existing motion generation models, we incorporate a skip connection and confirm its efficacy through experiments. Third, we have developed an action recognition system and a controllable motion generation system utilizing a pretrained generative model. Lastly, we demonstrate the differences between the motion interpolation model and our proposed generation model through experimental results.

II. RELATED WORK

A. Human Motion Data Generation Model

Motion generation models have employed various methods, including generative adversarial networks (GANs) [15], VAEs [16], normalizing flows (NFs), and generative pretrained transformers (GPTs) [17]. These methods provide diverse frameworks for generating motion data, each showcasing individual characteristics.

The GAN-based models consist of a generator and a discriminator [15]. The generator creates human-like motion data using noise data, while the discriminator assesses whether the data are real or fake. Ahn et al. [18] utilized GANs based on recurrent neural networks (RNNs) for generating human motion. ActFormer [11] includes transformers [19] and can generate multiperson motion data. Yan et al. [20] introduced the convolutional sequence generation network that utilizes a Gaussian process latent prior as input data. For GANs, training can be challenging, and the risk of model collapse is significant [4]. Some studies have shown a preference for VAEs over GANs in human motion generation tasks [4, 6].

The NF generates accurate motion data as it does not involve approximation in predicting probability distributions [21]. In stylized motion research [22], the NF method has been utilized to extract and apply a motion style to a new motion. The MoGlow [21] method employs the generative flow (Glow) [23] architecture. The quality of motion produced by MoGlow [21] was observed to be high. However, the NF method faces limitations due to strict design requirements for the transition function, which must possess an easily computable inverse and Jacobian determinant to enable NF application. Consequently, the NF is less frequently adopted compared to other methods such as GANs and VAEs.

Like the NF, VAEs predict probability density functions but differ in that they utilize approximations in some aspects. VAEs feature a structure akin to autoencoders, where the encoder processes the input data to extract the μ and σ values, and the decoder uses these μ and σ variables to generate new motion data. Habibie et al. [24] employed RNN-based VAE architectures. The CVAE [25] was introduced to include control factors for generating desired motion data. Various studies have imposed constraints to produce specific motion outcomes. The action-conditioned transformer (ACTOR) [3] produces a variety of motion data using action-class labels. Similarly, Action2Motion [5] utilizes action-class labels to generate human motion data. Li et al. [26] proposed Audio2Gestures, which creates gestures from speech audio. Due to the ease of applying constraints and training VAEs, these have found widespread use across numerous studies. Limited attempts have been made to employ GPT [17] in motion generation. In PoseGPT [4], although the overall structure is based on the autoencoder design, GPT is integrated internally to predict new sequences. The input motion data are quantized by an encoder and codebook, while GPT sequentially predicts the latent vector for the subsequent sequence.

In summary, VAE-based approaches are extensively researched within motion generation models, with recent explorations extending to NF and GPT-based methods. We have also employed the CVAE structure to generate motion data and have introduced skip connections to address issues related to posterior collapse and motion convergence.

B. Human Motion Data Generation Constraints

In motion generation research, various inputs serve as constraints. Researchers have employed conditions such as text [6, 7, 27], action-class labels [3, 4, 11], music [28-31], and styles to produce motion [22]. Predominantly, studies involving action-class labels are utilized. The earlier mentioned PoseGPT [4] and ACTOR [3] models also create new motion types using action-class labels. Action2Motion [5] inputs a sequence of vectors composed of action classes into a VAE to generate continuous motion.

Next, several studies have created motion using text input. The motion contrastive language-image pre-training (MotionCLIP) [8] generates motion where a CLIP [32] encoder processes the input text, and a pretrained motion CLIP decoder reconstructs the motion. In research by Ghosh et al. [7], input text data is vectorized using the bidirectional encoder representations from transformers (BERT) [33] algorithm and encoded by the sentence encoder. The system alternately inputs the encoded vectors from the hierarchical pose encoder and the sentence encoder into the decoder. In the testing environment, the hierarchical pose encoder is omitted, and a sentence encoder based on BERT is used alongside a hierarchical pose decoder for motion generation. Analogous to this, TEMOS [6] derives features from the input text using the pretrained natural language processing algorithm distilled BERT (DistilBERT) [34]. The encoder processes these features into a latent vector and conveys them to the decoder. Furthermore, TEMOS [6] incorporates text and motion encoders. The suggested system references the aforementioned architectures and utilizes keyframe and full-frame encoders, which are trained alternately, with only the keyframe encoder operational during the testing phase.

C. Motion Interpolation

Research in motion interpolation has predominantly focused on keyframe-based systems. The primary aim of these studies is to generate a smooth transitional motion between keyframes. Typically, this transitional motion is created using a linear interpolation system. Commercial software such as Blender and Unity employ spline interpolation to transition between two keyframe actions. Historically, researchers have used statistical models to forecast natural transition behavior between two actions [35-37]. Motion interpolation has seen considerable developments with the advent of deep learning [38-40]. Recently, a number of researchers have been investigating the use of transformers for interpolating between keyframes [41, 42]. In [13], they formulated three stages of transformer-based models to create continuous motion. Similar to our work, Qin et al. [14] considered context information to create transitional motions, dividing the motion interpolation model into two stages: the context transformer and the detail transformer. The context transformer generates coarse transitional motions based on context, while the detail transformer refines them. Their approach is similar to our proposed system in terms of using context information. However, it requires continuous context frames to extract this information. Moreover, their definition of ‘context’ pertains to motion style, as opposed to our definition involving motion class. Consequently, our proposed system and motion interpolation research pursue different objectives in motion generation. Our system aims to generate varied intermediate motions from input data, whereas interpolation systems primarily strive to produce smoothly connected motions. We confirm these distinctions through experimental results.

III. PROPOSED SYSTEM

This section outlines the proposed motion generation model structure and its training methods. Subsequently, we discuss the application of the pretrained model in developing systems for motion recognition and controllable motion generation.

A. Key Pose-based Motion Generation System

The proposed system generates all motion data using only a limited set of pose data. It is designed around a CVAE [25] architecture that includes two encoders and a decoder. Fig. 1 illustrates the overall architecture of the proposed system. Additionally, TEMOS [6] and the research by Ghosh et al. [7] influenced this design. The two encoders consist of a full-motion encoder (E_F) that processes the entire frame and a key-motion encoder (E_K) that focuses on a small number of frames. Each encoder features a transformer encoder and calculates both the mean (μ) and covariance (σ). We employed the reparameterization technique for generation.

Latent vectors (Z) are derived from the μ and σ values. During the training phase, both encoders were employed to train the key-motion encoder, while only the key-motion encoder was used to generate motion during the inference stage. Ultimately, utilizing a transformer decoder, the decoder can produce the complete motion data through the latent vector (Z) and the positional encoding vector.

1) Motion Encoder

Data preprocessing converted the skinned multi-person linear model (SMPL) format data into feature vectors. We excluded rotational data for both hands and transformed the SMPL's axis-angle format into a six-dimensional representation [43]. We concatenated the root joint translation $\mathbf{r}_{trans} \in \mathbb{R}^3$ with the body joint rotation $\mathbf{p} \in \mathbb{R}^{6\times 22}$ to form a feature vector $\mathbf{f} \in \mathbb{R}^{135}$. Consequently, we input the feature vectors $\mathbf{F} = \{\mathbf{f}_1, \mathbf{f}_2, \cdots \mathbf{f}_T\} \in \mathbb{R}^{135\times T}$ into the generation system, where T represents the frame length of the motion.

