# Cross-attention Masked Auto-Encoder for Human 3D Motion Infilling and Denoising

David Björkstrand[1, 2]
dbjorks@kth.se

Josephine Sullivan[1]
sullivan@kth.se

Lars Bretzner[2]
lars.bretzner@tracab.com

Gareth Loy[2]
gareth.loy@tracab.com

Tiesheng Wang[2]
tiesheng.wang@tracab.com

[1] Robotics, Perception and Learning
KTH Royal Institute of Technology
Stockholm, Sweden

[2] Tracab
Tegeluddsvägen 3
Stockholm, Sweden

**Abstract**

Human 3D pose and motion capture have numerous applications in fields such as augmented and virtual reality, animation, robotics and sports. However, even the best capturing methods suffer from artifacts such as missed joints and noisy or inaccurate joint positions. To address this we propose the Cross-attention Masked Auto-Encoder (XMAE) for human 3D motion infilling and denoising. XMAE extends the original Masked Auto-Encoder design by introducing cross-attention in the decoder to deal with the train-test gap common in methods utilizing masking and mask tokens. Furthermore, we introduce joint displacement as an additional noise source during training, enabling XMAE to learn to correct incorrect joint positions. Through extensive experiments, we show XMAE's effectiveness compared to state-of-the-art approaches across three public datasets and its ability to denoise real-world data, reducing limb length standard deviation by 28% when applied on our in-the-wild professional soccer dataset.

## 1 Introduction

There is a vast body of work detailing various solutions for capturing human 3D pose and motion [2, 8, 14, 21] with applications in fields such as augmented and virtual reality, animation, robotics and sports. However, even the best capturing methods suffer from artifacts such as missed joints and noisy or inaccurate joint positions. As such, numerous infilling and denoising methods have been proposed to deal with these artifacts [3, 15, 22, 23].

Our work focuses on addressing artifacts observed in human 3D motion, captured with a real-world multi-view human 3D pose and motion capture system deployed in soccer arenas [24]. These estimated 3D poses and motions are valuable for understanding the state of the game and enables, or improves, applications such as video-assisted rulings, injury prevention and event detection. The motions, represented as 3D joint positions over time, suffer from missing joints due to self-occlusion or occlusion from other players, low-frequency high-magnitude displacements from mistaken joint identification in crowded situations, and high-frequency low magnitude noise. However, due to the multi-view system and often high pace of soccer games, these artifacts typically do not persist for extended periods of time. Importantly, these type of artifacts are by no means unique to our system or soccer [14], and mitigating them has broader implications for human 3D pose and motion capture systems across various domains.

While some previous work [3] focuses on achieving plausible motions, in situations where multiple plausible motions might be considered correct, plausibility is not sufficient for some applications. Instead, we concentrate on recovering the correct pose with high accuracy, in situations where this is possible. Fast inference is also crucial, as many applications either benefit from or require rapid processing, leading us to propose a direct prediction framework over those that use test time optimization [22, 23]. Furthermore, knowing the global position of joints is critical, as understanding the pose alone is not enough. The relative positions of players, and their joints, to one another provide valuable insight into the state of the game.

Inspired by the success of Masked Auto-Encoders (MAE) in learning useful features from incomplete data across various domains [7, 9, 12], and their prior application in infilling missing joints [14], we propose the Cross-attention Masked Auto-Encoder (XMAE) for human 3D motion infilling and denoising. XMAE directly predicts accurate 3D joint positions, including global translation and rotation, from partial and noisy inputs. Our contributions are as follows:

- A conceptually simple and computationally efficient MAE-based framework for human 3D motion infilling and denoising. This framework relies on two key extensions from prior work, see Section 3.

- Extensive quantitative evaluations on three standard publicly available datasets and a proprietary in-the-wild professional soccer dataset. The results confirms both the importance of the extensions and the effectiveness, given a limited computational budget, of XMAE on a broad range of motions over prior methods for short to medium gap infilling and denoising.

## 2 Related work

XMAE is a MAE based method applied to human 3D motion infilling and denoising. We review related work in both of these areas.

