H_2OT: Hierarchical Hourglass Tokenizer for Efficient Video Pose Transformers

Wenhao Li, Mengyuan Liu, Hong Liu, Pichao Wang, Shijian Lu, and Nicu Sebe

Abstract—Transformers have been successfully applied in the field of video-based 3D human pose estimation. However, the high computational costs of these video pose transformers (VPTs) make them impractical on resource-constrained devices. In this paper, we present a hierarchical plug-and-play pruning-and-recovering framework, called **H**ierarchical **Ho**urglass **T**okenizer (H_2OT), for efficient transformer-based 3D human pose estimation from videos. H_2OT begins with progressively pruning pose tokens of redundant frames and ends with recovering full-length sequences, resulting in a few pose tokens in the intermediate transformer blocks and thus improving the model efficiency. It works with two key modules, namely, a Token Pruning Module (TPM) and a Token Recovering Module (TRM). TPM dynamically selects a few representative tokens to eliminate the redundancy of video frames, while TRM restores the detailed spatio-temporal information based on the selected tokens, thereby expanding the network output to the original full-length temporal resolution for fast inference. Our method is general-purpose: it can be easily incorporated into common VPT models on both *seq2seq* and *seq2frame* pipelines while effectively accommodating different token pruning and recovery strategies. In addition, our H_2OT reveals that maintaining the full pose sequence is unnecessary, and a few pose tokens of representative frames can achieve both high efficiency and estimation accuracy. Extensive experiments on multiple benchmark datasets demonstrate both the effectiveness and efficiency of the proposed method. Code and models are available at https://github.com/NationalGAILab/HoT.

Index Terms—3D Human Pose Estimation, Video Pose Transformer, Token Pruning, Token Recovering

1 Introduction

3D Human pose estimation (HPE) from videos has numerous applications, such as action recognition [1], [2], [3], human-robot interaction [4], [5], [6], and computer animation [7], [8]. Current video-based 3D HPE methods mainly follow the pipeline of 2D-to-3D pose lifting [9], [10], [11], [12]. This two-stage pipeline first utilizes an off-the-shelf 2D HPE model to detect 2D body joints for each video frame and then employs a separate lifting model to estimate 3D pose sequences from the detected 2D poses.

Recently, transformer-based architectures [13], [14], [15], [16], [17] have shown state-of-the-art (SOTA) performance in the field of video-based 3D HPE, thanks to their competency in modeling the long-range dependencies among video frames. These video pose transformers (VPTs) typically regard each video frame as a pose token and utilize extremely long video sequences to achieve superior HPE performance (*e.g.*, 81 frames in [13], 243 frames in [15], [16], [18], or 351 frames in [14], [19], [20]). However, these methods inevitably suffer from high computational costs since the complexity of the self-attention in VPT grows quadratically with respect to the number of tokens (*i.e.*, frames), hindering the deployment of these heavy VPTs in many real-world applications.

Two factors are crucial for achieving efficient VPTs. **First**, directly reducing the frame number can boost VPTs' efficiency, but it results in a small temporal receptive field that hinders the model from capturing sufficient spatio-temporal information in pose estimation [22], [23]. Hence, it is essential to design an efficient solution while maintaining a large temporal receptive field for accurate estimation. **Second**, adjacent frames in a video sequence contain redundant information due to the similarity of nearby poses (50 Hz cameras used in Human3.6M [24]). Moreover, recent studies [25], [26] found that many tokens tend to be similar in the deep transformer blocks. Hence, using full-length pose tokens in these blocks tends to introduce redundant calculations but contributes little to the pose estimation.

Based on these observations, we propose to prune pose tokens in the deep transformer blocks to improve the efficiency of VPTs. Although token pruning can reduce the number of tokens and improve efficiency, it also makes it difficult to estimate the consecutive 3D pose of all frames, as each token corresponds to one frame in existing VPTs [14], [15], [16]. Additionally, for efficient inference, a real-world 3D HPE system should be able to estimate the 3D poses of all frames at once in an input video. Therefore, it is necessary to recover the full-length tokens to estimate 3D poses for all frames so that the model can achieve fast inference and be compatible with existing VPT frameworks.

Driven by this analysis, we present a novel hierarchical *pruning-and-recovering* framework for efficient transformer-based 3D HPE from videos. Different from existing VPTs [13], [14], [15], [16] that maintain the full-length sequence

- *Wenhao Li is with the State Key Laboratory of General Artificial Intelligence, Peking University, Shenzhen Graduate School, China, and the School of Computer Science and Engineering, Nanyang Technological University, Singapore. E-mail: wenhao.li@ntu.edu.sg.*
- *Mengyuan Liu and Hong Liu are with the State Key Laboratory of General Artificial Intelligence, Peking University, Shenzhen Graduate School, China. E-mail: {liumengyuan, hongliu}@pku.edu.cn.*
- *Pichao Wang is with Amazon Prime Video, USA. The work does not relate to author's position at Amazon. E-mail: pichaowang@gmail.com.*
- *Shijian Lu is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. E-mail: Shijian.Lu@ntu.edu.sg.*
- *Nicu Sebe is with the University of Trento, Italy. E-mail: nicu-lae.sebe@unitn.it.*
- *Corresponding author: Mengyuan Liu, Hong Liu.*

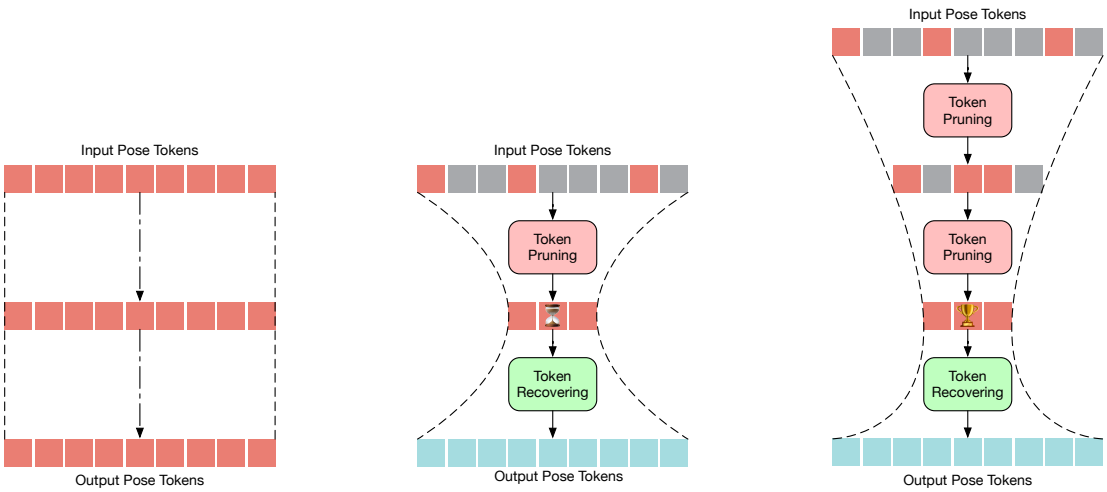


Fig. 1: Illustration of different VTP frameworks. **(a)** Existing VPTs [13], [14], [15], [16] follow a "rectangle" paradigm that retains the full-length sequence across all blocks, which incurs expensive and redundant computational costs. **(b)** HoT [21] follows an "hourglass" paradigm that prunes the pose tokens and recovers the full-length tokens, which keeps a few tokens in the intermediate transformer blocks and thus improves the model efficiency. **(c)** Our H_2OT extends HoT [21] by introducing a hierarchical pruning design, forming an "trophy-shaped (pyramidal)" paradigm. The gray squares represent the pruned tokens.

across all blocks, our method begins with progressively pruning the pose tokens of redundant frames and ends with recovering the full-length tokens. With such pruning and recovering designs, we can keep only a few tokens in the intermediate transformer blocks and effectively improve the model efficiency (see Fig. 1). Specifically, we design a Token Pruning Module (TPM) to select a few representative tokens, which can maintain rich information while reducing video redundancy. In addition, we design a Token Recovering Module (TRM) to expand the low temporal resolution caused by the pruning operation to the full temporal resolution. This strategy enables the network to estimate consecutive 3D poses of all frames and clearly improves the inference speed.

Our method can be easily integrated into existing VPTs [14], [15], [16], [27] with minimal modifications (see Fig. 2). Specifically, the first few transformer blocks of VPTs remain unchanged to obtain pose tokens with comprehensive information from full video frames. These pose tokens are then progressively pruned by the proposed TPM with its hierarchical pruning design. The remaining tokens, serving as representative tokens, are further fed into the subsequent transformer blocks. Finally, TRM recovers the full-length tokens after the last transformer block, while the intermediate transformer blocks still use representative tokens.

To validate the effectiveness and efficiency of our method, we deploy it on top of SOTA VPTs, including MHFormer [14], MixSTE [15], MotionBERT [16], and MotionAGFormer [27]. Extensive experiments demonstrate that the proposed method can greatly reduce the computational costs without sacrificing the pose estimation performance.

This work can be summarized as follows:

- We present H_2OT, a hierarchical plug-and-play *pruning-and-recovering* framework for efficient transformer-based 3D HPE from videos. Our H_2OT reveals that maintaining the full-length pose sequence is redundant, and identifying and keeping a few representative pose tokens can achieve both high efficiency and performance.
- To accelerate VPTs, we propose a Token Pruning Module (TPM) to select a few representative tokens for video redundancy reduction and a Token Recovering Module (TRM) to restore the original temporal resolution for fast inference.
- Extensive experiments conducted on four recent VPTs show that H_2OT achieves highly competitive or even superior results while significantly improving efficiency.

Part of the material presented in the paper appeared in our CVPR 2024 conference paper [21]. This journal version extends [21] in several ways:

- Unlike [21] that lacks a hierarchical design, we propose a novel hierarchical pruning strategy that gradually reduces the number of tokens as the layer gets deeper. Such a hierarchical design introduces a pyramidal feature hierarchy to VPTs, enabling our method to reduce video redundancy more effectively and further improve the efficiency of VPTs.
- We observe that cluster pruning and attention recovering strategies in [21] require additional parameters and impose a significant burden on the inference time compared to the original VPT models. To address this issue, we introduce sampling pruning and interpolation recovering strategies that are parameter-free and fast which makes H_2OT more robust and suitable for real-world applications.
- With the above new designs, H_2OT enables more efficient VPTs as compared with [21], as validated in extensive ablation studies.
- We extend the experiments by comparing our H_2OT with very recent works. Extensive experiments show that the proposed framework achieves both high efficiency and estimation accuracy compared with SOTA VPTs.