To train the key-motion encoder, the extraction of key poses ($\mathbf{F}_{key}$) from the complete motion sequence $\mathbf{F}_{full}$ is necessary. Key pose frames were selected by dividing the full frames evenly according to the number of key poses (K). During the extraction of key pose data, the unselected frame data were minimized to preserve the format of the input data. Consequently, the key-motion encoder was trained to generate the latent vector $\mathbf{Z}$ using sparse motion data ($\mathbf{F}_{key}$). Although the encoder consistently receives a fixed number (K) of key pose data during training, K can be adjusted flexibly during testing. For example, the system trained with ten key poses can still generate accurate motion with only six key poses. The completeness of the motion improves proportionally to how accurately the six key poses represent the intended motion characteristics.

Although the data from the feature vectors differed, the internal operation process remained identical for both encoders. The feature vectors ($\mathbf{F}$) were transformed into embedding vectors ($\mathbf{H}$) by an embedding network consisting of a straightforward multilayer perceptron:

$$H = MLP(\mathbf{F}) \quad \mu, \sigma = E(\mathbf{H}), \tag{1}$$

where $E(\,)$ represents the motion encoder, and $MLP(\,)$ denotes the embedding network.

The transformed embedding vector was sequentially input into the transformer encoder, where iterative self-attention operations were performed. The spatiotemporal relationships in the input data were captured in the output through these self-attention processes in the encoder. Subsequently, only the μ and σ values were extracted from the output data, which were then used to construct a latent vector using ($\mathbf{Z} \in \mathbb{R}^{256\times T}$) the reparameterization technique. Finally, the latent vector was obtained using Eq. (2) from the normal distribution:

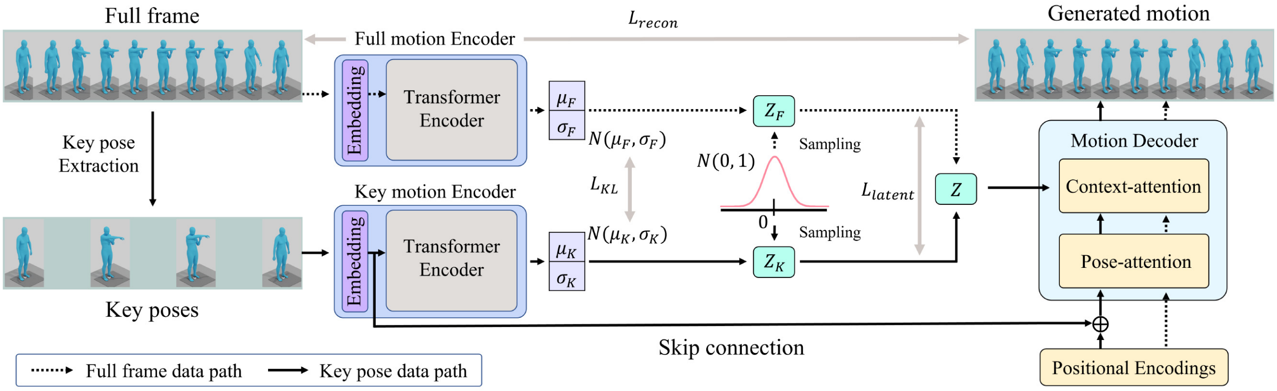


Fig. 1. Overall architecture of the proposed motion generation system.

$$\mathbf{Z} = Sample(dist(0,1)) * \sigma + \mu, \tag{2}$$

where $dist(0, 1)$ represents the normal distribution, and $Sample(\,)$ refers to a sampling function.

2) Motion Decoder and Skip Connection

We utilized a transformer decoder that includes two distinct attention layers as a motion decoder. These layers, pose attention (Att_{pose}) and context attention ($Att_{context}$), perform separate functions. The initial Att_{pose} layer distributes pose information across the entire action dataset from a selected subset of action data when activated by the key-motion encoder. The vectors inputted into the pose attention layer, denoted as **S**, are the sum of the initial embedding vectors **H** and positional vectors **P**. When the full-motion encoder is activated, only the positional vectors **P** are inputted to the Att_{pose} layer. The embedding vectors **H** are transmitted via a skip connection, while the positional values **P** are computed using a periodic wave function. The variable **S** encapsulates both positional information for each frame and pose-specific information from targeted keyframes. Within the pose attention layer, the input **S** undergoes repeated self-attention operations, thereby allowing pose information from keyframes to permeate across the entire sequence. As a result, the output from the pose attention layer embodies the prior pose information based on **H**.

Next, the second context attention ($Att_{context}$) layer was applied to the latent vector (**Z**) from the motion encoder and the output of the previous self-attention operation. At this point, **Z** encapsulates the context information of the action to be generated, while the output from the first Att_{pose} contains the input pose information. The cross-attention operation facilitates the effective integration of these two information streams, resulting in the final output of the decoder accurately captures the context and pose information, resulting in highly expressive motion. The decoder output matches **F**. Thus, we compared the decoder output with the input **F** to calculate the reconstruction loss.

3) Training Method

In the training phase, we trained two motion encoders and one motion decoder. The proposed training method is inspired by the TEMOS [6] and work by Ghosh et al. [7]. The loss function used during training consists of three distinct components: reconstruction loss (L_{recon}), Kullback-Leibler (KL) loss (L_{KL}), and latent vector loss (L_{latent}). The reconstruction loss L_{recon} measures the difference between the decoder-output feature vector and the ground-truth feature vector, we calculated two reconstruction losses from two encoders: the full-motion and key-motion encoders. Additionally, L_{recon} is the sum of the reconstruction losses from both encoders. The generative model can simulate human-like motion through L_{recon}. Next, L_{KL} represents the loss predicting the probability distribution derived from the μ and σ of each encoder, aiming to align the probability distributions extracted from both encoders. We applied the KL divergence loss terms (L_{KL}) to encourage similarity between the predicted probability distributions from each encoder. Further, L_{latent} calculates the L1 distance between the latent vectors of each encoder. Owing to these two loss terms, the key-motion encoder, which uses a limited dataset, can derive a latent vector similar to that of the full-motion encoder:

$$L = L_{recon} + \lambda_{KL} L_{KL} + \lambda_{latent} L_{latent}, \tag{3}$$

where λ_{KL} and λ_{latent} represent hyperparameters determined through iterative experimentation.

B. Application Systems using the Pretrained Generative Model

This section addresses the construction of application systems (i.e., the action recognition and controllable motion

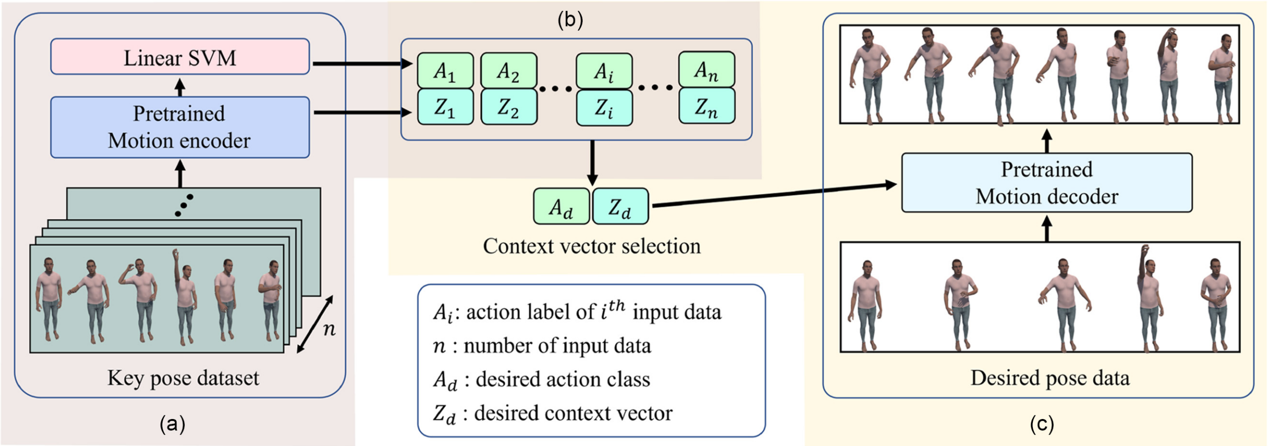


Fig. 2. Overall architecture of the application systems, comprising the action recognition (a) and controllable motion generation systems (c). (b) describes the recognized latent vectors.

generation systems) using the pretrained model. The overall structure of each application system is illustrated in Fig. 2. An advantage of this approach is that it requires no additional training procedures to implement each system.