**Masked Auto-Encoders** MAEs learn data representations by masking parts of the input and predicting the concealed data. This approach can be regarded as a specialized form of Denoising Auto-Encoders (DAE) [26], where masking serves as the type of noise. In natural language processing, BERT [6] has utilized this technique as a pre-training task by replacing some words with mask tokens and others with random alternatives. In the domain of image processing, He *et al.* [9] applied an MAE to predict masked patches of images as a pre-training task. They also introduce an asymmetric encoder-decoder architecture where the encoder operates solely on patches that have not been masked, while mask tokens are introduced between the encoder and decoder. This methodology has also been effectively

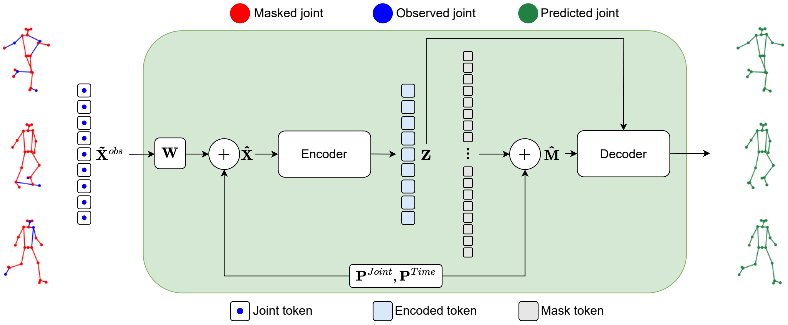


Figure 1: **Overview of XMAE.** The encoder operates solely on the embeddings of observed joint tokens. The query input to the decoder is a sequence of mask tokens with positional encoding added to it. The encoded tokens get fed to the decoder by cross-attention.

employed in pre-training tasks on video and audio [7, 12]. Moreover, the Dual-Masked Auto-Encoder (DMAE) proposed by Jiang *et al.* [14] uses an MAE to infill 3D joints missed by their 3D human pose estimation algorithm.

**Human 3D motion infilling and denoising** Various strategies have been proposed to address this problem. Kaufmann *et al.* [15] introduced a Convolutional Auto-Encoder (ConvAE), where they mask entire skeletons in the middle of the input sequence. Cai *et al.* [3] proposed a Conditional Variational Auto-Encoder (CVAE) method as a unified solution for different infilling tasks. They employed multiple masking strategies during training, focusing on generating plausible and realistic motions for situations with diverse potential outcomes. Both Rempe *et al.* [22] and Tiwari *et al.* [23] suggested using a pose prior in conjunction with test-time optimization for various human pose tasks, including infilling and denoising. Rempe *et al.* presented a CVAE-based prior, while Tiwari *et al.* noted that CVAE-based priors tend to generate poses near the mean of the computed Gaussian, which might not be accurate, and instead proposed a pose prior modeled as a signed distance field. Aliakbarian *et al.* [2] proposed a normalizing flow-based method for infilling of poses from very sparse inputs, enabling them to optimize the exact negative log-likelihood.

# 3 Method

XMAE is largely inspired by the work of He *et al.* [9] and its previous applications to human motion data [14]. However, we introduce two key extensions, cross-attention in the decoder and joint displacement as an additional noise source. Figure 1 contains an overview of XMAE. We describe its components in detail in the following sections.

## 3.1 Input to XMAE

**Tokenization of the input trajectory** Following the DMAE approach [14], we represent each 3D position of a joint $j \in [1, J]$ at each timestep $t \in [1, T]$ as an independent *joint token* $\mathbf{x}_{jt} \in \mathbb{R}^3$. The full motion sequence $\mathbf{X} \in \mathbb{R}^{3 \times JT}$ is expressed as a sequence of joint tokens:

$$\mathbf{X} = [\mathbf{x}_{11}, \dots, \mathbf{x}_{J1}, \dots, \mathbf{x}_{1T}, \dots, \mathbf{x}_{JT}] \tag{1}$$

where $J$ denotes the total number of joints in the skeleton definition, and $T$ represents the total number of timesteps in the sequence.