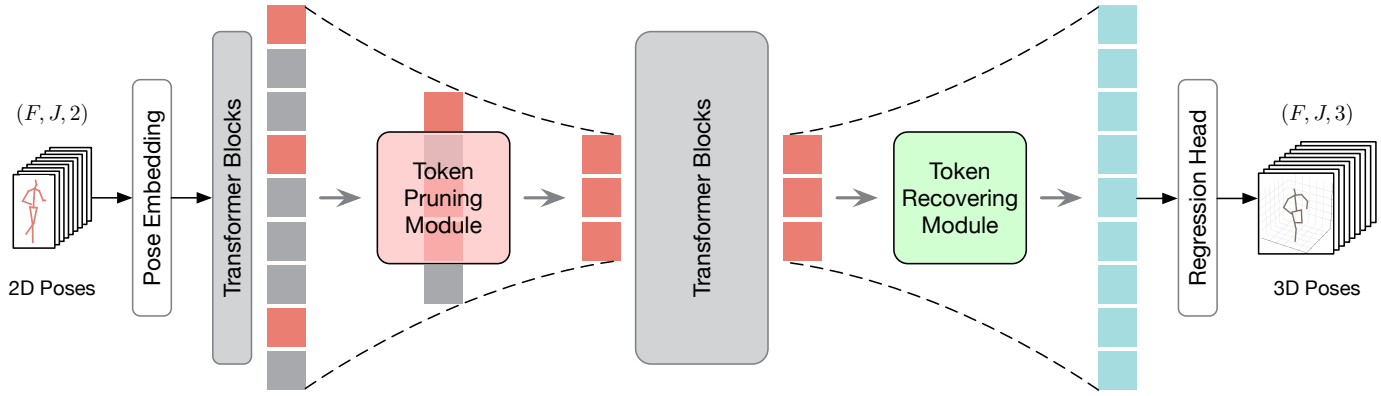


Fig. 2: Overview of the proposed Hierarchical Hourglass Tokenizer (H_2OT). H_2OT mainly consists of a Token Pruning Module (TPM) and a Token Recovering Module (TRM). TPM selects the pose tokens of representative frames after the first few transformer blocks and TRM recovers the full-length tokens after the last transformer block. Note that TRM can be added either before or after the regression head and we add it before the regression head here.

2 Related Work

Transformer-based 3D HPE. Transformers are firstly proposed in [28] have been successfully applied to different computer vision tasks [29], [30], [31], [32] and video-based 3D HPE [14], [15], [16], [33], [34]. These video pose transformers (VPTs) are often built to capture spatial and temporal information for 3D HPE using transformers. PoseFormer [13] designs a transformer-based architecture to capture joint correlations and temporal dependencies. MHFormer [14] learns spatio-temporal multi-hypothesis representations of 3D human poses via transformers. MixSTE [15] proposes a mixed spatio-temporal transformer to capture the temporal motion of different body joints. MotionBERT [16] presents a dual-stream spatio-temporal transformer to model long-range spatio-temporal relationships among skeletal joints.

However, the improved performance of these VPTs comes with a heavy computation burden, which limits their applications in practice. In this work, we improve the efficiency of existing VPTs by keeping a few representative tokens in the intermediate transformer blocks.

Efficient 3D HPE. Efficient 3D HPE is critical in computing resource-constrained environments. Existing explorations mainly focus on efficient architecture design [22], [35], [36] and data redundancy reduction [18], [19], [20], [37]. VPose [22] presents a fully convolutional architecture that processes multiple frames in parallel. Strided [19] designs a strided transformer encoder to aggregate redundant sequences. Recently, several studies [18], [20], [37] have attempted to improve model efficiency from the input sequence. For example, P-STMO [18] proposes a temporal downsampling strategy on the input side to diminish data redundancy. DeciWatch [37] proposes a flow that takes sparsely sampled frames as inputs. Einfalt *et al.* [20] designs a joint uplifting and upsampling transformer architecture to generate dense 3D poses from sparse sequences of 2D poses. Besides, many efficient methods [19], [20], [38] are designed for a specific model and some methods [37], [39] are built on single-frame estimators [35], [40], [41], [42].

However, none of them unifies the efficient design for different VPTs. We are the first to propose a plug-and-play framework for efficient VPTs, which can be plugged into common VPT models.

Token Pruning for Transformers. The self-attention complexity in transformers grows quadratically with the number of tokens, making it infeasible for high spatial or temporal resolution inputs. Several studies [43], [44], [45], [46] attempt to alleviate this issue with token pruning, aiming to select significant tokens from different inputs. The studies show that discarding less informative tokens in the deep transformer blocks only leads to a slight performance drop. DynamicViT [25] proposes a learnable prediction module to estimate the scores of tokens and prune redundant tokens. PPT [47] selects important tokens based on the attention score. TCFormer [48] presents a token clustering transformer to cluster and merge tokens. VTC-LFC [26] compresses ViTs from frequency domain and prunes both parameters and tokens. GTPT [49] groups human keypoint tokens and prunes visual tokens to reduce redundancy.

In this work, we are the first to perform token pruning in VPTs for model acceleration. Unlike these studies that aim to reduce less related information (*e.g.*, image background) from images in the spatial domain, we focus on reducing video redundancy by selecting a few pose tokens of representative frames in the temporal domain. Furthermore, we propose to restore the full-length temporal resolution to meet the domain-specific requirement of efficient video-based 3D HPE.

3 Method

Fig. 2 illustrates the overview of our Hierarchical Hourglass Tokenizer (H_2OT). We propose a Token Pruning Module (TPM) and a Token Recovering Module (TRM) for token pruning and recovering. Our H_2OT is a general-purpose *pruning-and-recovering* framework, where TPM and TRM can use different token pruning and token recovering strategies, and we insert them into SOTA VPTs [14], [15], [16], [27]. In the following, we give details about the proposed TPM and TRM and show how to apply them to existing VPTs.

3.1 Token Pruning Module

We observe that the existing VPTs [14], [15], [16], [27] take long video sequences as input and maintain the full-length

Algorithm 1 Hierarchical Pruning Strategy

Input: Pose tokens $x \in \mathbb{R}^{F \times J \times C}$
Output: Representative tokens $\tilde{x} \in \mathbb{R}^{r_M \times J \times C}$

```
for layer index l = 1 in L do
    for stage index m = 1 in M do
        if l = b_m then
            Token pruning at b_m-th transformer block
        end if
    end for
    Feature extraction at l-th transformer block
end for
return x̃
```

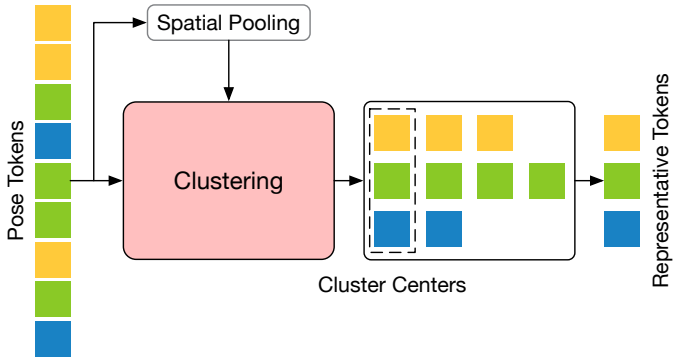


Fig. 3: Illustration of our Token Pruning Cluster (TPC) architecture. Given the input pose tokens, we pool them in the spatial dimension, cluster the input tokens into several groups according to the feature similarity of the pooled tokens, and select the cluster centers as the representative tokens.

sequence across all blocks (Fig. 1 (a)), which is computationally expensive for high temporal resolution inputs. To tackle this issue, we propose a Token Pruning Module (TPM) that prunes the pose tokens of video frames to improve the efficiency of VPTs. In our earlier conference work [21], we maintain the full-length sequence in the early stages and perform token pruning in large chunks which often leads to clear information loss.

Inspired by the hierarchical design in CNNs [50], [51], we design hierarchical pruning that gradually reduces the number of tokens across network layers, leading to a pyramidal feature hierarchy of VPTs which helps preserve more useful information. Algorithm 1 shows the overall algorithm of the proposed hierarchical token pruning. Compared to the one-time pruning design (as in our published conference paper [21]) that maintains the full-length sequence in the early stages, this hierarchical pruning design reduces the sequence at various stages, creating a pyramidal feature hierarchy for VPTs (see Figure 1). Let $r_m \in \{r_1, r_2, \ldots, r_M\}$ be the number of representative tokens, $b_m \in \{b_1, b_2, \ldots, b_M\}$ be the block index of representative tokens, and L be the number of transformer blocks at the m-th stage. TPM takes the pose tokens $x_{b_m} \in \mathbb{R}^{r_{m-1} \times J \times C}$ of b_m-th transformer block as inputs and outputs a few representative tokens $\tilde{x}_{b_m} \in \mathbb{R}^{r_m \times J \times C}$ ($r_m < r_{m-1}$). Here, J is the number of body joints, C is the feature dimension, and $r_0 = F$ is the number of input frames. Note that our method only prunes the tokens along the temporal dimension since the frame number F is much larger than the joint number J (*e.g.*, $F = 243$ and $J = 17$), *i.e.*, the expensive and redundant computational costs are dominated by the frame number in the temporal domain.

Another challenging question is how to select a few pose tokens that maintain rich information for accurate 3D HPE. In the following, we explore four token pruning strategies to answer this question, including Token Pruning Cluster (TPC), Token Pruning Attention (TPA), Token Pruning Motion (TPMo), and Token Pruning Sampler (TPS).

3.1.1 Token Pruning Cluster

We propose a simple, effective, and parameter-free Token Pruning Cluster (TPC) that dynamically selects a few pose tokens of representative frames to eliminate video redundancy. The architecture of TPC is illustrated in Figure 3. Given the input pose tokens of the b_m-th transformer block $x_{b_m} \in \mathbb{R}^{r_{m-1} \times J \times C}$, an average spatial pooling is used along the spatial dimension to remove spatial redundancy, resulting in pooled tokens $\overline{x}_{b_m} \in \mathbb{R}^{r_{m-1} \times C}$. Then, we apply an efficient density peaks clustering based on k-nearest neighbors (DPC-kNN) algorithm [52]. This algorithm clusters the input pose tokens into several groups according to the feature similarity of the pooled tokens without requiring an iterative process.

The cluster centers of tokens are characterized by a higher density compared to their neighbors, as well as a relatively large distance from other tokens with higher densities. For a token $x^i \in \overline{x}_{b_m}$, the local density of tokens ρ is calculated by:

$$\rho_i = \exp\left(-\frac{1}{k} \sum_{x^j \in k\mathrm{NN}(x^i)} \left\|x^i - x^j\right\|_2^2\right), \tag{1}$$

where $k\mathrm{NN}\left(x^i\right)$ are the k-nearest neighbors of a token x^i.

We then define the δ_i that measures the minimal distance between the token x^i and other tokens with higher density. The δ_i of the token with the highest density is set to the maximum distance between it and any other tokens. The δ_i of each token is calculated by:

$$\delta_i = \begin{cases} \min_{j:\rho_j > \rho_i} \left\|x^i - x^j\right\|_2, & \text{if } \exists \rho_j > \rho_i \\ \max_j \left\|x^i - x^j\right\|_2, & \text{otherwise} \end{cases}. \tag{2}$$

The clustering center score of a token x^i is computed by multiplying the local density ρ_i and the minimal distance δ_i (*i.e.*, as $\rho_i \times \delta_i$). A higher score indicates that the token has a large density and distance, and so a higher potential to be the cluster center. The top-r_m-scored input pose tokens are selected as cluster centers, and the remaining tokens are assigned to the nearest cluster center with higher density.

The cluster centers have high semantic diversity, containing more informative information than the other tokens. Therefore, the cluster centers serve as the representative tokens $\tilde{x}_{b_m} \in \mathbb{R}^{r_m \times J \times C}$ for efficient estimation, and the remaining tokens are discarded for reduction of video redundancy.

3.1.2 Token Pruning Attention

VPTs [13], [14], [15], [16], [19] are based on the transformer architecture [28], which is effective at capturing long-range spatio-temporal dependencies. The core of the transformer is

the self-attention mechanism, which matches a query with a set of key-value pairs to produce an output [28]. Given the input x, three linear projections are used to transform x into three matrices: queries Q, keys K, and values V. The self-attention operation can then be computed as follows:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V = \alpha V, \quad (3)$$

where $Q \in \mathbb{R}^{n_q \times d}$, $K \in \mathbb{R}^{n_k \times d}$, and $V \in \mathbb{R}^{n_v \times d}$. d is the dimension and $n_q = n_k = n_v = n$ is the number of tokens. $\alpha \in \mathbb{R}^{n \times n}$ is the attention score that determines how much information of each pose token is fused into the output. Thus, the attention score α can be used to select informative tokens.