1) Action Recognition System

We developed an action recognition system using the pretrained motion encoder. We hypothesized that if the latent vector **Z** encapsulates sufficient contextual information, it can uniquely map to the latent space for each action. Consequently, **Z** becomes a valuable asset for an action recognition system. To validate our hypothesis, we classified the latent vectors from the motion encoder using the simplest recognition algorithm, the linear support vector machine (SVM). We present the recognition system in Fig. 2(a). The pretrained motion encoder converts the key pose dataset into latent vectors. The linear SVM then predicts action class labels for all input latent vectors, resulting in recognized latent vectors as depicted in Fig. 2(b). We employed a linear classifier to avoid re-extracting features and focused on verifying if the latent vectors contained contextual information. The detailed setup of the environment is discussed in the experimental section for the recognition system.

2) Controllable Motion Generation System

We developed a controllable motion generation system by utilizing a pretrained motion decoder. Fig. 2(c) depicts this system. The decoder uses pose data and a context vector to generate and control actions based on these inputs. Pose data specify the poses to be included in the output action. We selected the latent vector of the desired action by leveraging the recognized latent vector. This selected latent vector serves as the context vector and inputs to the pretrained motion decoder. Despite the sparsity of desired pose data, the generated motions are continuous and exhibit an upward boxing motion. In the experimental section, we maintained consistent pose inputs while varying the context vectors to compare the generated motions with the interpolated motion.

IV. EXPERIMENTS

We conducted experiments to evaluate and verify the proposed generation model. We used different datasets for each experiment, tailored to specific purposes. Initially, we compared the proposed model with existing research concerning the accuracy of the generated motion. To facilitate this comparison, we utilized the Karlsruhe Institute of Technology (KIT) Whole-body Human Motion Database and the Anidance benchmark. Subsequently, we trained the generation model using a subset of the Archive of Motion Capture as Surface Shapes (AMASS) dataset [44] to assess the effectiveness of the skip connection. We evaluated the motion generation system both with and without the skip connection. Finally, we explored the applications of the pretrained generative model, including action recognition and controllable motion generation systems. For building the motion recognition system, we employed the HumanAct12 dataset [5], which consists of motion data with class labels. We compared the pretrained generative model with the standard interpolation system in the final experiment.

A. Quantitative Experimental Results

In this section, we quantitatively evaluated our models. We hypothesized that directly inputting poses would yield more accurate motion than text input. The initial experiment aimed to test this hypothesis. Subsequently, we compared our model with existing motion interpolation models.

1) Training Dataset

As previously mentioned, we evaluated our method on the KIT data and Anidance. The KIT dataset includes the KIT Whole-body Human Motion Database [45] and the Carnegie Mellon University (CMU) Graphics Lab Motion Capture Database [46] as a paired dataset of sentences and actions. The KIT data comprises raw motion capture data, which are processed using the master motor map framework [47]. It contains 3,911 motion sequences and 6,353 sequence-level annotations. The Language2Pose [27] method parsed the KIT dataset into 1,784 sets for training, 566 validation, and 587 testing motion instances were recorded. Additionally, TEMOS [6] and the study by Ghosh et al. [7] utilized 520 sequences for testing. Previous studies downsampled the 100 Hz KIT motion data to 12.5 Hz. We also conducted tests under the same conditions using the 520-sequence dataset.

Next, Tang et al. [39] introduced the Anidance dataset for generating dance sequences from music. Duan et al. [41] excluded audio features and employed the dataset for motion completion tasks. The Anidance dataset was then used for infilling tasks, involving the generation of motion frames between sparse keyframes. It includes 61 dance sequences across four genres, recorded at 25 frames per second and comprises 101,390 frames capturing the global positional coordinates of skeletal joints. Following previous evaluation settings, we used a window size of 128 and an offset size of 64, resulting in 1,117 training sequences and 323 evaluation sequences.

2) Comparison Results with Existing Models

We compared our models with existing text-based motion generation models using the average positional error and average variance error as quantitative evaluation metrics. These metrics calculate errors based on each joint's position and covariance using the L2-norm. This evaluation method,

Table 1. Quantitative generated-motion accuracy comparison of the proposed model with text-based generation models

	Average position error				Average variance error			
	Root joint	**Trajectory**	**Meal (L)**	**Mean (G)**	**Root joint**	**Trajectory**	**Meal (L)**	**Mean (G)**
JL2P [27]	1.622	1.616	0.097	1.630	0.669	0.669	0.006	0.672
Ghosh et al. [7]	1.291	1.242	0.206	1.294	0.564	0.548	0.024	0.563
TEMOS [6]	0.963	0.955	0.104	0.976	0.445	0.445	0.005	0.448
Without Skip	0.141	0.139	0.035	0.147	0.047	0.047	0.001	0.049
With Skip	0.137	0.135	0.034	0.143	0.048	0.048	0.001	0.049

L and G represent local and global, respectively.

widely used in previous studies [6, 7, 27], indicates that a smaller error value signifies better performance. Furthermore, TEMOS [6] modified the previously applied evaluation method. We assessed our system's performance according to the TEMOS [6] evaluation protocol.

Table 1 lists the comparative results of quantitative performance. In our performance evaluation, we distinguished between configurations of the proposed system with and without the skip connection. The results in Table 1 show that our method performs exceptionally well compared to other systems, particularly the configuration using the skip connection. Using only the KIT dataset [45], the method without the skip connection also exhibited excellent results, confirming that direct use of pose data can significantly enhance motion generation and more accurate motion data than indirectly transmitting text input data.

Next, we compared our models with motion interpolation models. In this evaluation, we adopted L2P metrics that calculate the L2-norm on global position data. We set transition motion lengths to 5, 15, and 30 frames. Table 2 represents the comparison results [40, 48, 49]. A model that extracts lower L2P values can generate more accurate transition motion. Thus, lower L2P values indicate better model performance. Our model achieved the best performance at 30 frames and yielded comparable results at 5 and 15 frames. Because our model primarily focuses on generating motion, it performs slightly worse in interpolation. However, our model holds an advantage in generating motions with long-time dependencies due to its CVAE structure. The latent vector preserves the motion context and provides evidence for the generated motion. Therefore, we achieved the best performance in the 30-frame evaluation.