**Masking & displacement** Similar to prior work [7, 9, 12, 14] we select a large fraction of joint tokens to mask by uniformly sampling from $\mathbf{X}$. Masked joint tokens are removed, yielding a sequence of observed joint tokens $\mathbf{X}^{obs} \in \mathbb{R}^{3\times N}$ where $N$ is the number of unmasked tokens. Similarly, we displace joint tokens by uniformly sampling a subset of observed joint tokens from $\mathbf{X}^{obs}$ and applying offsets to their positions, resulting in $\tilde{\mathbf{X}}^{obs} \in \mathbb{R}^{3\times N}$. Each offset is uniformly sampled from positions within a sphere with a radius of a maximum joint displacement distance. Incorporating both masking and displacements of varying magnitudes enable XMAE to learn to infill missing joints and correct displaced joint positions, regardless of the magnitude of the offsets.

**Token embedding** We use a joint token embedding consisting of a linear projection and two distinct learnable positional encodings, one for the different joints and one for the different time steps. This involves three learnable matrices: $\mathbf{W} \in \mathbb{R}^{d_{model}\times 3}$, $\mathbf{P}^{Joint} \in \mathbb{R}^{d_{model}\times J}$, and $\mathbf{P}^{Time} \in \mathbb{R}^{d_{model}\times T}$, where $d_{model}$ denotes the encoder's width. Each embedded joint token $\hat{\mathbf{x}}_{jt} \in \mathbb{R}^{d_{model}}$ in the embedded joint token sequence $\hat{\mathbf{X}} \in \mathbb{R}^{d_{model}\times N}$ is given by the equation

$$\hat{\mathbf{x}}_{jt} = \mathbf{W}\tilde{\mathbf{x}}_{jt} + \mathbf{P}^{Joint}_{:,j} + \mathbf{P}^{Time}_{:,t}, \quad \forall \tilde{\mathbf{x}}_{jt} \in \tilde{\mathbf{X}}^{obs} \tag{2}$$

where $\mathbf{P}^{Joint}_{:,j}$ and $\mathbf{P}^{Time}_{:,t}$ represents the $j$-th and $t$-th columns of $\mathbf{P}^{Joint}$ and $\mathbf{P}^{Time}$ respectively. Following previous work [7, 9, 12, 14] the input to XMAE consists of the embedded joint tokens $\hat{\mathbf{X}}$ exclusively derived from the observed joint tokens $\tilde{\mathbf{X}}^{obs}$.

## 3.2 XMAE

We use the original Transformer architecture [25] for XMAE. Our encoder takes the embedded joint token sequence $\hat{\mathbf{X}}$ as input and produces the encoded token sequence $\mathbf{Z} \in \mathbb{R}^{d_{model}\times N}$. The input to the decoder consists of the encoded tokens $\mathbf{Z}$ and a sequence of embedded mask tokens $\hat{\mathbf{M}} \in \mathbb{R}^{d_{model}\times JT}$ the same length as the full unmasked joint token sequence:

$$\hat{\mathbf{M}} = [\hat{\mathbf{m}}_{11}, \dots, \hat{\mathbf{m}}_{J1}, \dots, \hat{\mathbf{m}}_{1T}, \dots, \hat{\mathbf{m}}_{JT}] \tag{3}$$

where each element $\hat{\mathbf{m}}_{jt}$ is a copy of a learnable vector $\mathbf{m} \in \mathbb{R}^{d_{model}}$ with the same positional encoding as used in the joint token embedding added to it:

$$\hat{\mathbf{m}}_{jt} = \mathbf{m} + \mathbf{P}^{Joint}_{:,j} + \mathbf{P}^{Time}_{:,t} \tag{4}$$

The embedded mask token sequence $\hat{\mathbf{M}}$ is the query input to the decoder, while the encoded token sequence $\mathbf{Z}$ is fed into the decoder through each cross-attention layer, where it's used as the source for the key and value vectors.

In contrast to XMAE, prior work [7, 9, 12, 14] employs Transformer blocks without cross-attention throughout their architectures. In these models, the decoder receives a mix of encoded tokens and mask tokens, each mask token signaling to the decoder that there is a missing token to be reconstructed. This design is not problematic if the decoder is only used during pre-training [7, 9, 12]. However, when the decoder is used at test time, as is the case for XMAE and DMAE, issues emerge. Specifically, a distributional shift occurs in the tokens received by the decoder, if the number of mask tokens during testing is different than during training [4, 6]. In the case of joint infilling, a high mask ratio during training improves

| | Hyperparameter | | | |
|---|---|---|---|---|
| Network | $d_{model}$ | depth | MLP width | Att. head dim. |
| Encoder | 256 | 6 | 1024 | 64 |
| Decoder | 128 | 3 | 512 | 64 |