To this end, we propose a Token Pruning Attention (TPA) module that uses the attention score to select a few pose tokens of representative frames. Given the input pose tokens of the b_m-th transformer block $x_{b_m} \in \mathbb{R}^{r_{m-1} \times J \times C}$ and the attention score $\alpha_{b_m} \in \mathbb{R}^{r_{m-1} \times r_{m-1}}$, we use $\overline{\alpha}_{b_m} \in \mathbb{R}^{r_{m-1}} = \sum_j \alpha_{b_m}^{i,j}$ as the criterion and select the top-r_m-scored pose tokens as representative tokens $\tilde{x}_{b_m} \in \mathbb{R}^{r_m \times J \times C}$. This attention score based pruning strategy is learnable and can adaptively select the most informative tokens for efficient estimation.

3.1.3 Token Pruning Motion

Human motion can reveal variations in body movement. Therefore, it is reasonable to assume that by using human motion to identify keyframes in a motion sequence, we can retain frames with significant changes while removing those with minimal changes, thereby reducing redundant information.

With this assumption, we propose a Token Pruning Motion (TPMo) module that selects a few pose tokens of representative frames based on human motion. Specifically, given the input 2D pose sequence $p \in \mathbb{R}^{F \times J \times 2}$ detected by an off-the-shelf 2D HPE detector from a video, we first transform it into $s \in \mathbb{R}^{F \times (J \cdot 2)}$. For the b_m-th transformer block, the human motion can be formulated as:

$$M_{b_m} = \left\{0, s_{b_m}^2 - s_{b_m}^1, s_{b_m}^3 - s_{b_m}^2, \cdots, s_{b_m}^{t+1} - s_{b_m}^t\right\}, \quad (4)$$

where $s_{b_m}^t$ is the t-th frame of the sequence s in the b_m-th transformer block, and $M_{b_m} \in \mathbb{R}^{r_{m-1} \times (J \cdot 2)}$ is the motion matrix. Then, we use $\overline{M}_{b_m} \in \mathbb{R}^{r_{m-1}} = \sum_j M_{b_m}^{i,j}$ as the criterion and select the top-r_m-scored pose tokens as representative tokens $\tilde{x}_{b_m} \in \mathbb{R}^{r_m \times J \times C}$. This human motion based pruning strategy can consider redundancy and context simultaneously to select frames with a large variation in motion.

3.1.4 Token Pruning Sampler

Our conference version [21] chooses the TPC module for token pruning, which is simple, effective, and parameter-free. However, we observe some disadvantages in TPC used in [21]: **(i)** TPC utilizes a cluster-based algorithm for token pruning, but it consumes a big burden on the inference time compared to the original VPT models. **(ii)** Employing clustering necessitates token selection in an unordered manner which clearly degrades the interpolation efficiency. This issue also exists in the dynamic token pruning strategies like TPA and TPMo.

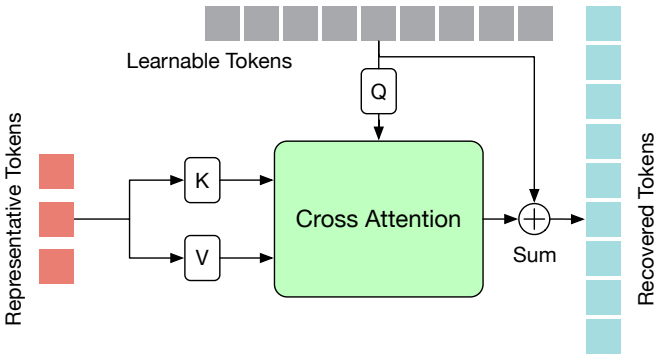


Fig. 4: Illustration of our Token Recovering Attention (TRA). TRA takes the representative tokens of the last transformer block, along with learnable tokens that are initialized to zero, as input to recover the full-length tokens.

To address these issues, we propose a Token Pruning Sampler (TPS) module that uses a linear sampling strategy to select a few pose tokens. Given the input pose tokens of b_m-th transformer block $x_{b_m} \in \mathbb{R}^{r_{m-1} \times J \times C}$, TPS uniformly samples pose tokens along the temporal dimension and output a few representative tokens $\tilde{x}_{b_m} \in \mathbb{R}^{r_m \times J \times C}$. Such a simple sampling strategy is reasonable due to data redundancy (i.e., nearby poses are similar) on Human3.6M dataset [24] (captured by 50 Hz cameras). TPS is parameter-free, efficient, and requires no additional computational costs and inference time. Besides, it enables the token selection in an ordered manner, which is more suitable for efficient interpolation.

3.2 Token Recovering Module

A large number of pose tokens have been pruned by TPM, which significantly reduces the computational costs. However, for fast inference, a real-world 3D HPE system should be capable of estimating the consecutive 3D poses of all frames in a given video (this is called *seq2seq* pipeline in [15]). Therefore, different from some token pruning methods in vision transformers that can use a few selected tokens to directly perform classification [53], [54], [55], [56], we need to recover the full-length tokens to keep the same number of tokens as the input video frames (in existing VPTs, each token corresponds to a frame). Meanwhile, for efficiency purposes, the recovering module should be lightweight.

3.2.1 Token Recovering Attention

We develop a lightweight Token Recovering Attention (TRA) module to restore the spatio-temporal information from the selected pose tokens, as shown in Fig. 4. It only contains one multi-head cross-attention (MCA) layer without any additional networks. The dot-product attention [28] in the MCA is defined in Eq. 3.

Our MCA takes the learnable tokens $x' \in \mathbb{R}^{F \times C}$ that are initialized to zero as queries and the j-th joint representative tokens of the last transformer block $x_L^j \in \mathbb{R}^{r_M \times C}$ as keys and values, followed by a residual connection:

$$\hat{x}^j = x' + \text{MCA}(x', x_L^j, x_L^j), \quad (5)$$

where L is the number of transformer blocks. $\text{MCA}(\cdot)$ is the function of MCA, and its inputs are queries, keys, and values.

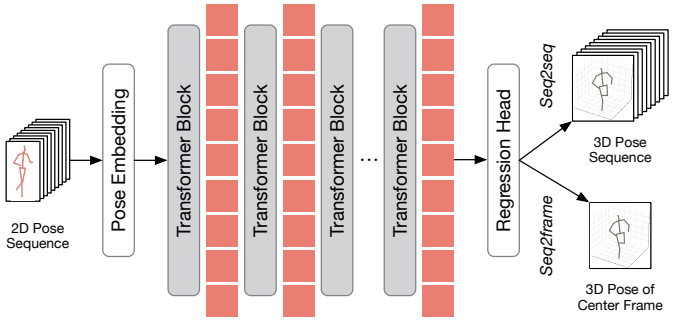


Fig. 5: Summary of existing VPT architectures. Existing VPTs typically contain a pose embedding module, a stack of transformer blocks, and a regression head module. The outputs of the regression head can be either the 3D poses of all frames for the *seq2seq* pipeline or the 3D pose of the center frame for the *seq2frame* pipeline.

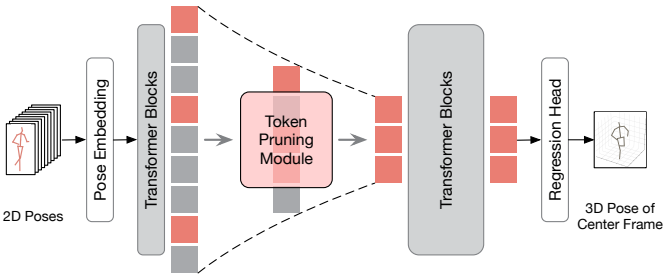


Fig. 6: Illustration of our framework on *seq2frame* pipeline. The pose tokens are fed into TPM to select representative tokens. After the regression head, the 3D pose of the center frame is selected as the output for evaluation.

$\hat{x}^j \in \mathbb{R}^{F\times C}$ is the j-th joint recovered token, whose temporal dimension is the same as the queries (*i.e.*, the designed learnable tokens).

The TRA performs a reverse operation of selecting representative tokens, which recovers tokens of full-length temporal resolution from low ones using high-level spatio-temporal semantic information.

3.2.2 Token Recovering Interpolation

Although TRA is an effective design choice for token recovering, it still requires additional parameters and computational costs. To address this issue in our conference paper [21], we design a Token Recovering Interpolation (TRI) module that uses a simple interpolation operation to recover the full-length 3D human poses. The tokens of the last transformer block $x_L \in \mathbb{R}^{r_M\times J\times C}$ are fed into a regression head to estimate the 3D pose sequence $\hat{q} \in \mathbb{R}^{r_M\times J\times 3}$. Then, a linear interpolation operation is used to recover the 3D poses of all frames $q \in \mathbb{R}^{F\times J\times 3}$. Note that efficient interpolation with parallel computing requires that the interpolated sequence should be ordered. Therefore, in TPM, it is necessary to use TPS in conjunction with TRI.

3.3 Applying to Video Pose Transformers

3.3.1 Video Pose Transformers

Recent studies of video pose transformers (VPTs) [13], [15], [16], [18], [19], [57] are mainly designed to estimate 3D poses from 2D pose sequences. These VPTs share a similar architecture, which includes a pose embedding module (often containing only a linear layer) to embed spatial and temporal information of pose sequences, a stack of transformer blocks to learn global spatio-temporal correlations, and a regression module to predict 3D human poses. We summarize the architecture in Fig. 5.

Each transformer block consists of a multi-head self-attention (MSA) layer and a feed-forward network (FFN) layer. Let N be the number of tokens, D be the dimension, and $2D$ be the expanding dimension in the FFN (the expanding ratio in VPTs is typically 2). The calculational costs of MSA and FFN are $\mathcal{O}\left(4ND^2+2N^2D\right)$, and $\mathcal{O}\left(4ND^2\right)$, respectively. Thus, the total computational complexity is $\mathcal{O}\left(8ND^2+2N^2D\right)$, which makes VPTs computationally expensive. Since the dimension D is important to determine the modeling ability and most recent VPTs employ a D of 512 or 256, we follow their hyperparameter settings and propose to prune pose tokens of video frames (*i.e.*, reducing N) to reduce the computational cost of VPTs.

There are two types of pipelines based on VPT's inference outputs: *seq2frame* [13], [14], [18], [19] and *seq2seq* [15], [16], [27] pipelines. The *seq2frame* pipeline outputs the 3D pose of the center frame and requires repeated inputs of 2D pose sequences with significant overlap to predict the 3D poses of all frames, which makes it less efficient but can achieve better performance by considering both past and future information. In contrast, the *seq2seq* pipeline outputs 3D poses of all frames from the input 2D pose sequence at once, making it more efficient but less accurate. As a result, these two pipelines have their unique strengths, and we need to develop two strategies to better accommodate their different inference manners.