Table 2. Quantitative comparison results of the proposed model with interpolation models

Method	L2P		
	5 frames	**15 frames**	**30 frames**
Autoencoder [40]	3.57	3.69	3.93
SSMCT [48]	0.84	1.46	1.64
Δ-Interpolator [49]	0.60	0.74	1.01
Proposed	0.67	0.84	0.99

This experiment quantitatively evaluated the generated motion using the proposed model. We observed that the proposed model can create accurate motion data based on a small number of pose data. The pose-based system exhibited superior accuracy in generating motion compared to the text input method. Using pose data as input should inherently yield better performance than text input; however, we confirmed the possibility and advantages of the motion generation system based on a small number of pose data. In addition, we compared our model with interpolation models. The comparison results demonstrate that our model can achieve comparable performance to interpolation models. In particular, our model achieves the best performance in long-range motion. These results originated from our model's purpose and architecture. Our model can detect the user's intent from key poses and generate desired motions using latent vectors. Consequently, our model exhibits robustness in generating long-range motion by utilizing the detected latent vector.

3) Ablation Studies

In ablation studies, we confirmed the effectiveness of model structures such as skip connections (Skip), a full-motion encoder (Encoder), and linear interpolation (LERP). We used the Anidance data and compared L2P for each structure setup. Table 3 shows the results of ablation studies. The first row represents the result of the original CVAE model, which is the worst performance. When we applied the skip connection to the original model, the performance significantly increased. Thus, skip connections play an important role in enhancing the accuracy of the generated motion. Next, we used both the key- and full-motion encoders in the training process and applied full-motion encoder-related loss terms. The model performance slightly increased. Lastly, we interpolated sparse keyframe data and input it to our model. This case also did not increase the performance dramatically. In short, the skip connection is more powerful than other structures in

Table 3. Ablation of the proposed model structure

Skip	Encoder	LERP	L2P		
			5 frames	15 frames	30 frames
x	x	x	0.84	1.34	1.96
√	x	x	0.70	0.87	1.06
x	√	x	0.82	1.34	1.93
x	x	√	0.82	1.30	1.81
√	√	√	0.67	0.84	0.99

terms of the accuracy of output motion.

B. Qualitative Experimental Results

To assess the influence of the skip connection, we conducted a qualitative evaluation of the generated motion and compared the results with and without the skip connection, utilizing multiple training datasets.

1) Training Dataset

In this experiment, we utilized the AMASS dataset [44] in SMPL-X format [50]. The datasets were categorized into training, validation, and testing data according to BABEL protocol. For the multiple training datasets, we used the Bio Motion Lab (BML) MoVi [51], BMLrub [52], and CMU datasets [46]. The BML MoVi dataset [51] encompasses daily human and sports motions captured through various sensing systems, including cameras, inertial measurement units, and optical motion capture systems. We allocated 845 for training, 267 for validation, and 266 for testing from the BML MoVi [51] dataset in the experiment. Additionally, BMLrub [52] includes 1,678 training, 526 validation, and 532 testing data. Finally, the CMU dataset [46], previously included in the KIT, features motion data collected in diverse environments such as two-person interactions, sports, and scenario motions. We allocated 996 for training, 386 for validation, and 320 for testing in the CMU dataset [46]. With the multiple training datasets, the experiment utilized 5,816 motion data, classified into 3,519 training, 1,179 validation, and 1,118 testing data.

2) Experimental Results and Discussion

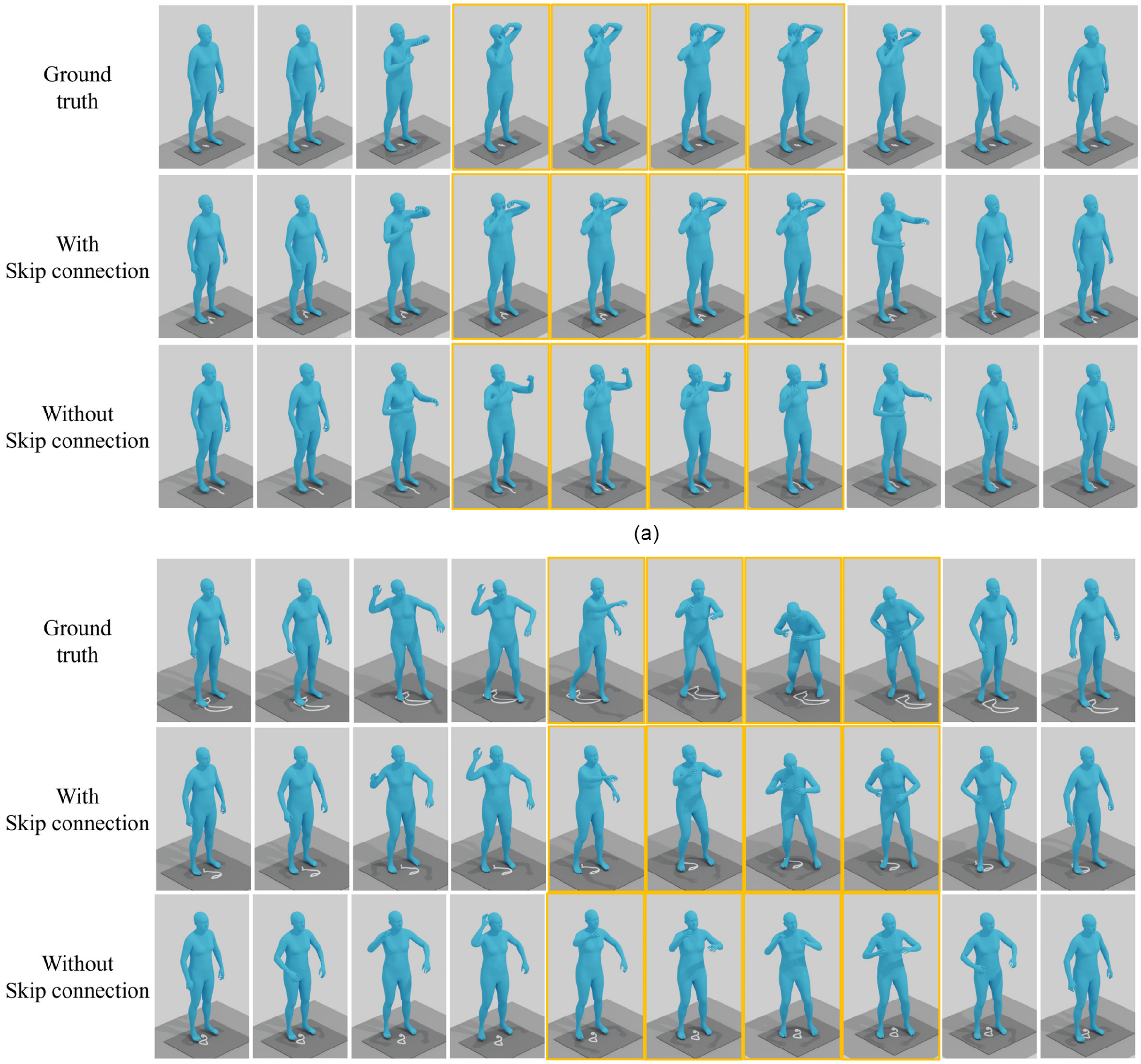
We explored the effects of the skip connection on identical inputs. We assessed the results for two actions using visualization tools from TEMOS [6]. The first row in Fig. 3 displays the ground-truth data for each action. We input ten pose data from the ground-truth into the proposed generation model and analyzed the impact of the skip connection by comparing the middle and last rows in Fig. 3.

Fig. 3 illustrates the significant differences in generated motion depending on the application of the skip connection. Without the skip connection, the input pose data characteristics vanish, and the motion range decreases. Specifically, in the motion for putting in contact lenses (Fig. 3(a)), using the skip connection yields motion similar to the ground-truth, while the third row of motion differs significantly from the ground-truth. Both hands fail to reach the avatar’s face in the motion sequence without the skip connection. Moreover, in the throwing motion (Fig. 3(b)), the arm movement range is narrow, and there is almost no waist movement. Similarly, the range of both arms' movement in the receiving motion is narrow. This observation aligns with the average motion convergence noted in previous studies [11]. We addressed these issues using the skip connection. The fifth row in Fig. 3 depicts the outcomes using the skip connection for both the throwing and receiving motions. Here, the arm motion range is noticeably broader. Subsequently, the waist dynamically bends in the receiving motion, enabling both hands to catch the item effectively. The results qualitatively confirm that the skip connection enhances the joint motion range and preserves the input pose data’s expressiveness.