Table 1: Network hyperparameters used unless stated otherwise.

performance when there are a high number of missing joints at test time (see Figure 5), which implies a degradation in performance when faced with the easier task of infilling a small number of missing joints (see Figure 7). XMAE circumvents this problem by ensuring that the query input to the decoder is always the same sequence of mask tokens $\hat{\mathbf{M}}$. A varying number of encoded tokens in $\mathbf{Z}$ as the source for key and value vectors in the cross-attention layers does not appear to pose a problem. We hypothesize the reason for the cross-attention mechanism's robustness to a varying number of key and value vectors is that the output of the cross-attention layers is a linear combination, with weights that add up to 1.

The output of the decoder is linearly projected back into 3D, and similar to DMAE, we directly regress the 3D joint positions. As in previous work the loss is calculated on the masked positions, however, we also include the displaced joint positions.

# 4 Experiments

## 4.1 Setup

**Datasets** We evaluate XMAE on three publicly available datasets, Human3.6M [13], Holden *et al.* [11] and AMASS[18], as well as a proprietary Soccer dataset.

*Human3.6M* is a widely used dataset for human motion synthesis and 3D pose estimation. It consists of 7 subjects performing various actions with 3D annotations for each joint, recorded at a 50Hz frequency. Following previous work [3] we use the standard training and validation split and the 17 joints commonly used. All numbers reported are on the validation split. We train for 60k iterations of which 1,2k are warm-up.

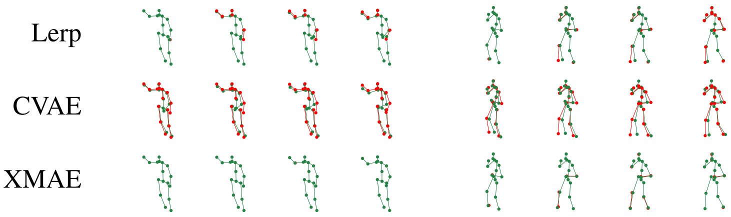
*Holden et al.* The dataset introduced by Holden *et al.* [11] is a collection of other publicly available datasets [5, 10, 19, 20]. We utilize the code implemented by Kaufmann *et al.* [15] to pre-process the data. However, unlike Kaufmann *et al.*'s method, XMAE takes global orientation and translation into account and we thus create a version of the data with these and one without. We manually set the seed to 0 to ensure the same training/validation split between the two versions. The skeletons are recorded at 120 Hz but downsampled to 60 Hz during pre-processing. We train for 100k iterations of which 2.2k are warm up.

*AMASS* is a large and diverse dataset of human motion, parameterized using SMPL [17]. We only consider the 21 SMPL [17] joints for training and testing. We only use sequence recorded at 60Hz or 120Hz, and 120Hz sequences are downmsampled to 60Hz. We use the standard training, validation and test splits [18] and the numbers reported are on the test split. We train for 240k iterations of which 5k are warm-up.

*Soccer dataset* is an in-the-wild proprietary dataset consisting of skeletons of professional soccer players and referees during games, captured by a real-world human 3D pose and motion capture system deployed in soccer arenas. The skeletons are created through 2D

| | | Gap (s) | | | | |
|---|---|---|---|---|---|---|
| Method | MPJPE | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
| Lerp | Rel. | **2.1** | 9.5 | 18.5 | 27.5 | 35.8 |
| CVAE [3] | Rel. | 15.7 | 35.5 | 48 | 57.4 | 65.7 |
| XMAE | Rel. | 3.9 | **6.7** | **11.1** | **17.2** | **26.5** |
| XMAE | Global | 3.9 | **6.7** | 11.5 | 19.1 | 30.2 |

Table 2: **Full skeleton infilling on Human3.6M validation set.** We compare Lerp, CVAE and XMAE on the full skeleton infilling task using MPJPE calculated in root-relative space. We additionally report MPJPE calculated in global space for XMAE.



a) Pointing motion.

b) Taking a step.