3.3.2 Applying to Seq2seq Pipeline

For the *seq2seq* pipeline, the outputs are all frames of the input video, and hence we need to restore the original temporal resolution. TPM and TRM are inserted into VPTs, where TPM prunes the tokens after a few transformer blocks and TRM recovers the full-length tokens after the last transformer block, as shown in Fig. 2. Given the input 2D pose sequence $p \in \mathbb{R}^{F\times J\times 2}$ detected by an off-the-shelf 2D HPE detector from a video, we first feed p into a pose embedding module to embed spatial and temporal information, resulting in tokens $x \in \mathbb{R}^{F\times J\times C}$. The embedded tokens are then fed into first few transformer blocks. Next, TPM selects representative tokens $\tilde{x} \in \mathbb{R}^{r_M\times J\times C}$ after several hierarchical pruning stages, and use them as the inputs of subsequent transformer blocks. TRM has two different design choices: TRA and TRI. **(i)** TRA recovers the full-length tokens from the tokens of the last transformer block $x_L \in \mathbb{R}^{r_M\times J\times C}$, resulting in recovered tokens $\hat{x} \in \mathbb{R}^{F\times J\times C}$. It finally includes a regression head to estimate the 3D poses of all frames $q \in \mathbb{R}^{F\times J\times 3}$. **(ii)** For TRI, the tokens of the last transformer block $x_L \in \mathbb{R}^{r_M\times J\times C}$ are fed into the regression head to estimate the 3D pose sequence $\hat{q} \in \mathbb{R}^{r_M\times J\times 3}$. Then TRI utilizes the interpolation operation to recover the 3D poses of all frames $q \in \mathbb{R}^{F\times J\times 3}$.

3.3.3 Applying to Seq2frame Pipeline

For the *seq2frame* pipeline, the output is the 3D pose of the center frame. Therefore, TRM is unnecessary and we only insert TPM into VPTs. Since the token of the center frame directly corresponds to the output and can provide crucial information to the final estimation, we concatenate it with the selected tokens to make this pipeline work better. As shown in Fig. 6, the early stages of both pipelines share the same workflow. After the last transformer block, the tokens are directly sent to the regression head to perform regression and the 3D pose of center frame $q_{center} \in \mathbb{R}^{1\times J\times 3}$ is selected as the final prediction.

4 Experiments

4.1 Datasets and Evaluation Metrics

Datasets. We evaluate our method on three 3D HPE benchmark datasets: Human3.6M [24], MPI-INF-3DHP [58], and H3WB [59]. Human3.6M is the most widely used dataset for 3D HPE. It consists of 3.6 million video frames recorded by four RGB cameras at 50 Hz in an indoor environment. This dataset includes 11 actors performing 15 daily actions. Following [57], [60], [61], [62], subjects S1, S5, S6, S7, S8 are used for training and subjects S9, S11 are used for testing. MPI-INF-3DHP is another popular 3D HPE dataset. This dataset contains 1.3 million frames collected in indoor and outdoor scenes. It is smaller than Human3.6M but more challenging due to its diverse scenes, viewpoints, and motions. H3WB extends Human3.6M by introducing comprehensive whole-body annotations, including 133 detailed keypoints that capture precise information about hands, face, and feet.
Evaluation Metrics. For Human3.6M, we use the most commonly used mean per joint position error (MPJPE) as the evaluation metric, which measures the average Euclidean distance between estimated and ground truth 3D joint coordinates in millimeters. For MPI-INF-3DHP, we follow previous works [13], [14], [15] to report metrics of MPJPE, percentage of correct keypoint (PCK) with the threshold of 150mm, and area under curve (AUC). For H3WB, we report the MPJPE metric.

4.2 Implementation Details

The network is implemented using the PyTorch framework on one consumer-level NVIDIA RTX 3090 GPU with 24G memory. Our method builds upon MHFormer [14], MixSTE [15], MotionBERT [16], and MotionAGFormer [27] for their largest frame number (*i.e.*, $F = 351, 243, 243, 243$) models. We adopt the same optimal hyperparameters and training strategies used in [14], [15], [16], [27], as detailed in Table 1. We also use the same loss functions for training, such as MPJPE loss for MHFormer, and weighted MPJPE loss, temporal consistency loss (TCLoss), and mean per-joint velocity error (MPJVE) for MixSTE. By default, we set $\{F = 351, r = [175, 117], b = [0, 1]\}$ for MHFormer, $\{F = 243, r = [121, 81], b = [0, 3]\}$ for MixSTE, $\{F = 243, r = [121, 81], b = [0, 1]\}$ for MotionBERT, and $\{F = 243, r = [121, 81], b = [0, 7]\}$ for MotionAGFormer. Note that MHFormer is designed for *seq2frame* pipeline, so we only implement the TPM on it. MixSTE, MotionBERT, and MotionAGFormer are designed for *seq2seq* pipeline and can be implemented on both *seq2frame* (with TPM) and *seq2seq* (with H_2OT) pipelines.

TABLE 1: Implementation details of our method on MHFormer [14], MixSTE [15], MotionBERT [16], and MotionAGFormer [27]. (L) - number of transformer blocks, (C) - dimension, (LR) - initial learning rate, (Flip) - horizontal flip augmentation, (CPN) - Cascaded Pyramid Network [63], (SH) - Stack Hourglass [64].

Config	MHFormer [14]	MixSTE [15]	MotionBERT [16]	MotinAGFormer [27]
L	3	8	5	16
C	512	512	256	128
Training Epoch	20	160	120	120
Batch Size	210	4	4	4
LR	1×10^{-3}	4×10^{-5}	5×10^{-4}	5×10^{-4}
Optimizer	Amsgrad	Adam	Adam	Adam
Augmentation	Flip	Flip	Flip	Flip
2D Detector	CPN	CPN	SH	SH

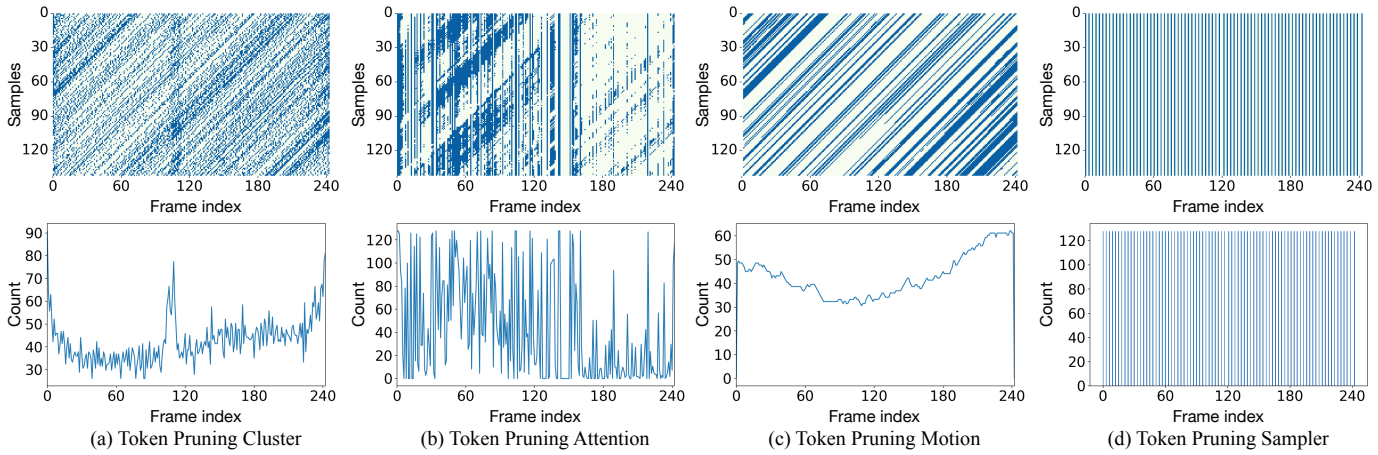
4.3 Ablation Study

To validate the effectiveness of our method, we conduct extensive ablation studies on Human3.6M dataset.
Token Pruning and Recovering Designs. Our method is a general-purpose *pruning-and-recovering* framework that can be directly inserted into various token pruning and recovering techniques. In Table 2, we compare different combinations of TPC, TPA, TPMo, and TPS for token pruning, and TRA and TRI for token recovering. The hyper-parameters are all set by $F = 243$, $r = 81$, and $b = 3$ for a fair comparison. Since efficient interpolation requires the sequence to be ordered, TRI can only be combined with TPS. To measure the quality of selected tokens, we define a frame noise metric that calculates the MPJPE of the 2D poses of input frames corresponding to the selected indexes. As the table shows, the frame noises across these methods are similar (around 6.6mm) except for the motion pruning (7.0mm). This is because selecting tokens with top-k large motion introduces noisy frames that differ significantly from clean frames, which can adversely affect performance. Moreover, the studies show that the combination of TPS and TRI requires the fewest parameters (33.78 M) and FLOPs (161.73 G) but achieves the highest FPS per frame (66). It also has the lowest relative ratio of time spent on token pruning and recovery (0.00) and achieves competitive performance (41.1 mm). As a comparison, TPC consumes more inference time (56 FPS/Frame and 0.33 Ratio) and TRA requires additional parameters and computational costs (35.00M Param and 167.52G FLOPs), while TPS and TRI are clearly more efficient. We therefore adopt TPS and TRI to create more powerful and faster VPT models.

Furthermore, we statistically visualize selected tokens of four token pruning strategies: TPC, TPA, TPMo, and TPS. For better observation, we take samples of consecutive video sequences as input with a temporal interval of 1 between neighboring samples. The frame indexes and the frequency count of frame indexes of the selected tokens are shown in Figure 7 (top) and Figure 7 (bottom). TPS and TPMo are static pruning methods because the former selects tokens at a fixed frame interval (equidistance in the top of Figure 7 (d)), while the latter selects tokens with top-k large motions that move with the input sequence (oblique triangle in the top of Figure 7 (c)). Instead, TPA and TPC are dynamic methods that consider the significance of input tokens. The bottom of Figure 7 (b) shows that TPA tends to select tokens

TABLE 2: Ablation study on the different combinations of token pruning and recovering strategies under *seq2seq* pipeline. "FPS/Frame" denotes frame per second per each output frame. "FN" denotes the frame noise that calculates the MPJPE of selected frames. "Ratio" denotes the ratio of time spent on token pruning and recovering with the overall inference time.

		Token Pruning				Token Recovering						
Method	FN	TPC	TPA	TPMo	TPS	TRA	TRI	Param (M)	FLOPs (G)	FPS/Frame	Ratio	MPJPE ↓
MixSTE [15]	6.61							33.78	277.25	41	-	40.9
Ours	6.63	✓				✓		35.00	167.52	56	0.33	41.0
Ours	6.56		✓			✓		35.00	167.52	63	0.04	42.1
Ours	7.00			✓		✓		35.00	167.52	62	0.03	42.8
Ours	6.61				✓	✓		35.00	167.52	64	0.03	41.4
Ours	6.61				✓		✓	**33.78**	**161.73**	**66**	**0.00**	41.1



(a) Token Pruning Cluster (b) Token Pruning Attention (c) Token Pruning Motion (d) Token Pruning Sampler

Fig. 7: Statistics visualization of selected tokens for different token pruning strategies. **Top**: Frame indexes of selected tokens for some samples (140 samples) of consecutive video sequences (243 frames). Blue points are selected tokens and white points are pruned tokens. **Bottom**: Frequency count of frame indexes of selected tokens for these samples.

in the left half of a sequence, indicating that the selected tokens tend to be similar to each other [26]. TPC primarily selects tokens at the beginning, center, and end of a sequence (the bottom of Figure 7 (a)). This is reasonable since these three parts can represent the rough motion of an entire sequence, which contributes a lot to accurate estimation. These findings highlight that these different token pruning strategies have different characteristics, and all of them can select representative pose tokens to eliminate the redundancy of video frames.

