A Implementation Details

We will provide the implementation details including the training and evaluations of PoseAugment, for future researchers to reproduce our work.

A.1 VAE Model Structure.

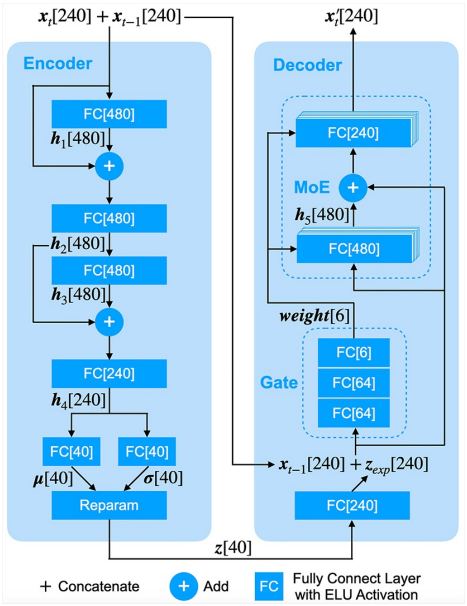


Fig. 4: The VAE model structure details. Two adjacent frames are first input to the encoder with two separate residual blocks. After reparameterization, predictions of the current frame $\boldsymbol{x}_t'$ will be reconstructed by the decoder with the MoE architecture.

Fig. 4 demonstrates the structure details of our VAE model. The current frame $\boldsymbol{x}_t$, together with the condition frame $\boldsymbol{x}_{t-1}$, are first input to the encoder network to capture a latent representation of their differences. The encoder is comprised of 4 FC layers connected by two separate residual blocks. Then, the encoding heads will output the mean and the standard deviation of the latent vector, representing its distribution. The reparameterization stage finally adds noises obeying the standard deviation $\boldsymbol{\sigma}$ to the mean vector $\boldsymbol{\mu}$, generating disturbed latent vectors.

Next, the latent vector is first decompressed to $\boldsymbol{z}_{exp}$, which will be decoded together with the condition frame $\boldsymbol{x}_{t-1}$ by the decoder with a MoE architecture [23]. The MoE network consists of 6 identical expert networks. Their output is smoothed by the weight from the gate network, generating a more stable

motion prediction. Finally, the predicted frame $\boldsymbol{x}'_t$ will the current frame in the reconstructed poses as well as the next condition frame in the next prediction.

In total, the VAE model contains 2950k parameters, which is lightweight and easy to train.

A.2 Training VAE Model.

To make the autoregressive prediction stable, we adopted the *scheduled sampling* technique proposed by [23]. First, the pose sequences are cut into mini-batches with lengths equal to 30. Within each mini-batch, each pose prediction $\boldsymbol{x}'_t$ will be used as the next conditioned frame $\boldsymbol{x}_{t-1}$ with probability $(1-p)$, while the ground truth $\boldsymbol{x}_t$ will be used with probability p. The total training epochs are divided into three stages, including the supervised stage ($p=1$), the transition stage (p decreasing from 1 to 0 linearly), and the autoregressive stage ($p=0$). Since the VAE model during inferencing is purely autoregressive, this design will make our model robust to self-prediction errors. In practice, since the sampling rate is relatively high (60Hz, which means the frame difference is small for each prediction), we choose $L=30$ (0.5s) and the lengths of the three stages to be 50, 150, and 200 epochs.

In total, we trained the VAE model first with 10 warm-up epochs, where the learning rate increased from 2×10^{-6} to 2×10^{-5} linearly. Then, the model was trained with 400 scheduled sampling epochs in total, where the learning rate started from 2×10^{-5} and decayed exponentially with a factor of 0.99 for each epoch. We used the Adam optimizer, with a batch size equal to 512.

A.3 Baseline Methods

Our baselines include Jitter, MotionAug [26], ACTOR [34], and MDM [41] (including MDM-M2M and MDM-T2M). Jitter is a universal data augmentation method, that could be applied to any data modalities. MotionAug is a VAE/IK-based method, trained on HDM05, and only supports 8 motion types. ACTOR and MDM are conditioned on action labels and text descriptions, which also require specific annotated datasets. Therefore, we first applied these methods on their corresponding datasets to generate the basic datasets, as described in Sec. 4.3. After that, we applied these methods again, together with Jitter and PoseAugment to generate the augmented datasets.

For Jitter, we add random noise $\sim N(0,\sigma^2)$ to the IMU data, following [49,50]. To find the best σ, we first conducted a pilot study, using different σ to train the TransPose model, and selected the best σ with minimal reconstruction errors. We searched σ within $\{1\times10^x, 2\times10^x, 5\times10^x\}, x\in\{-1,-2,-3,-4\}$, and got the best $\sigma=0.002$.

For MotionAug, since the IK-based method needs users to annotate the motion keyframes (referred to as "semi-automatic" in the original paper), we directly used 1/5 of the released dataset as the basic dataset, and 4/5 as the augmented dataset.