Figure 2: **Lerp, CVAE and XMAE on the 0.4s full skeleton infilling task.** Infilled motion is depicted in green, with corresponding ground truth in red when the distance between infilling and ground truth exceeds 1cm. Visualized at a frequency of 10Hz.

detection and multi-view 3D reconstruction at a frequency of 50 Hz. The dataset consists of 437,414 training sequences and 43,692 testing sequences. The split is team based, ensuring no overlap of players. We train for 700k iterations of which 14k are warm-up.

**Evaluation tasks** We evaluate XMAE on the tasks of infilling, correcting constant arm displacement, and correcting Gaussian noise. These tasks align with the types of artifacts we have observed in 3D skeleton tracking of soccer players.

*Infilling task.* We remove joints, within a time interval ranging from 0.1s to 0.9s, centred at the middle of the sequence. The task is to predict the removed joints. We remove either the full skeleton, arm joints or leg joints. We only remove one limb at a time, for a full test epoch. Reported numbers are an average over left and right limbs.

*Constant arm displacement task.* We randomly sample a 3D direction and a length in the range of 0.5m to 1m, and displace the joints of an arm within a time interval ranging from 0.1s to 0.9s, centred at the middle of the sequence, maintaining the same offset throughout the entire duration. The task is to predict correct positions for the displaced joints. As on the infilling task, we only displace on arm at a time for a full test epoch. Reported numbers are an average over left and right limbs.

*Gaussian noise task.* We add Gaussian noise to each joint throughout the entire sequence. On the Human3.6M dataset, we add noise to the 3D joint positions, while on the AMASS dataset we add noise to the axis-angle rotations. The task is to remove the noise.

**Evaluation Metrics** We employ two metrics: Mean Per Joint Positional Error (MPJPE) and

| | | Gap (s) | | | | |
|---|---|---|---|---|---|---|
| Method | MPJPE | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
| Lerp | Rel. | 8.8 | 21.1 | 34.9 | 47.6 | 58.3 |
| ConvAE [15] | Rel. | 26.7 | 26.5 | 42.3 | 38.9 | **40.8** |
| XMAE | Rel. | **8.1** | **13.2** | **21.2** | **32.3** | 48.5 |
| XMAE | Global | **8.1** | 14 | 23.9 | 38.7 | 62.4 |

Table 3: **Full skeleton infilling on Holden *et al.* validation set.** We compare Lerp, ConvAE and XMAE on the full skeleton infilling task using MPJPE calculated without global translation and rotation. We additionally report MPJPE calculated with global translation and rotation for XMAE.

| | Noise $\sigma$ (Rad.) | | |
|---|---|---|---|
| Method | 0.05 | 0.10 | 0.15 |
| Pose-NDF[23] | **20.2** | **28.2** | **35.4** |
| XMAE | 34.2 | 62.3 | 93.4 |

Table 4: **Gaussian noise task on AMASS.** We report MPJPE (root-relative) for Pose-NDF and XMAE.

| Skeletons | Limb $\sigma$ (mm) |
|---|---|
| Original | 8.6 |
| XMAE predicted | 6.2 |

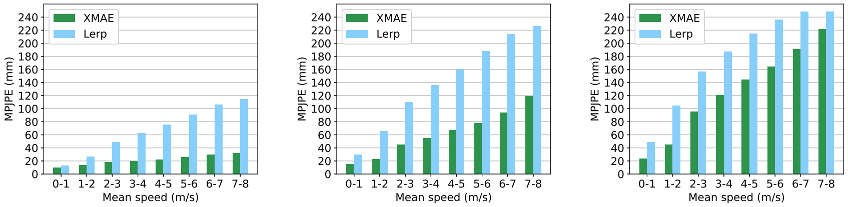
Table 5: **Summary statistics on the skeletons in the Soccer test split** before and after applying XMAE.

Limb Length Standard Deviation (Limb $\sigma$). The latter metric evaluates qualitative motion characteristics. MPJPE is our primary metric, as we want to closely match ground truth, when we predict joint positions that are either missing or noisy.

*MPJPE* is the mean Euclidean distance, reported in millimeters, between predicted and ground truth 3D joint positions. We only calculate MPJPE over displaced or removed joints, except for the Gaussian noise task where it is calculated over the whole sequence. Results for XMAE are reported w.r.t. absolute position unless stated otherwise.

*Limb $\sigma$* is the standard deviation of the length of each limb in a sequence. We report the mean over the dataset in millimeters.