Inference Pipelines. In Table 3, we compare the efficiency and accuracy of different inference pipelines (mentioned in Sec 3.3). We conduct experiments on MixSTE [15] and MotionBERT [16] because both are designed for *seq2seq* pipeline and can be evaluated on both *seq2frame* and *seq2seq* pipelines. As shown in the table, the *seq2frame* can achieve better estimation accuracy by taking advantage of past and future information but lower efficiency due to repeated computation, *i.e.*, 40.7mm vs. 40.9mm and 41 FPS vs. 9982 FPS for MixSTE (about 243× lower). Moreover, our method can reduce the computational costs and improve the inference speed of these two pipelines, while maintaining or obtaining better performance.

For the *seq2seq*, our method can reduce the FLOPs of MixSTE and MotionBERT by 57.4% and 63.4% and improve the FPS by 87.8% and 47.6%, respectively. Meanwhile, our method can improve performance by 0.5mm for MixSTE and only decrease it by 0.1mm (0.25%) for MotionBERT. For the *seq2frame*, our TPM w. MixSTE can reduce the FLOPs by 57.4% and improve the FPS by 68.3%, while bringing 0.3mm improvement. Additionally, our TPM w. MotionBERT can reduce 63.4% FLOPs and improve 46.2% FPS, while the estimation errors are reduced from 39.5mm to 39.3mm. Note that our method with TPM outperforms the one utilizing H_2OT, largely because TRM in H_2OT is a reverse operation that uses inadequate information to recover the full-length tokens.

We also compare our method with our conference version [21]. As shown in Table 3, compared with [21], our method can further reduce the FLOPs, GPU memory cost, and training time, while improving the FPS and maintaining the performance. For example, our H_2OT w. MixSTE outperforms HoT w. MixSTE [21] by reducing FLOPs by 29.4% (118.24G vs. 167.52G), GPU memory cost by 26.3% (5.6G vs. 7.6G), and training time by 25.0% (8.4 min/epoch vs. 11.2 min/epoch), while improving the FPS by 31.1% (18745 vs. 14297) and MPJPE by 0.5mm (from 41.0mm to 40.5mm). This demonstrates the effectiveness of our H_2OT in reducing the computational costs, enabling stronger and faster VPT models on resource-limited devices.

In the following ablations, we conduct experiments on the *seq2seq* pipeline since it is more efficient and widely used in practice, and we choose MixSTE [15] as the baseline since it is the first *seq2seq* transformer-based architecture, and MotionBERT [16] and MotionAGFormer [27] are its follow-ups.

TABLE 3: Comparison of efficiency and accuracy between *seq2seq* (∗) and *seq2frame* (†) inference pipelines. FPS, GPU memory cost (G), and training time (min/epoch) are computed on a single GeForce RTX 3090 GPU.

Method	Param (M)	FLOPs (G)	FPS	GPU Memory	Training Time	MPJPE ↓
MixSTE [15] (∗)	33.78	277.25	9982	11.4	17.9	40.9
HoT w. MixSTE [21] (∗)	35.00	167.52 (↓ **39.6%**)	14297 (↑ **43.2%**)	7.6 (↓ **33.3%**)	11.2 (↓ **37.4%**)	41.0
H_2OT w. MixSTE (**Ours**) (∗)	33.78	118.23 (↓ **57.4%**)	18745 (↑ **87.8%**)	5.6 (↓ **52.3%**)	8.4 (↓ **53.1%**)	40.5
MixSTE [15] (†)	33.78	277.25	41	11.4	17.9	40.7
TPC w. MixSTE [21] (w/o TRM) (†)	33.78	161.73 (↓ **41.7%**)	61 (↑ **48.8%**)	7.3 (↓ **36.9%**)	10.7 (↓ **40.2%**)	40.4
TPM w. MixSTE (**Ours**, w/o TRM) (†)	33.78	118.23 (↓ **57.4%**)	69 (↑ **68.3%**)	5.6 (↓ **50.9%**)	8.7 (↓ **51.4%**)	40.4
MotionBERT [16] (∗)	16.00	131.09	12719	10.7	18.5	39.8
HoT w. MotionBERT [21] (∗)	16.35	63.21 (↓ **51.8%**)	15289 (↑ **20.2%**)	6.1 (↓ **43.0%**)	11.5 (↓ **37.8%**)	39.8
H_2OT w. MotionBERT (**Ours**) (∗)	16.00	47.98 (↓ **63.4%**)	18779 (↑ **47.6%**)	4.4 (↓ **58.9%**)	9.0 (↓ **51.4%**)	39.9
MotionBERT [16] (†)	16.00	131.09	52	10.7	18.5	39.5
TPC w. MotionBERT [21] (w/o TRM) (†)	16.00	61.04 (↓ **53.4%**)	68 (↑ **30.8%**)	5.7 (↓ **46.7%**)	10.9 (↓ **41.1%**)	39.2
TPM w. MotionBERT (**Ours**, w/o TRM) (†)	16.00	47.98 (↓ **63.4%**)	76 (↑ **46.2%**)	4.4 (↓ **58.9%**)	8.9 (↓ **51.9%**)	39.3

TABLE 4: Ablation studies on the number of representative tokens (r) and the block index of representative tokens (b) under *seq2seq* pipeline. Frame per second (FPS) was computed on a single GeForce RTX 3090 GPU.

Method	r	b	Param (M)	FLOPs (G)	FPS/Frame	GPU Memory	Training Time	MPJPE ↓
MixSTE [15]	243	-	33.78	277.25	41	11.4	17.9	40.9
HoT w. MixSTE [21]	81	3	35.00	167.52 (↓ **39.6%**)	56 (↑ **36.6%**)	7.6 (↓ **33.3%**)	11.2 (↓ **37.4%**)	41.0
H_2OT w. MixSTE	[81]	[3]	33.78	161.73 (↓ **41.7%**)	66 (↑ **61.0%**)	7.3 (↓ **36.0%**)	10.7 (↓ **40.2%**)	41.0
H_2OT w. MixSTE	[121, 81]	[0, 3]	33.78	118.23 (↓ **57.4%**)	82 (↑ **100.0%**)	5.6 (↓ **50.9%**)	8.0 (↓ **55.3%**)	**40.5**
H_2OT w. MixSTE	[121, 81, 49]	[0, 3, 5]	33.78	104.54 (↓ **62.3%**)	85 (↑ **107.3%**)	5.1 (↓ **55.3%**)	7.9 (↓ **55.9%**)	40.7
H_2OT w. MixSTE	[189, 121, 81, 49]	[0, 3, 5, 7]	33.78	149.32 (↓ **46.1%**)	74 (↑ **80.5%**)	6.7 (↓ **41.2%**)	10.1 (↓ **43.6%**)	40.8
H_2OT w. MixSTE	[189, 81]	[0, 3]	33.78	142.48 (↓ **48.6%**)	75 (↑ **82.9%**)	6.4 (↓ **43.9%**)	10.2 (↓ **43.0%**)	40.7
H_2OT w. MixSTE	[121, 81]	[0, 3]	33.78	118.23 (↓ **57.4%**)	82 (↑ **100.0%**)	5.6 (↓ **50.9%**)	8.0 (↓ **55.3%**)	**40.5**
H_2OT w. MixSTE	[121, 49]	[0, 3]	33.78	95.41 (↓ **65.6%**)	82 (↑ **100.0%**)	4.8 (↓ **57.9%**)	6.8 (↓ **62.0%**)	41.2
H_2OT w. MixSTE	[121, 27]	[0, 3]	33.78	79.73 (↓ **71.2%**)	83 (↑ **102.4%**)	4.4 (↓ **61.4%**)	6.5 (↓ **63.7%**)	41.3
H_2OT w. MixSTE	[121, 81]	[0, 3]	33.78	118.23 (↓ **57.4%**)	82 (↑ **100.0%**)	5.6 (↓ **50.9%**)	8.0 (↓ **55.3%**)	**40.5**
H_2OT w. MixSTE	[121, 81]	[0, 5]	33.78	129.64 (↓ **53.2%**)	82 (↑ **100.0%**)	5.7 (↓ **50.0%**)	8.7 (↓ **51.4%**)	40.8
H_2OT w. MixSTE	[121, 81]	[0, 7]	33.78	141.05 (↓ **49.1%**)	80 (↑ **95.1%**)	6.4 (↓ **43.9%**)	9.5 (↓ **46.9%**)	40.6
H_2OT w. MixSTE	[121, 81]	[3, 5]	33.78	173.14 (↓ **37.6%**)	64 (↑ **56.1%**)	7.8 (↓ **31.6%**)	11.3 (↓ **36.9%**)	40.9
H_2OT w. MixSTE	[121, 81]	[3, 7]	33.78	184.55 (↓ **33.4%**)	62 (↑ **51.2%**)	8.2 (↓ **28.1%**)	12.0 (↓ **33.0%**)	40.8
H_2OT w. MixSTE	[189, 121, 81]	[0, 3, 5]	33.78	153.89 (↓ **44.5%**)	72 (↑ **75.6%**)	6.9 (↓ **39.5%**)	10.8 (↓ **39.7%**)	40.9
H_2OT w. MixSTE	[121, 81, 49]	[0, 3, 5]	33.78	104.54 (↓ **62.3%**)	81 (↑ **97.6%**)	5.1 (↓ **55.3%**)	7.2 (↓ **59.8%**)	40.7
H_2OT w. MixSTE	[121, 81, 27]	[0, 3, 5]	33.78	95.13 (↓ **65.7%**)	84 (↑ **104.9%**)	4.9 (↓ **57.0%**)	6.7 (↓ **62.6%**)	41.6
H_2OT w. MixSTE	[121, 81, 49]	[0, 3, 5]	33.78	104.54 (↓ **62.3%**)	84 (↑ **104.9%**)	5.1 (↓ **55.3%**)	7.2 (↓ **59.8%**)	40.7
H_2OT w. MixSTE	[121, 81, 49]	[0, 3, 7]	33.78	113.67 (↓ **59.0%**)	83 (↑ **102.4%**)	5.4 (↓ **52.6%**)	7.8 (↓ **56.4%**)	40.9
H_2OT w. MixSTE	[121, 81, 49]	[0, 5, 7]	33.78	125.08 (↓ **54.9%**)	71 (↑ **73.2%**)	5.8 (↓ **49.1%**)	9.0 (↓ **49.7%**)	40.8
H_2OT w. MixSTE	[121, 81, 49]	[3, 5, 7]	33.78	168.57 (↓ **39.2%**)	66 (↑ **61.0%**)	7.6 (↓ **33.3%**)	11.1 (↓ **38.0%**)	40.7
H_2OT w. MixSTE	[189, 121, 81, 49]	[0, 3, 5, 7]	33.78	149.32 (↓ **46.1%**)	74 (↑ **80.5%**)	6.7 (↓ **41.2%**)	10.2 (↓ **43.0%**)	40.8
H_2OT w. MixSTE	[121, 81, 49, 27]	[0, 3, 5, 7]	33.78	101.40 (↓ **63.4%**)	84 (↑ **104.9%**)	5.1 (↓ **55.3%**)	7.7 (↓ **57.0%**)	41.3

Hyperparameters (r and b). The number of representative tokens (r) and the block index of representative tokens (b) can be flexibly adjusted, thereby achieving the trade-off between computational costs and accuracy. Table 4 shows ablation studies on r and b in a hierarchical design under *seq2seq* pipeline. We can see that decreasing b and r can reduce the FLOPs, and the best accuracy is achieved while $r = [121, 81]$ and $b = [0, 3]$. This indicates that appropriate b and f can bring a good trade-off between retaining important information and reducing redundant information for both the pruning and recovering stages. Therefore, the optimal hyper-parameters for our H_2OT w. MixSTE are $r = [121, 81]$ and $b = [0, 3]$. We can also flexibly adjust r and b to achieve a speed-accuracy trade-off according to specific requirements.