For ACTOR, we generated 200 motion clips for each action (12 actions in total) as the basic dataset, and 800 motions for each action as the augmented dataset. Each motion clip is sampled at 20Hz and contains 70 frames. Then, all motion clips are upsampled to 60Hz using quadratic interpolation, to be consistent with TransPose.

For MDM-T2M, we generate 1 motion clip for each text in the HumanML3D text split (4384 texts in total) as the basic dataset, and 4 motions for each text as the augmented dataset. Each motion clip is sampled at 20Hz and contains 120 frames. We upsampled the motions to 60Hz as well to train the TransPose model.

For MDM-M2M, we modified the diffusion model in the original MDM, such that it can denoise partially noised motions. We chose the partial noising steps to be 500 (half of the original MDM model), such that the generated motion diversity ($d_{pos} = 0.71cm, d_{rot} = 2.42°$, the same metric as in Sec. 4.1) is comparable with PoseAugment ($d_{pos} = 0.82cm, d_{rot} = 1.91°$). Then, we generated the basic and augmented datasets using the modified MDM-M2M model in a similar way as MDM-T2M.

To generate the augmented datasets for Jitter and PoseAugment, we used them on all of the basic datasets generated by MotionAug, ACTOR, MDM-M2M, and MDM-T2M. As last, we got the basic and augmented datasets for all baselines and PoseAugment to evaluate their data augmentation performance.

A.4 MoCap Model Training

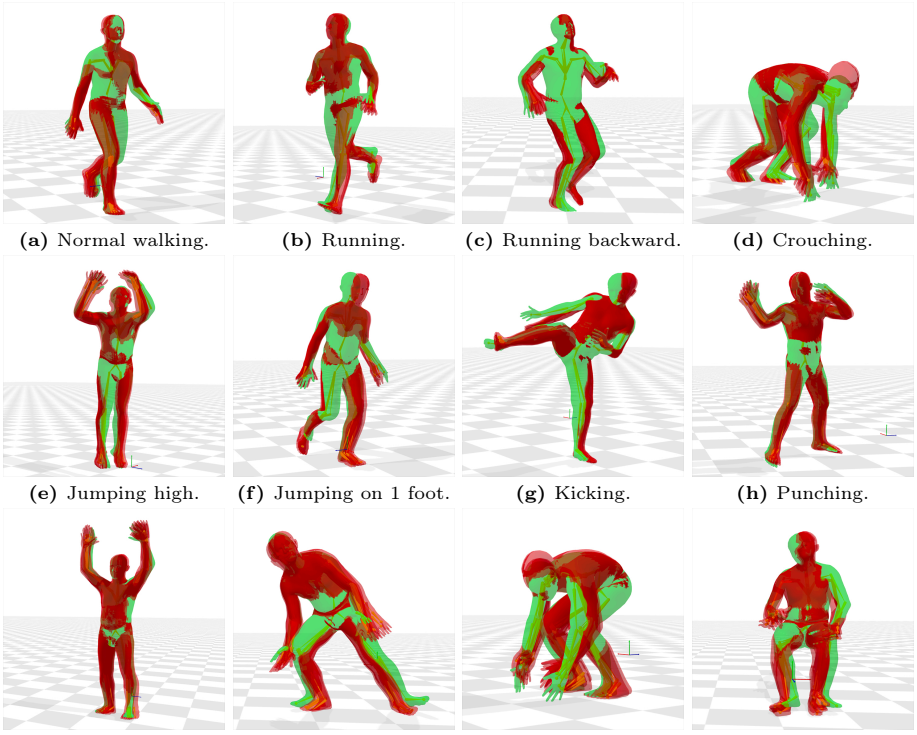
To evaluate data augmentation methods for training MoCap models, two key factors need to be addressed: (1) How much basic data should be used? (2) How much data should be augmented? They both affect the actual dataset size to train the model, which is essential for the data augmentation performance, as explored in [56].

For the basic data, the lengths of the datasets generated by MotionAug, ACTOR, and MDM are 2.65h, 2.22h, and 8.89h respectively, which are chosen to be comparable with the training dataset size of these methods. They simulate the situation of using small or big datasets in real practice. For the augmented data, we define the augmentation scale n_{aug}, which represents using $1\times$ of the basic dataset together with $(n_{aug} - 1)\times$ of the augmented dataset to train the model. We found the data augmentation performance would converge quickly when n_{aug} reaches about 5 ($4\times$ of augmented data), and more data would not improve the model performance. Therefore, we set n_{aug} to be $2 - 5$ in our evaluation and selected the best model in each training to simulate the tuning process on n_{aug}.

The training data are all resampled to 60Hz and are first cut to a fixed window size of 200 before training. For the basic dataset generated by ACTOR (2.22h), we trained the TransPose model for 200 epochs with batch size 64. When using other datasets or a different n_{aug}, we modified the training epochs accordingly to make sure the total training steps would be the same. We used the Adam optimizer, with the learning rate decreasing linearly from 5×10^{-4} to 5×10^{-5}.