**Implementation details** We employ the Adam optimizer [16], with a maximum learning rate of $3e-4$, a $\beta_1$ of 0.9, and a $\beta_2$ of 0.999. Our learning rate schedule consists of a linear warm-up followed by a cosine decay to zero. We use ReLU [1] as our activation function and employ L1 loss. The batch size is set to 128. Network hyperparameters can be found in Table 1. Assuming no masked joints and depending on the dataset, the FLOPs vary between 8B and 14B. It has 5.6M parameters. We train and test exclusively with 1-second sequences. Unless otherwise specified our training process involves masking 85% of the joint tokens and displacing 5% with a maximum joint displacement distance of 1m. We employ pre-normalization [27] in all Transformer blocks. We consider global translation, but to avoid arbitrarily high magnitude 3D coordinate values we normalize by subtracting x and y coordinates by their corresponding mean over the sequence. The mean is computed only on observed joint tokens, including displaced ones. For data augmentation, we rotate the entire sequence around the z-axis after normalization.



a) 0.2s b) 0.4s c) 0.6s

Figure 3: **Full skeleton infilling on Soccer dataset**. MPJPE (mm) split by average speed.

## 4.2 Comparison to other methods

We compare XMAE to four approaches, a Linear Interpolation (Lerp) baseline, the CVAE by Cai *et al.* [3], ConvAE by Kaufmann *et al.* [15] and Pose-NDF [23]. We run all experiments but utilize the code released by the authors. In the case of CVAE we also use the authors code to train a model without action modulation, in the cases of ConvAE and Pose-NDF we utilize the released pre-trained models.

Table 2 shows XMAE significantly outperforming CVAE in infilling gaps ranging from 0.1s to 0.9s. Lerp is most effective when the gap is very short, but its performance degrades more rapidly as the gap size increases. Figure 2 demonstrates that although CVAE generates plausible poses, it does not closely adhere to the ground truth. Lerp performs well, but Figure 2 also shows that it does not handle complex motions as effectively as XMAE.

In Table 3 we compare XMAE and ConvAE, another direct regression model. XMAE performs better on gap sizes ranging from 0.1s to 0.7s, even when global translation and rotation is included in the MPJPE calculations. On the 0.9s gap size, close to what ConvAE is specifically trained for, ConvAE outperforms XMAE.

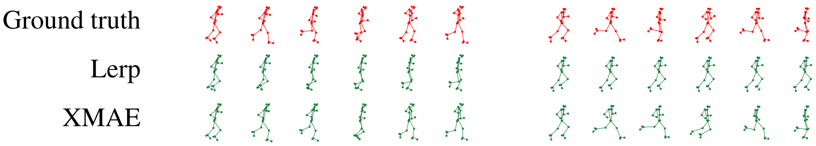
Table 4 contains a comparison between XMAE and Pose-NDF on the Gaussian noise task. Pose-NDF only considers limb angles and disregards limb lengths. To enable a comparison, we add noise to the axis-angle rotations in the SMPL parameterization. At the low noise level, XMAE performs comparably to Pose-NDF, but its performance degrades more rapidly with increasing noise. However, XMAE substantially surpasses Pose-NDF in computational efficiency, taking 6ms to denoise a 60-frame sequence, as opposed to approximately 3.5s for Pose-NDF, when benchmarked on a Geforce RTX 2080 Ti.

We applied XMAE to the Soccer data test set without artificially masking or adding noise and recorded Limb $\sigma$, as shown in Table 5. XMAE corrects noise in the dataset and reduces Limb $\sigma$ by 28%, reducing it from 8.6mm to 6.2mm. We also evaluate using the full-body infilling task on the same dataset. Figure 3 displays the results for different gap lengths, split by average speed. Higher speeds make infilling more challenging. Figure 4a illustrates that the model can handle infilling 0.6s gaps when the player is running at moderate speed, while it misses some motion when the player is running fast in Figure 4b.

## 4.3 Ablation studies

We conducted ablation studies on the Human3.6M dataset to evaluate the impact of mask ratio, cross-attention, joint displacement and network hyperparameters.

**Mask ratio** We experimented with masking 80%-95% of all joints during training, akin to



a) Running at moderate speed. b) Running fast.