We also compare our method with MixSTE [15] using the same number of representative tokens and approximately the same number of FLOPs. To achieve this, we set the input frame number of the original MixSTE to $F = 81, 105, 121$. The results in Table 5 show that our method obtains better results, further demonstrating the importance of large receptive fields and the effectiveness of our method. Additionally, we compare our method with our conference version HoT [21] using similar hyperparameters and approximately the same number of FLOPs. Table 5 shows that our H_2OT w. MixSTE using fewer parameters and FLOPs achieves better performance than HoT w. MixSTE [21]. This further demonstrates the effectiveness of our hierarchical design and efficient TPS and TRI.

Number of Recovered Tokens. In Table 6, we conduct the ablation study on the number of recovered tokens (f) under *seq2frame* pipeline. Since f differs from the input frames, we evaluate the performance under the *seq2frame* pipeline, which selects the 3D pose of the center frame as the final estimation. The results show that the performance remains almost unchanged when adjusting f. Therefore, we choose $f = 243$, which is more efficient and can be evaluated under the *seq2seq* pipeline.

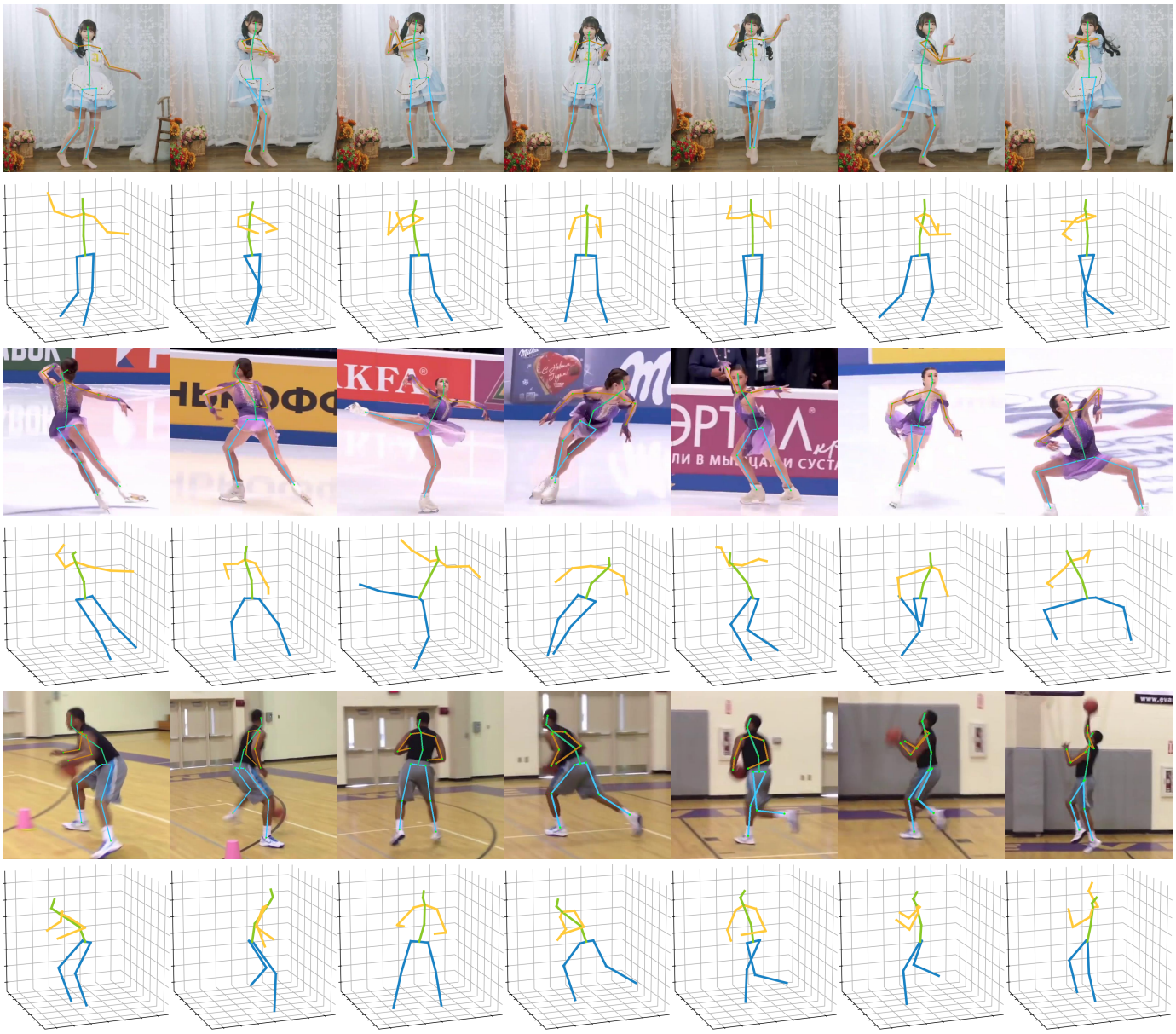


Fig. 8: Qualitative results of our method on challenging in-the-wild videos.

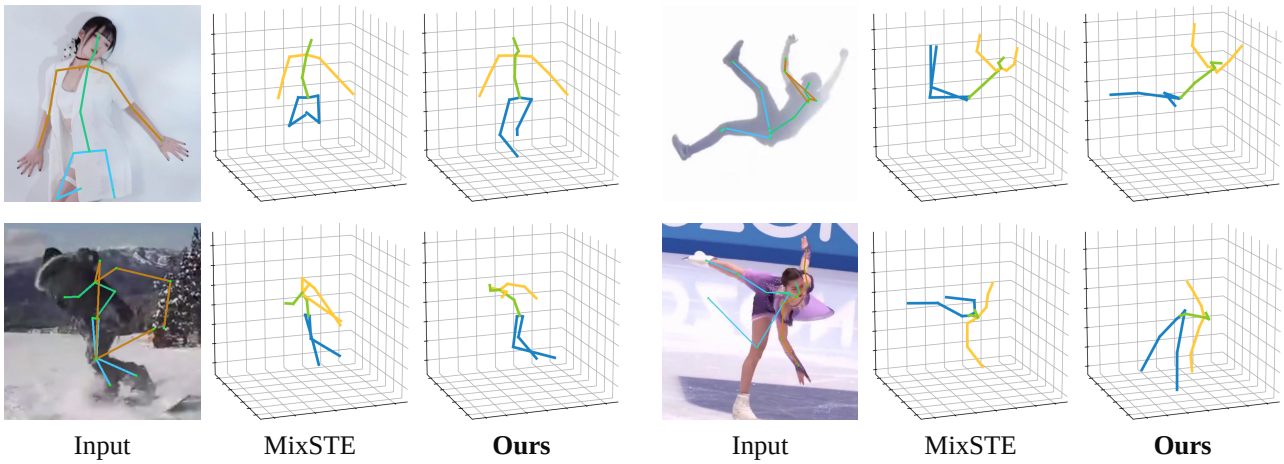


Fig. 9: Failure cases in challenging scenarios.

Fig. 10: Visualization of input images, estimated 3D poses (cyan), and ground truth 3D poses (black) from three video sequences. The 2D poses of selected frames are colored in red, and the 2D poses of pruned frames are colored in gray. The 3D poses of selected frames are highlighted with red rectangular boxes.

4.4 Comparison with state-of-the-art methods

Current SOTA performance on Human3.6M is achieved by transformer-based architectures. We compare our method with them by adding it to four very recent VPTs: MHFormer [14], MixSTE [15], MotionBERT [16], and MotionAGFormer [27]. These four models significantly outperform previous works at the cost of high computational complexity, thus we choose them as baselines to evaluate our method. The comparisons are shown in Table 7. For example, compared with MixSTE [15], our H_2OT w. MixSTE obtains better performance with 0.5mm improvements while reducing computational costs by 57.4% in FLOPs. We also compare our method with diffusion-based VPTs (D3DP [67] and DiffPose [68]) in Table 8. It can be found that our H_2OT with D3DP reduces FLOPs by 57.4% and increases FPS

TABLE 5: Comparison with MixSTE [15] and HoT [21].

Method	F	b	r	Param (M)	FLOPs (G)	MPJPE ↓
MixSTE [15]	81	81	-	33.70	92.42	42.7
MixSTE [15]	121	121	-	33.72	138.05	42.0
MixSTE [15]	105	105	-	33.71	119.80	42.3
HoT w. MixSTE	243	81	0	35.00	109.76	41.4
HoT w. MixSTE	243	81	3	35.00	167.52	41.0
HoT w. MixSTE	243	121	0	35.02	153.26	41.3
HoT w. MixSTE	243	121	3	35.02	196.75	41.1
HoT w. MixSTE	243	81	1	35.00	121.31	41.2
HoT w. MixSTE	243	89	0	35.00	118.46	41.7
HoT w. MixSTE	243	15	3	34.97	119.28	42.9
H_2OT w. MixSTE	243	[121, 81]	[0, 3]	33.78	118.23	**40.5**

TABLE 6: Ablation study on the number of recovered tokens (f) under *seq2frame* pipeline.

Method	Param (M)	FLOPs (G)	MPJPE ↓
MixSTE [15]	33.78	277.25	40.9
H_2OT w. MixSTE ($f = 9$)	33.78	118.23	41.2
H_2OT w. MixSTE ($f = 27$)	33.78	118.23	40.7
H_2OT w. MixSTE ($f = 81$)	33.78	118.23	40.7
H_2OT w. MixSTE ($f = 243$)	33.78	118.23	**40.4**

by 40.7%, with only a 0.1mm decrease in performance. Additionally, H_2OT with DiffPose achieves a 57.4% reduction in FLOPs and a 44.5% improvement in FPS with a 0.2mm decrease in performance. Moreover, we conduct experiments on H3WB in Table 9. It can be observed that our H_2OT w. MixSTE reduces FLOPs by 57.4% and improves FPS by 138.9%, with only a 0.2mm drop in accuracy. These experiments demonstrate the effectiveness and efficiency of our method, while revealing that existing VPTs incur redundant computational costs that contribute little to the estimation accuracy and even decrease the accuracy.

Furthermore, we conduct experiments on Human3.6M at varying FPS (50Hz, 25Hz, 16.7Hz, and 12.5Hz), corresponding to downsampling rates of 1, 2, 3, and 4, respectively. The results are shown in Table 10. We can see a performance improvement at lower FPS, which can be attributed to reduced temporal redundancy and an expanded temporal receptive field [18]. However, we observe that our H_2OT w. MixSTE shows slightly inferior performance compared to the original MixSTE at low FPS. This can be attributed to the fact that in low-FPS scenarios, temporal information is sparse, leaving less redundant data that can be removed through token pruning. Despite this, our method maintains its core advantage: a remarkable 57.4% reduction in computational cost while preserving competitive performance. This makes our method particularly valuable for real-world applications where computational efficiency is paramount.

We further evaluate our method on MPI-INF-3DHP dataset in Table 11. For a fair comparison, following [14], [15], [27], we implement our method on them with the same number of input frames: MHFormer with $\{F = 9, r = [5, 3], b = [0, 1]\}$, MixSTE with $\{F = 27, r = [13, 9], b = [0, 3]\}$, and MotionAGFormer with $\{F = 81, r = [41, 27], b = [0, 7]\}$. It can be found that our method (TPM w. MHFormer, H_2OT w. MixSTE, and H_2OT w. MotionAGFormer) achieves competitive performance compared with our original version

TABLE 7: Comparison of parameters (M), FLOPs (G), and MPJPE with SOTA VPTs on Human3.6M. Here, F denotes the number of input frames. * indicates our re-implementation.