This experiment confirmed the effectiveness of the skip connection and demonstrated the need for structural improvements in the basic VAE to handle substantial training data. We explored a generative model capable of learning extensive human motion data to construct a substantial latent space. Thus, our objective was to train the model on a large volume of motion data, but this reduced the motion expressiveness. We resolved this issue by implementing the skip connection in the VAE, which enhanced motion expressiveness. In summary, training VAE-based generative models with copious amounts of data revealed the difficulty of conveying all motion information solely through latent vectors. Consequently, we used the skip connection to transfer the input motion data directly to the decoder.

C. Pretrained Generative Model Applications

This section establishes two application systems and verifies the differences between the proposed generation system and existing motion interpolation models. Like motion interpolation systems, the proposed system uses pose data as input. However, the generation system can produce distinct actions based on contextual information, even with identical pose input. To verify this capability, we confirmed whether a motion recognition system can classify latent vectors according to action-class labels in experiments. Subsequently, we utilized the classified latent vectors and pose data to generate motions and compared these with interpolated motions. Additionally, we experimentally validated the roles of attention modules in the motion decoder.



(a)

(b)

Fig. 3. Qualitative comparison results: generative model trained with multiple datasets. We tested two actions: (a) putting in contact lenses and (b) the throwing and receiving motions. The top three rows show the putting in contact lenses motion, and the bottom three rows depict the throwing and receiving motion.

1) Action Recognition System

This experiment utilizes the HumanAct12 dataset [5], which contains labeled human motion data divided into 12 action classes and further subdivided into 34 classes. We extracted SMPL data formats from the HumanAct12 dataset [5], and the preprocessing steps followed those used in previous datasets.

The data was divided into training, validation, and testing sets according to previous studies' methodologies. For the HumanAct12 dataset [5], we used 1,191 training, 61 validation, and 171 testing datasets.

As previously mentioned, we employed the pretrained motion encoder and linear SVM to develop an action recognition system. The encoder processes input pose data into latent vectors, which the SVM classifies. The results are summarized in Table 4, with each row indicating a different training setup of the generative model. For example, the system in the first row was trained with 5 key pose data and without a skip connection. Initially, we compared recognition accuracy between generative models with and without skip connection. Recognition rates decreased when skip connections were used because the decoder processes both latent vector and pose data. We then examined the differences in recognition rates between the key-motion encoder ($K = 5$) and the full-motion

Table 4. Recognition comparison results according to the motion encoder and the number of key poses

Skip connection	Key pose number (K)	Accuracy (%)	
		Key encoder	Complete encoder
x	5	83.62	84.79
√	5	81.87	84.21
√	10	87.71	88.88
√	15	88.88	90.05

encoder, the latter displaying a higher recognition rate. Additionally, increasing the number of key pose data enhanced the recognition rate. This indicates potential data loss in the key pose-based latent vectors. Nonetheless, even with a limited number of frames, the system achieved a reasonable action recognition performance around 80%. These results demonstrate the efficacy of latent vectors in classifying motion.

From these recognition rates, we concluded that the latent vectors capture contextual information capable of distinguishing different motion. This confirms that the latent vectors provide essential contextual information in the system's decoder, and the context attention module generates motion based on this embedded information.

2) Controllable Motion Generation System

This experiment utilizes the pretrained decoder of the generative model to generate desired motion using the input: pose data and latent vectors. We determined that the innovation of pose-based generation systems distinctly differs from motion interpolation systems. The motion decoder receives latent vectors and pose data derived from the HumanAct12 dataset. In the experiment, we selected motion classes with similar poses and extracted commonly shared pose data.

To clearly identify the differences, we compared the generated motions and interpolated motions using their trajectories in Figs. 4 and 5, which illustrate motion data projected onto an SMPL unity avatar, where the trajectories visualize the avatar's right hand positions. The first rows in Figs. 4 and 5 illustrate the pose data input into the decoder (yellow rectangles). The blue points indicate the avatar's right hand positions relative to key pose data. Consequently, we used 5 key pose data points to generate motion data. The yellow trajectory in the first rows depicts

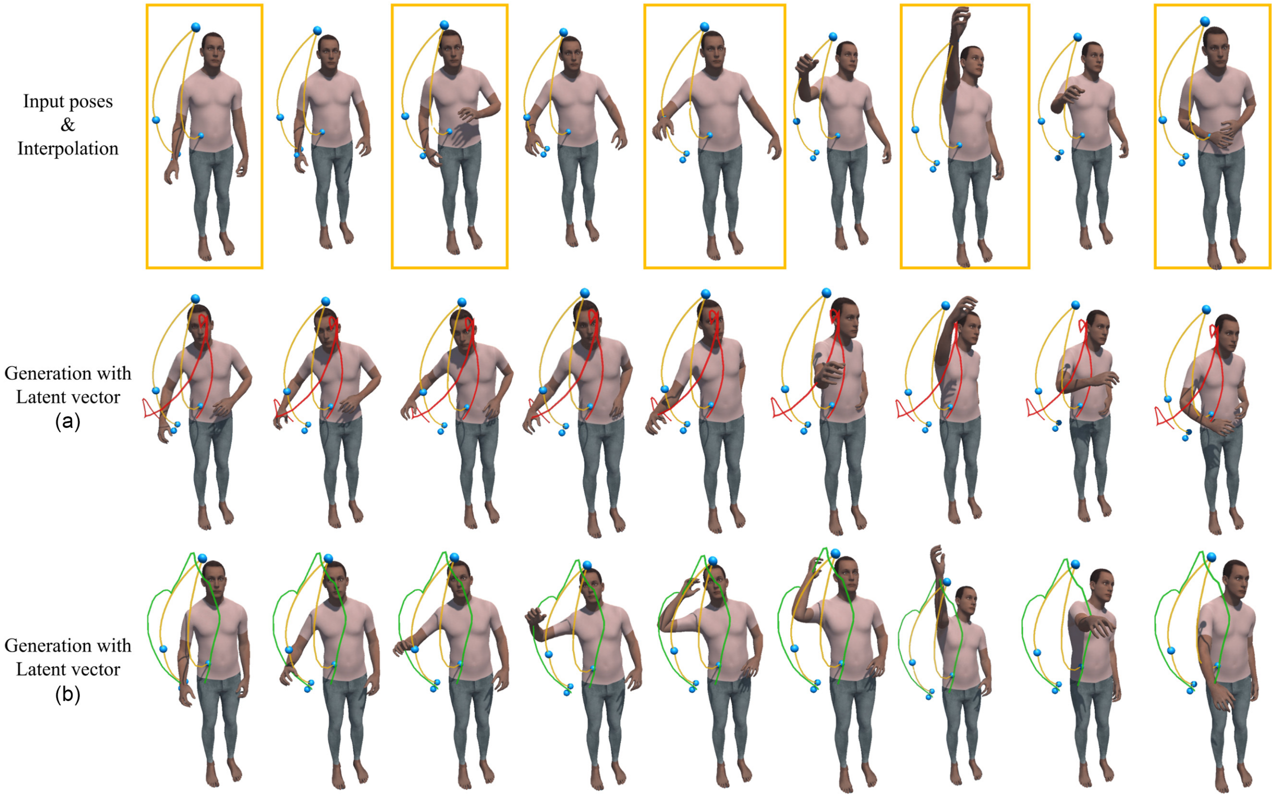

Input poses & Interpolation
Generation with Latent vector (a)
Generation with Latent vector (b)

Fig. 4. Qualitative comparison results: Generated motions using the controllable motion generation system, testing two actions comprising (a) boxing right upward and (b) throwing. The first row illustrates the input posed data (yellow rectangles) and interpolated motion. Subsequent rows display the results generated using the latent vectors for boxing right upward and throwing.