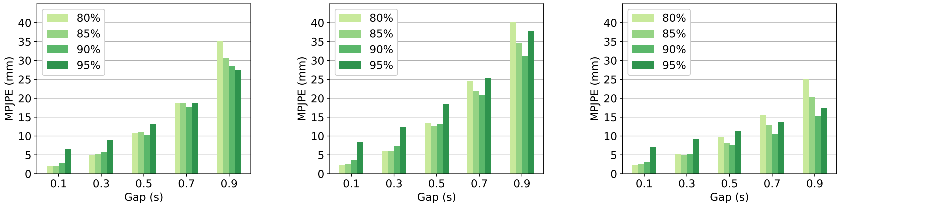
Figure 4: **Lerp and XMAE on the 0.6s full skeleton infilling task.** Infilled motions in green and removed ground truth sequence in red. Visualized at a frequency of 10Hz.

masking ratios used in previous work [7, 9, 12]. Figure 5 illustrates the results for XMAE trained with various masking ratios. The lower half of the masking ratios perform slightly better on the small gap sizes but degrades more rapidly as the gap size increase. Similarly the highest masking ratio of 95% performs the worst on the small gap size, however, it does not make up for it on the larger gap size. We choose a 90% total corruption ratio (85% mask ratio and 5% displacement ratio) as it performs well over all gap sizes.

**Joint displacement** We examined the effect of joint displacement during training by comparing XMAE trained with a 90% masking ratio to three trained with 85% masking ratios and 5% displacement ratios and varying maximum joint displacement lengths. Figure 6a shows a slight degradation in performance on the full skeleton infilling task but Figure 6b and Figure 6c demonstrates significant gains on the Gaussian noise task and Constant arm displacement task when adding any displacement. There is not a big difference between the maximum joint displacement lengths other than the 0.5m version showcasing more rapid degradation on the constant arm displacement task. We decide to employ a 5% displacement ratio with a 1m maximum joint displacement length.

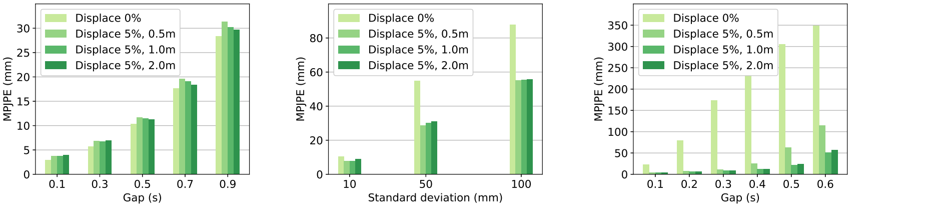
**Cross-attention** We evaluated the impact of cross-attention by comparing it to the original MAE design as well as the strategy used in BERT [6]. In BERT, they mitigate the train-test gap by replacing, during training, a percentage of the masked words with random words or the original correct word, but maintaining them in the loss function. Even though the primary aim of joint displacement during training is for XMAE to learn to correct displaced joints, we view it as a joint-specific analogue to random word replacement and train a model with an 85% masking ratio, a 5% displacement ratio, and by including 5% (of the total amount) of the clean joints in the loss calculations. We refer to this strategy as *BERT masking*. In Figure 7 we compare XMAE trained with 90% mask ratio with the original MAE design [9], also trained with a 90% mask ratio, and the BERT masking strategy. The original MAE design struggles with short gaps, but improves as the fraction of missing joints comes closer to its training mask ratio. The BERT masking strategy provides some improvement, however, XMAE works significantly better.

**Network hyperparameters**. We evaluated the impact of decoder depths, specifically using 1, 3 and 5 Transformer decoder blocks. A decoder depth of 3 yields the marginally best results on average across all tasks. In addition, we have examined the effect of network widths. We have compared the widths specified in Table 1 with half and double those, referred to as XMAE-Small and XMAE-Large. The differences are slight, with three exceptions. XMAE-Small increases MPJPE by 6.4mm and 7.6mm on average over the infilling tasks and the constant arm displacement task respectively. XMAE-Large decreases MPJPE by 4.8mm on



a) Full skeleton infilling b) Arm joints infilling c) Leg joints infilling

Figure 5: **Mask ratio ablations.** Effect of mask ratios evaluated on infilling tasks.



a) Full skeleton infilling b) Gaussian noise c) Constant arm displacement

Figure 6: **Joint displacement impact.** Evaluated on three tasks.