Method	F	Param	FLOPs	MPJPE ↓
PoseFormer (ICCV'21) [13]	81	9.60	1.63	44.3
Strided (TMM'22) [19]	351	4.35	1.60	43.7
P-STMO (ECCV'22) [18]	243	7.01	1.74	42.8
Einfalt *et al.* (WACV'23) [20]	351	10.36	1.00	44.2
HDFormer (IJCAI'23) [10]	96	3.70	1.18	42.6
STCFormer (CVPR'23) [65]	243	18.93	156.22	40.5
KTPFormer (CVPR'24) [66]	243	33.65	277.27	40.1
MHFormer (CVPR'22) [14]	351	31.52	14.15	43.0
TPC w. MHFormer (CVPR'24) [21]	351	31.52	8.22 (↓ **41.9%**)	43.0
TPM w. MHFormer (**Ours**)	351	31.52	6.74 (↓ **52.4%**)	43.2
MixSTE (CVPR'22) [15]	243	33.78	277.25	40.9
HoT w. MixSTE (CVPR'24) [21]	243	35.00	167.52 (↓ **39.6%**)	41.0
H_2OT w. MixSTE (**Ours**)	243	33.78	118.23 (↓ **57.4%**)	40.5
MotionBERT (ICCV'23) [16]	243	16.00	131.09	39.2
MotionBERT (ICCV'23) [16]*	243	16.00	131.09	39.8
HoT w. MotionBERT (CVPR'24) [21]	243	16.35	63.21 (↓ **51.8%**)	39.8
H_2OT w. MotionBERT (**Ours**)	243	16.00	47.99 (↓ **63.4%**)	39.9
MotionAGFormer (WACV'24) [27]	243	11.72	95.99	38.4
HoT w. MotionAGFormer (CVPR'24) [21]	243	11.83	60.56 (↓ **36.9%**)	38.9
H_2OT w. MotionAGFormer (**Ours**)	243	11.72	38.87 (↓ **59.5%**)	38.5

TABLE 8: Comparison of efficiency and accuracy with diffusion-based methods on Human3.6M. Frame per second (FPS) was computed on a single GeForce RTX 3090 GPU.

Method	Param (M)	FLOPs (G)	FPS	MPJPE ↓
D3DP [67] ($H = 5, K = 5$)	34.84	277.26	820	39.7
H_2OT w. D3DP	34.84	118.25 (↓ **57.4%**)	1154 (↑ **40.7%**)	39.8
DiffPose [68]	33.78	277.25	6522	39.5
H_2OT w. DiffPose	33.78	118.23 (↓ **57.4%**)	9426 (↑ **44.5%**)	39.7

[21] and the original VPTs [14], [15], [27]. These results demonstrate the effectiveness of our method in both indoor and outdoor scenes, and our method can also work well with a small temporal receptive field.

4.5 Qualitative Results

Figure 8 shows the qualitative results on challenging in-the-wild videos. These results confirm the ability of our method to produce accurate 3D pose estimations. However, in challenging scenarios, there are some failure cases where our method cannot accurately estimate 3D human poses due to factors such as partial body visibility, rare poses, and significant errors in the 2D detector (Figure 9). We also provide visualizations of recovering 3D human poses in Figure 10, which illustrate that our method can predict realistic 3D human poses of the entire sequence.

5 Conclusion

This paper presents Hierarchical Hourglass Tokenizer (H_2OT), a hierarchical plug-and-play *pruning-and-recovering* framework for efficient transformer-based 3D human pose estimation from videos. Our method reveals that maintaining the full pose sequence is unnecessary, and using a few pose tokens of representative frames can achieve both high efficiency and estimation accuracy. Comprehensive experiments demonstrate that our method is compatible and general. It can be easily incorporated into common VPT models on both

TABLE 9: Comparison of efficiency and accuracy with other methods on H3WB. Frame per second (FPS) was computed on a single GeForce RTX 3090 GPU.

Method	Param (M)	FLOPs (G)	FPS	MPJPE ↓
MixSTE	33.84	2185.36	1417	54.5
H_2OT w. MixSTE	33.84	931.96 (↓ **57.4%**)	3385 (↑ **138.9%**)	54.7

TABLE 10: Results on Human3.6M at different FPS settings.

FPS	50.0	25.0	16.7	12.5
MixSTE	40.9	39.6	39.3	39.2
H_2OT w. MixSTE	40.5	40.2	40.0	40.8

seq2seq and *seq2frame* pipelines while effectively accommodating various token pruning and recovery strategies, thereby highlighting its potential for using future ones. We hope H_2OT can enable the creation of stronger and faster VPTs.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (No. 62373009), the Natural Science Foundation of Shenzhen (No. JCYJ20230807120801002), the EU Horizon project ELIAS (No. 101120237), and by the MUR PNRR project FAIR (PE00000013) funded by the NextGenerationEU.

REFERENCES

[1] M. Liu, H. Liu, and C. Chen, "Enhanced skeleton visualization for view invariant human action recognition," *PR*, vol. 68, pp. 346–362, 2017.
[2] Y. Zhou, X. Yan, Z.-Q. Cheng, Y. Yan, Q. Dai, and X.-S. Hua, "BlockGCN: Redefine topology awareness for skeleton-based action recognition," in *CVPR*, 2024, pp. 2049–2058.
[3] D. C. Luvizon, D. Picard, and H. Tabia, "Multi-task deep learning for real-time 3D human pose estimation and action recognition," *IEEE TPAMI*, vol. 43, no. 8, pp. 2752–2764, 2020.
[4] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, and T. Brox, "3D human pose estimation in rgbd images for robotic task learning," in *ICRA*, 2018, pp. 1986–1992.
[5] M. Garcia-Salguero, J. Gonzalez-Jimenez, and F.-A. Moreno, "Human 3D pose estimation with a tilting camera for social mobile robot interaction," *Sensors*, vol. 19, no. 22, p. 4943, 2019.
[6] H. Li, M. Li, Z.-Q. Cheng, Y. Dong, Y. Zhou, J.-Y. He, Q. Dai, T. Mitamura, and A. Hauptmann, "Human-aware vision-and-language navigation: Bridging simulation to reality with dynamic human interactions," in *NeurIPS*, vol. 37, 2024, pp. 119 411–119 442.
[7] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt, "VNect: Real-time 3D human pose estimation with a single rgb camera," *ACM TOG*, vol. 36, no. 4, pp. 1–14, 2017.
[8] S. Tu, Q. Dai, Z.-Q. Cheng, H. Hu, X. Han, Z. Wu, and Y.-G. Jiang, "MotionEditor: Editing video motion via content-aware diffusion," in *CVPR*, 2024, pp. 7882–7891.
[9] Y. You, H. Liu, T. Wang, W. Li, R. Ding, and X. Li, "Co-evolution of pose and mesh for 3D human body estimation from video," in *ICCV*, 2023, pp. 14 963–14 973.
[10] H. Chen, J.-Y. He, W. Xiang, Z.-Q. Cheng, W. Liu, H. Liu, B. Luo, Y. Geng, and X. Xie, "HDFormer: High-order directed transformer for 3D human pose estimation," in *IJCAI*, 2023, pp. 581–589.
[11] H. Liu, J.-Y. He, Z.-Q. Cheng, W. Xiang, Q. Yang, W. Chai, G. Wang, X. Bao, B. Luo, Y. Geng *et al.*, "PoSynDA: Multi-hypothesis pose synthesis domain adaptation for robust 3D human pose estimation," in *ACM MM*, 2023, pp. 5542–5551.
[12] J. Zhang, M. Liu, H. Liu, G. Wang, and W. Li, "APP: Adaptive pose pooling for 3D human pose estimation from videos," in *ACM MM*, 2024, pp. 1672–1681.

TABLE 11: Quantitative comparison with SOTA VPTs on MPI-INF-3DHP.

Method	F	PCK ↑	AUC ↑	MPJPE ↓
PoseFormer (ICCV'21) [13]	9	88.6	56.4	77.1
P-STMO (ECCV'22) [18]	81	97.9	75.8	32.2
Einfalt *et al.* (WACV'23) [20]	81	95.4	67.6	46.9
HDFormer (IJCAI'23) [10]	81	98.7	72.9	37.2
STCFormer (CVPR'23) [65]	81	98.7	83.9	23.1
KTPFormer (CVPR'24) [66]	81	98.9	85.9	16.7
MHFormer (CVPR'22) [14]	9	93.8	63.3	58.0
TPC w. MHFormer (CVPR'24) [21]	9	94.0	63.3	58.4
TPM w. MHFormer (**Ours**)	9	94.1	63.6	57.8
MixSTE (CVPR'22) [15]	27	94.4	66.5	54.9
HoT w. MixSTE (CVPR'24) [21]	27	94.8	66.5	53.2
H_2OT w. MixSTE (**Ours**)	27	95.2	66.4	53.0
MotionAGFormer (WACV'24) [27]	81	98.3	84.2	18.2
HoT w. MotionAGFormer (CVPR'24) [21]	81	99.0	84.5	18.8
H_2OT w. MotionAGFormer (**Ours**)	81	99.1	85.2	18.0