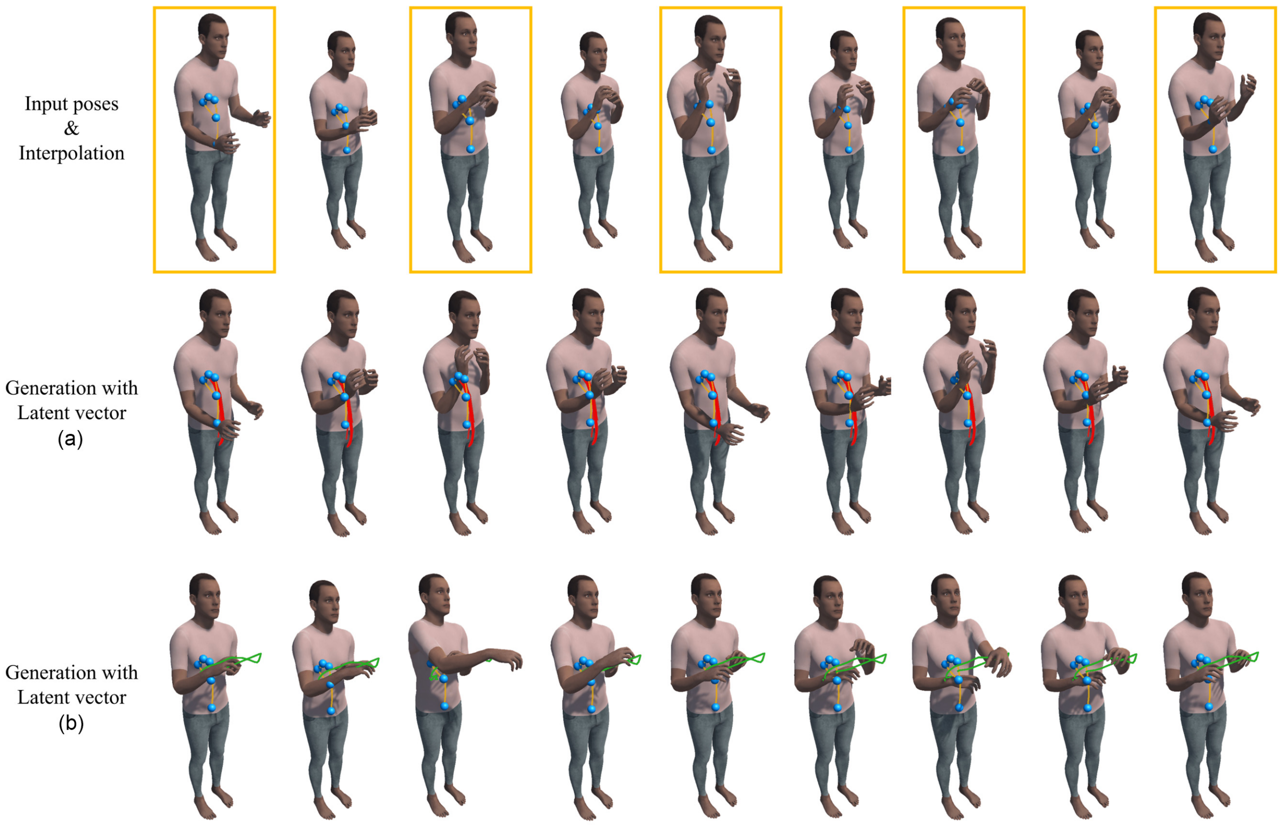


Fig. 5. Qualitative comparison results: Generated motions using the controllable motion generation system, testing two actions comprising (a) both hands lifting and (b) right and left punching. The first row shows input posed data (yellow rectangles) and interpolated motion. The second row displays the generated motion using the latent vector for both hands lifting. The last row illustrates the generation results for the right and left punching using the latent vector.

the interpolated motion data, generated by linearly connecting the input poses using spherical linear interpolation. The latent vectors were identified using the previous motion recognition system. Subsequently, we selectively chose recognized latent vectors corresponding to the action class and fed them into the decoder. The second and third rows display the results after inputting the latent vectors for selected motion classes. Despite the consistent use of the same pose data, the generated motions vary based on the latent vectors.

Initially, we compared the generated motions and interpolated motion in Fig. 4. The yellow trajectory and avatar illustrate a straightforward movement where the avatar's right hand linearly follows the trajectory between blue points. Subsequent to this, the red trajectory signifies the results when latent vectors for the boxing right upward motion are input. In Contrast to the yellow trajectory, this trajectory exhibits a downward curve depicting an upward punching motion. Even with slight modifications to the input poses, the red trajectory adheres closely to positions akin to those input (blue points). Finally, the green trajectory signifies the generated motion using identical pose data and latent vectors for the throwing motion. Unlike prior cases, the green trajectory forms a distinctive upper semicircular curve as it transitions from the back to the front, simulating a throwing motion. The comprehensive avatar motion in the third row mimics throwing an object toward the front, with positions of the input poses marginally adjusted to match the throwing motion's characteristics.

The comparison results are shown in Fig. 5, where motions involving both hands and selected shared pose data with minimized differences between each pose were utilized. Except for the first key pose, blue points cluster at similar positions. Due to this setup, the range of interpolated motion is restricted in the first row of Fig. 5, causing the avatar's hands to consistently maintain positions in front of the chest. The second row depicts generated motion employing latent vectors for the motion of lifting both hands. The avatar's hands move up and down rhythmically. Furthermore, the red trajectory is significantly longer than the yellow trajectory. The last row shows the motion generated using identical pose data and latent vectors for both right and left punching motions. Here, the avatar's hands sequentially advance and retract. Unlike the red trajectory, the green trajectory extends directly forward. This experiment allowed us to clearly discern differences between the generated motion and the interpolated motion derived from pose data with minimal movement alteration. Unlike simple interpolation methods, the generative model produces varied trajectories based on contextual vectors, enabling the generation of diverse motion types using the same pose data with latent vectors.

The experimental results verify that the controllable motion generation model effectively manipulates the resultant motion data through both latent vectors and input pose data. Users can define the motion's context using latent vectors and manipulate its form by altering the input poses. For example, in the process of throwing, the alignment of arm height and trajectories with the input poses is precise.

V. CONCLUSION

We proposed a motion generation model using a small number of pose data as input. The experimental results confirm that the proposed model can accurately generate human motion data. Furthermore, we incorporated a skip connection into the CVAE model, enabling extensive learning from substantial training data. The qualitative experiments suggest a marked enhancement in the expressiveness of the generated motion following the implementation of the skip connection.

The pretrained generative model supports the development of action recognition and controllable motion generation systems. The encoded latent vector captures essential context information applicable to the recognition system. The controllable motion generation systems can produce specific motions by modifying input poses and latent vectors. Thus, virtual environment content creators can employ the proposed system to generate desired motions. Finally, we compared the proposed generation system with the interpolation system using a controllable motion generation system. The generation system can produce a variety of motions using identical input poses, unlike the interpolation system.