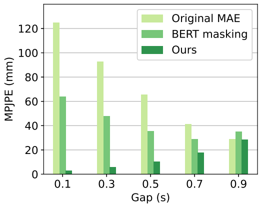


Figure 7: **Cross-attention impact.** Comparison of strategies to reduce the train-test gap.

average over the constant arm displacement task but adds 14B FLOPs.

# 5 Conclusion and future work

In this work we presented the XMAE for human 3D motion infilling an denoising. We introduced two key extensions to the original MAE design. Cross-attention in the decoder and joint displacement as an additional noise source, enabling XMAE to deal with the train-test gap and to correct inaccurate joint positions. We have demonstrated the importance of both through our ablation studies. Furthermore, we have showcased XMAE's effectiveness compared to state-of-the-art methods across three public datasets and its capabilities on real-world data. We hope that our work may offer insights for any future research aiming to utilize MAEs for human 3D motion infilling and denoising as well as other reconstruction purposes. We believe an interesting avenue for future research would be exploring integration of MAEs with generative models or otherwise circumventing direct 3D joint regression.

# Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

# References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[2] Sadegh Aliakbarian, Pashmina Cameron, Federica Bogo, Andrew Fitzgibbon, and Thomas J Cashman. Flag: Flow-based 3d avatar generation from sparse observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13253–13262, 2022.

[3] Yujun Cai, Yiwei Wang, Yiheng Zhu, Tat-Jen Cham, Jianfei Cai, Junsong Yuan, Jun Liu, Chuanxia Zheng, Sijie Yan, Henghui Ding, et al. A unified 3d human motion synthesis model via conditional variational auto-encoder. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11645–11655, 2021.

[4] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.

[5] CMU. Carnegie mellon university graphics lab motion capture database, 2003. URL http://mocap.cs.cmu.edu/. Accessed: 2023-02-23.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[7] Christoph Feichtenhofer, Yanghao Li, Kaiming He, et al. Masked autoencoders as spatiotemporal learners. *Advances in neural information processing systems*, 35:35946–35958, 2022.

[8] Vladimir Guzov, Aymen Mir, Torsten Sattler, and Gerard Pons-Moll. Human poseitioning system (hps): 3d human pose estimation and self-localization in large scenes from body-mounted sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4318–4329, 2021.

[9] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[10] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, 2015.

[11] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. In *SIGGRAPH 2016*, 2016.

[12] Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen. *Advances in Neural Information Processing Systems*, 35:28708–28720, 2022.

[13] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7): 1325–1339, 2013.

[14] Junkun Jiang, Jie Chen, and Yike Guo. A dual-masked auto-encoder for robust motion capture with spatial-temporal skeletal token completion. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5123–5131, 2022.

[15] Manuel Kaufmann, Emre Aksan, Jie Song, Fabrizio Pece, Remo Ziegler, and Otmar Hilliges. Convolutional autoencoders for human motion infilling. In *2020 International Conference on 3D Vision (3DV)*, pages 918–927. IEEE, 2020.

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

[17] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

[18] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, October 2019.

[19] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, June 2007.

[20] Ferda Ofli, Rizwan Chaudhry, Gregorij Kurillo, René Vidal, and Ruzena Bajcsy. Berkeley mhad: A comprehensive multimodal human action database. In *2013 IEEE workshop on applications of computer vision (WACV)*, pages 53–60. IEEE, 2013.

[21] N Dinesh Reddy, Laurent Guigues, Leonid Pishchulin, Jayan Eledath, and Srinivasa G Narasimhan. Tessetrack: End-to-end learnable multi-person articulated 3d pose tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15190–15200, 2021.

[22] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. Humor: 3d human motion model for robust pose estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11488–11499, 2021.

[23] Garvita Tiwari, Dimitrije Antić, Jan Eric Lenssen, Nikolaos Sarafianos, Tony Tung, and Gerard Pons-Moll. Pose-ndf: Modeling human pose manifolds with neural distance fields. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*, pages 572–589. Springer, 2022.

[24] Tracab. URL https://tracab.com/.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[26] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[27] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533. PMLR, 2020.