[13] C. Zheng, S. Zhu, M. Mendieta, T. Yang, C. Chen, and Z. Ding, "3D human pose estimation with spatial and temporal transformers," in *ICCV*, 2021, pp. 11 656–11 665.
[14] W. Li, H. Liu, H. Tang, P. Wang, and L. Van Gool, "MHFormer: Multi-hypothesis transformer for 3D human pose estimation," in *CVPR*, 2022, pp. 13 147–13 156.
[15] J. Zhang, Z. Tu, J. Yang, Y. Chen, and J. Yuan, "MixSTE: Seq2seq mixed spatio-temporal encoder for 3D human pose estimation in video," in *CVPR*, 2022, pp. 13 232–13 242.
[16] W. Zhu, X. Ma, Z. Liu, L. Liu, W. Wu, and Y. Wang, "MotionBERT: A unified perspective on learning human motion representations," in *ICCV*, 2023, pp. 15 085–15 099.
[17] J. Liu, M. Liu, H. Liu, and W. Li, "TCPFormer: Learning temporal correlation with implicit pose proxy for 3D human pose estimation," in *AAAI*, 2025.
[18] W. Shan, Z. Liu, X. Zhang, S. Wang, S. Ma, and W. Gao, "P-STMO: Pre-trained spatial temporal many-to-one model for 3D human pose estimation," in *ECCV*, 2022.
[19] W. Li, H. Liu, R. Ding, M. Liu, P. Wang, and W. Yang, "Exploiting temporal contexts with strided transformer for 3D human pose estimation," *IEEE TMM*, vol. 25, pp. 1282–1293, 2022.
[20] M. Einfalt, K. Ludwig, and R. Lienhart, "Uplift and upsample: Efficient 3D human pose estimation with uplifting transformers," in *WACV*, 2023, pp. 2903–2913.
[21] W. Li, M. Liu, H. Liu, P. Wang, J. Cai, and N. Sebe, "Hourglass tokenizer for efficient transformer-based 3D human pose estimation," in *CVPR*, 2024, pp. 604–613.
[22] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, "3D human pose estimation in video with temporal convolutions and semi-supervised training," in *CVPR*, 2019, pp. 7753–7762.
[23] R. Liu, J. Shen, H. Wang, C. Chen, S.-c. Cheung, and V. Asari, "Attention mechanism exploits temporal contexts: Real-time 3D human pose reconstruction," in *CVPR*, 2020, pp. 5064–5073.
[24] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments," *IEEE TPAMI*, vol. 36, no. 7, pp. 1325–1339, 2013.
[25] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh, "DynamicViT: Efficient vision transformers with dynamic token sparsification," in *NeurIPS*, vol. 34, 2021, pp. 13 937–13 949.
[26] Z. Wang, H. Luo, P. Wang, F. Ding, F. Wang, and H. Li, "VTC-LFC: Vision transformer compression with low-frequency components," in *NeurIPS*, vol. 35, 2022, pp. 13 974–13 988.
[27] S. Mehraban, V. Adeli, and B. Taati, "MotionAGFormer: Enhancing 3D human pose estimation with a transformer-gcnformer network," in *WACV*, 2024, pp. 6920–6930.
[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008.
[29] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[30] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021, pp. 10 012–10 022.
[31] Z.-Q. Cheng, Q. Dai, S. Li, T. Mitamura, and A. Hauptmann, "GSRFormer: Grounded situation recognition transformer with alternate semantic attention refinement," in *ACM MM*, 2022, pp. 3272–3281.
[32] S. Tu, Q. Dai, Z. Wu, Z.-Q. Cheng, H. Hu, and Y.-G. Jiang, "Implicit temporal modeling with learnable alignment for video recognition," in *ICCV*, 2023, pp. 19 936–19 947.
[33] H. Shuai, L. Wu, and Q. Liu, "Adaptive multi-view and temporal fusing transformer for 3D human pose estimation," *IEEE TPAMI*, vol. 45, no. 4, pp. 4122–4135, 2022.
[34] W. Zhang, M. Liu, H. Liu, and W. Li, "SVTformer: Spatial-view-temporal transformer for multi-view 3d human pose estimation," in *AAAI*, vol. 39, no. 10, 2025, pp. 10 148–10 156.
[35] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A simple yet effective baseline for 3D human pose estimation," in *ICCV*, 2017, pp. 2640–2649.
[36] S. Choi, S. Choi, and C. Kim, "MobileHumanPose: Toward real-time 3D human pose estimation in mobile devices," in *CVPR*, 2021, pp. 2328–2338.
[37] A. Zeng, X. Ju, L. Yang, R. Gao, X. Zhu, B. Dai, and Q. Xu, "DeciWatch: A simple baseline for 10x efficient 2D and 3D pose estimation," in *ECCV*, 2022, pp. 607–624.
[38] Q. Zhao, C. Zheng, M. Liu, P. Wang, and C. Chen, "PoseFormerV2: Exploring frequency domain for efficient and robust 3D human pose estimation," in *CVPR*, 2023, pp. 8877–8886.
[39] Y. Sun, A. W. Dougherty, Z. Zhang, Y. K. Choi, and C. Wu, "MixSynthFormer: A transformer encoder-like structure with mixed synthetic self-attention for efficient human pose estimation," in *ICCV*, 2023, pp. 14 884–14 893.
[40] N. Kolotouros, G. Pavlakos, M. J. Black, and K. Daniilidis, "Learning to reconstruct 3D human pose and shape via model-fitting in the loop," in *ICCV*, 2019, pp. 2252–2261.
[41] H. Joo, N. Neverova, and A. Vedaldi, "Exemplar fine-tuning for 3D human model fitting towards in-the-wild 3D human pose estimation," in *3DV*, 2021, pp. 42–52.
[42] M. Kocabas, C.-H. P. Huang, O. Hilliges, and M. J. Black, "PARE: Part attention regressor for 3D human body estimation," in *ICCV*, 2021, pp. 11 127–11 137.
[43] Z. Dou, Q. Wu, C. Lin, Z. Cao, Q. Wu, W. Wan, T. Komura, and W. Wang, "TORE: Token reduction for efficient human mesh recovery with transformer," in *ICCV*, 2023, pp. 15 143–15 155.
[44] S. Long, Z. Zhao, J. Pi, S. Wang, and J. Wang, "Beyond attentive tokens: Incorporating token importance and diversity for efficient vision transformers," in *CVPR*, 2023, pp. 10 334–10 343.
[45] Z. Kong, P. Dong, X. Ma, X. Meng, W. Niu, M. Sun, X. Shen, G. Yuan, B. Ren, H. Tang *et al.*, "SPViT: Enabling faster vision transformers via latency-aware soft token pruning," in *ECCV*, 2022, pp. 620–640.
[46] S. Chang, P. Wang, M. Lin, F. Wang, D. J. Zhang, R. Jin, and M. Z. Shou, "Making vision transformers efficient from a token sparsification view," in *CVPR*, 2023, pp. 6195–6205.
[47] H. Ma, Z. Wang, Y. Chen, D. Kong, L. Chen, X. Liu, X. Yan, H. Tang, and X. Xie, "PPT: Token-pruned pose transformer for monocular and multi-view human pose estimation," in *ECCV*, 2022, pp. 424–442.
[48] W. Zeng, S. Jin, L. Xu, W. Liu, C. Qian, W. Ouyang, P. Luo, and X. Wang, "TCFormer: Visual recognition via token clustering transformer," *IEEE TPAMI*, 2024.
[49] H. Wang, J. Liu, J. Tang, G. Wu, B. Xu, Y. Chou, and Y. Wang, "GTPT: Group-based token pruning transformer for efficient human pose estimation," in *ECCV*, 2024.
[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.
[51] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.
[52] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on k-nearest neighbors and principal component analysis," *Knowledge-Based Systems*, vol. 99, pp. 135–145, 2016.
[53] H. Yin, A. Vahdat, J. M. Alvarez, A. Mallya, J. Kautz, and P. Molchanov, "A-ViT: Adaptive tokens for efficient vision transformer," in *CVPR*, 2022, pp. 10 809–10 818.
[54] D. Marin, J.-H. R. Chang, A. Ranjan, A. Prabhu, M. Rastegari, and O. Tuzel, "Token pooling in vision transformers for image classification," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 12–21.
[55] Y. Liang, C. Ge, Z. Tong, Y. Song, J. Wang, and P. Xie, "Not all patches are what you need: Expediting vision transformers via token reorganizations," in *ICLR*, 2022.
[56] M. Chen, W. Shao, P. Xu, M. Lin, K. Zhang, F. Chao, R. Ji, Y. Qiao, and P. Luo, "DiffRate: Differentiable compression rate for efficient vision transformers," in *ICCV*, 2023, pp. 17 164–17 174.
[57] W. Li, H. Liu, H. Tang, and P. Wang, "Multi-hypothesis representation learning for transformer-based 3D human pose estimation," *PR*, vol. 141, p. 109631, 2023.
[58] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt, "Monocular 3D human pose estimation in the wild using improved CNN supervision," in *3DV*, 2017, pp. 506–516.
[59] Y. Zhu, N. Samet, and D. Picard, "H3WB: Human3.6M 3D wholebody dataset and benchmark," in *ICCV*, 2023, pp. 20 166–20 177.
[60] W. Li, M. Liu, H. Liu, T. Guo, T. Wang, H. Tang, and N. Sebe, "GraphMLP: A graph MLP-like architecture for 3D human pose estimation," *PR*, vol. 158, p. 110925, 2025.
[61] H. Ci, X. Ma, C. Wang, and Y. Wang, "Locally connected network for monocular 3D human pose estimation," *IEEE TPAMI*, vol. 44, no. 3, pp. 1429–1442, 2020.
[62] Y. Xu, W. Wang, T. Liu, X. Liu, J. Xie, and S.-C. Zhu, "Monocular 3D pose estimation via pose grammar and data augmentation," *IEEE TPAMI*, vol. 44, no. 10, pp. 6327–6344, 2021.
[63] S. Li, L. Ke, K. Pratama, Y.-W. Tai, C.-K. Tang, and K.-T. Cheng, "Cascaded deep monocular 3D human pose estimation with evolutionary training data," in *CVPR*, 2020, pp. 6173–6183.
[64] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*, 2016, pp. 483–499.
[65] Z. Tang, Z. Qiu, Y. Hao, R. Hong, and T. Yao, "3D human pose estimation with spatio-temporal criss-cross attention," in *CVPR*, 2023, pp. 4790–4799.
[66] J. Peng, Y. Zhou, and P. Mok, "KTPFormer: Kinematics and trajectory prior knowledge-enhanced transformer for 3D human pose estimation," in *CVPR*, 2024, pp. 1123–1132.
[67] W. Shan, Z. Liu, X. Zhang, Z. Wang, K. Han, S. Wang, S. Ma, and W. Gao, "Diffusion-based 3D human pose estimation with multi-hypothesis aggregation," in *ICCV*, 2023, pp. 14 761–14 771.
[68] J. Gong, L. G. Foo, Z. Fan, Q. Ke, H. Rahmani, and J. Liu, "DiffPose: Toward more reliable 3D pose estimation," in *CVPR*, 2023, pp. 13 041–13 051.

Wenhao Li is currently a Postdoctoral Researcher at the School of Computer Science and Engineering, Nanyang Technological University, Singapore. He received the Ph. D. degree from the School of Computer Science, Peking University, China. His research interests lie in deep learning, machine learning, and their applications to computer vision.

Mengyuan Liu is an Assistant Professor at Peking University. He received a Ph.D. degree in 2017 under the supervision of Prof. H. Liu from the School of EE&CS, Peking University (PKU), China. His research interests include human action recognition, human motion prediction, and human motion generation using RGB, depth, and skeleton data. Related methods have been published in T-IP, T-CSVT, T- MM, PR, CVPR, ECCV, ACM MM, AAAI, and IJCAI. He has been invited to be a Technical Program Committee (TPC) member for ACPR, ACM MM, and AAAI.

Hong Liu received the Ph.D. degree in Mechanical Electronics and Automation in 1996, and serves as a Full Professor in the School of EE&CS, Peking University (PKU), China. Prof. Liu has been selected as Chinese Innovation Leading Talent supported by "National High-level Talents Special Support Pla" since 2013. His research interests include computer vision, pattern recognition and robot motion planning. He is also reviewers for many international journals such as Pattern Recognition, IEEE Trans. on Signal Processing, and IEEE Trans. on PAMI.

Pichao Wang received the Ph.D. degree in computer science from the University of Wollongong, Australia. He is currently a Senior Applied Scientist in Amazon AGI, USA. He has authored 120+ peer reviewed papers, including those in highly regarded journals and conferences such as IEEE TPAMI, IJCV, CVPR, ICCV, ECCV, ICLR, NeurIPS, AAAI, ACM MM, etc. He is the recipient of CVPR2022 Best Student Paper Award. He was named AI 2000 Most Influential Scholar during 2012-2022 by Miner, due to his contributions in the field of multimedia. He is also in the list of World's Top 1% Scientists named by Stanford University.

Shijian Lu is an Assistant Professor with the School of Computer Science and Engineering at Nanyang Technological University, Singapore. He received his PhD in electrical and computer engineering from the National University of Singapore. His major research interests include image and video analytics, visual intelligence, and machine learning.

Nicu Sebe is a Professor in the University of Trento, Italy, where he is leading the research in the areas of multimedia analysis and human behavior understanding. He was the General Co-Chair of the IEEE FG 2008 and ACM Multimedia 2013 and 2022. He was a program chair of ACM Multimedia 2011 and 2007, ECCV 2016, ICCV 2017 and ICPR 2020. He is a fellow of IAPR and of ELLIS.