The controllable motion generation system relies more on the latent vector compared to key pose data for motion generation. This characteristic stems from the design of the generation model that seeks to generate desired motions by detecting the user's intent. In this system, the user's intent is directly input as a latent vector. Therefore, when distinct key poses and latent vectors are intentionally input into the system, it generates new motions based on the latent vector. We can mitigate this tendency by expanding the scale of the training data. By training on a wide array of motion data, the model increasingly reflects key poses in the motions it generates. Consequently, we plan to progressively increase the number of training datasets. We currently work with a maximum of three datasets; however, future work could utilize additional motion data from AMASS [44]. We anticipate that training on extensive motion data could significantly enhance both the encoding capabilities of the encoder and the performance of the decoder in generating motion data.

CONFLICT OF INTEREST

The authors have declared that no competing interests exist.

ACKNOWLEDGEMENTS

This work was supported by the Ministry of Science & ICT of Korea, under the Software Star Lab. program (No. IITP-2018-0-00599) supervised by the Institute for Information & Communications Technology Planning & Evaluation and the Chung-Ang University research grant for the year 2024.

REFERENCES

1. F. Hong, M. Zhang, L. Pan, Z. Cai, L. Yang, and Z. Liu, "AvatarCLIP: zero-shot text-driven generation and animation of 3D avatars," 2022 [Online]. Available: https://arxiv.org/abs/2205.08535.
2. G. Xia, P. Xue, D. Zhang, and Q. Liu, "Keyframe-editable real-time motion synthesis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 7, pp. 4538-4551, 2022. https://doi.org/10.1109/TCSVT.2021.3129478
3. M. Petrovich, M. J. Black, and G. Varol, "Action-conditioned 3D human motion synthesis with transformer VAE," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Montreal, Canada, 2021, pp. 10965-10975. https://doi.org/10.1109/ICCV48922.2021.01080
4. T. Lucas, F. Baradel, P. Weinzaepfel, and G. Rogez, "PoseGPT: Quantization-based 3d human motion generation and forecasting," in *Computer Vision – ECCV 2022*. Cham, Switzerland: Springer, 2022, pp. 417-435. https://doi.org/10.1007/978-3-031-20068-7_24
5. C. Guo, X. Zuo, S. Wang, S. Zou, Q. Sun, A. Deng, M. Gong, and L. Cheng, "Action2motion: conditioned generation of 3D human motions," in *Proceedings of the 28th ACM International Conference on Multimedia*, Seattle, WA, USA, 2020, pp. 2021-2029. https://doi.org/10.1145/3394171.3413635
6. M. Petrovich, M. J. Black, and G. Varol, "TEMOS: generating diverse human motions from textual descriptions," in *Computer Vision – ECCV 2022*. Cham, Switzerland: Springer, 2022, pp. 480-497. https://doi.org/10.1007/978-3-031-20047-2_28
7. A. Ghosh, N. Cheema, C. Oguz, C. Theobalt, and P. Slusallek, "Synthesis of compositional animations from textual descriptions," in *Proceedings of the IEEE/CVF International*

Conference on Computer Vision, Montreal, Canada, 2021, pp. 1376-1386. https://doi.org/10.1109/ICCV48922.2021.00143

8. G. Tevet, B. Gordon, A. Hertz, A. H. Bermano, and D. Cohen-Or, "MotionCLIP: exposing human motion generation to clip space," in *Computer Vision – ECCV 2022*. Cham, Switzerland: Springer, 2022, pp. 358-374. https://doi.org/10.1007/978-3-031-20047-2_21
9. S. Raab, I. Leibovitch, P. Li, K. Aberman, O. Sorkine-Hornung, and D. Cohen-Or, "MoDi: unconditional motion synthesis from diverse data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Vancouver, Canada, 2023, pp. 13873-13883. https://doi.org/10.1109/CVPR52729.2023.01333
10. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770-778. https://doi.org/10.1109/CVPR.2016.90
11. L. Xu, Z. Song, D. Wang, J. Su, Z. Fang, C. Ding, et al., "ActFormer: a GAN-based transformer towards general action-conditioned 3D human motion generation," 2022 [Online]. Available: https://arxiv.org/abs/2203.07706.
12. D. Wei, H. Sun, B. Li, J. Lu, W. Li, X. Sun, and S. Hu, "Human joint kinematics diffusion-refinement for stochastic motion prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 5, pp. 6110-6118, 2023. https://doi.org/10.1609/aaai.v37i5.25754
13. C. A. Mo, K. Hu, C. Long, and Z. Wang, "Continuous intermediate token learning with implicit motion manifold for keyframe based motion interpolation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Vancouver, Canada, 2023, pp. 13894-13903. https://doi.org/10.1109/CVPR52729.2023.01335
14. J. Qin, Y. Zheng, and K. Zhou, "Motion in-betweening via two-stage transformers," *ACM Transactions on Graphics*, vol. 41, no. 6, article no. 184, 2022. https://doi.org/10.1145/3550454.3555454
15. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Couville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139-144, 2020. https://doi.org/10.1145/3422622
16. D. P. Kingma, "Auto-encoding variational Bayes," 2013 [Online]. Available: https://arxiv.org/abs/1312.6114v1.
17. A. Radford, "Improving language understanding by generative pre-training," 2018 [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.
18. H. Ahn, T. Ha, Y. Choi, H. Yoo, and S. Oh, "Text2Action: generative adversarial synthesis from language to action," in *Proceedings of 2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, 2018, pp. 5915-5920. https://doi.org/10.1109/ICRA.2018.8460608
19. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998-6008, 2017.
20. S. Yan, Z. Li, Y. Xiong, H. Yan, and D. Lin, "Convolutional sequence generation for skeleton-based action synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, South Korea, 2019, pp. 4394-4402. https://doi.org/10.1109/ICCV.2019.00449
21. G. E. Henter, S. Alexanderson, and J. Beskow, "MoGlow: probabilistic and controllable motion synthesis using normalising flows," *ACM Transactions on Graphics*, vol. 39, no. 6, article no. 236, 2020. https://doi.org/10.1145/3414685.3417836
22. Y. H. Wen, Z. Yang, H. Fu, L. Gao, Y. Sun, and Y. J. Liu, "Autoregressive stylized motion synthesis with generative flow," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13612-13621. https://doi.org/10.1109/CVPR46437.2021.01340
23. D. P. Kingma and P. Dhariwal, "Glow: generative flow with invertible 1x1 convolutions," *Advances in Neural Information Processing Systems*, vol. 31, pp. 10236-10245, 2018.
24. I. Habibie, D. Holden, J. Schwarz, J. Yearsley, and T. Komura, "A recurrent variational autoencoder for human motion synthesis," in *Proceedings of the 28th British Machine Vision Conference (BMVC)*, London, UK, 2017, pp. 1-13.
25. K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," *Advances in Neural Information Processing Systems*, vol. 28, pp. 3483-3491, 2015.
26. J. Li, D. Kang, W. Pei, X. Zhe, Y. Zhang, Z. He, and L. Bao, "Audio2Gestures: generating diverse gestures from speech audio with conditional variational autoencoders," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Montreal, Canada, 2021, pp. 11273-11282. https://doi.org/10.1109/ICCV48922.2021.01110
27. C. Ahuja and L. P. Morency, "Language2Pose: natural language grounded pose forecasting," in *Proceedings of 2019 International Conference on 3D Vision (3DV)*, Quebec City, Canada, 2019, pp. 719-728. https://doi.org/10.1109/3DV.2019.00084
28. H. Y. Lee, X. Yang, M. Y. Liu, T. C. Wang, Y. D. Lu, M. H. Yang, and J. Kautz, "Dancing to music," *Advances in Neural Information Processing Systems*, vol. 32, pp. 3581-3591, 2019.
29. J. Li, Y. Yin, H. Chu, Y. Zhou, T. Wang, S. Fidler, and H. Li, "Learning to generate diverse dance motions with transformer," 2020 [Online]. Available: https://arxiv.org/abs/2008.08171.
30. R. Li, S. Yang, D. A. Ross, and A. Kanazawa, "AI choreographer: music conditioned 3D dance generation with AIST++," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Montreal, Canada, 2021, pp. 13381-13392. https://doi.org/10.1109/ICCV48922.2021.01315
31. B. Li, Y. Zhao, S. Zhelun, and L. Sheng, "DanceFormer: music conditioned 3D dance generation with parametric motion transformer," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, pp. 1272-1279, 2022. https://doi.org/10.1609/aaai.v36i2.20014
32. A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, et al., "Learning transferable visual models from natural language supervision," *Proceedings of Machine Learning Research*, vol. 139, pp. 8748-8763, 2021. https://proceedings.mlr.press/v139/radford21a
33. J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," 2018 [Online]. Available: https://arxiv.org/abs/1810.04805v1.