B Qualitative Results

Here we further provide more motions generated by PoseAugment, to demonstrate the generalizability of our method.



(a) Normal walking. **(b)** Running. **(c)** Running backward. **(d)** Crouching.

(e) Jumping high. **(f)** Jumping on 1 foot. **(g)** Kicking. **(h)** Punching.

(i) Waving hands. **(j)** Dodging. **(k)** Bending down. **(l)** Sitting down.

Fig. 5: More poses generated by PoseAugment, including various motion types. In each subfigure, one ground truth pose (green) and 9 augmented poses (red) are visualized.

As shown in Fig. 5, we randomly selected some motions with different motion types from the AMASS dataset, and augmented 9 similar motions with PoseAugment for each of them. As a result, the augmented poses followed the original motions closely, but with more diversity to cover the motion space. It simulates the repetitions during data collection. Compared with other M2M, A2M, and T2M methods, our method is not limited to specific motion types, and maintains the original data distribution well, which is essential for the data augmentation task.

C Discussion

We would like to discuss the key factors of our data augmentation method, the comparison with diffusion-based methods, and the potential applications of our method, which will provide more insights into our design choices, and benefit future research that has similar goals to us.

C.1 Data Augmentation Performance

Fundamentally, all data-driven tasks expect the test data (in real use) to have a similar distribution with the training data, so that the knowledge the model acquires during training can be transferred to the test data. As a result, to achieve the best test performance, we need the training data distribution to be closer to the test data distribution, and the training data should cover this distribution comprehensively.

In our work, we found that the data augmentation performance is related to many factors, which confirms the above analysis. The first factor is the dataset size. With a fixed training time, models trained on the HumanML3D (8.89h) have a better accuracy compared with models trained with HumanAct12 (2.22h). Models trained with the augmented dataset would also outperform using the original dataset (Sec. 4.3). We also found the data augmentation performance of PoseAugment is generally higher on smaller datasets, due to the problem of overfitting. Thus, our method would benefit the tasks with high data collection costs or involve few-shot learning the most.

Another factor is the data distribution. We found direct manipulation of the IMU data (*e.g.* Jitter) would not improve the model performance much, since it may fail to capture the physical constraints of the body joints, thus lowering the data quality. Our method, on the other hand, synthesizes IMU signals from augmented poses with physical plausibility. This would best fit the distribution of the real MoCap data.

Last but not least, the data diversity is also important. From the experiments, we conclude that the best structure of the training dataset is "diverse motion types with proper repetitions", just like the normal data collection process. Since the ACTOR and MDM-T2M models are only conditioned on high-level information, the poses they generate have too much diversity and lack of repetitions. It would make the model hard to converge, resulting in underfitting. PoseAugment only generates poses with a high fidelity, while with appropriate diversity to simulate the motion repetitions during data collection. This would make the model easier to converge on each motion, but not overfit to a specific motion pattern compared with no data augmentation at all. As a result, it achieved the best performance in our experiments.

C.2 Compare with Diffusion-based Methods

Since diffusion models have been widely used in AIGC and human motion diffusion models [4,21,41,51,54] have also been proposed to generate poses, we would like to discuss why we chose VAE instead of diffusion models in our work.

Our first priority is data fidelity. Current human diffusion models are good at generating high-quality data from random noises and condition information, but not focus on reconstructing poses. The forward and reverse diffusion process may introduce large fluctuations to the pose distribution. Our method reconstructs motion frames with minimal errors to ensure the frame level consistency for synthesizing IMU data. Therefore, we choose VAE over diffusion models to achieve a higher data fidelity.

The second factor is data generation efficiency. Diffusion models are known for a longer inferencing time [4] due to the iterative diffusion process, thus taking a significantly longer time to generate data. Moreover, another practical reason is that current human diffusion models are trained on the HumanML3D dataset, which only generates global joint positions. They need first to be converted into local joint rotations using Inverse Kinematics (IK), which are even more time-consuming than generating joint positions. As a result, our method is 33× faster than MDM [41] and is a more pervasive data augmentation method.

C.3 Applications

PoseAugment aims to improve the model performance and alleviate the data collection burden for IMU-based motion capture. Furthermore, since the human pose is a general representation of human motion, our method can be used for any tasks driven by poses but is not limited to augmenting IMU data. For example, PoseAugment can directly benefit pose-based action recognition, anomaly detection, and motion rendering tasks. The augmented poses can also be converted to other modalities, like images or videos, to benefit CV-based motion capture and recognition. The physical module estimates the dynamic properties of the human body, which can be adopted for motion-related diseases and sports analysis. PoseAugment can also make contributions to early prototyping and explorations in research when there are few available data to use before the mass data collection. In a word, as long as data-driven approaches continue to be widely employed, PoseAugment will bring value to the aforementioned domains.