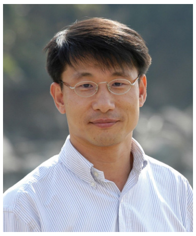
34. V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," 2019 [Online]. Available: https://arxiv.org/abs/1910.01108v1.
35. J. Chai and J. K. Hodgins, "Constraint-based motion optimization using a statistical dynamic model," in *Proceedings of the ACM SIGGRAPH 2007 Papers*, San Diego, CA, USA, 2007, pp. 8-es. https://doi.org/10.1145/1275808.1276387
36. J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 283-298, 2008. https://doi.org/10.1109/TPAMI.2007.1167
37. A. M. Lehrmann, P. V. Gehler, and S. Nowozin, "Efficient nonlinear Markov models for human motion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, 2014, pp. 1314-1321. https://doi.org/10.1109/CVPR.2014.171
38. L. Li, R. Villegas, D. Ceylan, J. Yang, Z. Kuang, H. Li, and Y. Zhao, "Task-generic hierarchical human motion prior using VAEs," in *Proceedings of 2021 International Conference on 3D Vision (3DV)*, London, UK, 2021, pp. 771-781. https://doi.org/10.1109/3DV53792.2021.00086
39. X. Tang, H. Wang, B. Hu, X. Gong, R. Yi, Q. Kou, and X. Jin, "Real-time controllable motion transition for characters," *ACM Transactions on Graphics*, vol. 41, no. 4, article no. 137, 2022. https://doi.org/10.1145/3528223.3530090
40. M. Kaufmann, E. Aksan, J. Song, F. Pece, R. Ziegler, and O. Hilliges, "Convolutional autoencoders for human motion infilling," in *Proceedings of 2020 International Conference on 3D Vision (3DV)*, Fukuoka, Japan, 2020, pp. 918-927. https://doi.org/10.1109/3DV50981.2020.00102
41. Y. Duan, T. Shi, Z. Zou, Y. Lin, Z. Qian, B. Zhang, and Y. Yuan, "Single-shot motion completion with transformer," 2021 [Online]. Available: https://arxiv.org/abs/2103.00776.
42. J. Kim, T. Byun, S. Shin, J. Won, and S. Choi, "Conditional motion in-betweening," *Pattern Recognition*, vol. 132, article no. 108894, 2022. https://doi.org/10.1016/j.patcog.2022.108894
43. Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 5745-5753. https://doi.org/10.1109/CVPR.2019.00589
44. N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black, "AMASS: archive of motion capture as surface shapes," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, South Korea, 2019, pp. 5441-5450. https://doi.org/10.1109/ICCV.2019.00554
45. M. Plappert, C. Mandery, and T. Asfour, "The kit motion-language dataset," *Big Data*, vol. 4, no. 4, pp. 236-252, 2016. https://doi.org/10.1089/big.2016.0028
46. F. De la Torre, J. Hodgins, A. Bargteil, X. Martin, J. Macey, A. Collado, and P. Beltran, "Guide to the Carnegie Mellon University Multimodal Activity (CMU-MMAC) database," 2008 [Online]. Available: http://humansensing.cs.cmu.edu/sites/default/files/11cmu-mad.pdf.
47. O. Terlemez, S. Ulbrich, C. Mandery, M. Do, N. Vahrenkamp, and T. Asfour, "Master Motor Map (MMM): framework and toolkit for capturing, representing, and reproducing human motion on humanoid robots," in *Proceedings of 2014 IEEE-RAS International Conference on Humanoid Robots*, Madrid, Spain, 2014, pp. 894-901. https://doi.org/10.1109/HUMANOIDS.2014.7041470
48. Y. Duan, Y. Lin, Z. Zou, Y. Yuan, Z. Qian, and B. Zhang, "A unified framework for real time motion completion," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 4, pp. 4459-4467, 2022. https://doi.org/10.1609/aaai.v36i4.20368
49. B. N. Oreshkin, A. Valkanas, F. G. Harvey, L. S. Menard, F. Bocquelet, and M. J. Coates, "Motion in-betweening via deep Δ-interpolator,". *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, no. 8, pp. 5693-5704, 2024. https://doi.org/10.1109/TVCG.2023.3309107
50. G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A Osman, D. Tzionas, and M. J. Black, "Expressive body capture: 3D hands, face, and body from a single image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 10975-10985. https://doi.org/10.1109/CVPR.2019.01123
51. S. Ghorbani, K. Mahdaviani, A. Thaler, K. Kording, D. J. Cook, G. Blohm, and N. F. Troje, "MoVi: a large multi-purpose human motion and video dataset," *PLOS One*, vol. 16, no. 6, article no. e0253157, 2021. https://doi.org/10.1371/journal.pone.0253157
52. N. F. Troje, "Decomposing biological motion: a framework for analysis and synthesis of human gait patterns," *Journal of Vision*, vol. 2, no. 5, pp. 317-387, 2002. https://doi.org/10.1167/2.5.2

Jaeyeong Ryu https://orcid.org/0000-0002-0186-0091

Jaeyeong Ryu received the Master's and Ph.D. degrees in Computer Graphics and Virtual Reality from Chung-Ang University, South Korea, in 2021 and 2024, respectively. He worked as a postdoctoral researcher in the Virtual Environments Lab at Chung-Ang University in 2024. His areas of interest are virtual reality, HCI, human motion generation, and generative AI.

Soungsill Park https://orcid.org/0000-0002-6113-6919

Soungsill Park received her Master's degree in Computer Graphics and Virtual Reality from Chung-Ang University, South Korea, in 2016. She is currently pursuing her Ph.D. degree in Virtual Environments Lab at Chung-Ang University. Her areas of interest are virtual reality and human action recognition.



Youngho Chai https://orcid.org/0000-0003-0513-7471

Youngho Chai holds an M.S. degree in mechanical engineering from SUNY Buffalo and was awarded a Ph.D. degree in mechanical engineering in 1997 from Iowa State University. From 2006 to 2007, he was with the Louisiana Immersive Technology Enterprise (LITE) at the University of Louisiana at Lafayette, USA. He is currently a professor with the Graduate School of Advanced Imaging Science, Multimedia, and Film, Chung-Ang University, Seoul, Korea where he leads the Virtual Environments Lab. Research interests include spatial sketching, HCI, HAR, and motion recognition.